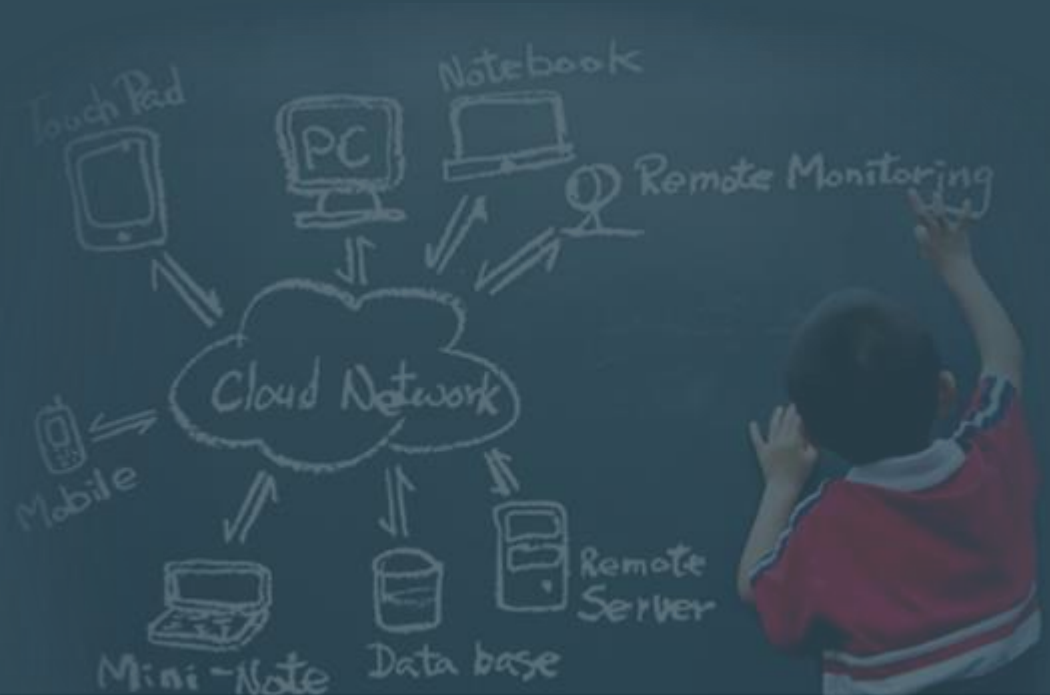
Semplificare lo sviluppo di applicazioni Cloud-Native con DAPR

Andrea Tosato
@ATosato86
Software Architect - HUDI

Microsoft® Most Valuable Professional



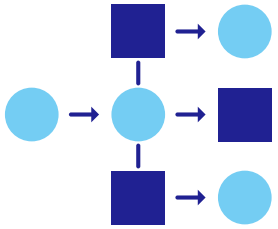
CommunityDays.it

Agenda

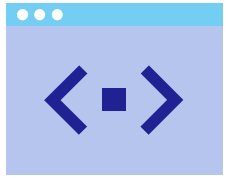
- Overview
- Building Block
- Tools
- Deployments
- Demo
- Osservability
- Closing

State of enterprise developers

Deploying scale-out apps for flexibility, cost, and efficiency

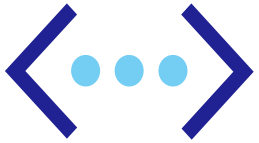


Developing resilient, scalable, microservice-based apps that interact with services



