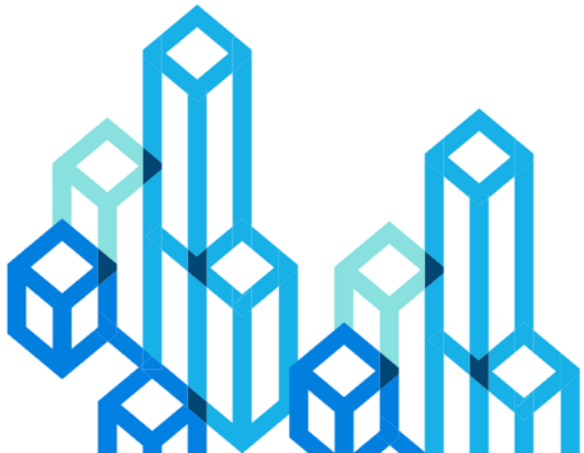
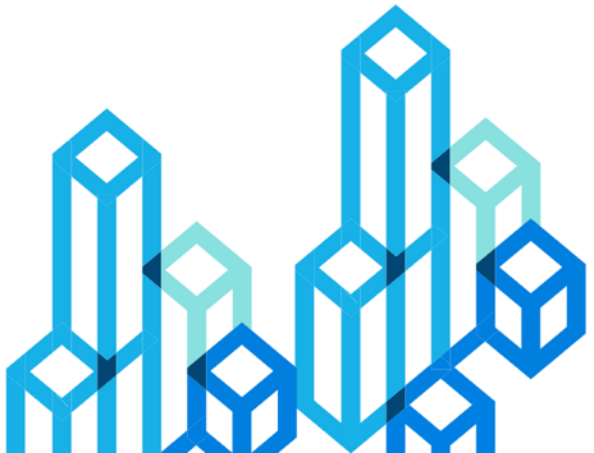
利用 dapr 轻易开发跨云云原生应用

朱永光





朱永光，从业21年，目前在微软咨询服务部任高级技术顾问。在软件架构、研发管理方面有长期的经验，近期在工作中重点关注DevOps和云原生/微服务的落地。曾连任微软最有价值专家（MVP）11届，喜欢在社区中和大家交流技术，闲暇之余会在自己的技术博客和微信公众号上分享一些经验总结文章。

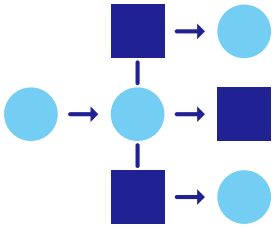
云原生经验谈

- 云原生开发的现状和挑战
- Dapr的起源、目标和发展情况
- Dapr的原理和组成
- 深入Dapr的7大构建块
- Dapr的托管环境
- 多环境Demo
- 其他话题
  - 和Service Mesh的关系
  - Traffic Control示例讲解

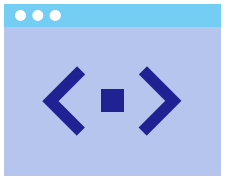
- 指构建软件的过程天生就利用公有云或私有云提供的服务和基础设施，包括……
  - 计算服务
  - 存储服务
  - 弹性伸缩
  - 并行处理
- 云原生应用能基于某种云平台上开发，但是内置的解耦模式让它们可以被部署到其他云平台
- 云原生应用通常使用微服务（以及分布式）架构来构建
- 云原生应用的总体目标就是提升敏捷性、开发速度、伸缩能力和最终收益
- **一切皆为服务**

企业应用开发者面临的挑战

满足灵活性、成本和效率的要求下，开发大规模伸缩的应用

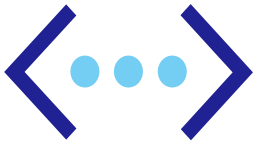


开发需要和其他服务交互的弹性、可伸缩、基于微服务的应用



更加关注于应用程序的构建而非基础设施配置

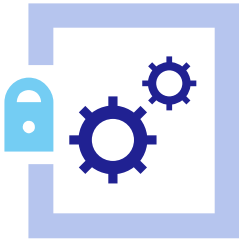
倾向于无服务器平台，将简单的代码部署到云中



在开发过程中需要使用多种语言和框架

开发微服务应用的路上布满荆棘

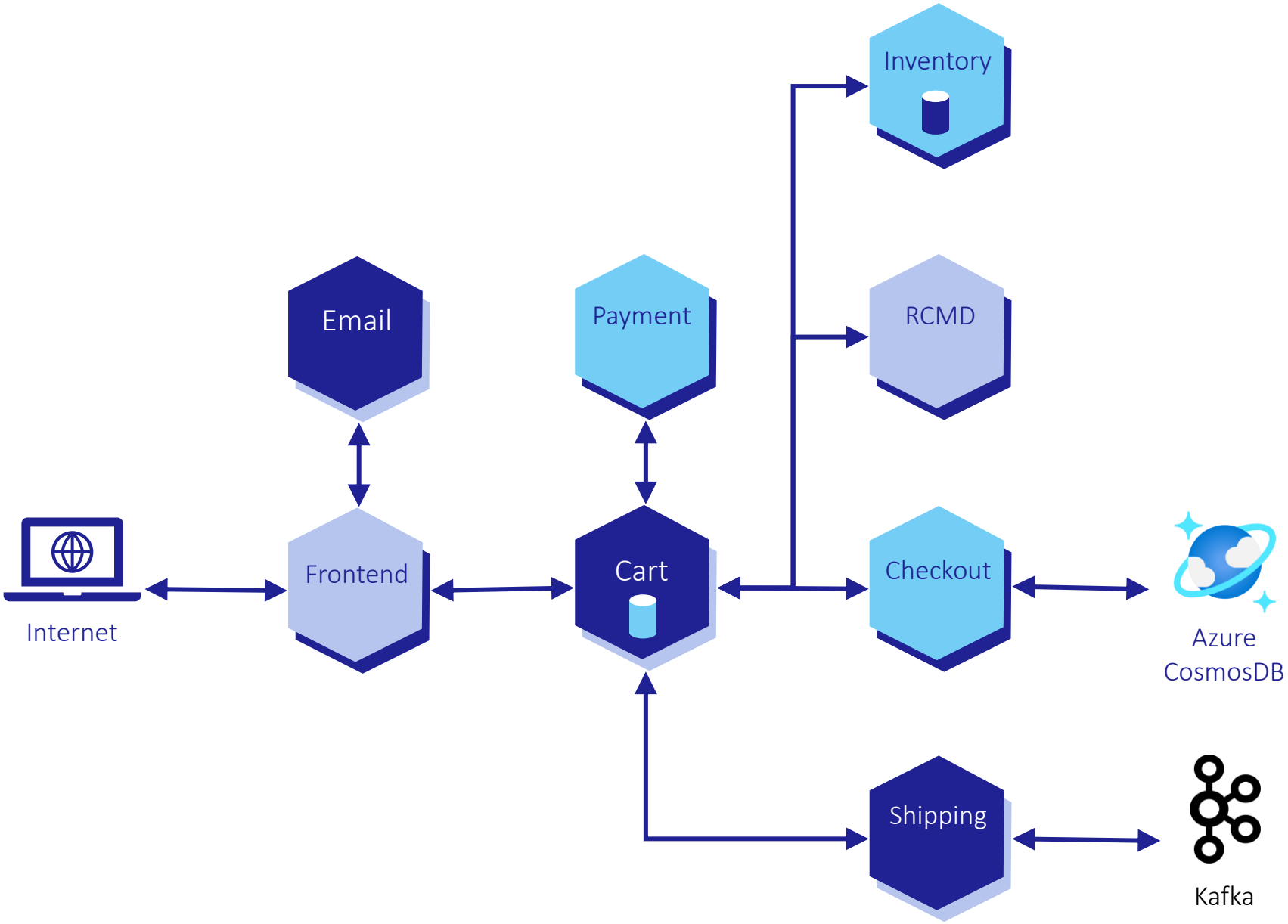
只能选用有限的工具和运行时来构建分布式应用程序



运行时只支持有限的语言，并和特性紧密绑定

运行时仅针对特定的基础设施

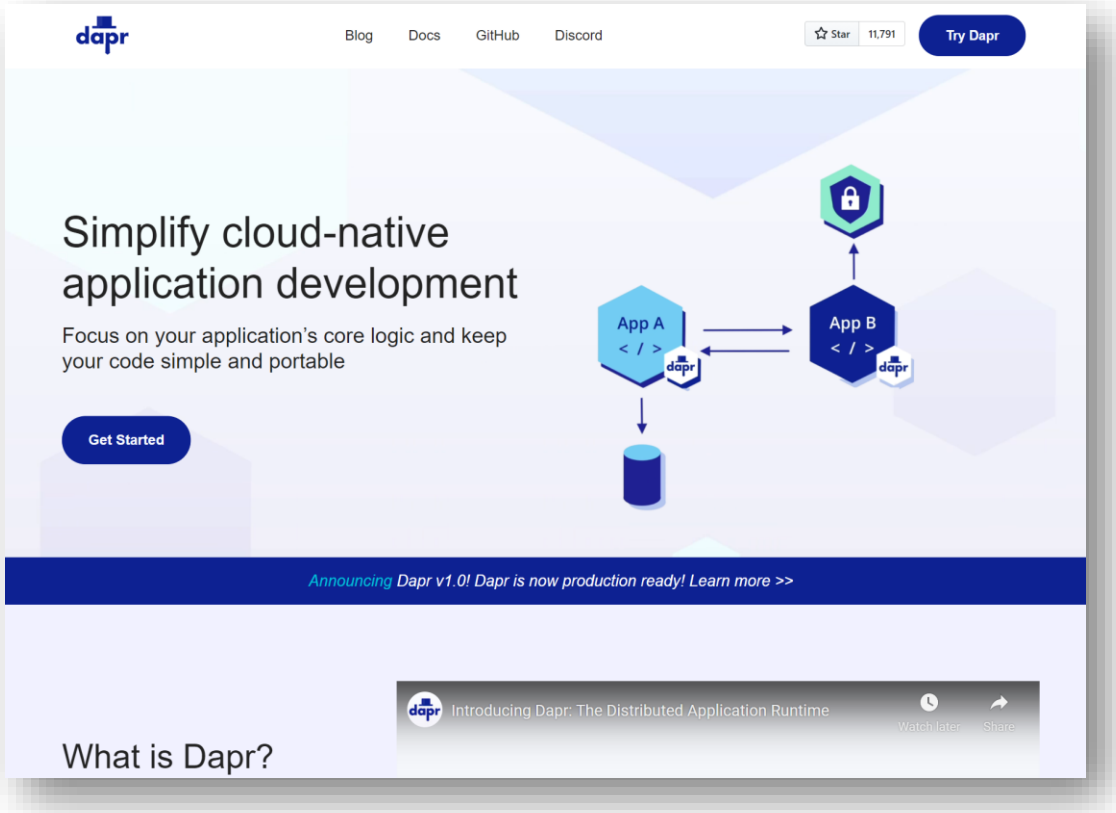
假如我们要开发如下这样一个电子商务应用



dapr

Distributed Application Runtime

dapr.io



可移植的，事件驱动的，用于构建跨云和边缘的**分布式应用程序**的**运行时**

Dapr的宏大目标

最佳实践的构建块

任何语言或框架

一致、可移植、开放的API

采用流行标准

可扩展和可插接的组件

平台无关，支持任意云和边缘

社区驱动，供应商中立

14.1K

贡献者131

ZEISS

Alibaba Cloud

IGNITION GROUP

LEGENTIC

Roadwork

morphir

社区客户（20+）	
EKEYNOW	小红书
StaringWorld	中国再保险风险管理公司
Solid Value Software	Endeavour Group
Tdcare	Man Group
腾讯	Border States Electric
钉钉	NashTech
高德	……

