Microsoft

Microsoft Ready

Microsoft

AZT337P

The Future of Cloud Native Applications

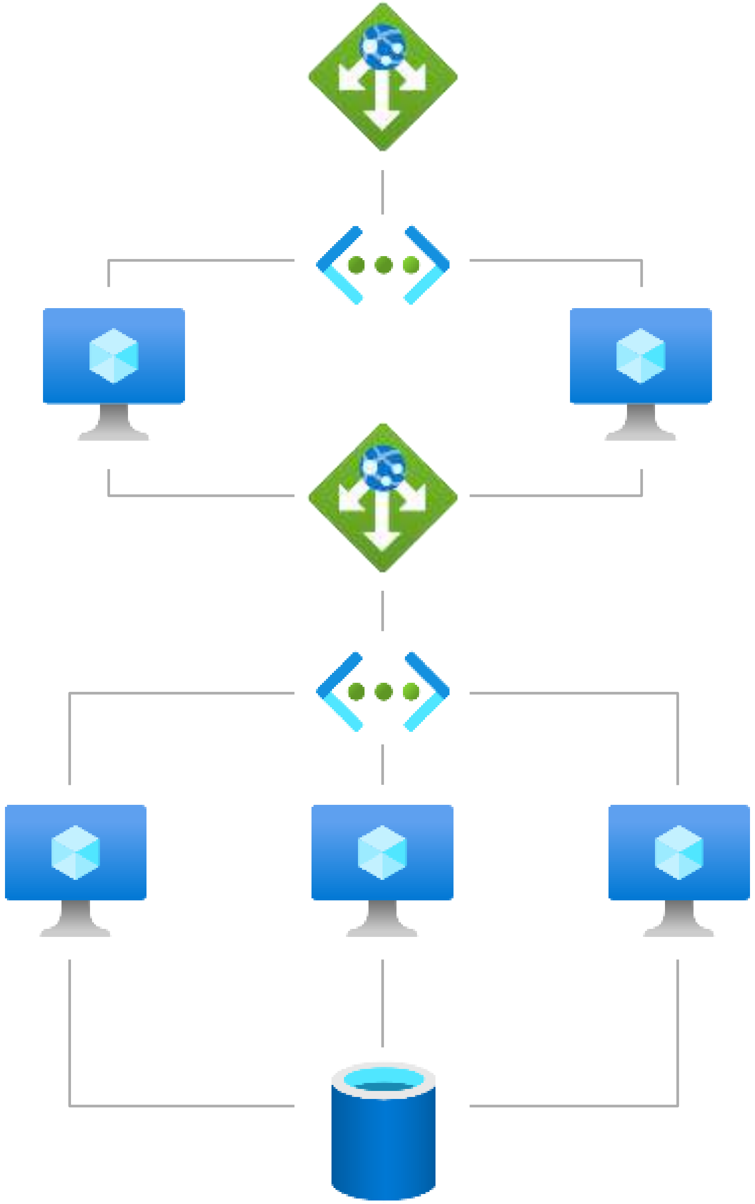
with Open Application Model (OAM) and Dapr

Mark Russinovich
Chief Technology Officer, Microsoft Azure

@markrussinovich

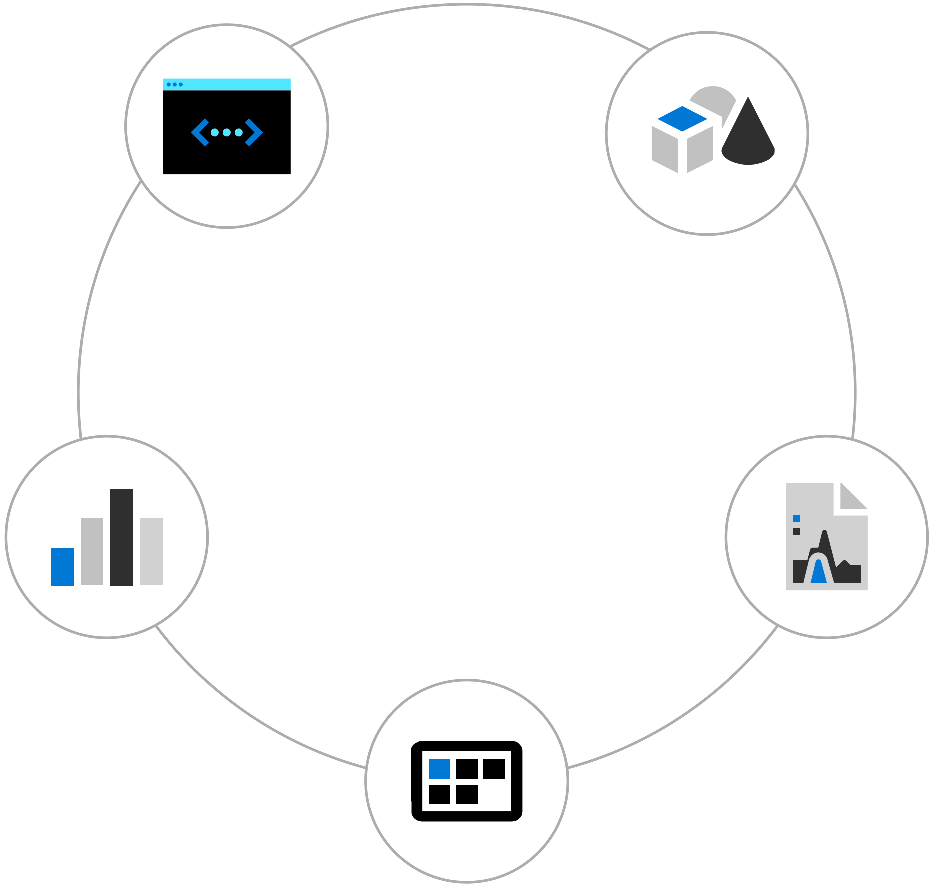
Application models

Describes the topology of your application and its components



Programming models

The way developers write their application to interact with other services and data stores



Open Application Model (OAM)

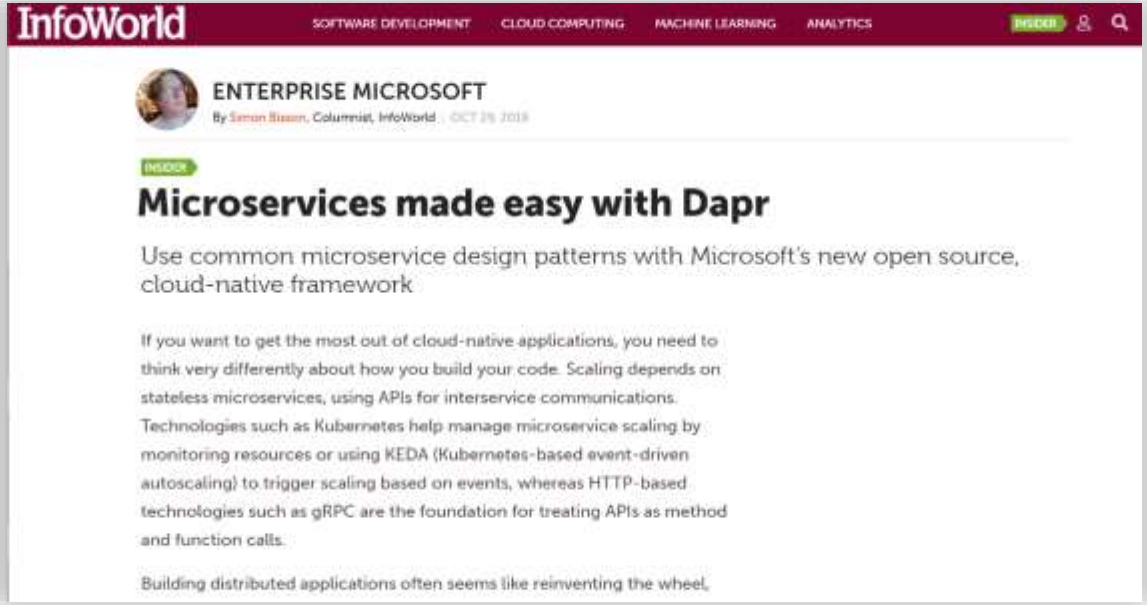
Open Application Model

Platform agnostic application model

Distributed Application Runtime (Dapr)

dapr

Building blocks for building scalable distributed apps



ENTERPRISE MICROSOFT

By Simon Bisson, Columnist, InfoWorld | OCT 29, 2019

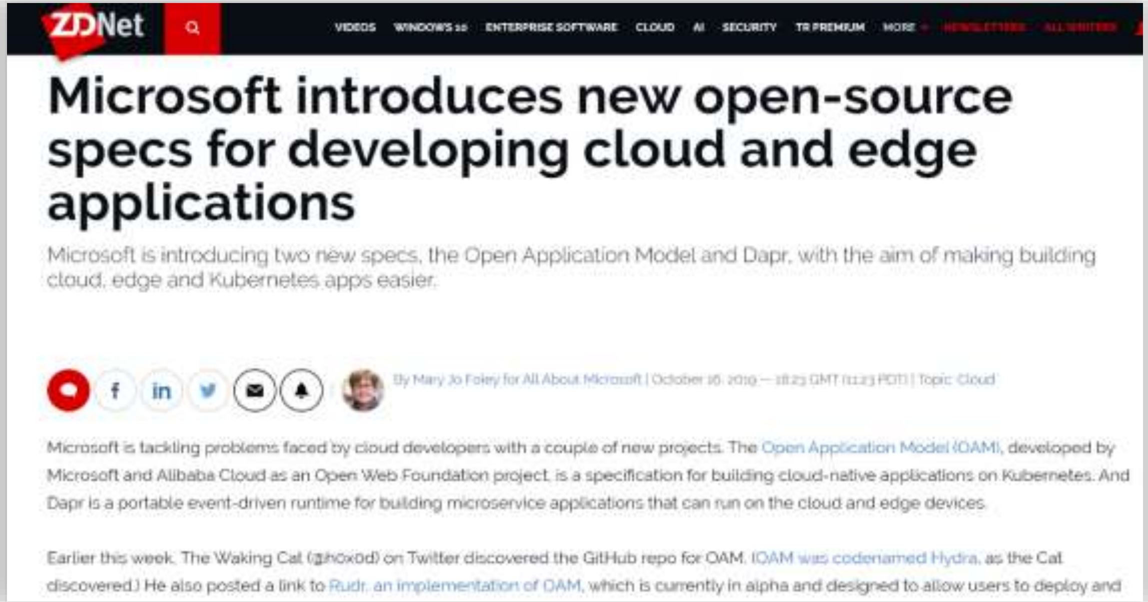
INSIDER

Microservices made easy with Dapr

Use common microservice design patterns with Microsoft's new open source, cloud-native framework

If you want to get the most out of cloud-native applications, you need to think very differently about how you build your code. Scaling depends on stateless microservices, using APIs for interservice communications. Technologies such as Kubernetes help manage microservice scaling by monitoring resources or using KEDA (Kubernetes-based event-driven autoscaling) to trigger scaling based on events, whereas HTTP-based technologies such as gRPC are the foundation for treating APIs as method and function calls.

Building distributed applications often seems like reinventing the wheel,



Microsoft introduces new open-source specs for developing cloud and edge applications

Microsoft is introducing two new specs, the Open Application Model and Dapr, with the aim of making building cloud, edge and Kubernetes apps easier.

By Mary Jo Foley for All About Microsoft | October 16, 2019 -- 18:23 GMT (11:23 PDT) | Topic: Cloud

Microsoft is tackling problems faced by cloud developers with a couple of new projects. The Open Application Model (OAM), developed by Microsoft and Alibaba Cloud as an Open Web Foundation project, is a specification for building cloud-native applications on Kubernetes. And Dapr is a portable event-driven runtime for building microservice applications that can run on the cloud and edge devices.

Earlier this week, The Waking Cat (@hoxod) on Twitter discovered the GitHub repo for OAM. (OAM was codenamed Hydra, as the Cat discovered.) He also posted a link to Rudr, an implementation of OAM, which is currently in alpha and designed to allow users to deploy and

Microsoft launches new open-source projects around Kubernetes and microservices

Frederic Lardinois @fredericl / 9:00 am PDT • October 16, 2019

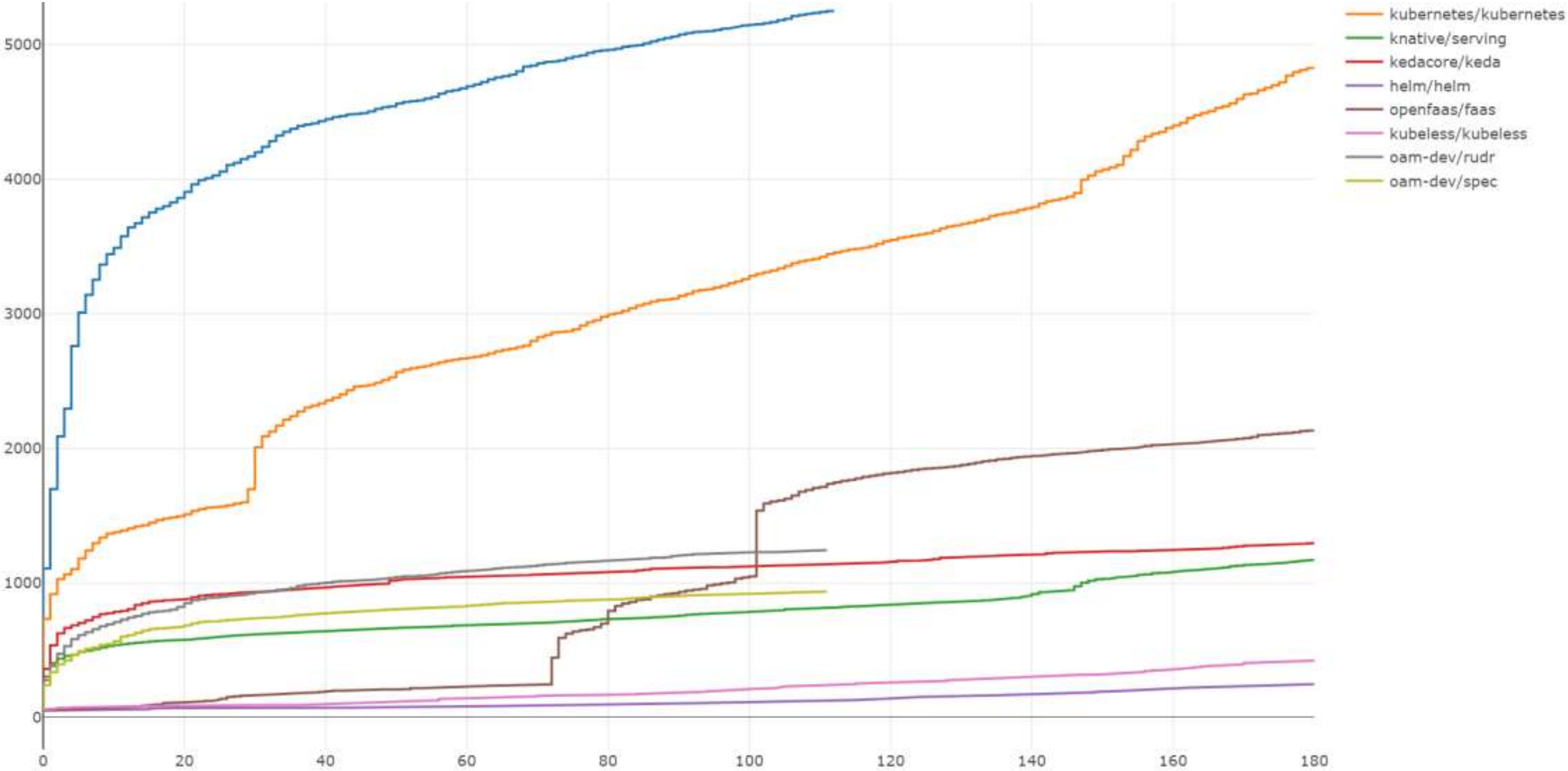
Image Credits: Alex Tai/SOPA Images/LightRocket via Getty Images / Getty Images