Focusing on building applications, not infrastructure

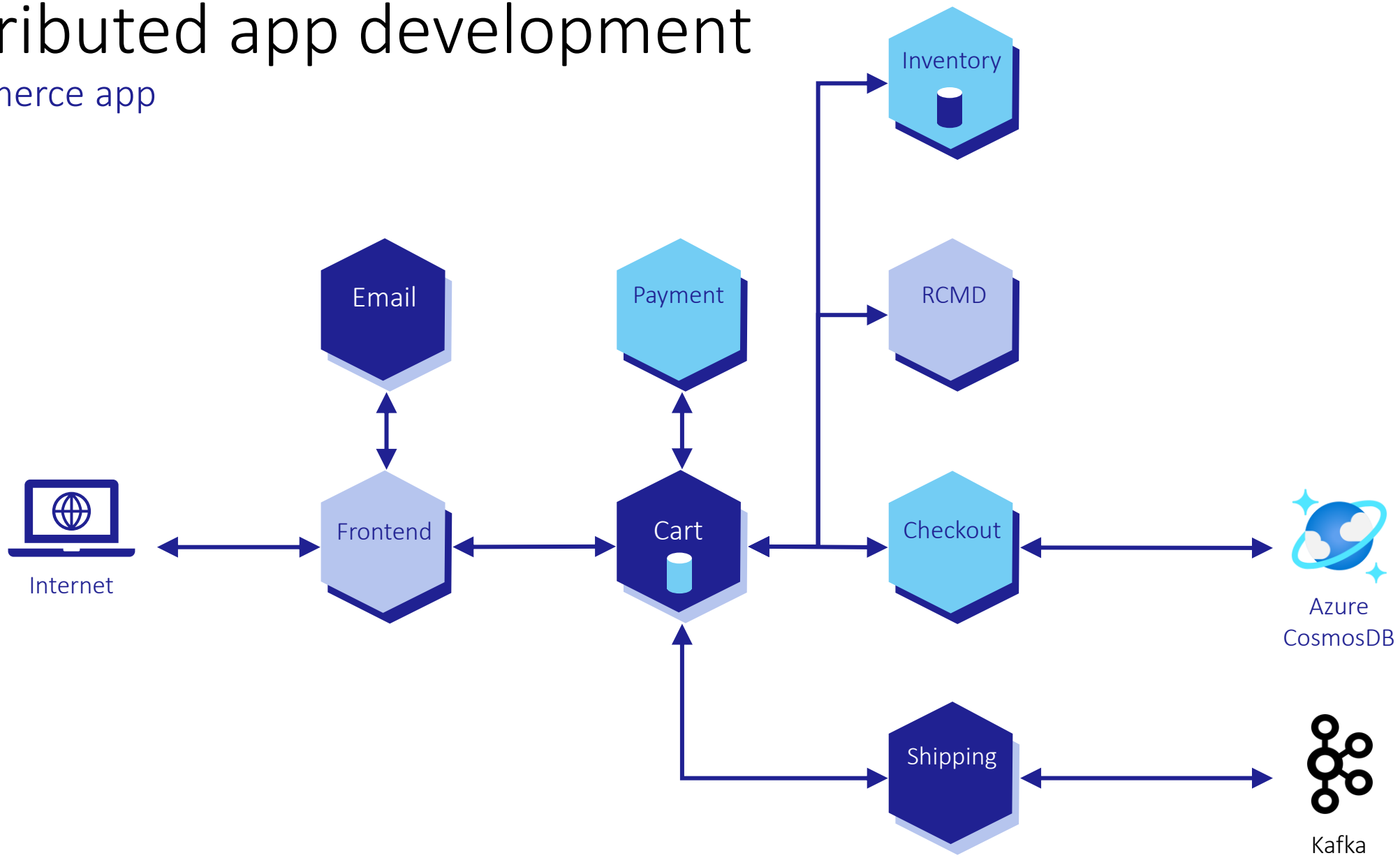
Trending toward serverless platforms with simple code to cloud pipelines