https://github.com/dapr/dapr/issues/3169

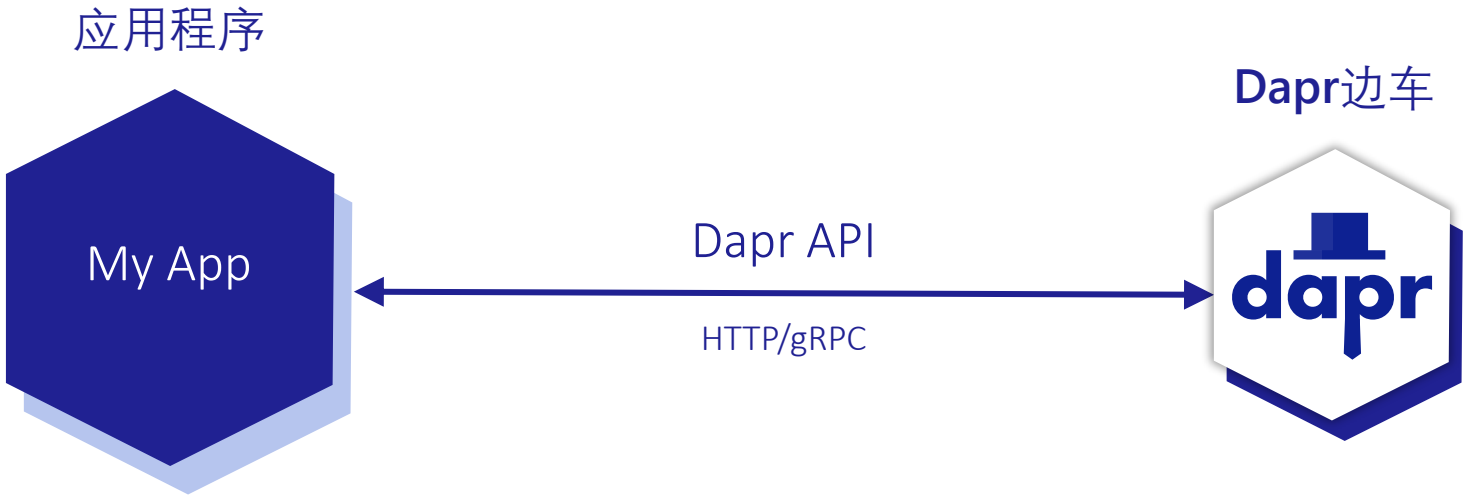
Dapr通过所谓的构建块来提供核心能力

利用开放协议和标准API支持任何语言和框架

- 在用户服务代码中通过HTTP/gRPC协议访问标准API
- 为每个服务动态加载一个本地运行的边车函数库

并托管到任意的环境下运行

边车模型



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

GET http://localhost:3500/v1.0/**state**/inventory/item67

POST http://localhost:3500/v1.0/**publish**/shipping/orders

GET http://localhost:3500/v1.0/**secrets**/keyvault/password

Dapr应用程序的运行模式

Dapr目前支持的组件

通过YAML来描述资源的连接信息

超过70个组件

可以创建自己的组件

状态存储
AWS DynamoDB
Azure CosmosDB
Firebase
Redis
Cassandra

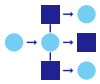
发布订阅代理
AWS SQS
Azure Service Bus
GCP Pub/Sub
Redis
RabbitMQ

绑定与触发器
AWS S3
Azure Storage
GCP Storage
Twilio
Kafka

机密存储
AWS Secrets Manager
Azure KeyVault
GCP Secret Manager
HashiCorp Vault
Kubernetes Secret

可观测性
Prometheus
AppInsights
Zipkin
Jaeger

Dapr的7大构建块



服务调用

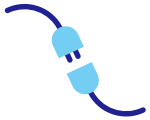
- 安全的执行服务到服务之间的直接方法调用

状态管理

- 创建长时间运行的有状态或无状态服务

发布订阅

- 在服务之间实现安全可伸缩的消息传递



绑定与触发器

- 绑定输入和输出到外部资源，比如数据库或消息队列
- 基于各类输入触发事件

Actors

- 把代码和数据封装到一个可重用的Actor对象当中，轻易实现很多常见的场景

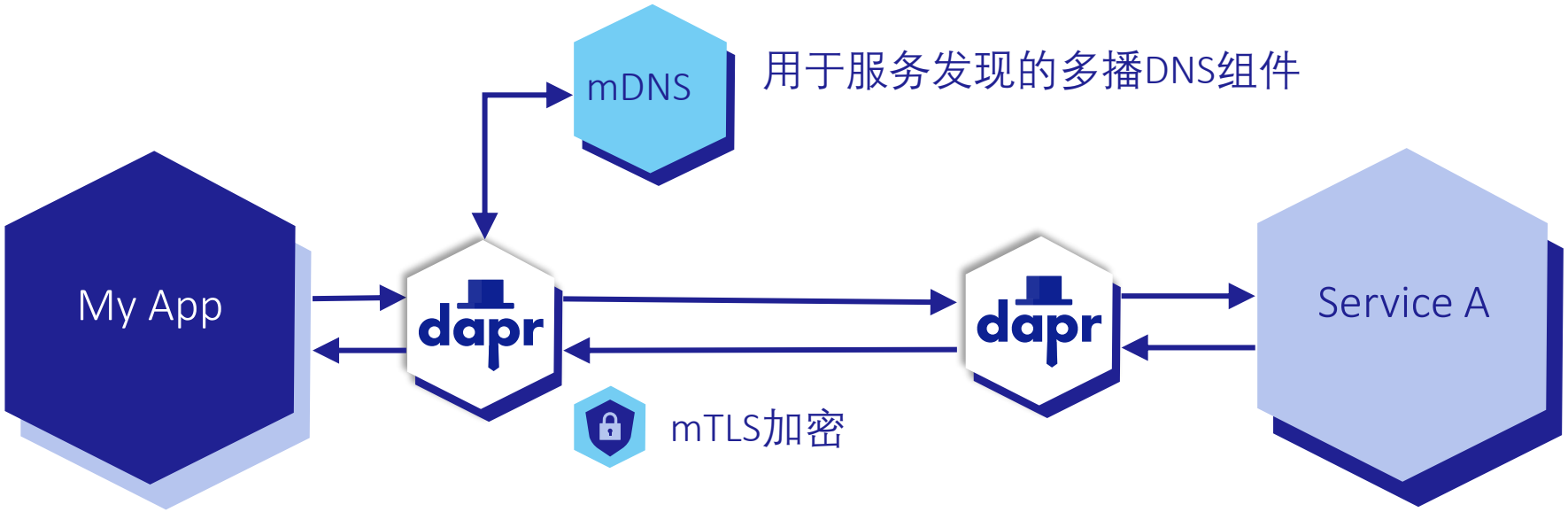
可观测性

- 观察和测量组件和服务的消息调用情况

机密管理

- 从你的应用程序中安全地访问机密

服务调用



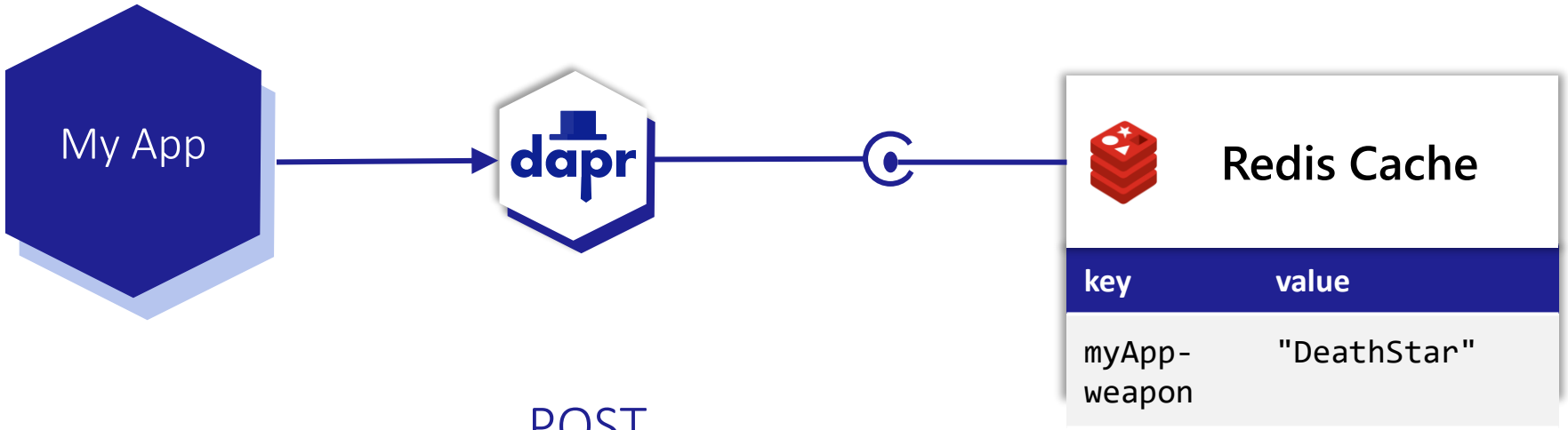
POST
http://localhost:3500/v1.0/invoke/servicea/method/neworder

```
{"data":"Hello World"}
```