OAM, dapr, and rudr

The future of cloud native applications

Mark Russinovich
CTO, Microsoft Azure
@markrussinovich

BRK3098

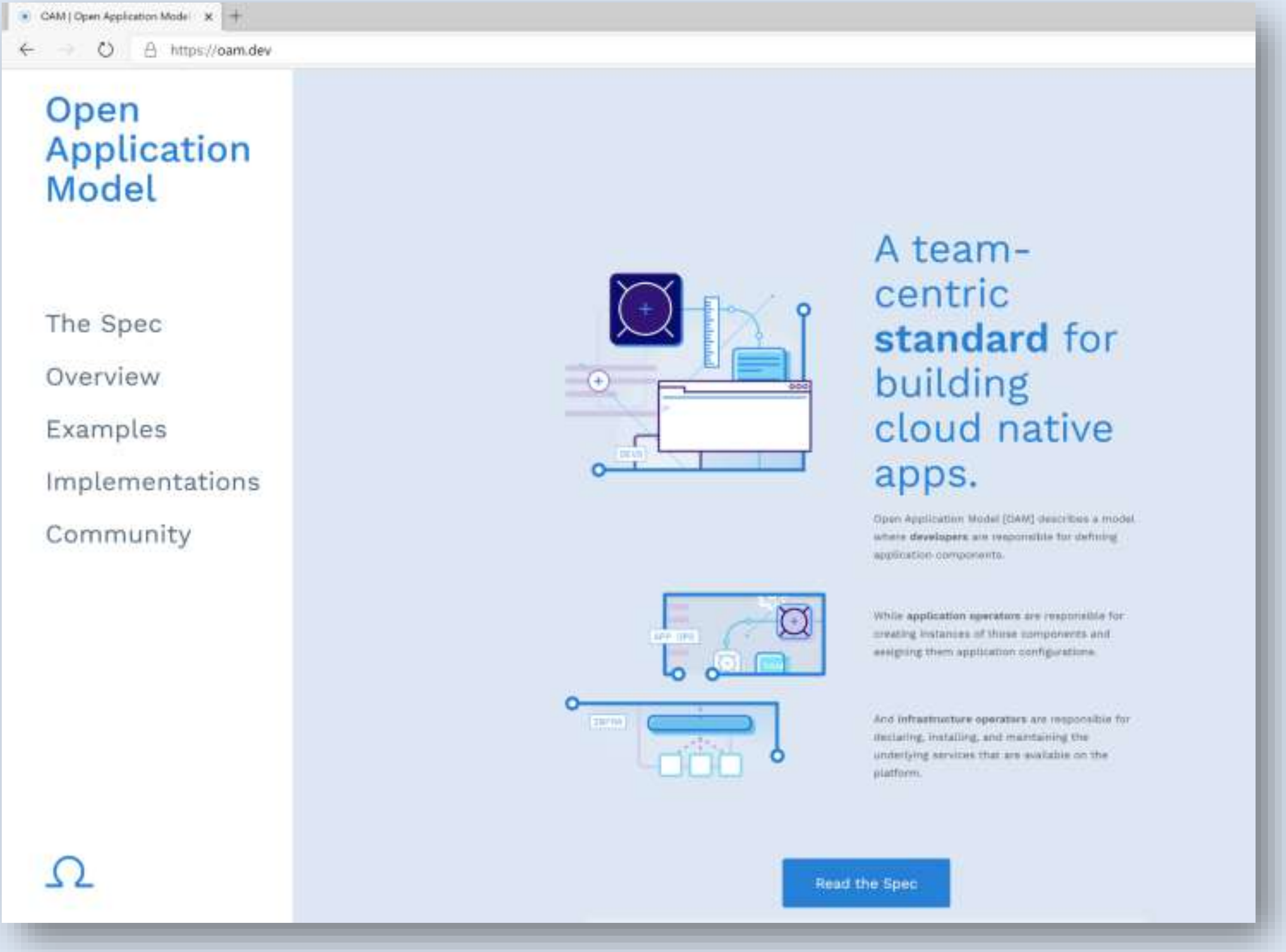


5000
4000
3000
2000
1000
0
0
20
40
60
80
100
120
140
160
180
kubernetes/kubernetes
knative/serving
kedacore/keda
helm/helm
openfaas/faas
kubeless/kubeless
oam-dev/rudr
oam-dev/spec

Open Application Model

Application model
for Cloud and Edge

https://oam.dev



Open Application Model

- The Spec
- Overview
- Examples
- Implementations
- Community

A team-centric **standard** for building cloud native apps.

Open Application Model [OAM] describes a model where **developers** are responsible for defining application components.

While **application operators** are responsible for creating instances of those components and assigning them application configurations.

And **infrastructure operators** are responsible for declaring, installing, and maintaining the underlying services that are available on the platform.

State of Cloud Native Application Platforms

The cloud is going serverless, but Kubernetes is the infrastructure on-premise and on-edge

App developers need to know and code for each infrastructure they deploy to

Kubernetes for applications

Kubernetes focuses on container infrastructure, not on applications

Application developers need to be experts in Kubernetes APIs

Production use of Kubernetes requires mastery of the broader cloud-native ecosystem

"[Kubernetes] is really hard to get into it and understand how all the parts play together, even for experienced people."

—Software Architect @ crisp RESEARCH A CLOUDFLIGHT COMPANY

"A key principle for us when it comes to choosing a platform is that we can maintain the size of our team."

—CTO @ Handled

Open Application Model

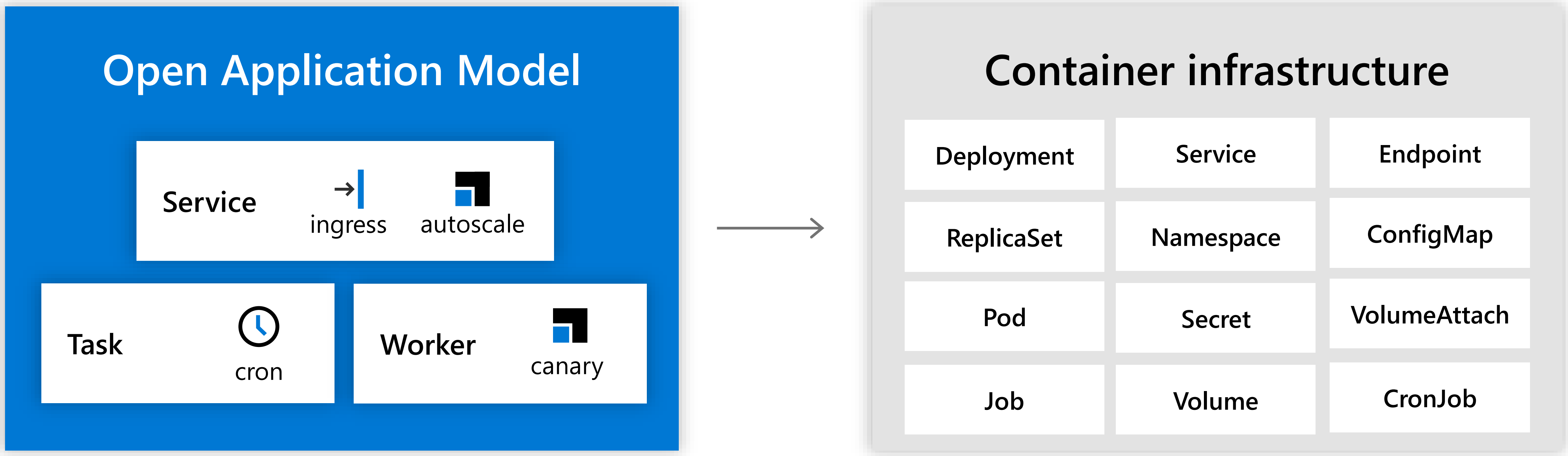
Application focused

Separation of concerns

Cloud + Edge

Application focused

- ✓ Describes application components and operations as first-class concepts without having to stitch together individual container primitives
- ✓ Flexible application modeling supports a wide range of application architectures
- ✓ Small and simple applications are easy, large and complex applications are manageable



Container infrastructure

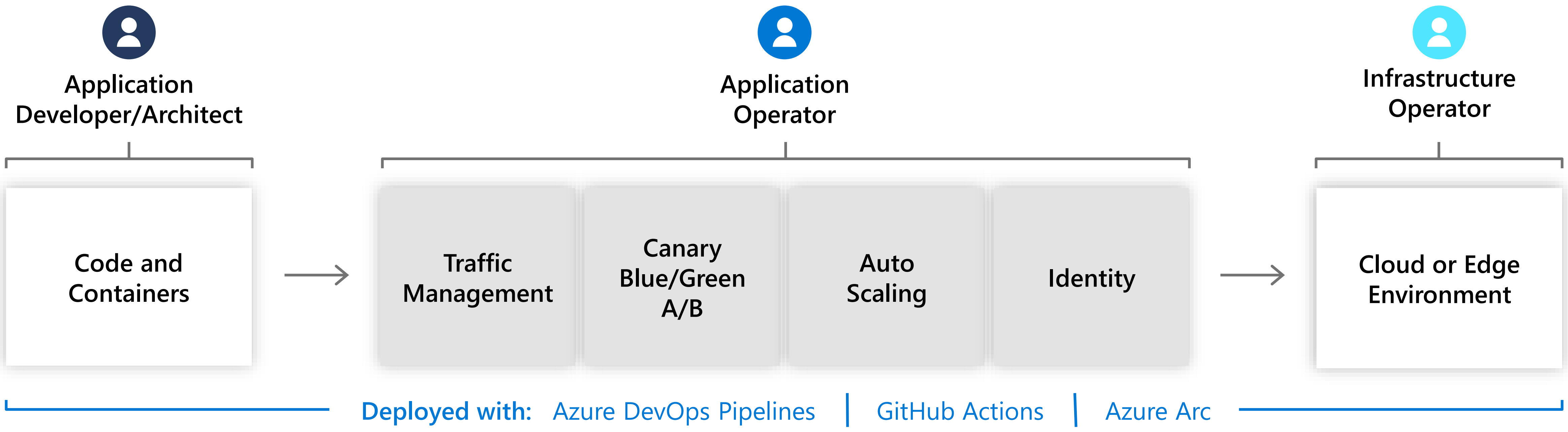
Deployment	Service	Endpoint
ReplicaSet	Namespace	ConfigMap
Pod	Secret	VolumeAttach
Job	Volume	CronJob

Separation of concerns

✓ Allows application developers to focus on their code in a platform-neutral setting to deliver business value

✓ Application operators use powerful and extensible operational traits consistently across platforms and environments

✓ Infrastructure operators can configure their environments to satisfy any unique operating requirements



Application Developer/Architect

Code and Containers

→

Application Operator

Traffic Management

Canary Blue/Green A/B

Auto Scaling

Identity

→

Infrastructure Operator

Cloud or Edge Environment

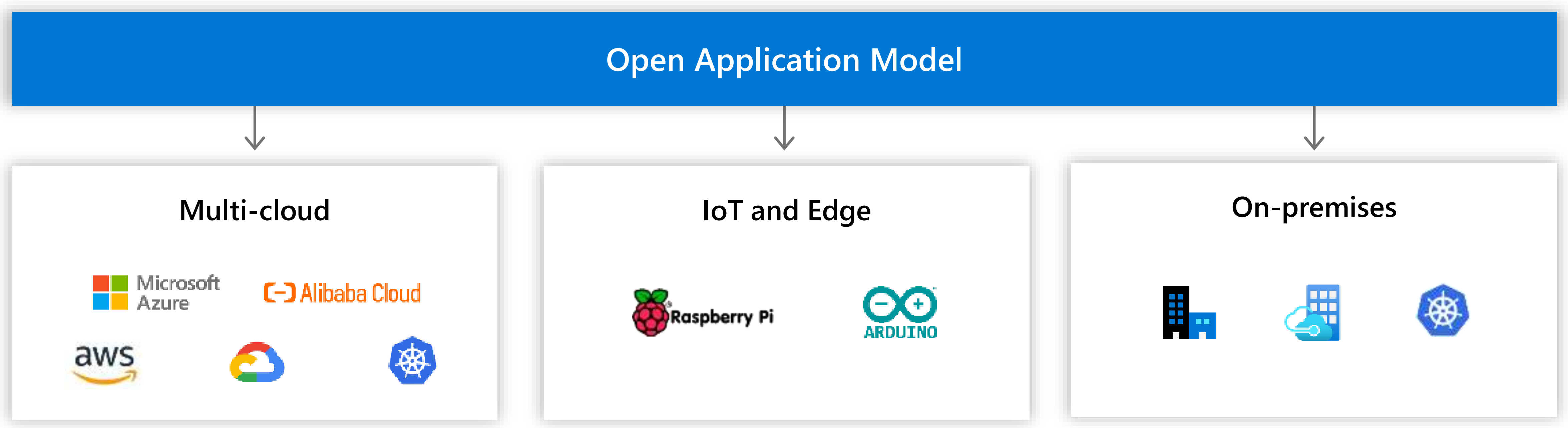
Deployed with: Azure DevOps Pipelines | GitHub Actions | Azure Arc