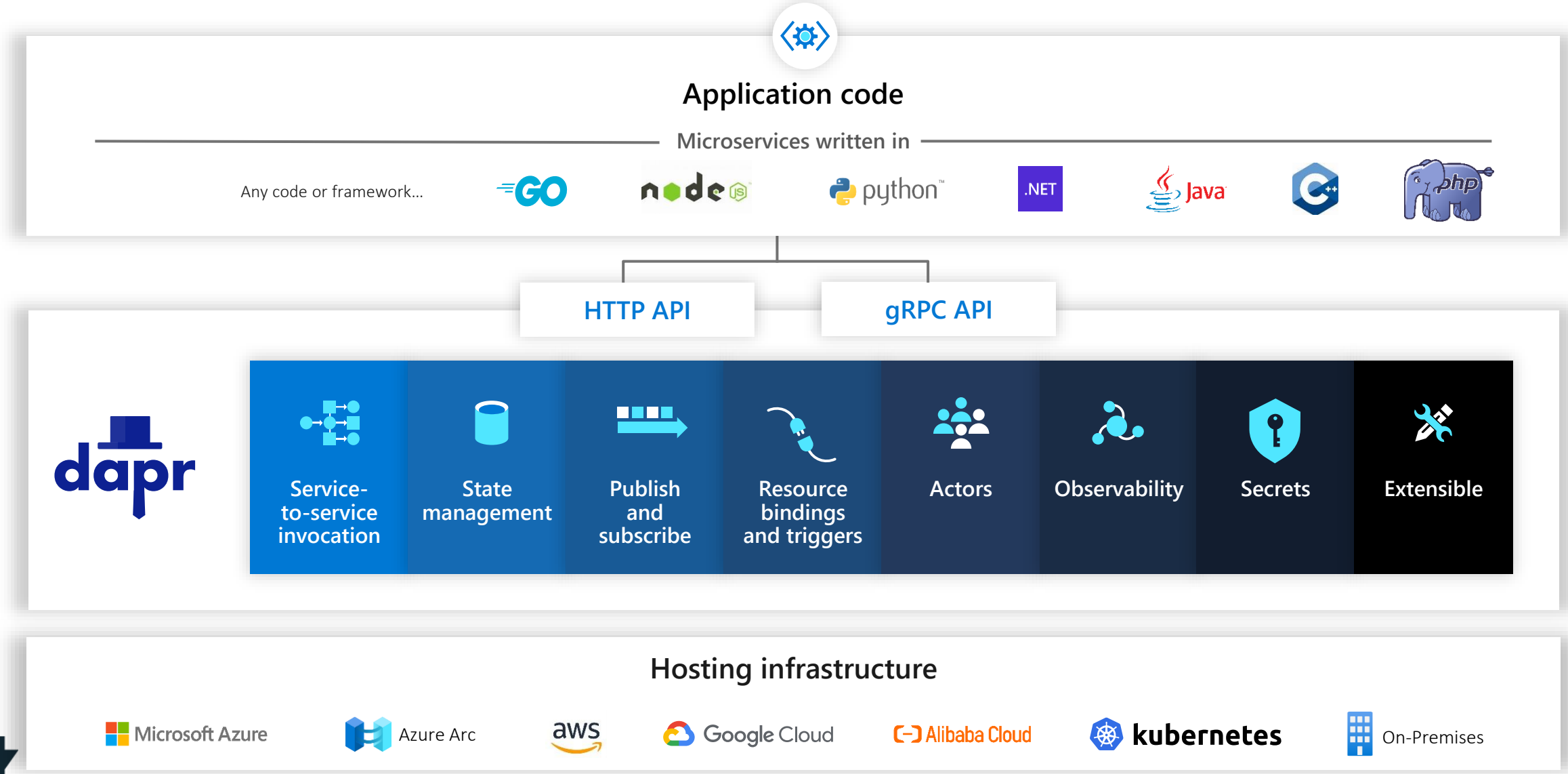
Using multiple languages and frameworks during development