POST
http://10.0.0.2:8000/neworder

```
{"data":"Hello World"}
```

状态管理（存）

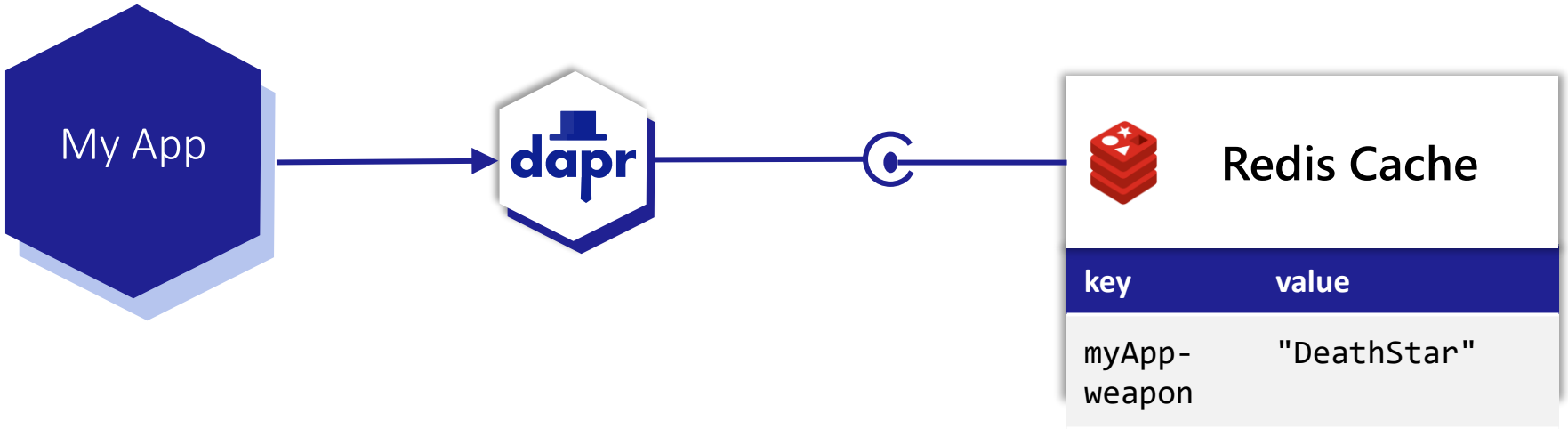


POST

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

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

状态管理（取）

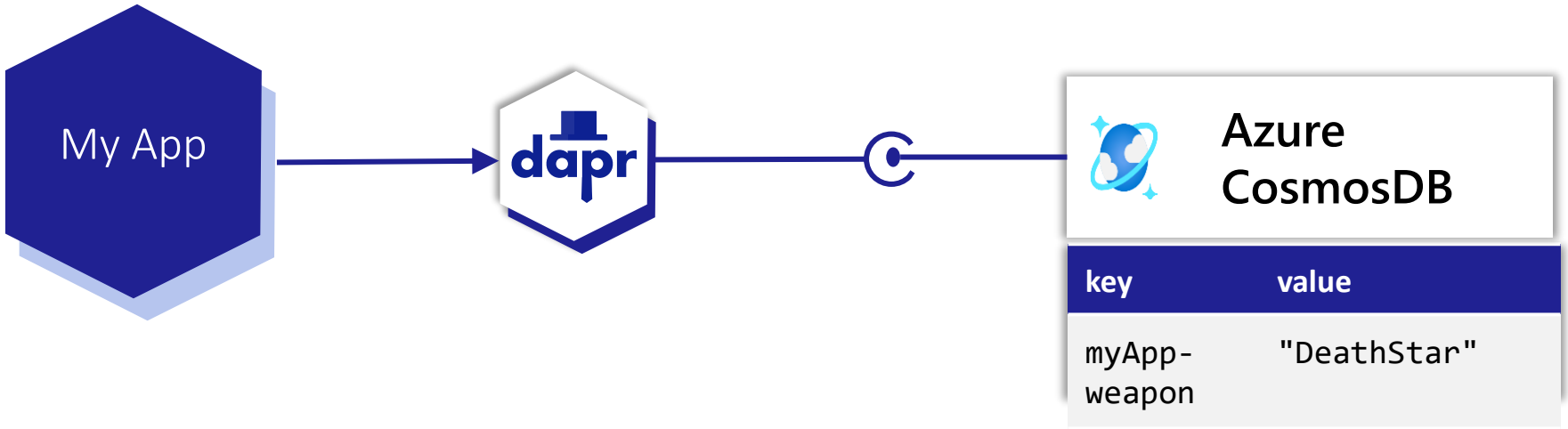


GET

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

```
"DeathStar"
```

状态管理（不同存储库）



GET

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

```
"DeathStar"
```

Dapr state API

Save state

```
POST /v1.0/state/corpdb
```

Retrieve state

```
GET /v1.0/state/corpdb/mystate
```

Delete state

```
DELETE /v1.0/state/corpdb/mystate
```

Get bulk state

```
POST /v1.0/state/corpdb/bulk
```

Submit multiple state transactions

```
POST /v1.0/state/corpdb/transaction
```

```
apiVersion: dapr.io/v1alpha1
kind: Component
metadata:
  name: corpdb
spec:
  type: state.azure.cosmosdb
  version: v1
  metadata:
  - name: url
    value: corpdb.documents.azure.com
  - name: masterKey
    secretKeyRef:
      name: master-key
      key: cosmos-key
  - name: database
    value: orders
  - name: collection
    value: processed
```

Dapr state API

Save state

```
POST /v1.0/state/corpdb
```

Retrieve state

```
GET /v1.0/state/corpdb/mystate
```

Delete state

```
DELETE /v1.0/state/corpdb/mystate
```

Get bulk state

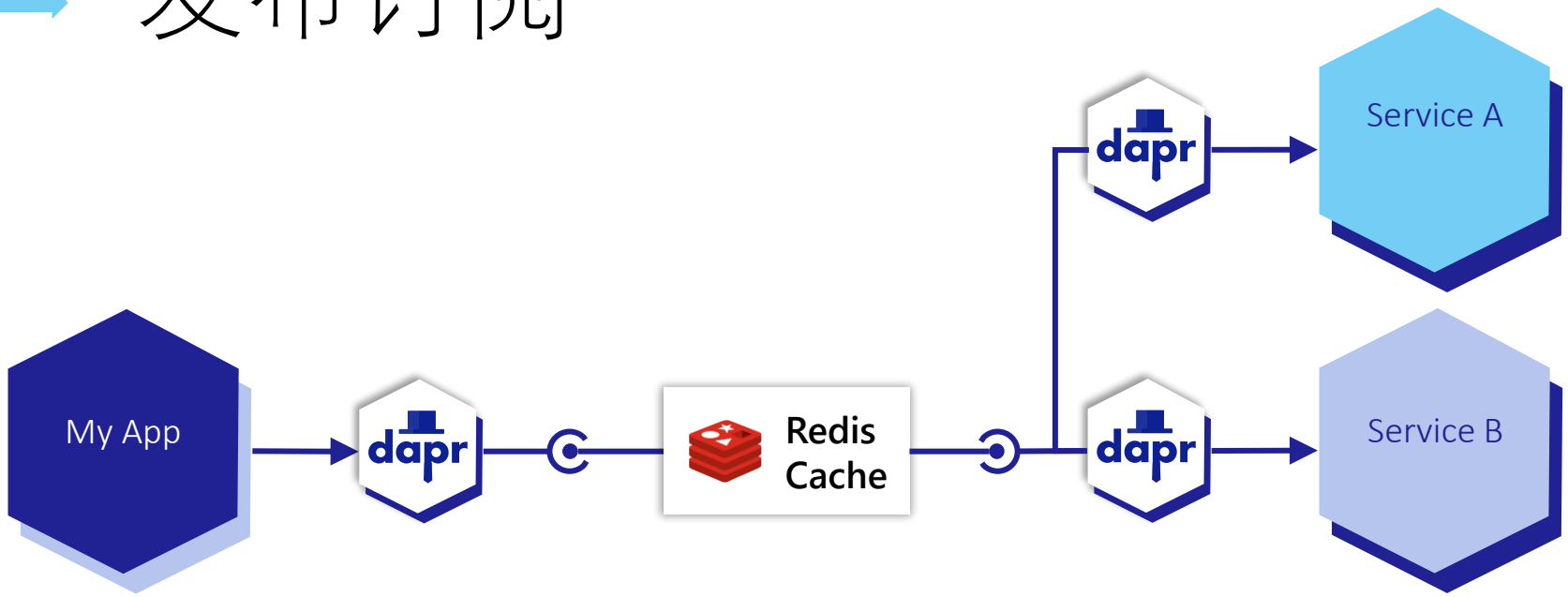
```
POST /v1.0/state/corpdb/bulk
```

Submit multiple state transactions

```
POST /v1.0/state/corpdb/transaction
```

```
apiVersion: dapr.io/v1alpha1
kind: Component
metadata:
  name: corpdb
spec:
  type: state.redis
  version: v1
  metadata:
  - name: redisHost
    value: redis-master.default.svc.cluster.local:6379
  - name: redisPassword
    secretKeyRef:
      name: redis-secret
      key: redis-password
```

发布订阅



POST
http://localhost:3500/v1.0/publish/orders/processed

```
{"data":"Hello World"}
```

POST
http://10.0.0.2:8000/orders
http://10.0.0.4:8000/factory/orders

```
{"data":"Hello World"}
```

Dapr pub/sub API

App-to-sidecar

Publish a message

`POST /v1.0/publish/orders/processed`

Sidecar-to-app

Get app subscriptions

`GET /dapr/subscribe`

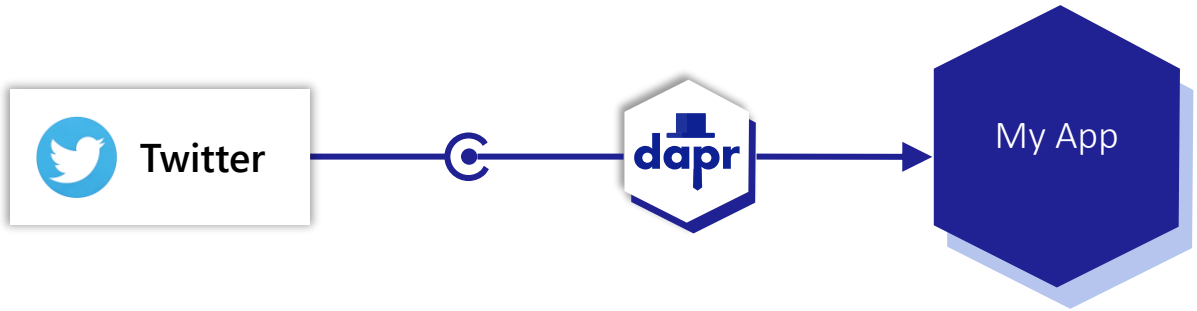
Publish to app

`POST /order-processing`

orders.yaml

```
apiVersion: dapr.io/v1alpha1
kind: Component
metadata:
 name: orders
spec:
 type: pubsub.redis
 metadata:
 - name: redisHost
  value: leader.redis.svc.cluster.local:6379
 - name: redisPassword
  secretKeyRef:
   name: redis-secret
   key: password
 - name: allowedTopics
  value: "processed,audit"
```