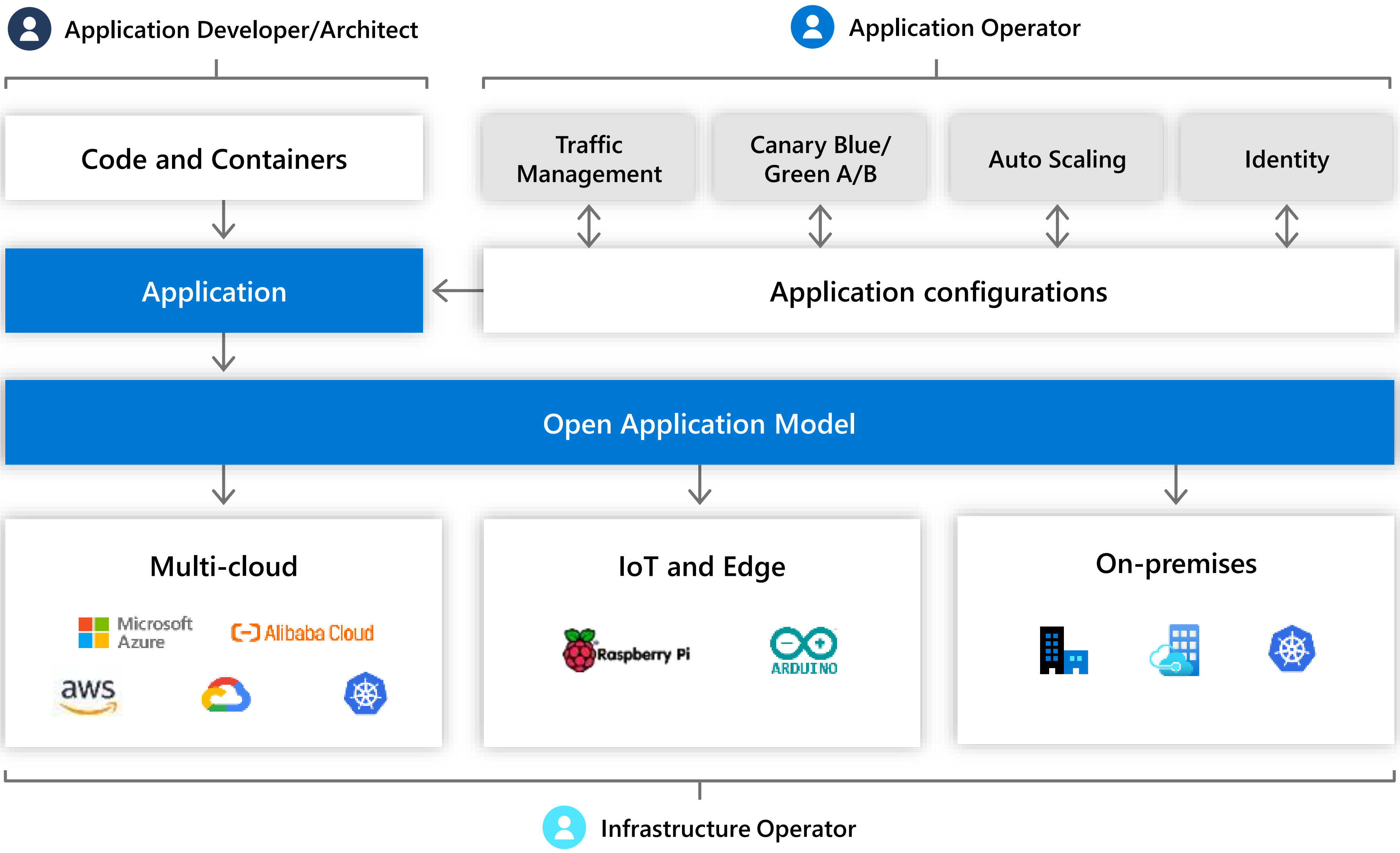
Cloud + Edge

✓ A standard, platform-agnostic application definition for any platform in any environment

✓ Consistent application modeling for small devices, Kubernetes on-premises or cloud, and fully-managed cloud environments

✓ Extendable by design to leverage the native APIs, tools, and unique features of platforms that users know and love





Application Developer/Architect

Application Operator

Code and Containers

Traffic Management

Canary Blue/ Green A/B

Auto Scaling

Identity

Application

Application configurations

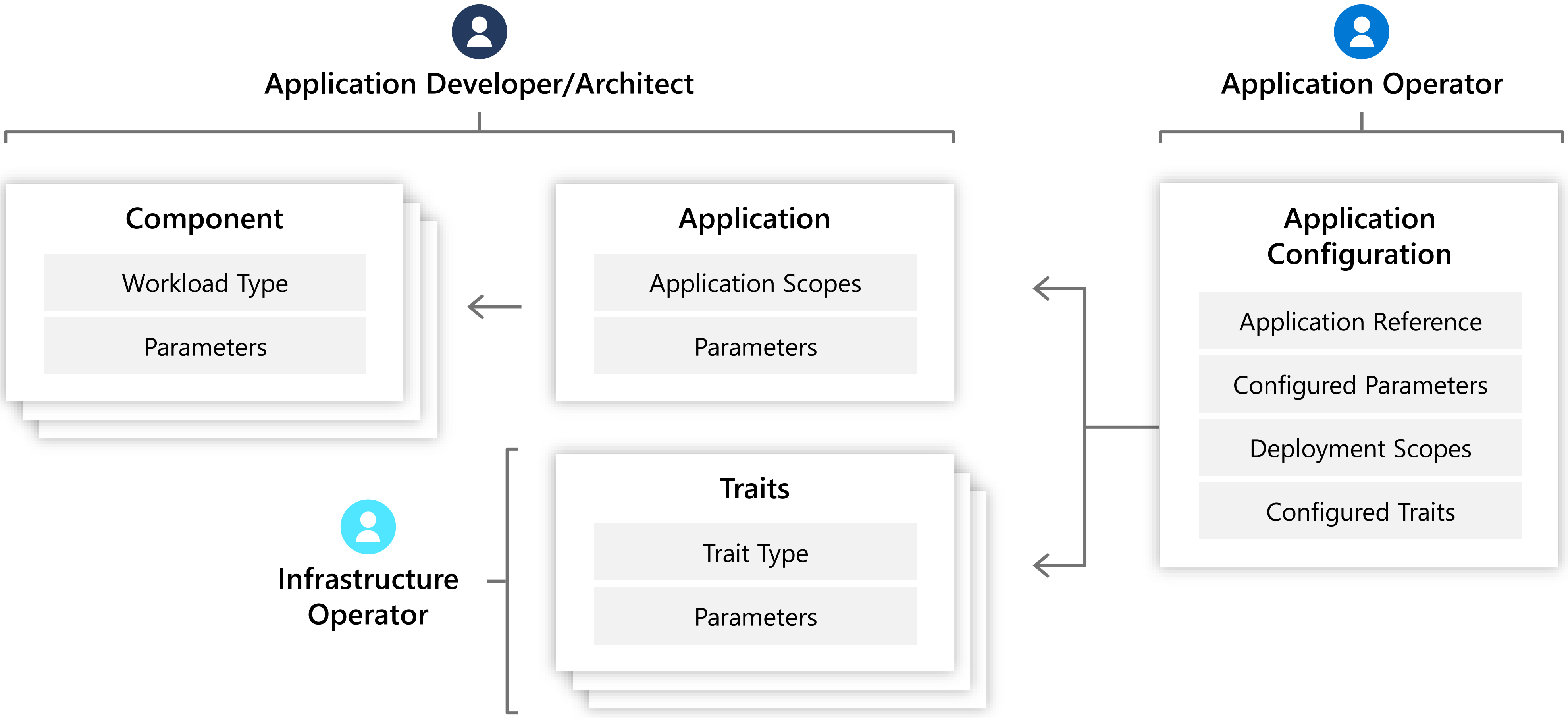
Open Application Model

Multi-cloud

IoT and Edge

On-premises

Infrastructure Operator



Application Developer/Architect

Application Operator

Component

Workload Type

Parameters

Application

Application Scopes

Parameters

Application Configuration

Application Reference

Configured Parameters

Deployment Scopes

Configured Traits

Infrastructure Operator

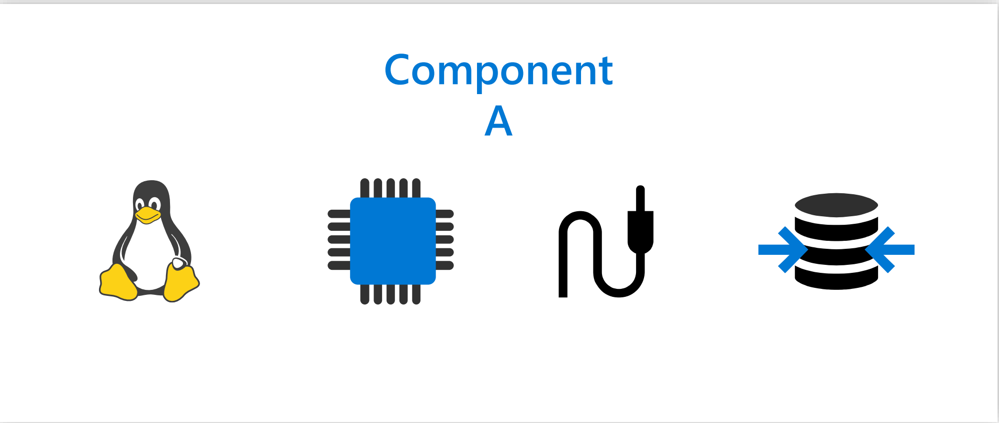
Traits

Trait Type

Parameters

Component

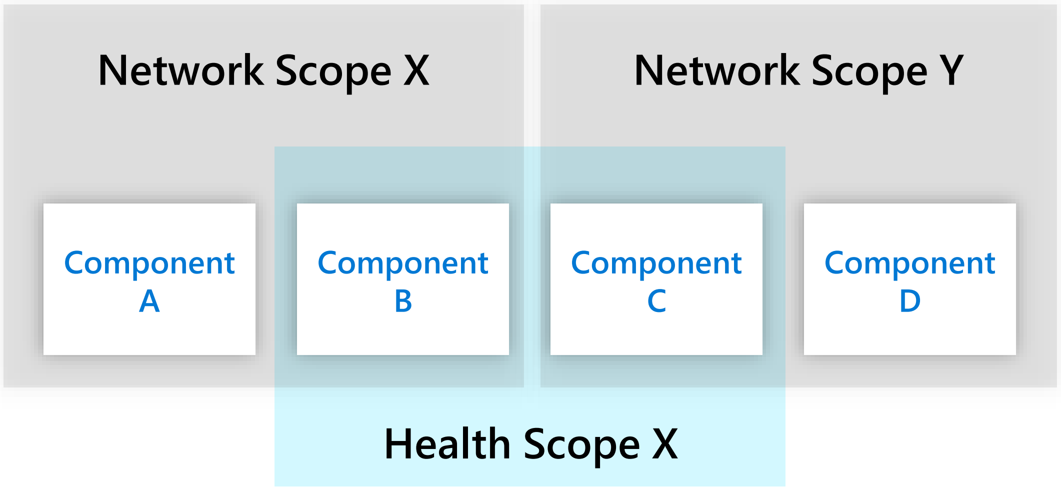
Where developers declare the operational characteristics of the code they deliver *in infrastructure neutral terms*.



```yaml
apiVersion: core.oam.dev/v1alpha1
kind: Component
metadata:
  name: oamfrontend
  version: "1.0.0"
  description: Simple OAM app
spec:
  workloadType: core.oam.dev/v1alpha1.Server
  os: linux
  arch: amd64
  parameters:
    - name: oam_texture
      type: string
      required: true
      default: texture.jpg
  containers:
    - name: frontend
      image: ready2020/hwfrontend:latest
      env:
        - name: OAM_TEXTURE
          value: texture.jpg
          fromParam: oam_texture
      ports:
        - containerPort: 8001
          name: http
          protocol: TCP
```

Application Scope

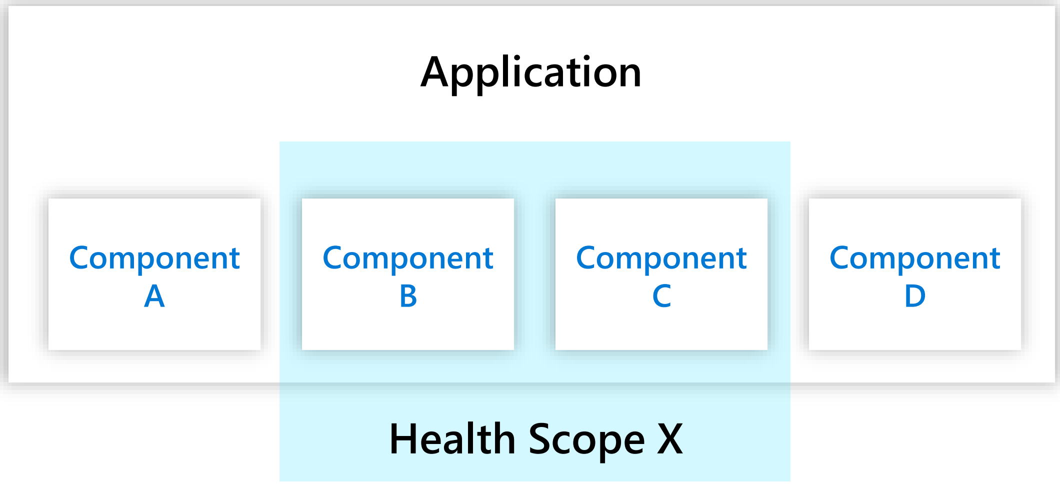
A way to loosely couple components into groups with common characteristics.



```yaml
apiVersion: core.oam.dev/v1alpha1
kind: ApplicationScope
metadata:
  name: network
  annotations:
    version: v1.0.0
    description: "network boundary that a
group of components reside in"
spec:
  type: core.oam.dev/v1.NetworkScope
  allowComponentOverlap: false
  parameters:
    - name: network-id
      description: The id of the network
      type: string
      required: Y
    - name: subnet-id
      description: The id of the subnet
      type: string
      required: Y
    - name: internet-gateway-type
      description: The type of the gateway.
      type: string
      required: N
```

Application

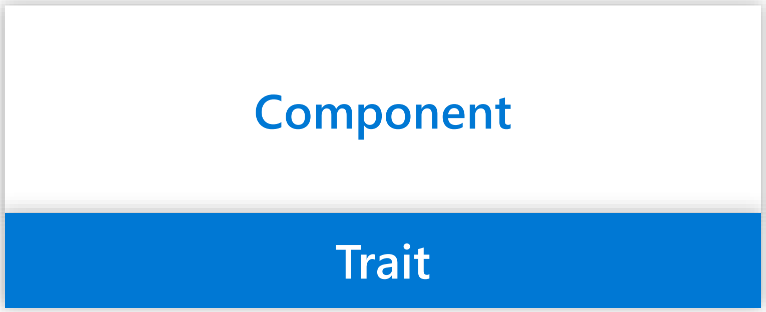
Where developers group components together into a single, deployable unit and specifies cross-component info, such as health scopes.



```
apiVersion: core.oam.dev/v1alpha1
kind: Application
metadata:
name: oam-helloworld-app
spec:
 components:
  - name: oamfrontend
  - name: oambackend
scopes:
  - name: oam-be-fe-metrics
   type: core.oam.dev/v1.HealthScope
   parameters:
    - name: metrics-endpoint
     protocol: https
     path: /metrics
```