Distributed app development

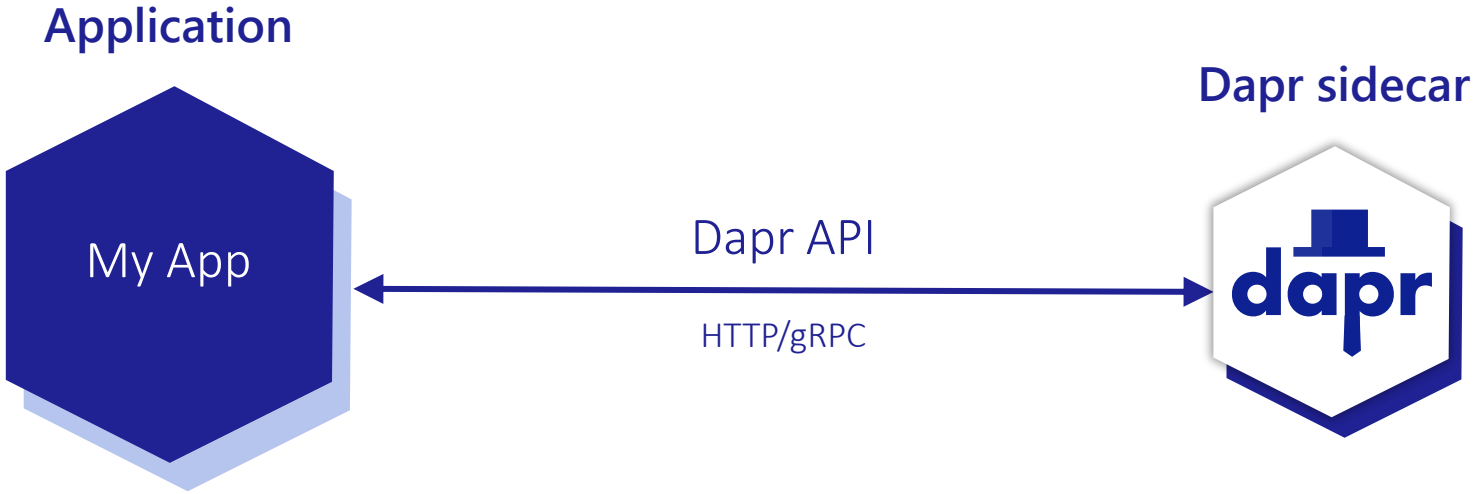
E-commerce app



Any cloud or edge infrastructure



Sidecar model




```
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 components

Swappable YAML files with resource connection details

Over 70 components available

Create components for your resource at: github.com/dapr/components-contrib

State Stores
AWS DynamoDB, Azure CosmosDB, Firebase, Redis, Cassandra

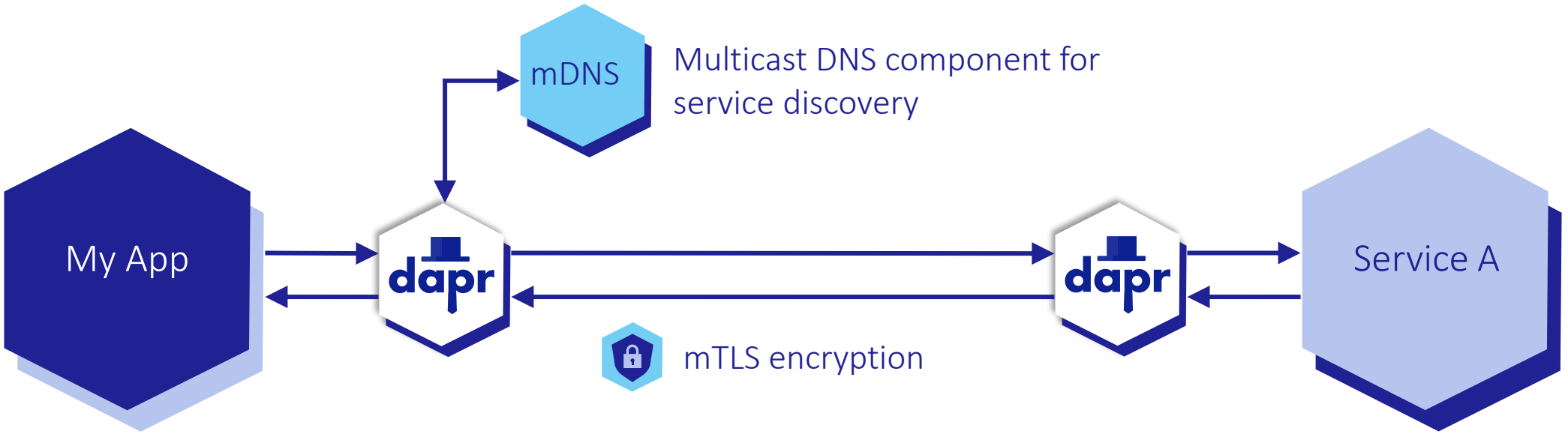
PubSub Brokers
AWS SQS, Azure Service Bus, GCP Pub/Sub, Redis, RabbitMQ

Bindings & Triggers
AWS S3, Azure Storage, GCP Storage, Twilio, Kafka

Secret Stores
AWS Secrets Manager, Azure KeyVault, GCP Secret Manager, HashiCorp Vault, Kubernetes Secret