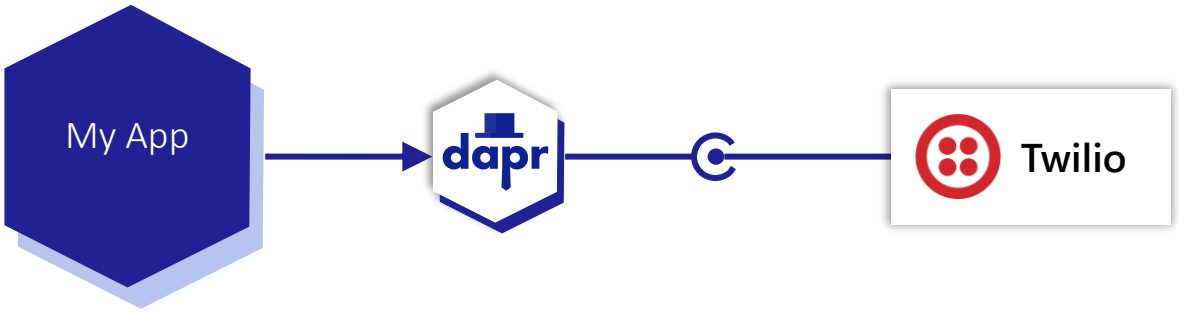
输入触发器



POST
`http://10.0.0.2:8000/newtweet`

```
{"data":"📣 We are excited
to announce the …"}
```

输出绑定



POST
http://localhost:3500/v1.0/bindings/twilio

```
{"data":"Hello World"}
```

Hello World

Dapr bindings API

App-to-sidecar

Invoke an output binding

`POST/PUT /v1.0/bindings/twitter`

Sidecar-to-app

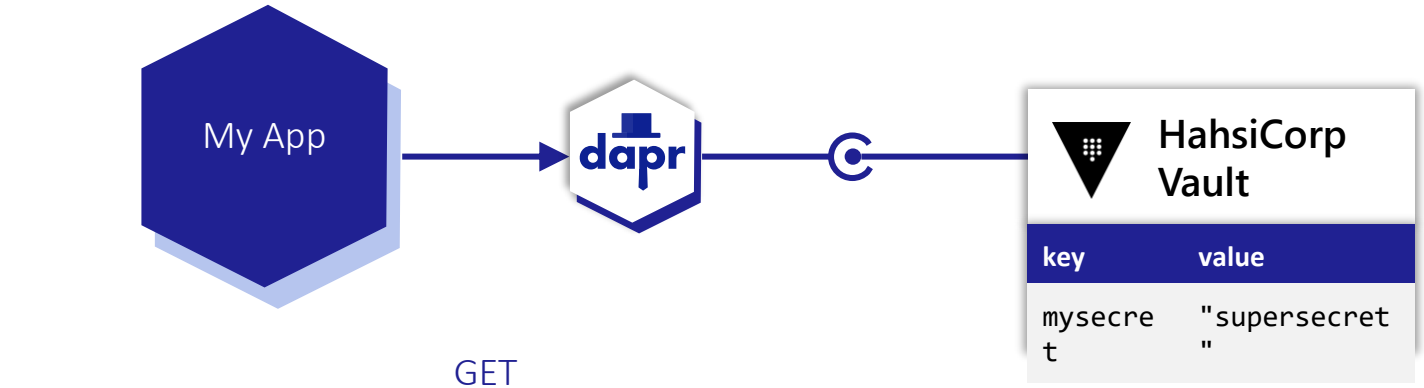
Trigger an app

`OPTIONS/POST /new-tweet`

twitter.yaml

```
apiVersion: dapr.io/v1alpha1
kind: Component
metadata:
 name: twitter
spec:
 type: bindings.twitter
 version: v1
 metadata:
 - name: consumerKey
  secretKeyRef:
    name: twitter-secret
    key: consumerKeys
 - name: consumerSecret
  secretKeyRef:
    name: twitter-secret
    key: consumerSecret
 - name: accessToken
  secretKeyRef:
```

机密



GET
http://localhost:3500/v1.0/secrets/vault/mysecret

```
"supersecret"
```

Dapr secrets API

App-to-sidecar

Retrieve a secret

```
GET /v1.0/secrets/vault/mysecret
```

Retrieve secrets in bulk

```
GET /v1.0/secrets/vault/bulk
```

vault.yaml

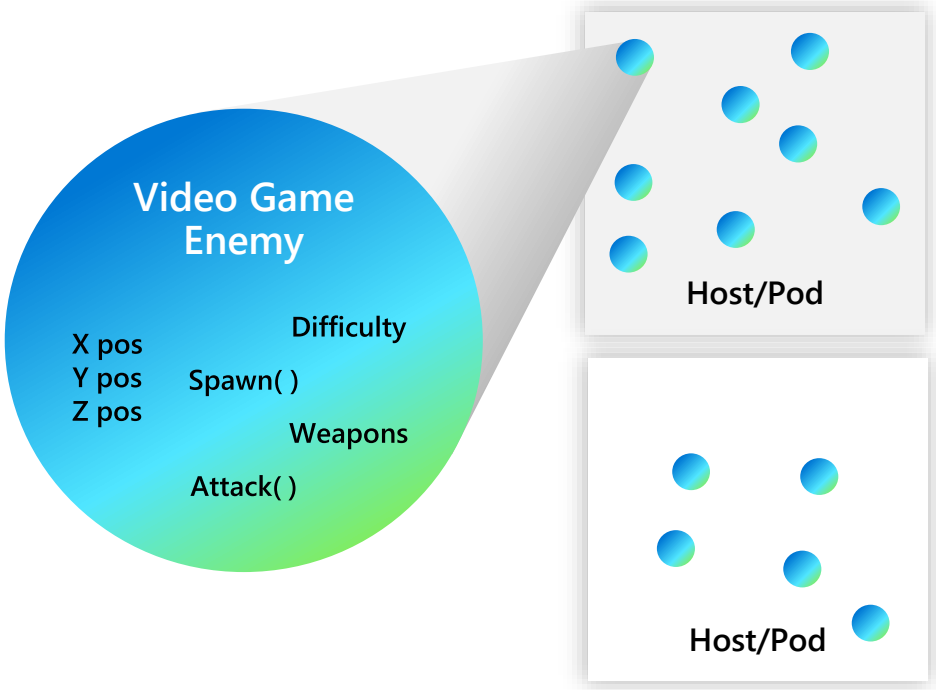
```
apiVersion: dapr.io/v1alpha1
kind: Component
metadata:
 name: vault
spec:
 type: secretstores.hashicorp.vault
 metadata:
 - name: vaultAddr
  value: https://127.0.0.1:8200
 - name: caCert
  value: "ca_cert"
 - name: caPath
  value: "/certs/cert.pem"
 - name: caPem
  value: "/certs/ca.pem”
```

虚拟Actors

有状态的，混合了存储和计算的对象实例

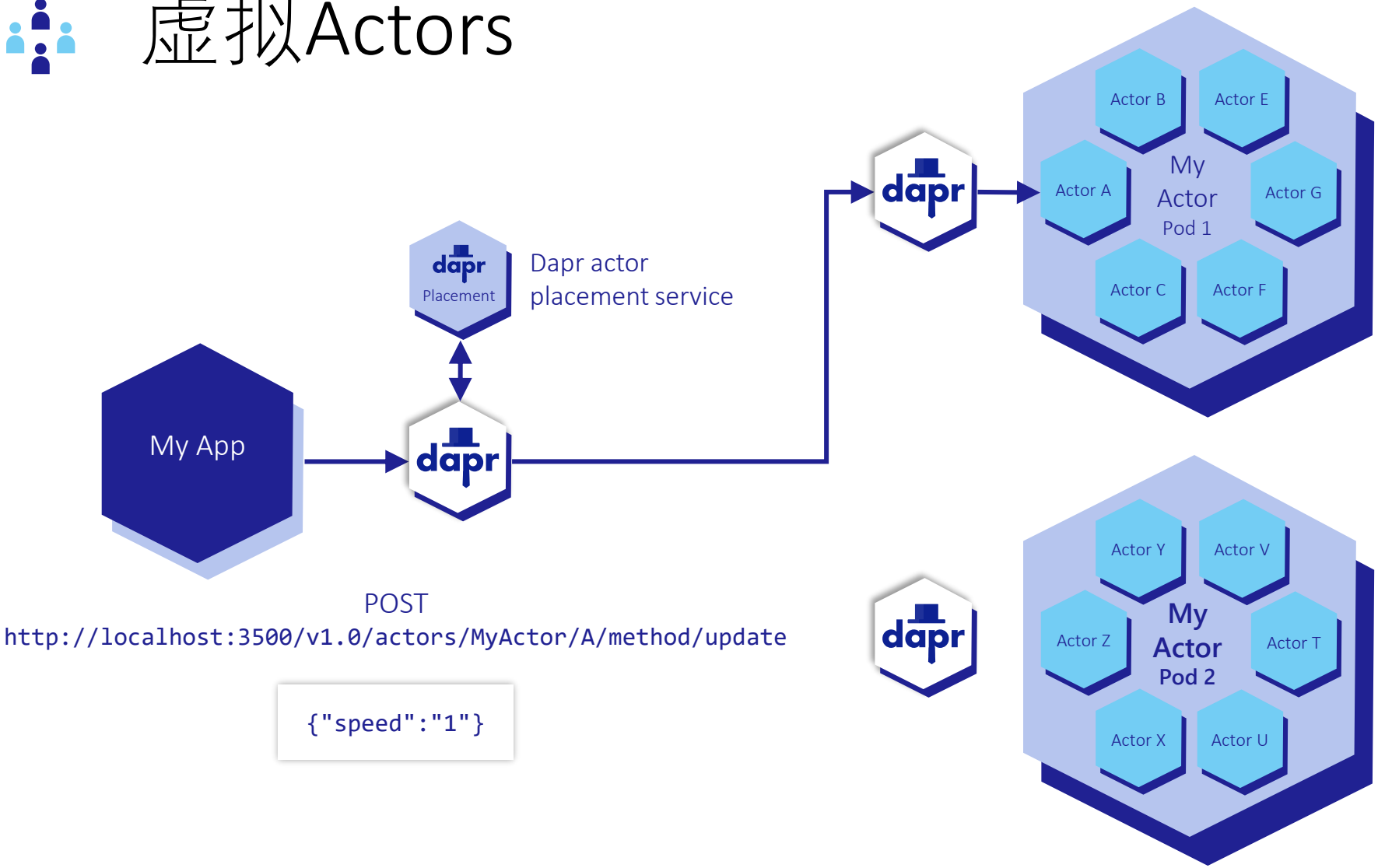
Dapr Actor 特性:

- ✓ 分布式和故障转移
- ✓ 基于轮次的并发处理
- ✓ 状态管理
- ✓ 计时器
- ✓ 提醒器



虚拟化的方式和Service Fabric Reliable Actors一致

虚拟Actors



POST
`http://localhost:3500/v1.0/actors/MyActor/A/method/update`

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

Dapr actors API

App-to-sidecar

Invoke an actor method

`POST/GET /v1.0/actors/MyActor/A/method/update`

Save actor state

`POST/PUT /v1.0/actors/MyActor/A/state`

Get actor state

`GET /v1.0/actors/MyActor/A/state/status`

Create actor reminder

`POST/PUT /v1.0/actors/MyActor/A/reminders/process`

Create actor timer

`POST/PUT /v1.0/actors/MyActor/A/timers/process`

Sidecar-to-actor

Invoke an actor method

`PUT /actors/MyActor/A/method/update`

Deactivate actor

`DELETE /actors/MyActor/A`

Invoke actor reminder

`PUT /actors/MyActor/A/method/remind/process`

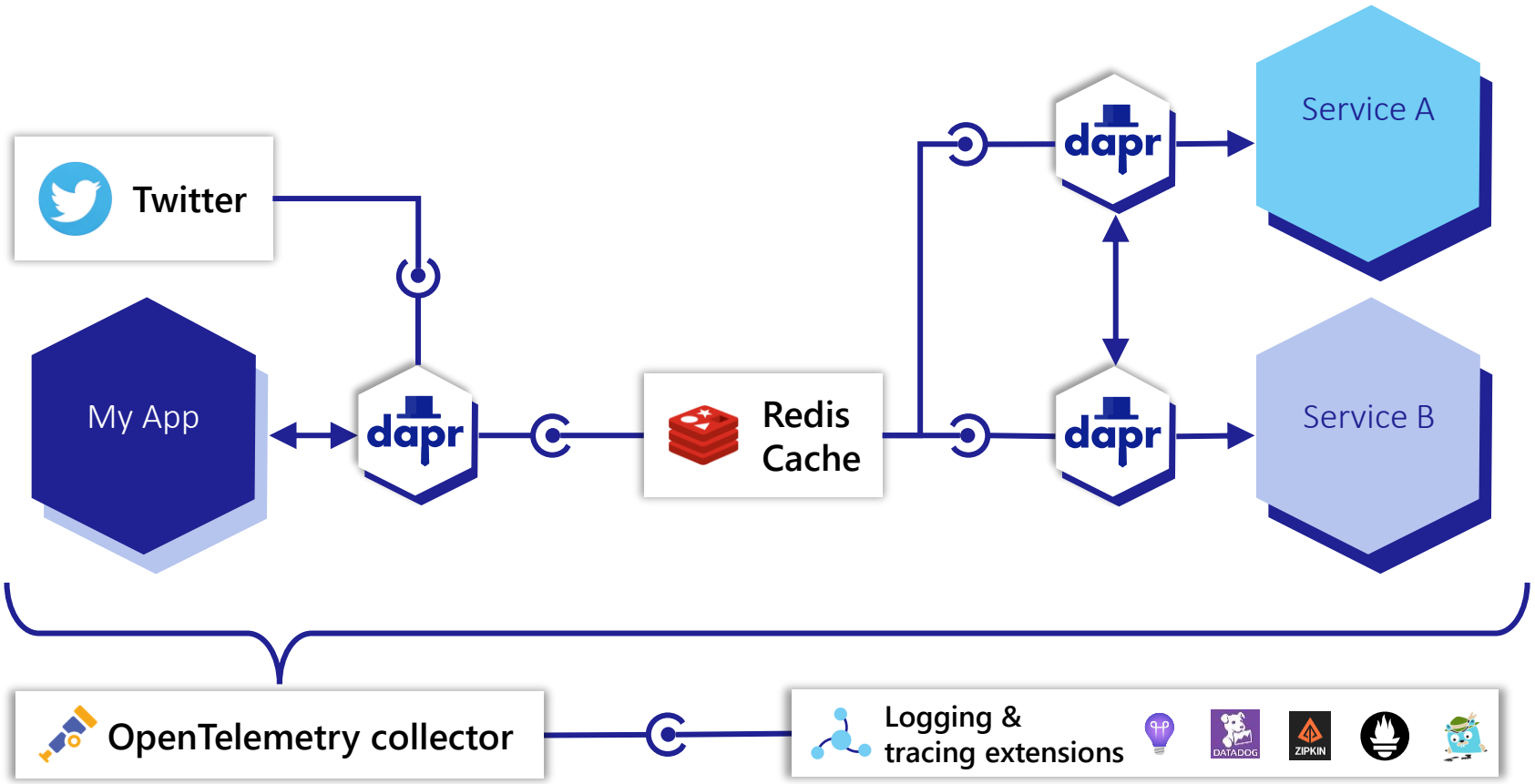
Invoke actor timer

`PUT /actors/MyActor/A/method/timer/process`

Health check

`GET /healthz`

可观测性

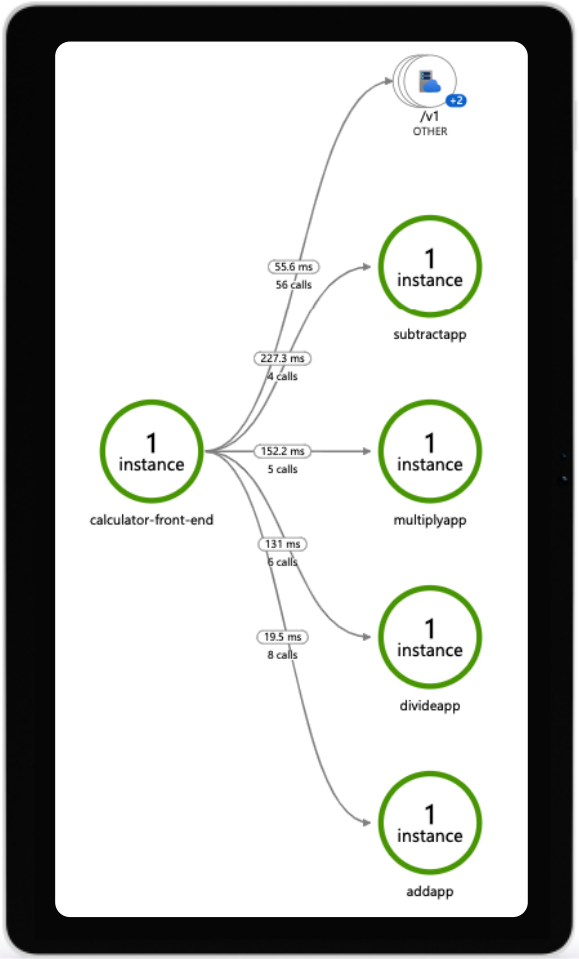

Twitter
My App
dapr
Redis Cache
dapr
dapr
Service A
Service B
OpenTelemetry collector
Logging & tracing extensions
DATADOG
ZIPKIN

分布式跟踪

收集**Dapr**边车和系统服务跟踪数据，轻易实现应用程序级的测量

Dapr 分布式跟踪特性:

- 内置Zipkin收集和查看器
- 可配置采样率
- TBD
- TBD
- TBD



度量

内置的监控能力可以了解**Dapr**边车和系统服务的行为

Dapr 度量特性:

- 调用延迟
- CPU/memory利用率
- 错误率
- 边车注入失败情况
- 系统健康

Dapr托管环境

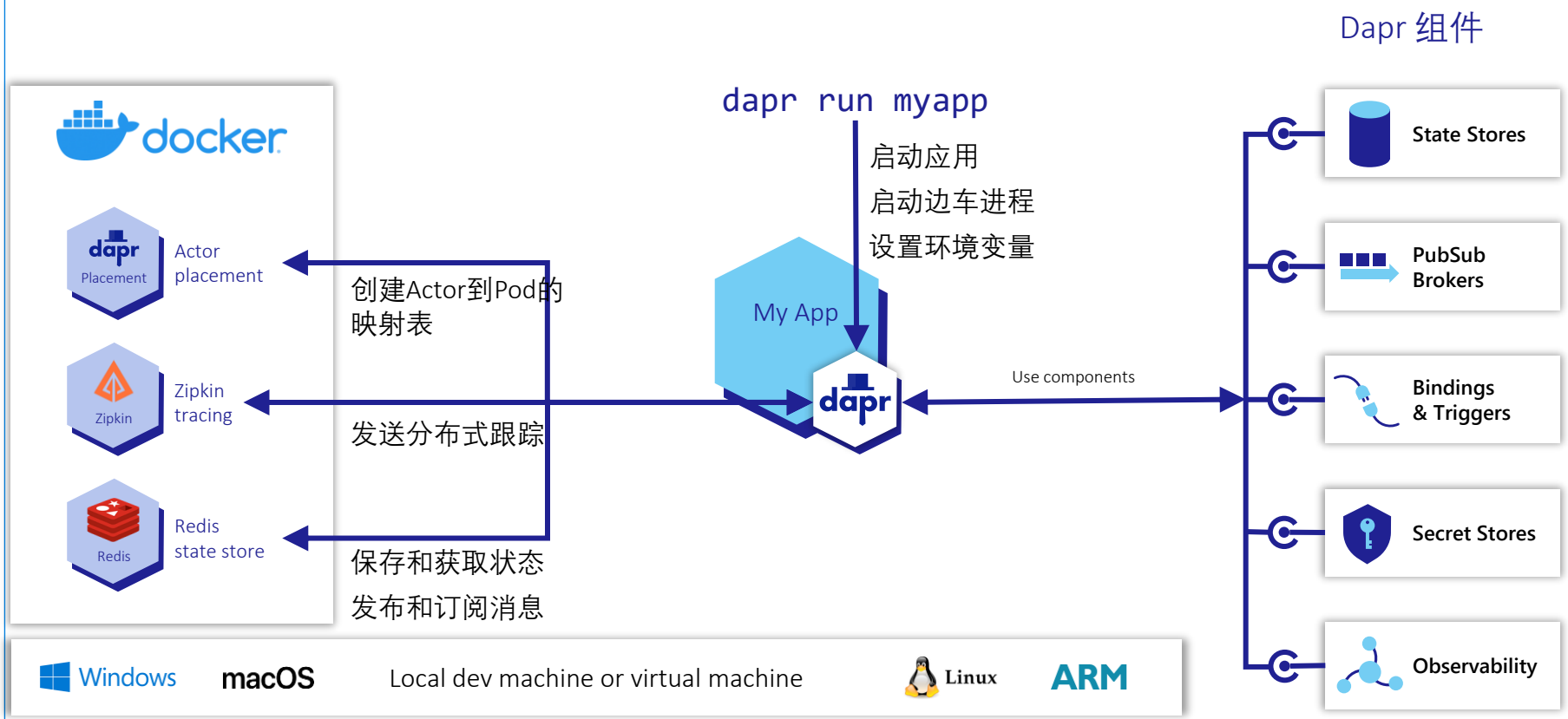
自托管

- 使用`dapr init`配置
- 利用Dockers镜像方便配置
  - 配置placement, Zipkin, Redis容器实例
  - `slim-init`可无需Docker
- 使用`dapr run`来启动任何附带Dapr边车的应用程序