Trait

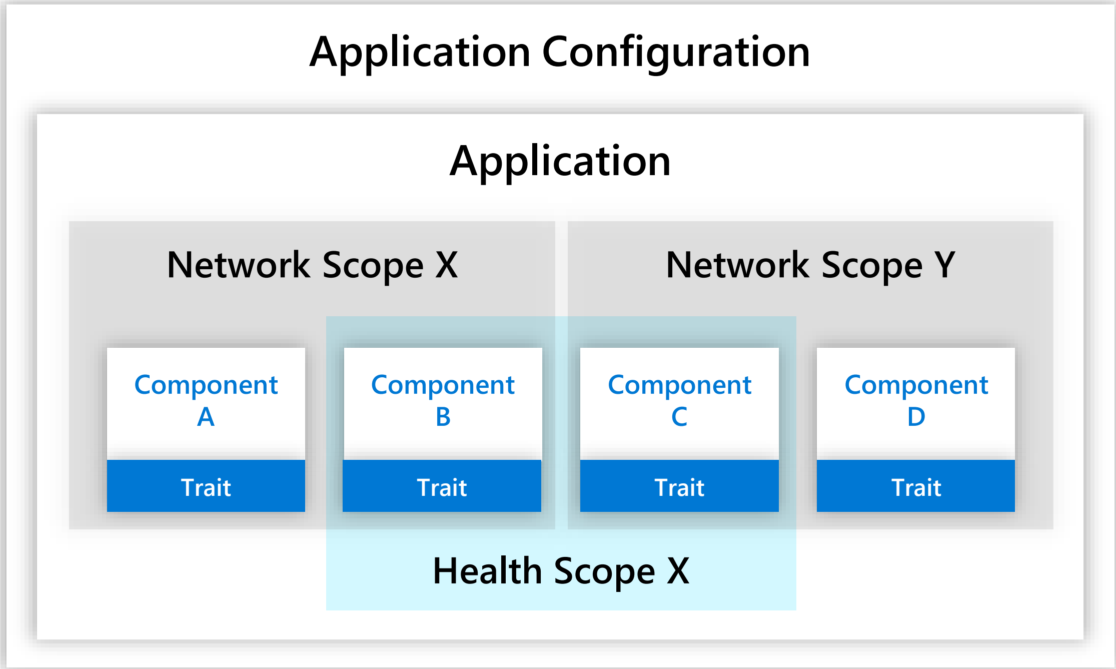
For assigning operational features to instances of components.



```yaml
apiVersion: core.oam.dev/v1alpha1
kind: ApplicationConfiguration
metadata:
  name: demo-scale
spec:
  components:
    - componentName: oamfrontend
      instanceName: oam-fe
      traits:
        - name: manual-scaler
          properties:
            replicaCount: 1
        - name: ingress
          properties:
            hostname: aks.azureocto.com
            path: /
            servicePort: 8001
```

Application Configuration

Defines a configuration of an application, its traits, and additional scopes, such as network scopes.



```
apiVersion: core.oam.dev/v1alpha1
kind: ApplicationConfiguration
metadata:
  name: oam-helloworld
spec:
  components:
    - componentName: oamfrontend
      instanceName: oam-fe1
      parameterValues:
        - name: oam_texture
          value: aks
      traits:
        - name: manual-scaler
          properties:
            replicaCount: 1
        - name: ingress.core.oam.dev/v1alpha1
          properties:
            hostname: aks.azureocto.com
            path: /
            servicePort: 8001
    - componentName: oambackend
      instanceName: oam-be1
```

rudr

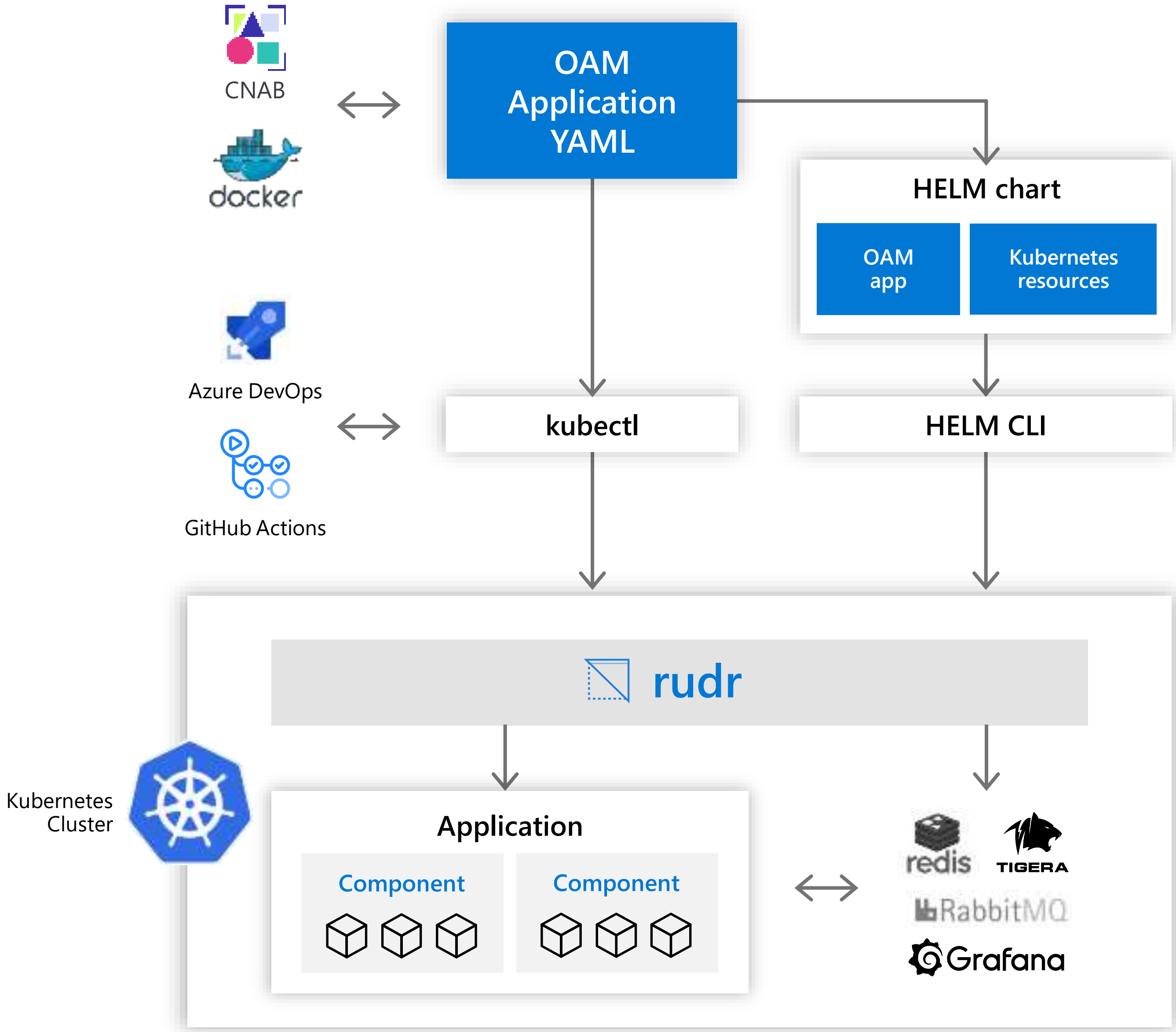
Open Application Model on Kubernetes

Build and operate cloud-native applications on the leading open source orchestrator

Application developers can focus on business value, not on container primitives and plumbing

CRDs combine high-level application modeling with familiar Kubernetes concepts

Infra operators continue to use familiar Kubernetes infrastructure, APIs, and domain knowledge



DEMO

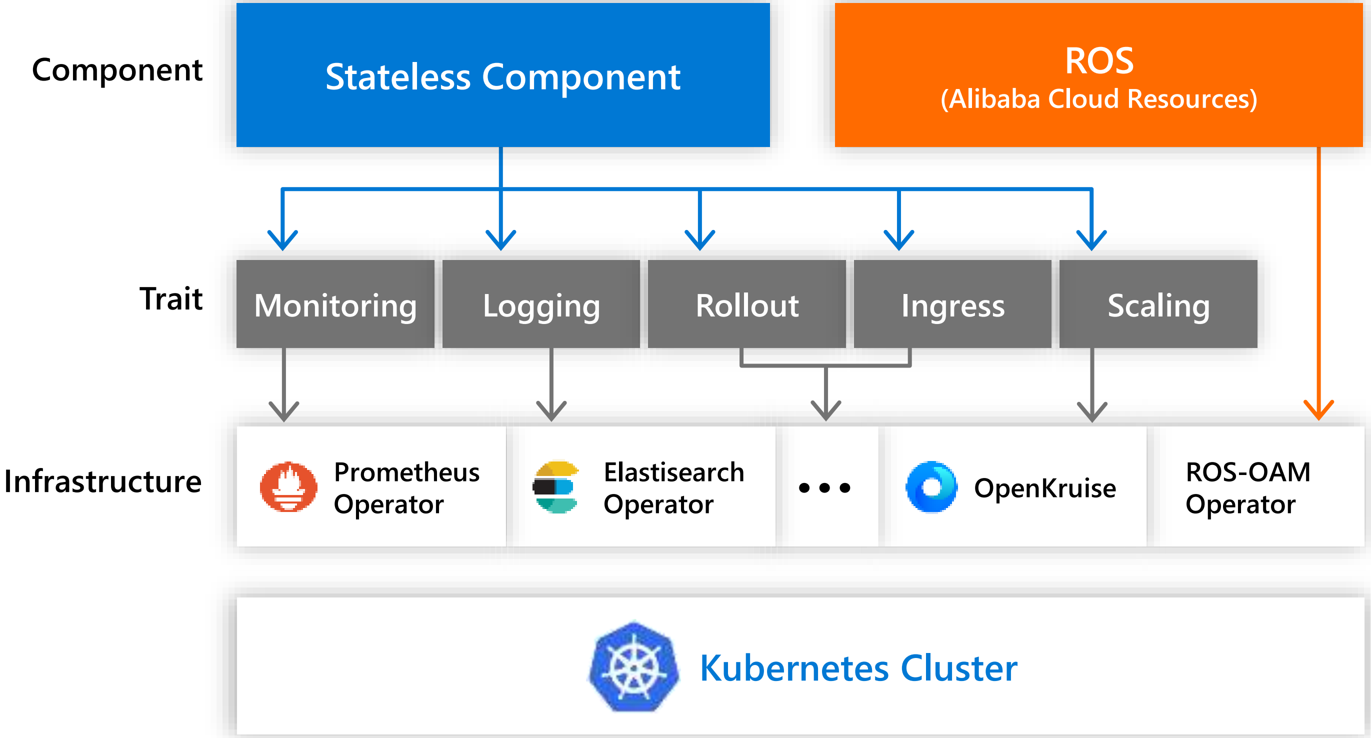
Deploying an OAM application to rudr

Alibaba Cloud

Enterprise Distributed Application Service (EDAS)

OAM-based PaaS implementation

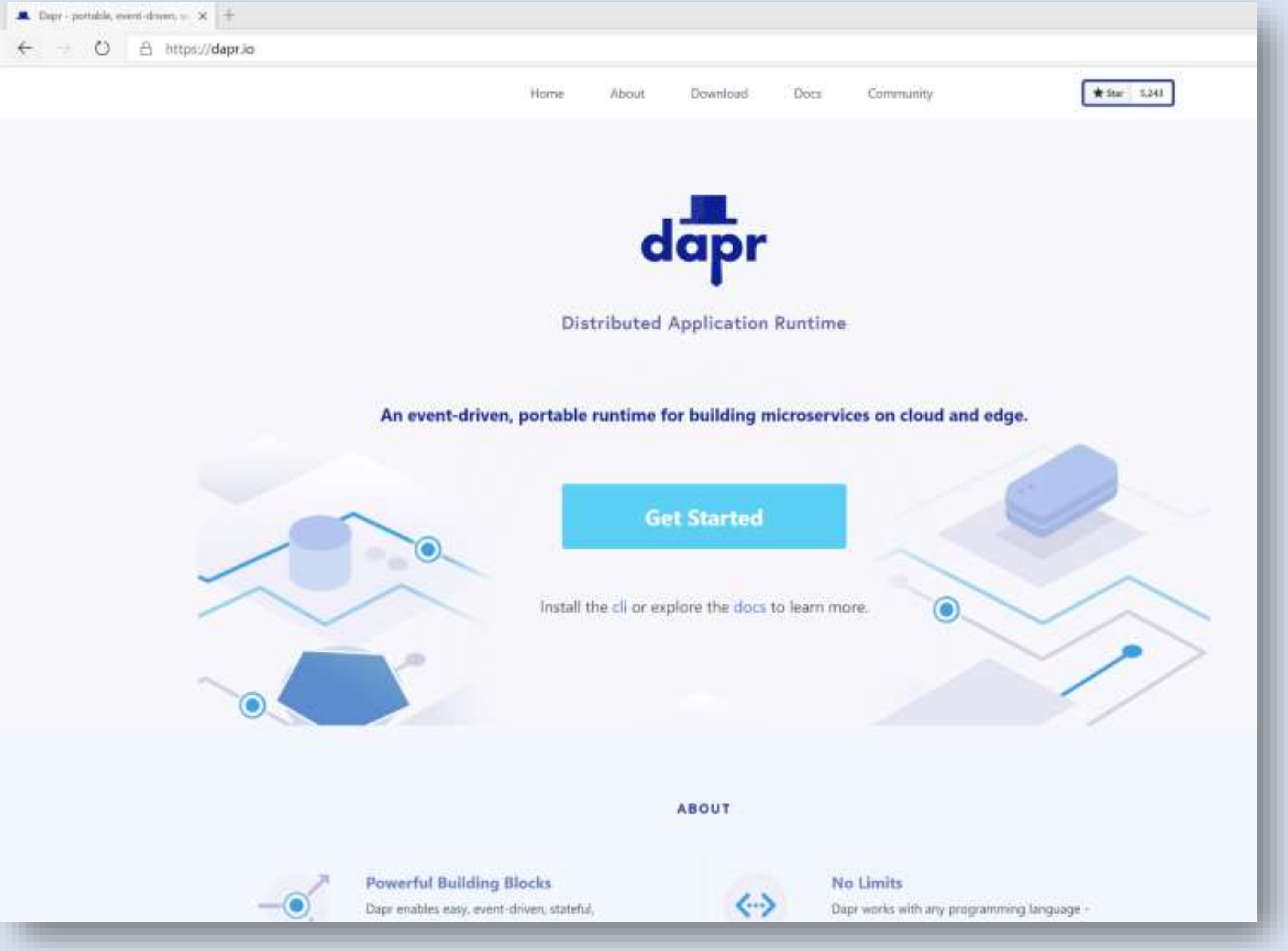
- ✓ Empower app developers to focus on building and delivering apps without concerning operations
- ✓ Provide manageability of CRDs, consistency of app model, portability of app profiles
- ✓ Give platform team flexibility to choose and operate the infra tools in their domain knowledge by adopting OAM



Distributed Application Runtime

Portable, event-driven, runtime for building distributed applications across cloud and edge

https://dapr.io



Distributed Application Runtime

An event-driven, portable runtime for building microservices on cloud and edge.