Observability
Prometheus, AppInsights, Zipkin, Jaeger

Service invocation



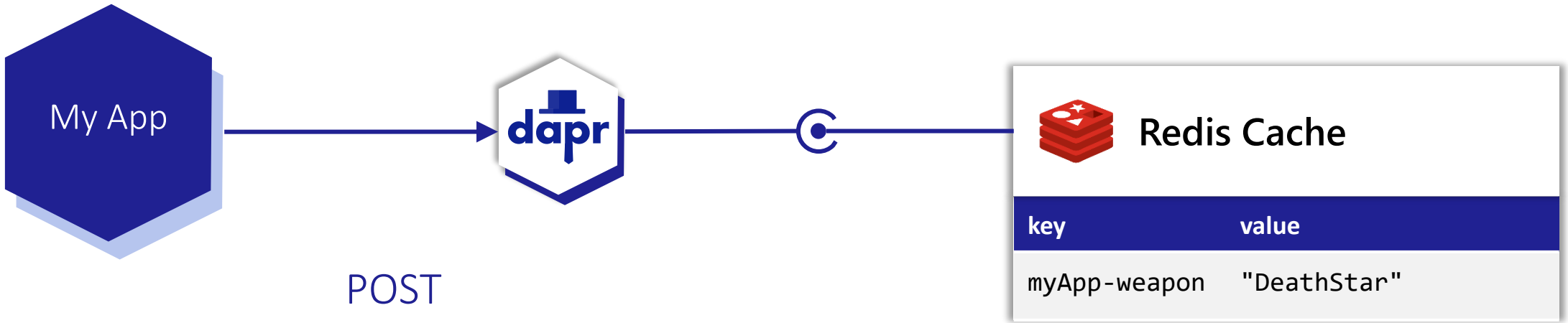
POST
`http://localhost:3500/v1.0/invoke/servicea/method/neworder`

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

POST
`http://10.0.0.2:8000/neworder`

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

State management

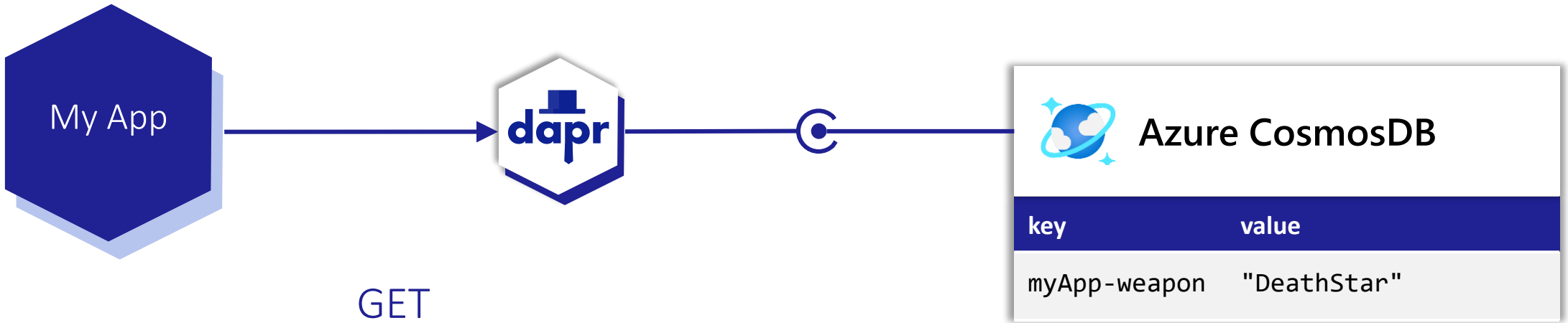


POST

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

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

State management



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
```

corpdb-redis.yaml

```yaml
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
```