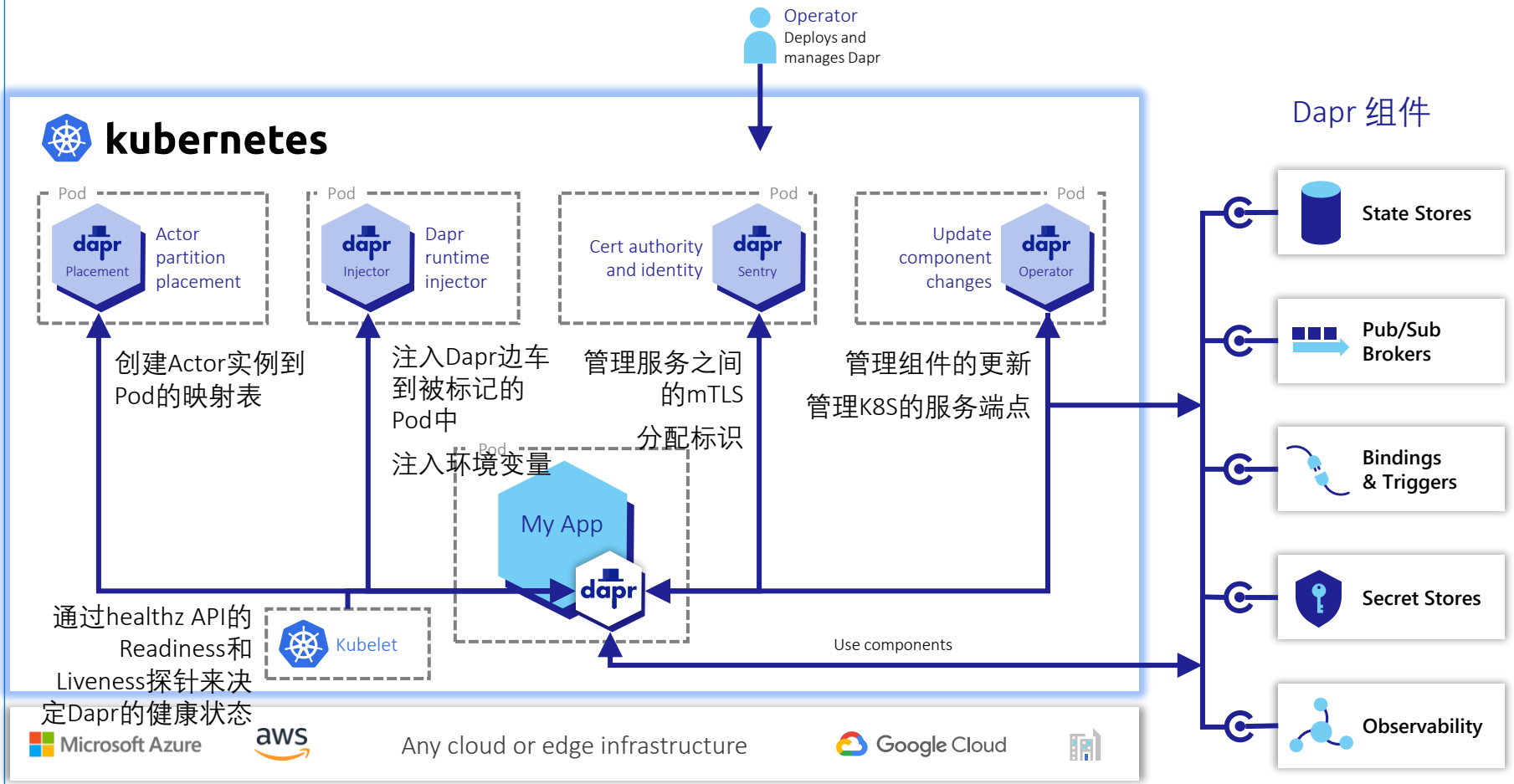
kubernetes

- 使用`dapr init -k`来配置
- 完全托管的Dapr控制面
  - 部署dashboard, placement, operator, sentry, and injector pods
- 自动注入Dapr边车到被标记的Pod之内
- 使用`dapr upgrade` or Helm来升级

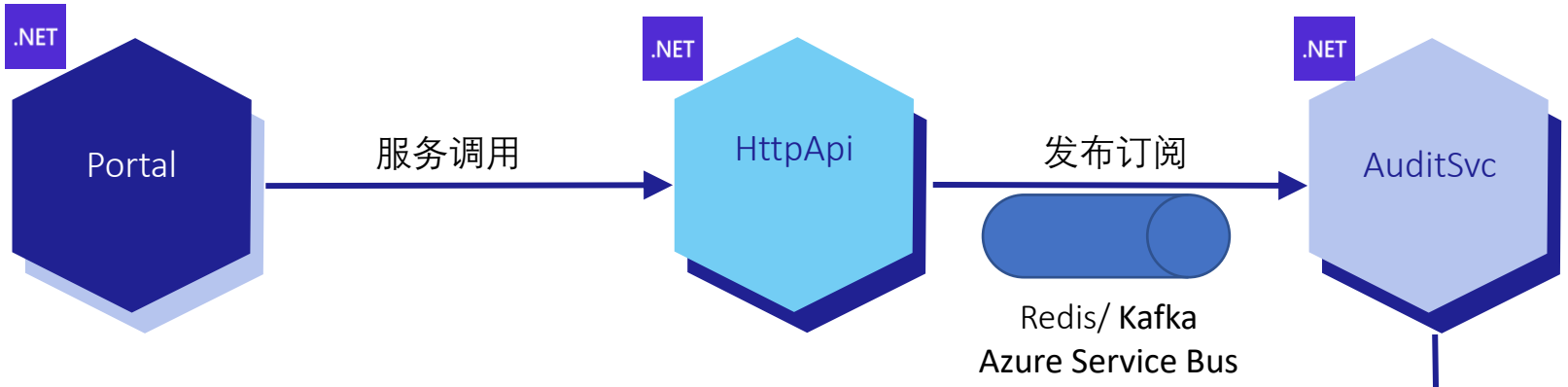
Dapr在自托管的Docker模式下



Dapr在Kubernetes之上



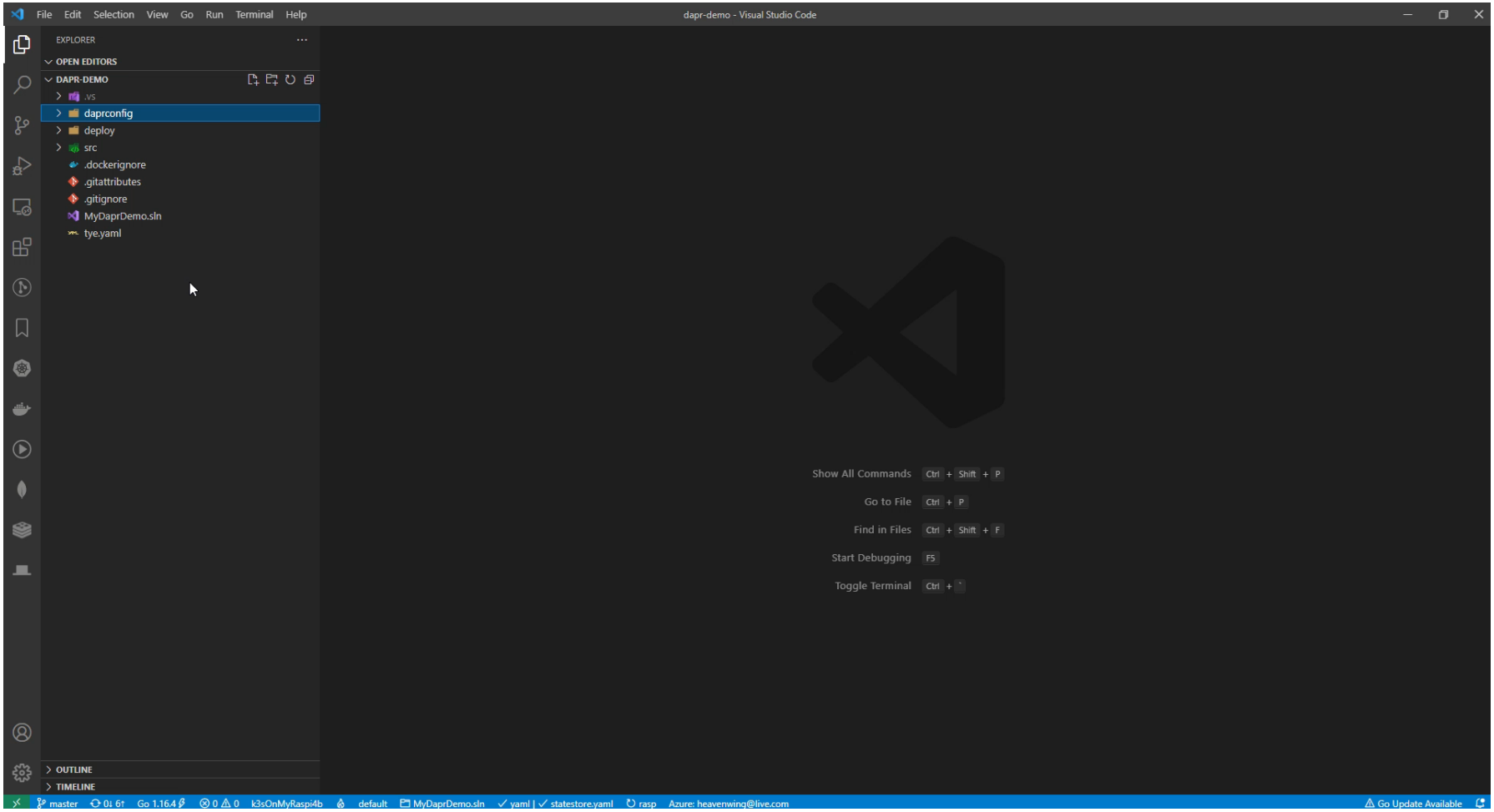
DaprDemo架构



- 托管环境及组件:
  - K3S on Raspberry Pi 4b
    - 集群内Redis支持发布订阅和状态管理
  - K3S on Azure VM
    - 集群内Kafka支持发布订阅
    - 集群内Memcached支持状态管理
  - Azure Kubernetes Service
    - Azure Service Bus支持发布订阅
    - Azure Blob Storage支持状态管理

https://github.com/heavenwing/dapr-demo

DaprDemo演示


File Edit Selection View Go Run Terminal Help
dapr-demo - Visual Studio Code
EXPLORER
OPEN EDITORS
DAPR-DEMO
.vs
daprconfig
deploy
src
.dockerignore
.gitattributes
.gitignore
MyDaprDemo.sln
tye.yaml
OUTLINE
TIMELINE
Show All Commands Ctrl + Shift + P
Go to File Ctrl + P
Find in Files Ctrl + Shift + F
Start Debugging F5
Toggle Terminal Ctrl + `
master
Go 1.16.4
k3sOnMyRaspi4b
default
MyDaprDemo.sln
yaml | statestore.yaml
rasp
Azure: heavenwing@live.com
Go Update Available