Install the cli or explore the docs to learn more.

ABOUT

Powerful Building Blocks

Dapr enables easy, event-driven, stateful,

No Limits

Dapr works with any programming language -

State of Enterprise Developers

Being asked to develop resilient, scalable, microservice-based apps

They write in many languages

They want to leverage existing code

Functions and Actors are powerful programming models

What is holding back micro-service development?

Hard to incrementally migrate from existing code to a microservices architecture

Programming model runtimes have narrow language support and tightly controlled feature sets

Runtimes only target specific infrastructure platforms with limited code portability across clouds and edge dapr

Sidecar architecture

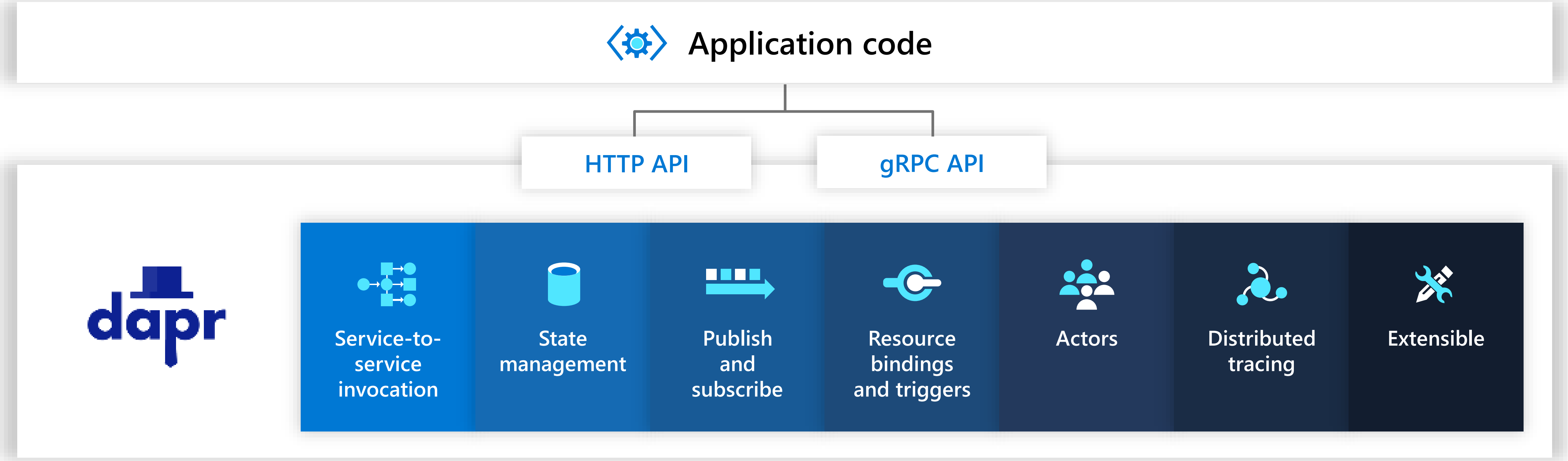
Cloud + Edge

Microservice building blocks

Sidecar architecture

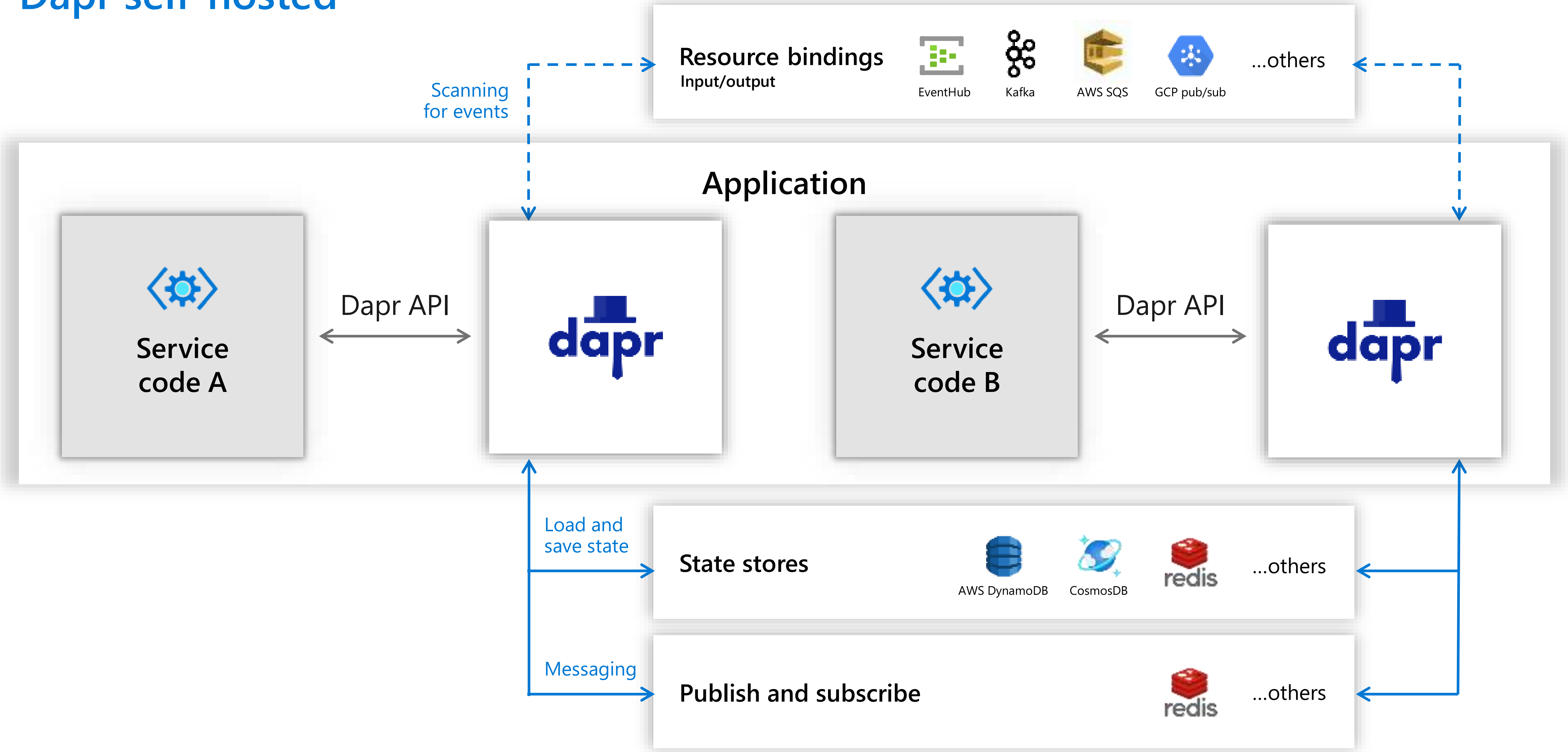
✓ Standard APIs accessed over http/gRPC protocols from user service code
e.g. http://localhost:3500/v1.0/state/inventory