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
```

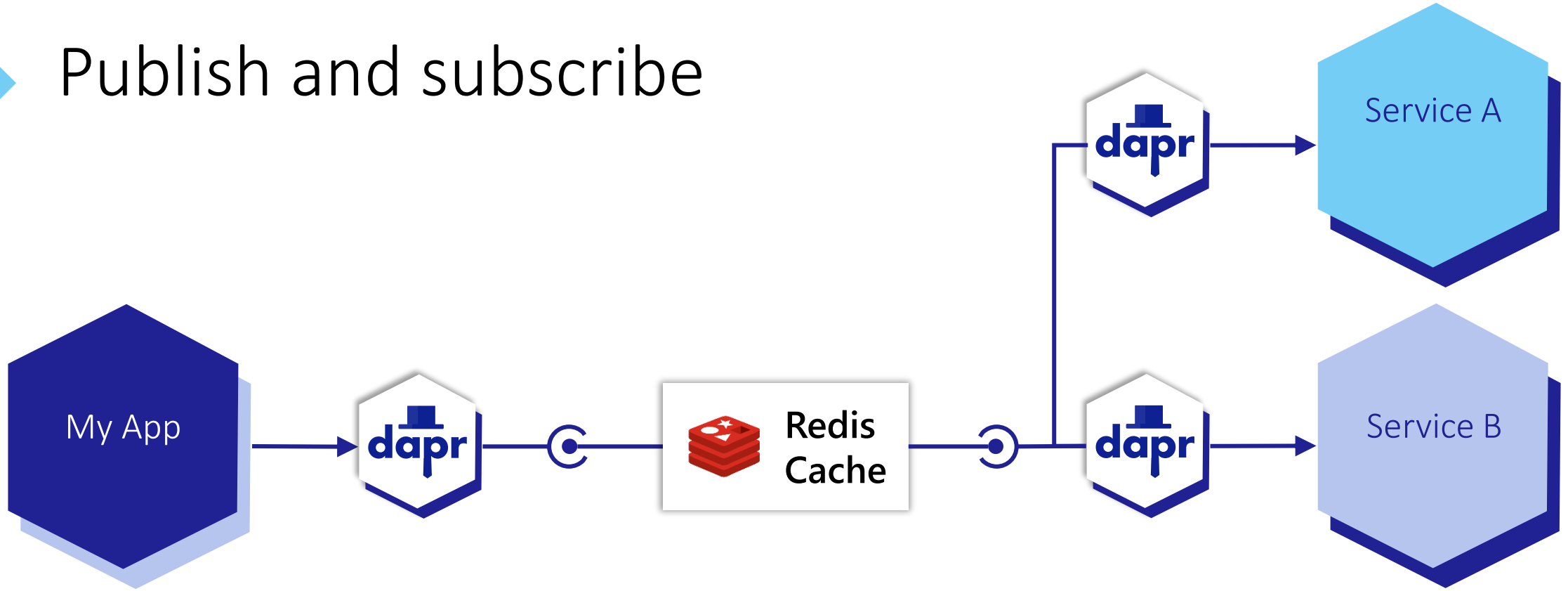
Submit multiple state transactions

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

corpdb-cosmosdb.yaml

```yaml
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
```

Publish and subscribe



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
```

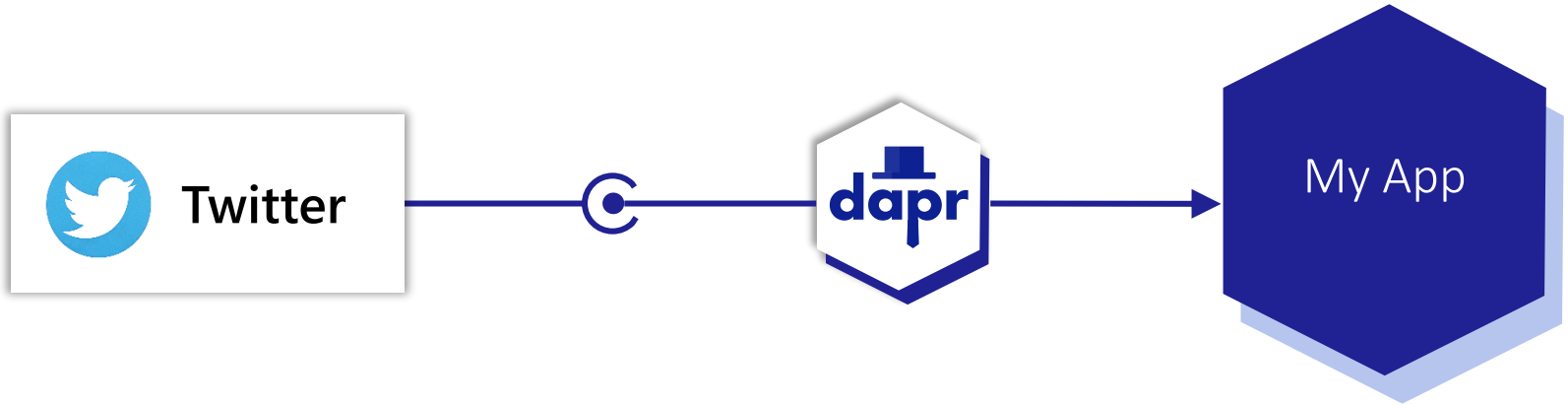
Publish to app

```
POST /order-processing
```