DaprDemo演示

MyUbuntu
虚拟机

搜索(Ctrl+/)

概述
活动日志
访问控制(标识和访问管理)
标记
诊断

连接 开始 重新启动 停止 捕获 删除 刷新

概要

资源组 (更改) : zyg
状态 : 正在运行
位置 : 中国东部 2

操作系统 : Linux
大小 : 标准 B2ms (2 vcpu，8 GiB 内存)
公共 IP 地址 : 52.131.228.83

zygaksww
Kubernetes service

设置

网络
连接
磁盘
大小
安全
顾问
扩展
可用
配置
标识
导出
属性

Search (Ctrl+/)

Overview
Activity log
Access control (IAM)
Tags
Diagnose and solve problems
Security

Kubernetes resources

Namespaces
Workloads
Services and ingresses
Storage
Configuration

Settings

Node pools
Cluster configuration
Networking
Deployment center (preview)
Policies
Properties
Locks

Monitoring

Insights

Connect Delete Refresh

Essentials

Resource group (change) : k8s
Status : Succeeded
Location : Southeast Asia
Subscription (change) : Visual Studio Enterprise
Subscription ID : 5842f786-d670-4bd9-b29b-f491c55c3ccf
Tags (change) : Click here to add tags

Kubernetes version : 1.20.5
API server address : [redacted]d5.hcp.southeastasia.azmk8s.io
Network type (plugin) : Azure CNI
Node pools : 1 node pool

Properties Capabilities

Kubernetes services
Encryption type : Encryption at-rest with a platform-managed key
Virtual node pools : Not enabled

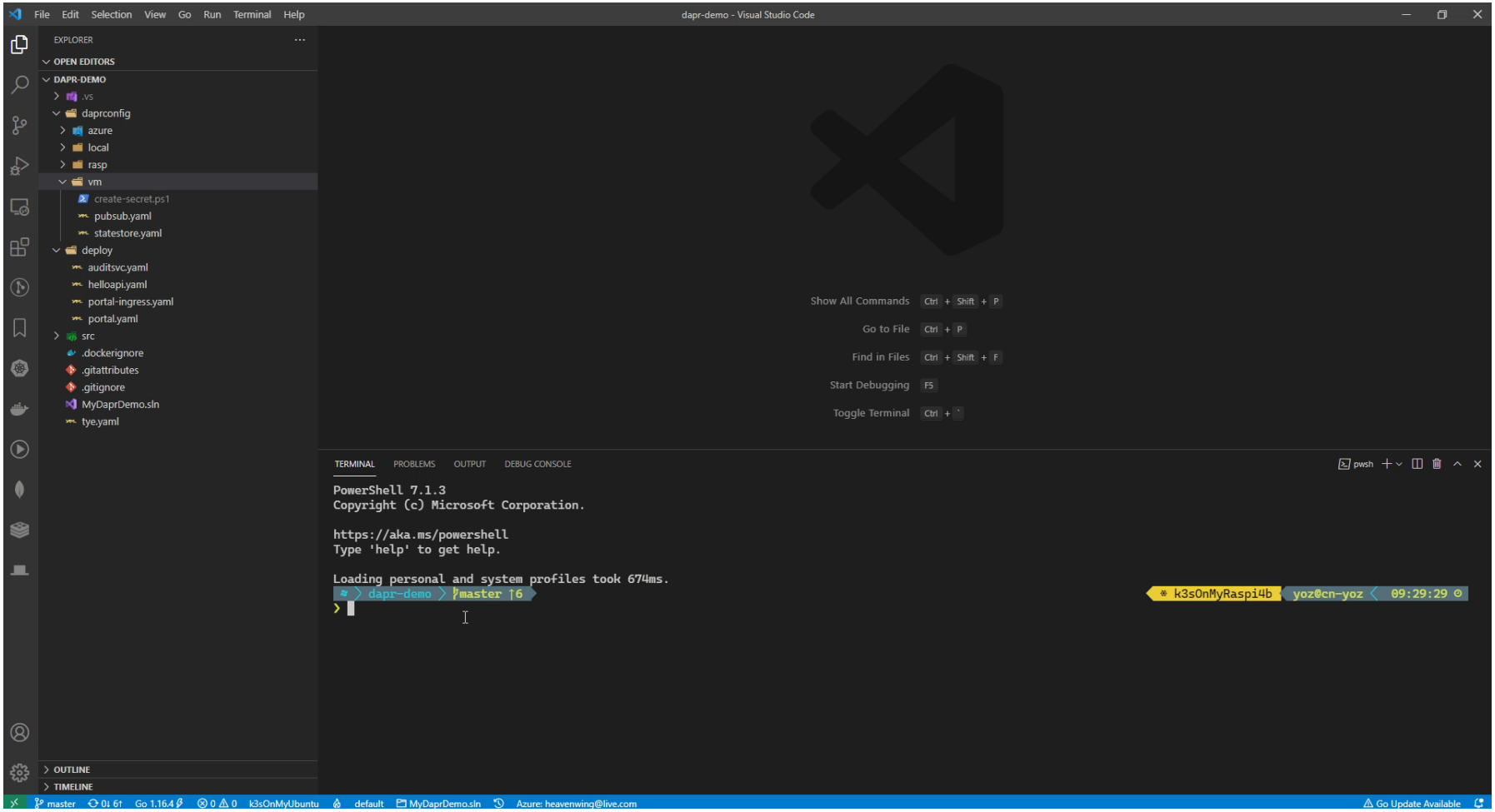
Node pools
Node pools : 1 node pool
Kubernetes versions : 1.20.5
Node sizes : Standard_B2s

Configuration
Kubernetes version : 1.20.5
Kubernetes RBAC : Enabled
AKS-managed AAD : Not enabled

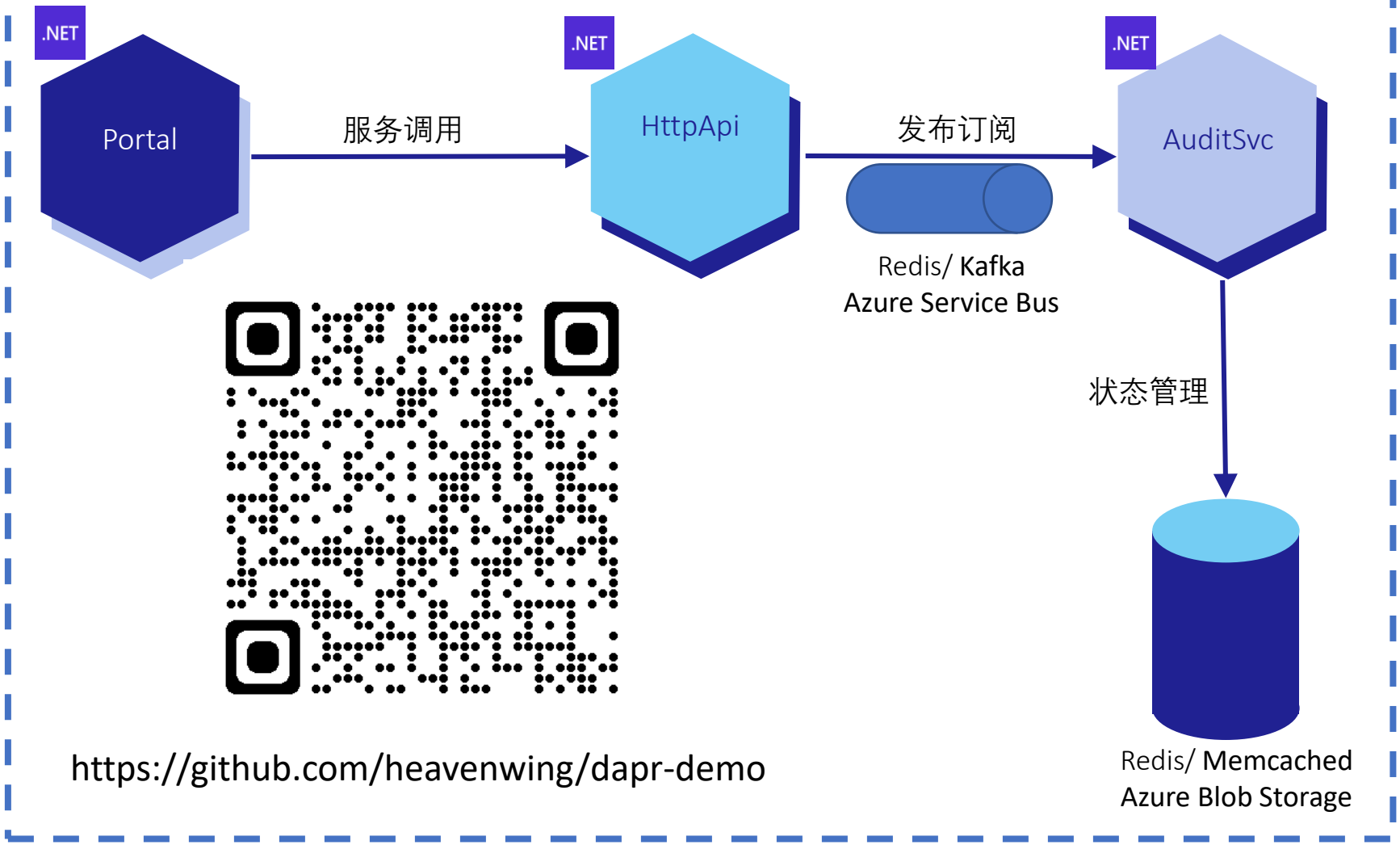
Networking
API server address : [redacted]d5.hcp.southeastasia.azmk8s.io
Network type (plugin) : Azure CNI
Pod CIDR : -
Service CIDR : 172.0.0.0/16
DNS service IP : 172.0.0.10
Docker bridge CIDR : 172.17.0.1/16
Network Policy : None
Load balancer : Standard
HTTP application routing : Not enabled
Private cluster : Not enabled
Authorized IP ranges : Not enabled
Application Gateway ingress controller : Not enabled