✓ Runs as local “sidecar library” dynamically loaded at runtime for each service



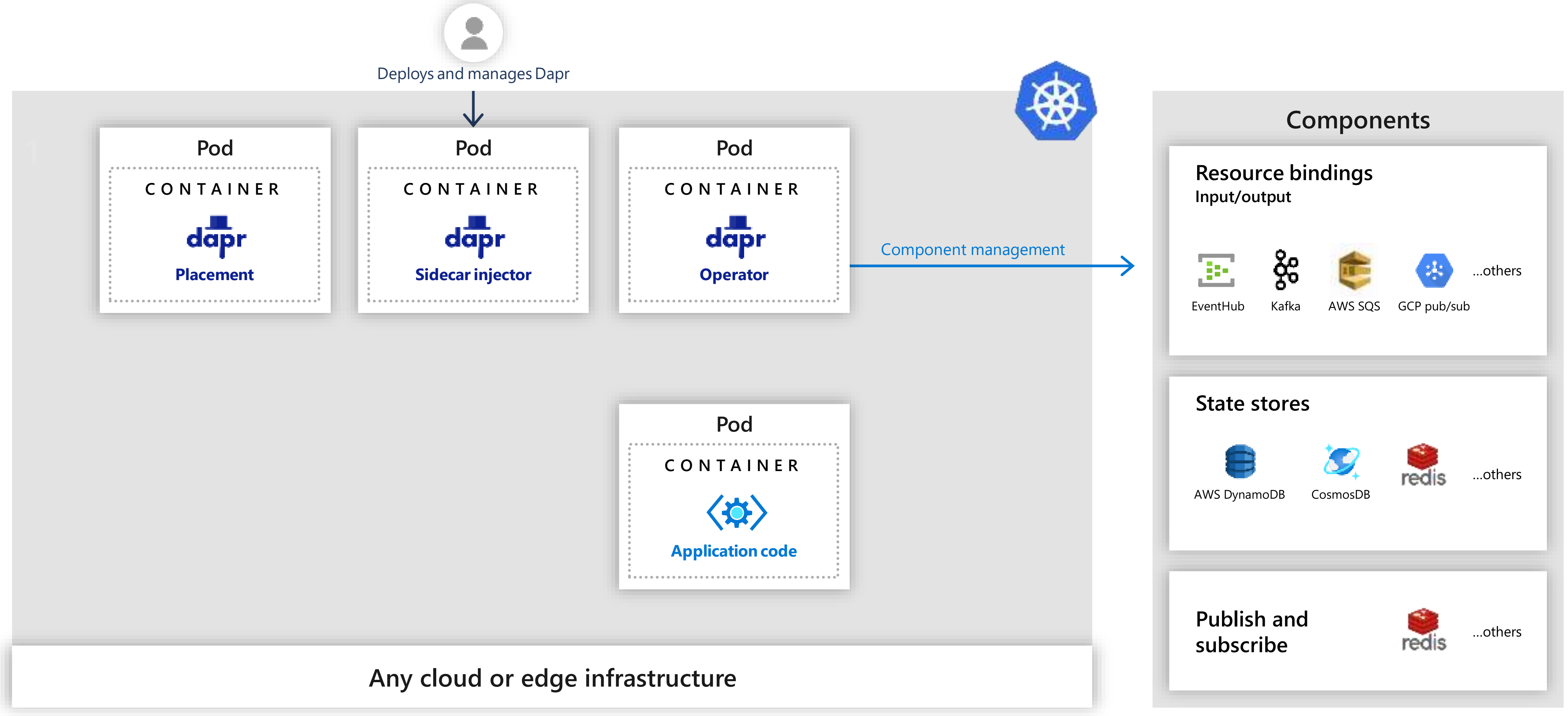
Sidecar architecture

Dapr self-hosted



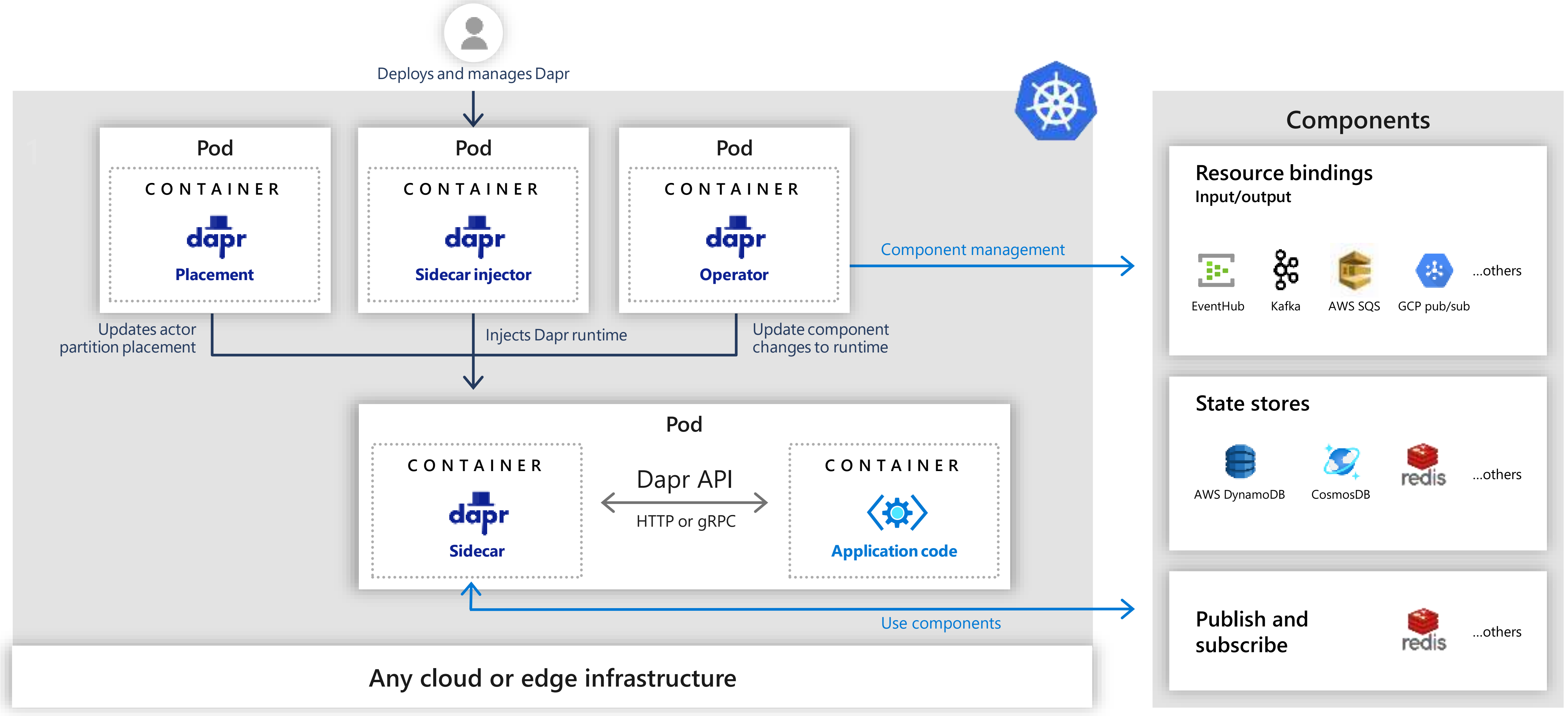
Sidecar architecture

Dapr Kubernetes-hosted



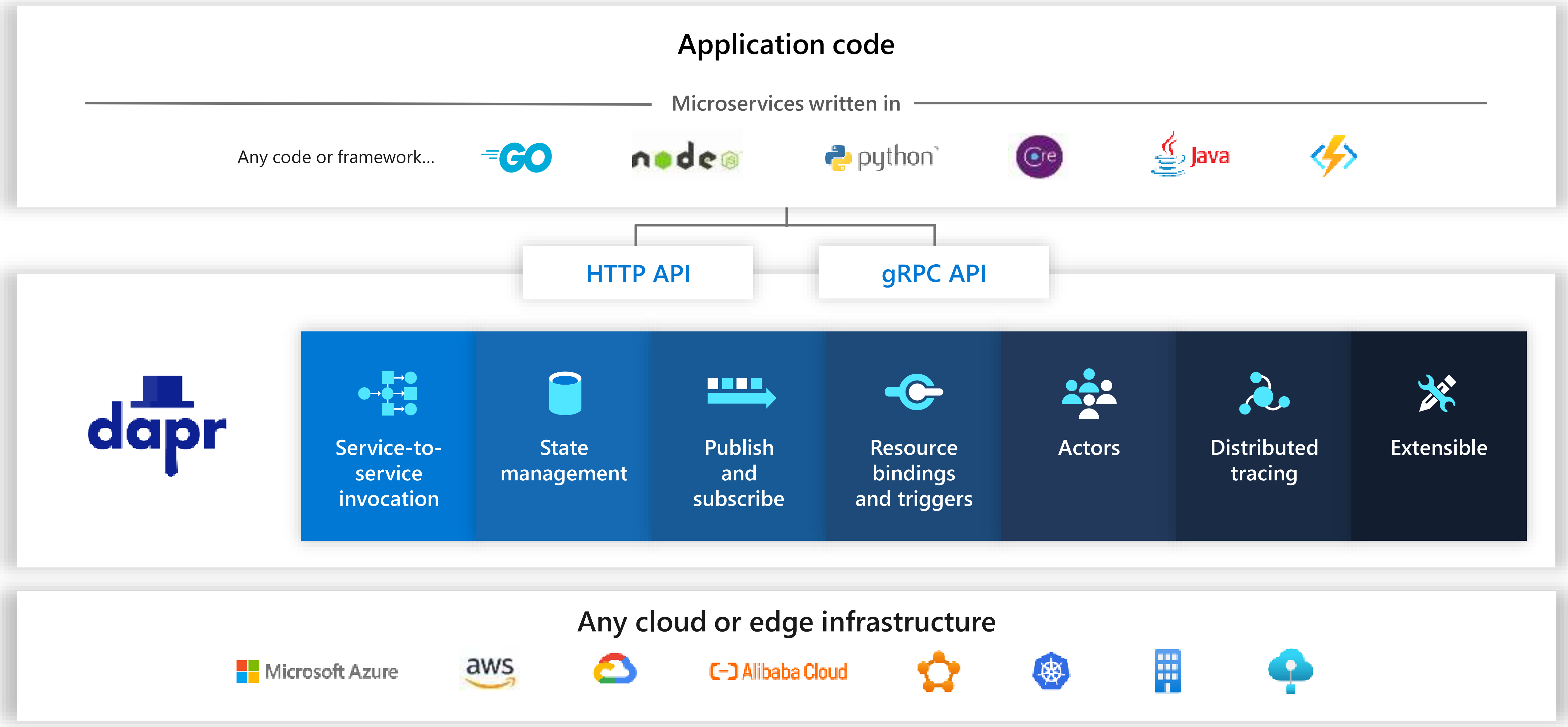
Sidecar architecture

Dapr Kubernetes-hosted



Cloud + Edge

Build apps using any language with any framework



Application code

Microservices written in

Any code or framework...

HTTP API

gRPC API

Service-to-service invocation

State management

Publish and subscribe

Resource bindings and triggers

Actors

Distributed tracing

Extensible

Any cloud or edge infrastructure

Microservice building blocks

Service-to-service invocation

Perform direct, secure, service-to-service method calls

State management

Create long running, stateless and stateful services

Publish and subscribe

Secure, scalable messaging between services

Resource bindings and triggers

Trigger code through events from a large array of inputs

Output bindings to external resources including databases and queues

Actors

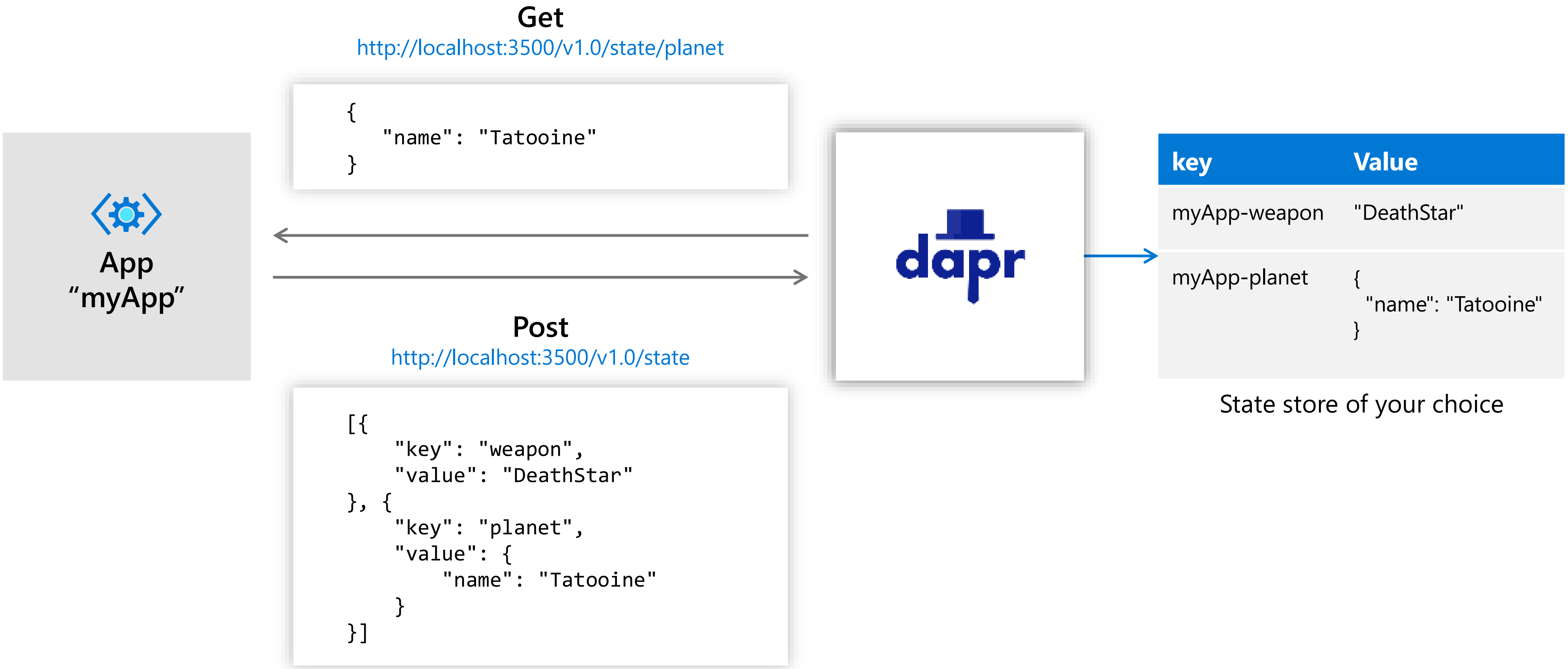
Encapsulate code and data in reusable actor objects as a common microservices design pattern

Distributed tracing

See and measure the message calls across components and networked services

Microservice building blocks

State management



Get

http://localhost:3500/v1.0/state/planet

```
{
   "name": "Tatooine"
}
```

Post

http://localhost:3500/v1.0/state

```
[{
    "key": "weapon",
    "value": "DeathStar"
}, {
    "key": "planet",
    "value": {
        "name": "Tatooine"
    }
}]
```

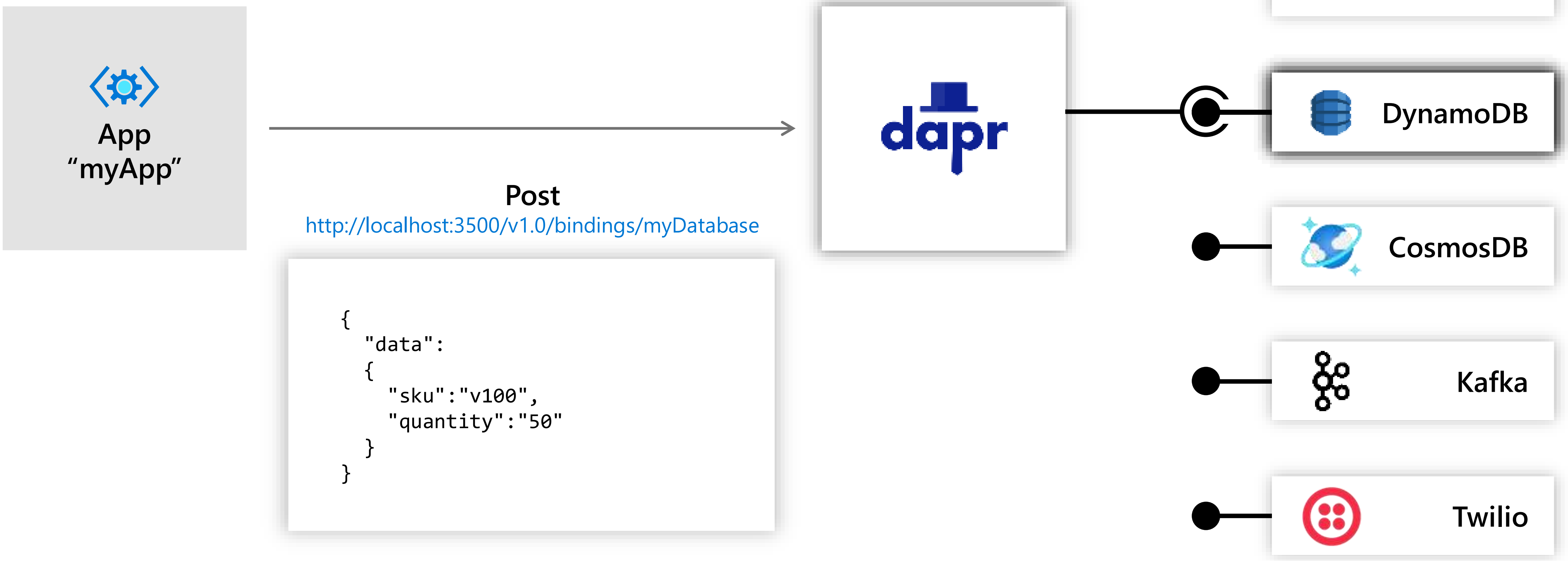
key	Value
myApp-weapon	"DeathStar"

myApp-planet	{
	"name": "Tatooine"
	}

State store of your choice

Microservice building blocks

Resource bindings: Output



App
“myApp”

Post
http://localhost:3500/v1.0/bindings/myDatabase

```
{
  "data":
  {
    "sku":"v100",
    "quantity":"50"
  }
}
```

dapr

Redis

Event Hubs

DynamoDB

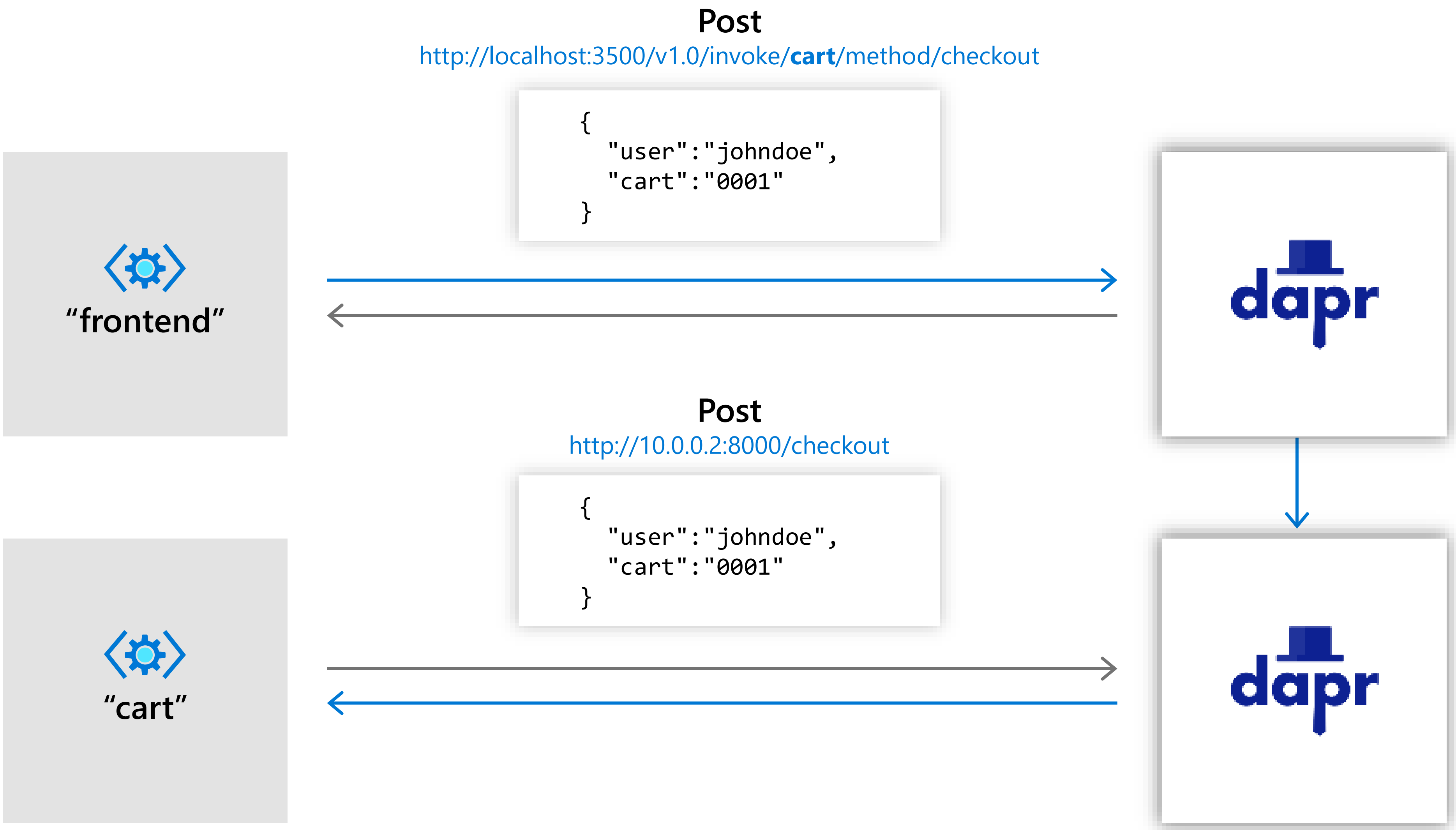
CosmosDB

Kafka

Twilio

Microservice building blocks

Service invocation



“frontend”