orders.yaml

```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"
```

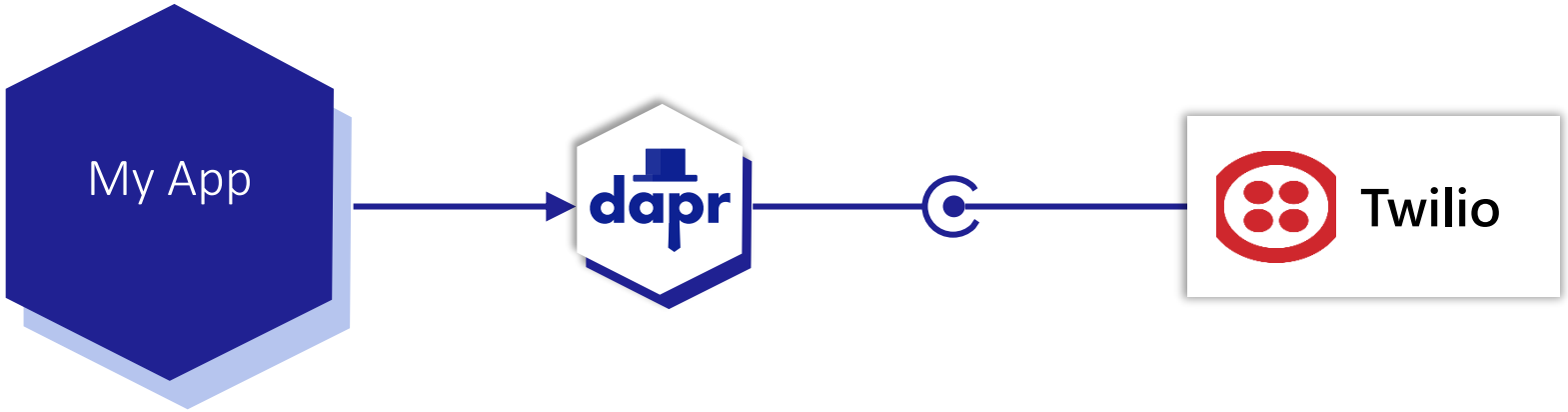

Input triggers



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

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

Output bindings

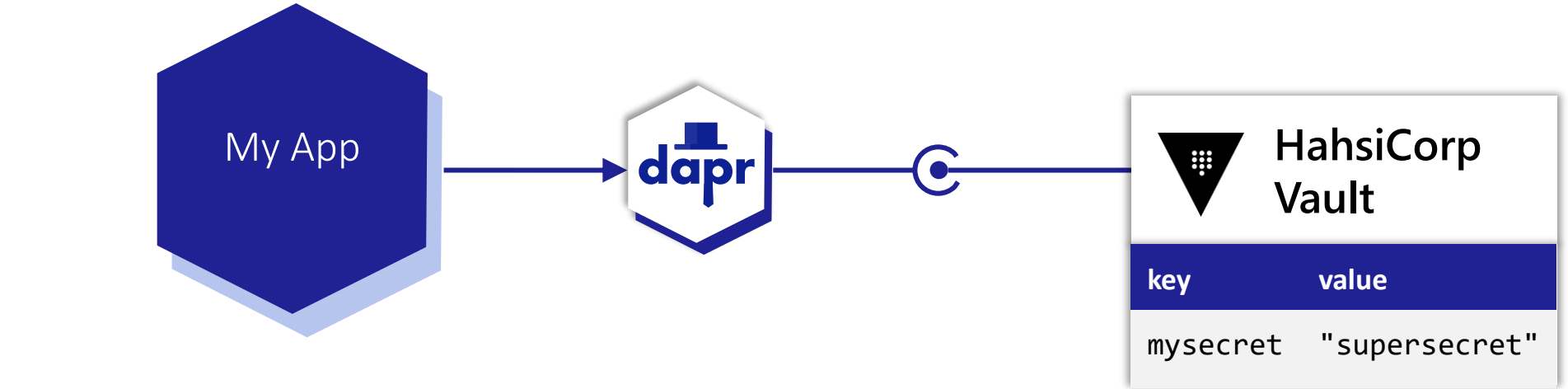