Integrations
Container insights : Enabled
Workspace resource ID : zyg-try

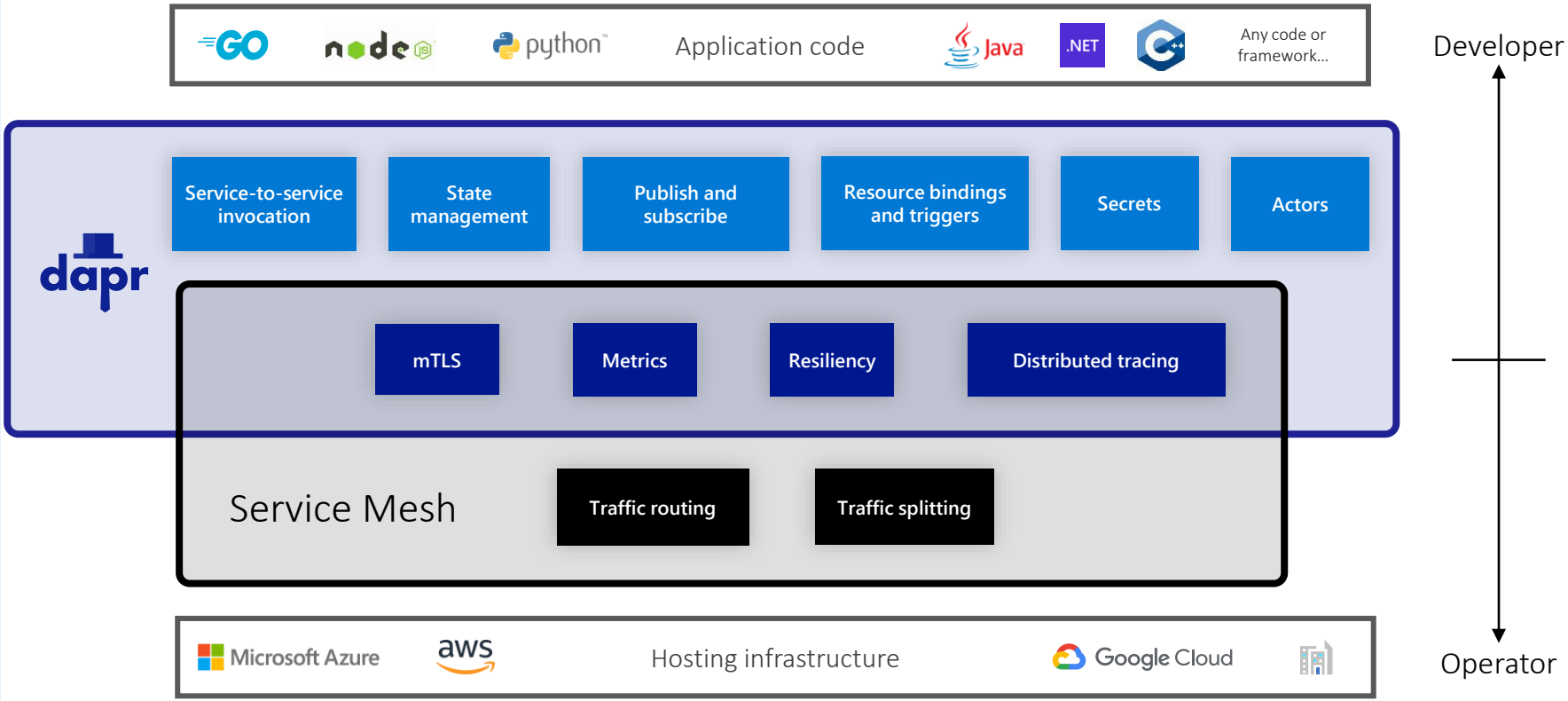
DaprDemo演示


File Edit Selection View Go Run Terminal Help
dapr-demo - Visual Studio Code
EXPLORER
OPEN EDITORS
DAPR-DEMO
.vs
daprconfig
azure
local
rasp
vm
create-secret.ps1
pubsub.yaml
statestore.yaml
deploy
auditsvc.yaml
helloapi.yaml
portal-ingress.yaml
portal.yaml
src
.dockerignore
.gitattributes
.gitignore
MyDaprDemo.sln
tye.yaml
Show All Commands Ctrl + Shift + P
Go to File Ctrl + P
Find in Files Ctrl + Shift + F
Start Debugging F5
Toggle Terminal Ctrl + `
TERMINAL PROBLEMS OUTPUT DEBUG CONSOLE
pwsh
PowerShell 7.1.3
Copyright (c) Microsoft Corporation.
https://aka.ms/powershell
Type 'help' to get help.
Loading personal and system profiles took 674ms.
dapr-demo master ↑6
k3sOnMyRaspi4b yoz@cn-yoz 09:29:29
OUTLINE
TIMELINE
master 0↓ 6↑ Go 1.16.4 0 0 k3sOnMyUbuntu default MyDaprDemo.sln Azure: heavenwing@live.com
Go Update Available

DaprDemo架构

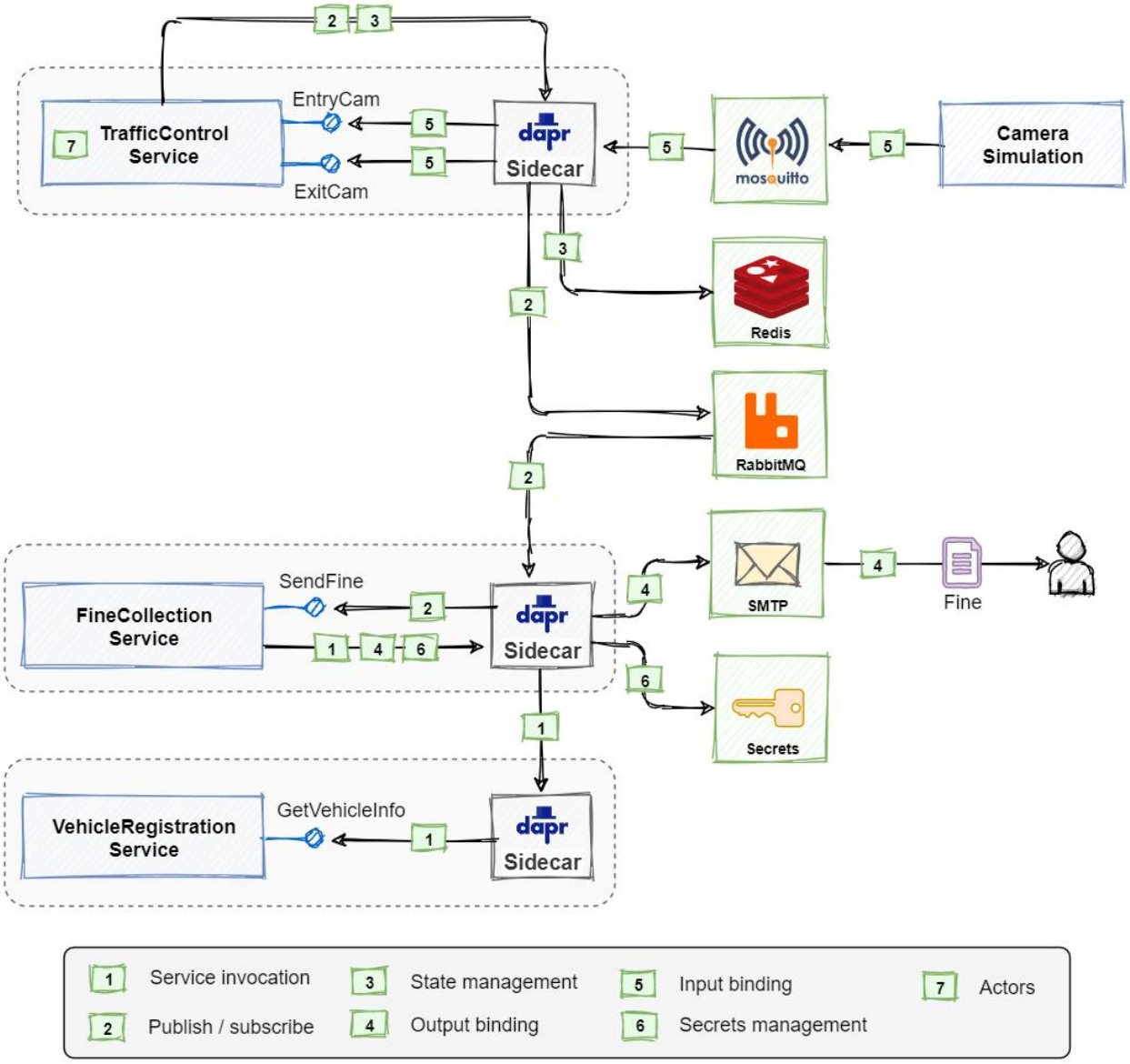

.NET
Portal
服务调用
.NET
HttpApi
发布订阅
.NET
AuditSvc
Redis/ Kafka
Azure Service Bus
状态管理
Redis/ Memcached
Azure Blob Storage
https://github.com/heavenwing/dapr-demo

Dapr 和 Service Meshes



Dapr和Open Service Mesh一起运行:
https://docs.dapr.io/developing-applications/integrations/open-service-mesh/

Traffic Control 示例



https://docs.microsoft.com/en-us/dotnet/architecture/dapr-for-net-developers/sample-application

谢谢你的时间

麦思博(msup)有限公司是一家面向技术型企业的培训咨询机构，携手2000余位中外客座导师，服务于技术团队的能力提升、软件工程效能和产品创新迭代，超过3000余家企业续约学习，是科技领域占有率第1的客座导师品牌，msup以整合全球领先经验实践为己任，为中国产业快速发展提供智库。

高可用架构公众号主要关注互联网架构及高可用、可扩展及高性能领域的知识传播。订阅用户覆盖主流互联网及软件领域系统架构技术从业人员。

高可用架构系列社群是一个社区组织，其精神是“分享+交流”，提倡社区的人人参与，同时从社区获得高质量的内容。