Post

http://localhost:3500/v1.0/invoke/**cart**/method/checkout

```
{
  "user":"johndoe",
  "cart":"0001"
}
```

Post

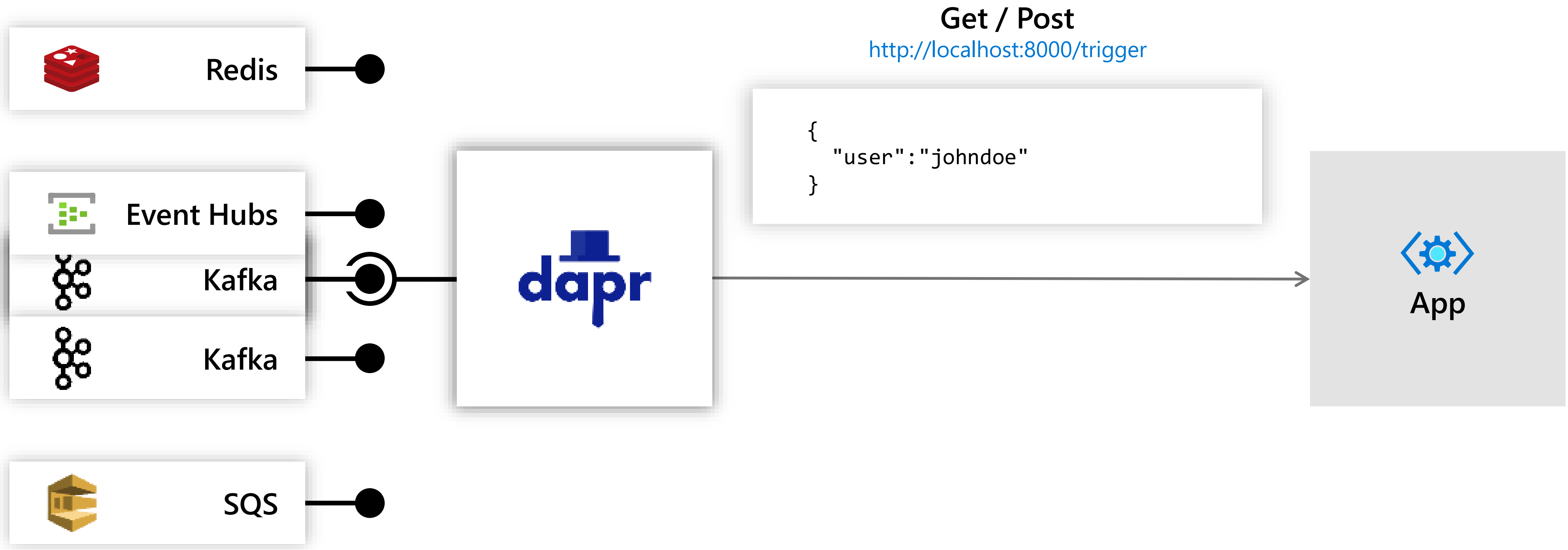
http://10.0.0.2:8000/checkout

```
{
  "user":"johndoe",
  "cart":"0001"
}
```

“cart”

Microservice building blocks

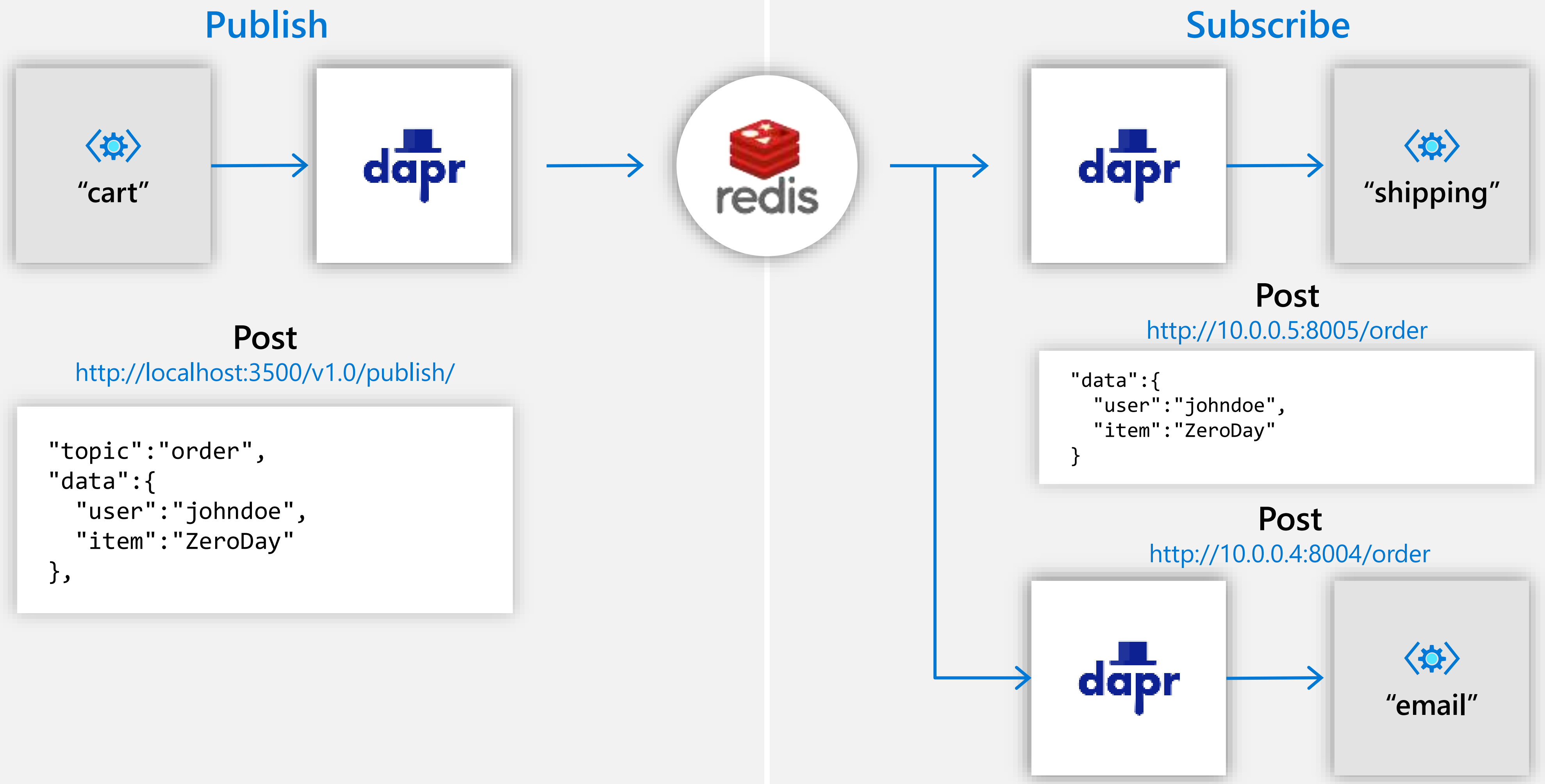
Resource triggers: Input



```
{
  "user":"johndoe"
}
```

Microservice building blocks

Publish and subscribe



Post
http://localhost:3500/v1.0/publish/

```
"topic":"order",
"data":{
  "user":"johndoe",
  "item":"ZeroDay"
},
```

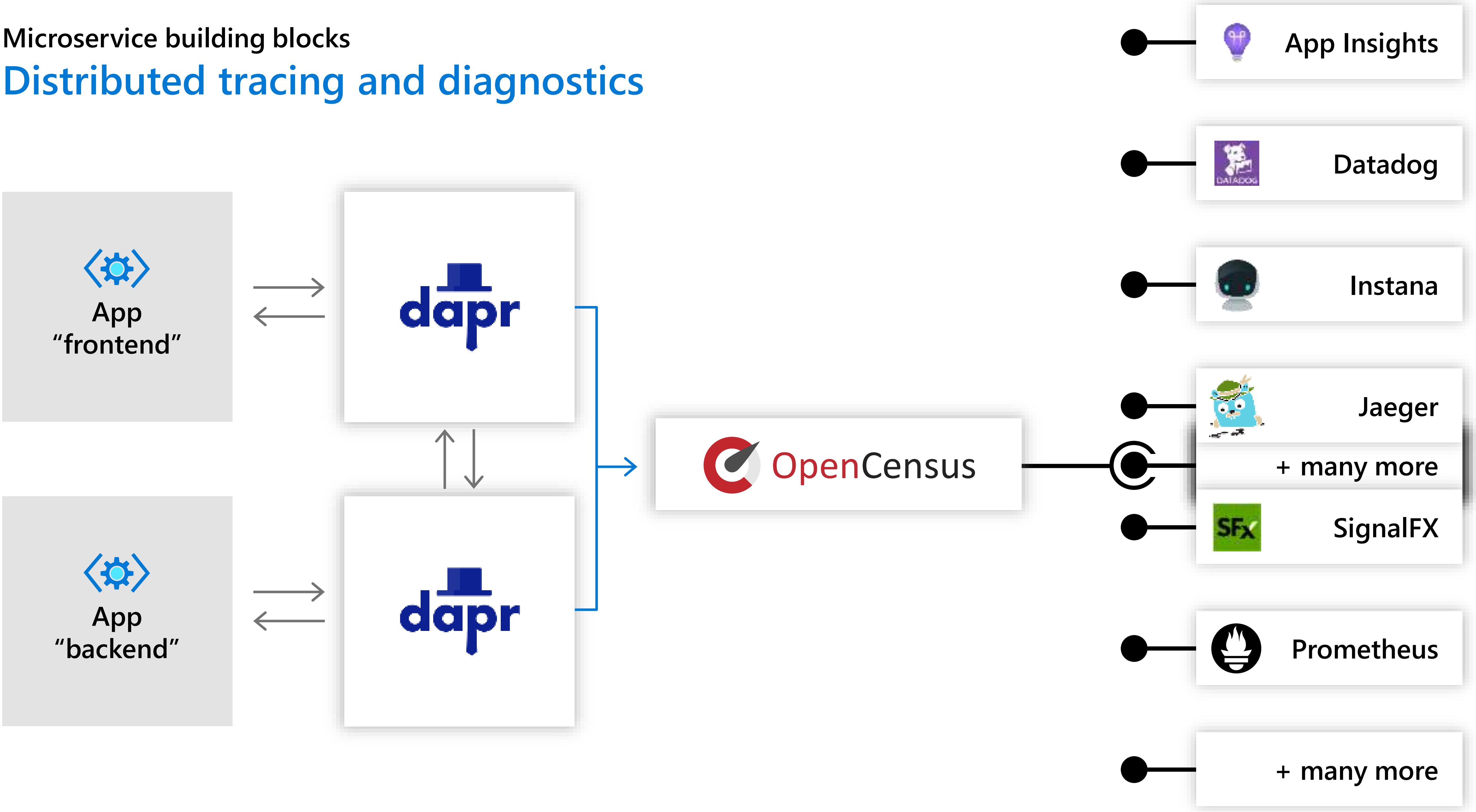
Post
http://10.0.0.5:8005/order

```
"data":{
  "user":"johndoe",
  "item":"ZeroDay"
}
```

Post
http://10.0.0.4:8004/order

Microservice building blocks

Distributed tracing and diagnostics

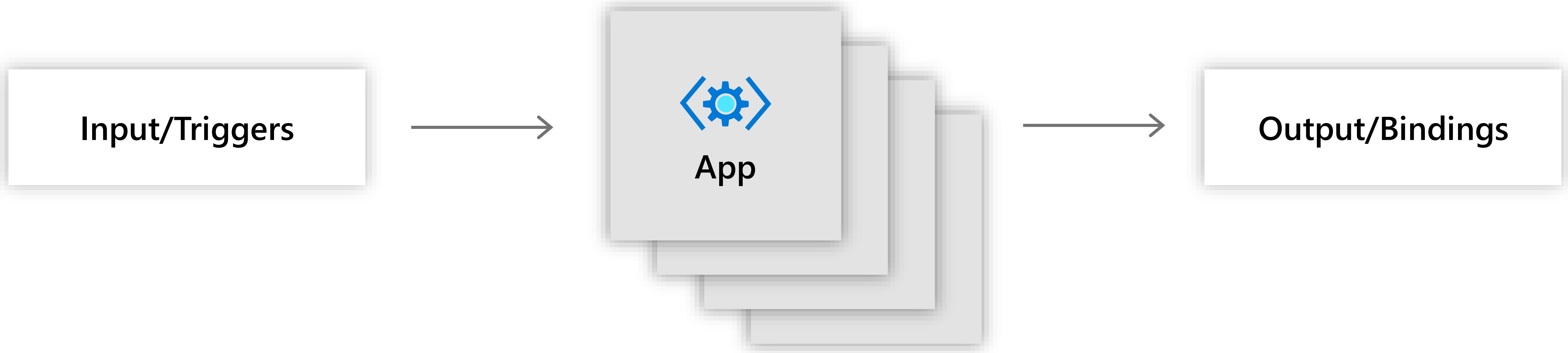


DEMO

Distributed Calculator

Functions with Dapr

✓ Event-driven ✓ Stateless ✓ Easy replication and sharing



DEMO

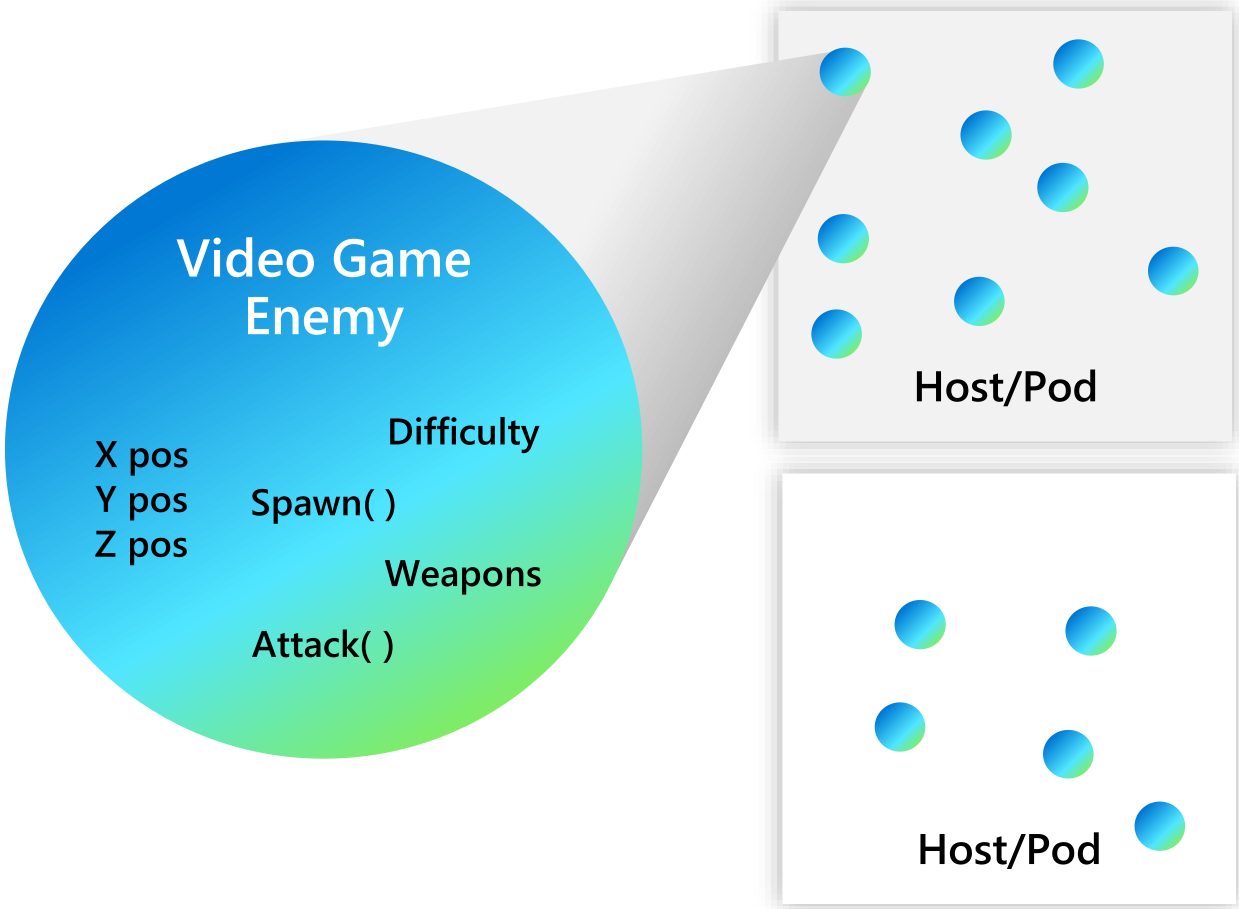
Functions with Dapr

Virtual Actors with Dapr

Stateful, objects of storage and compute

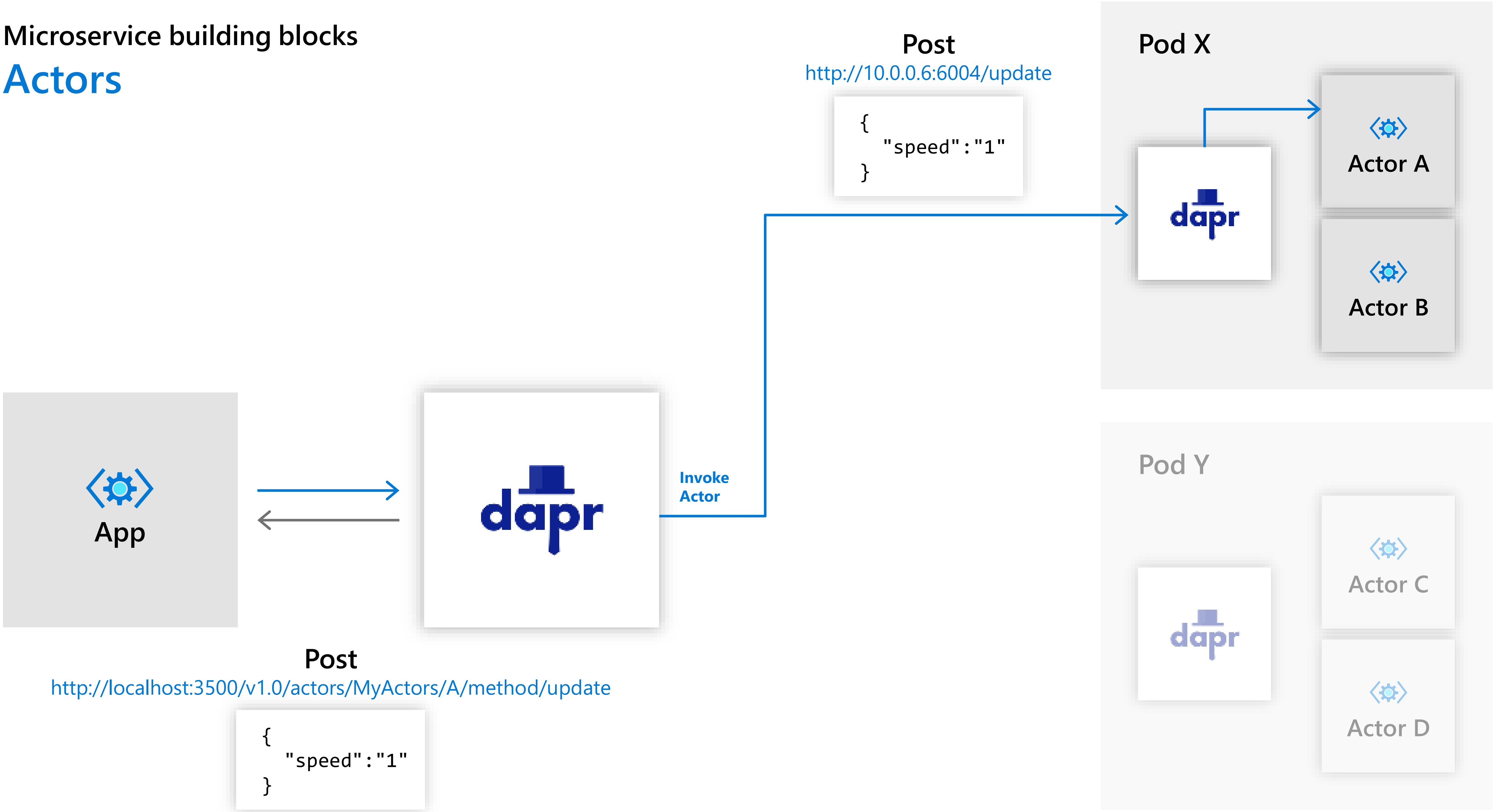
Dapr Actor features:

- ✓ Distribution and failover
- ✓ Turn-based concurrency
- ✓ State management
- ✓ Timers
- ✓ Reminders



Microservice building blocks

Actors



Post
http://10.0.0.6:6004/update

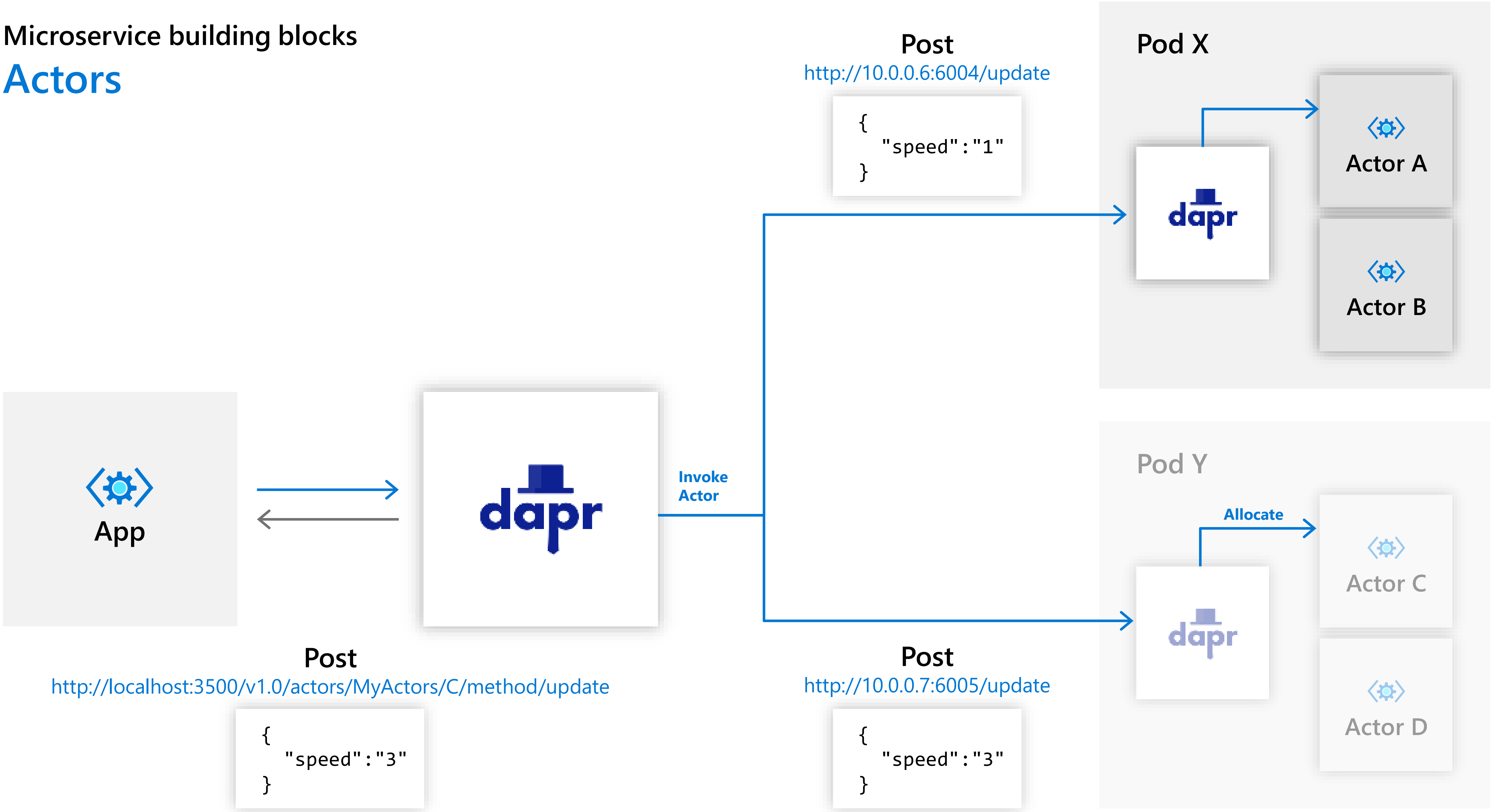
```
{
  "speed":"1"
}
```

Post
http://localhost:3500/v1.0/actors/MyActors/A/method/update

```
{
  "speed":"1"
}
```

Microservice building blocks

Actors



Post
http://10.0.0.6:6004/update

```
{
  "speed":"1"
}
```

Post
http://localhost:3500/v1.0/actors/MyActors/C/method/update

```
{
  "speed":"3"
}
```

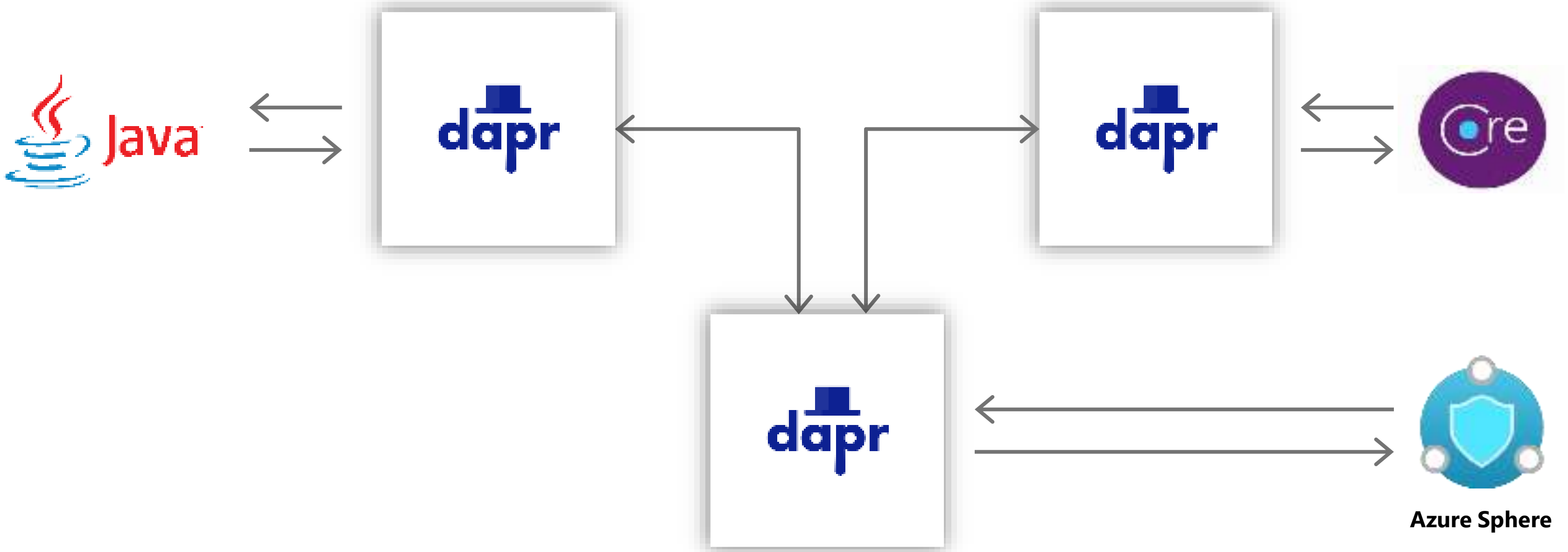
Post
http://10.0.0.7:6005/update

```
{
  "speed":"3"
}
```

DEMO

Cloud Native Parking Garage

Open Application Model



Java Microsoft building 37

Core Microsoft building 99

Community

dapr

GitHub | github.com/dapr

53 Contributors

25 new components added since launch

v1.0 coming later this year

5.2k GitHub stars in under 4 months

rudr

GitHub | github.com/oam-dev

35 Contributors (rudr)
25 Contributors (spec)

Beta draft proposal in review

Roadmap

dapr

GitHub | github.com/dapr

Operability and observability

Integration with more languages

Java/Python SDKs

Integration with Microsoft frameworks

ASP.NET, Functions, Blazor

Integration with more platforms

Kubernetes, IoT Edge, Azure Stack Edge

Production ready

V1.0 later this year

Looking to partner with customers to bring to production

rudr

GitHub | github.com/oam-dev

Specification updates to Open Application Model

External services support

Model more workload types

Integration with more platforms

Blue/green updates

Production ready

V1.0 later this year

Looking to partner with customers to bring to production

Microsoft