POST
`http://localhost:3500/v1.0/bindings/twilio`

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

Hello World

Secrets



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

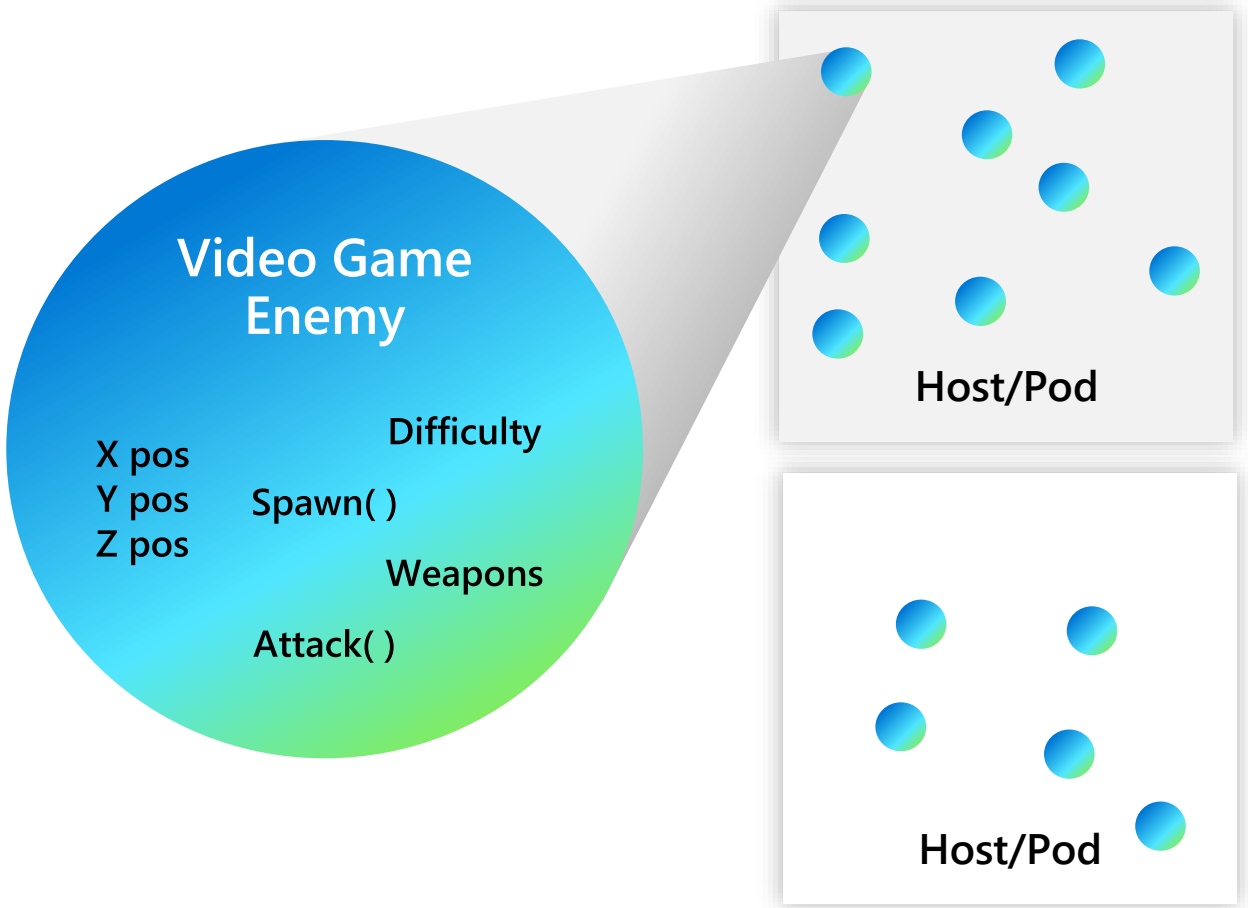
```
"supersecret"
```

Virtual actors

Stateful, objects of storage and compute

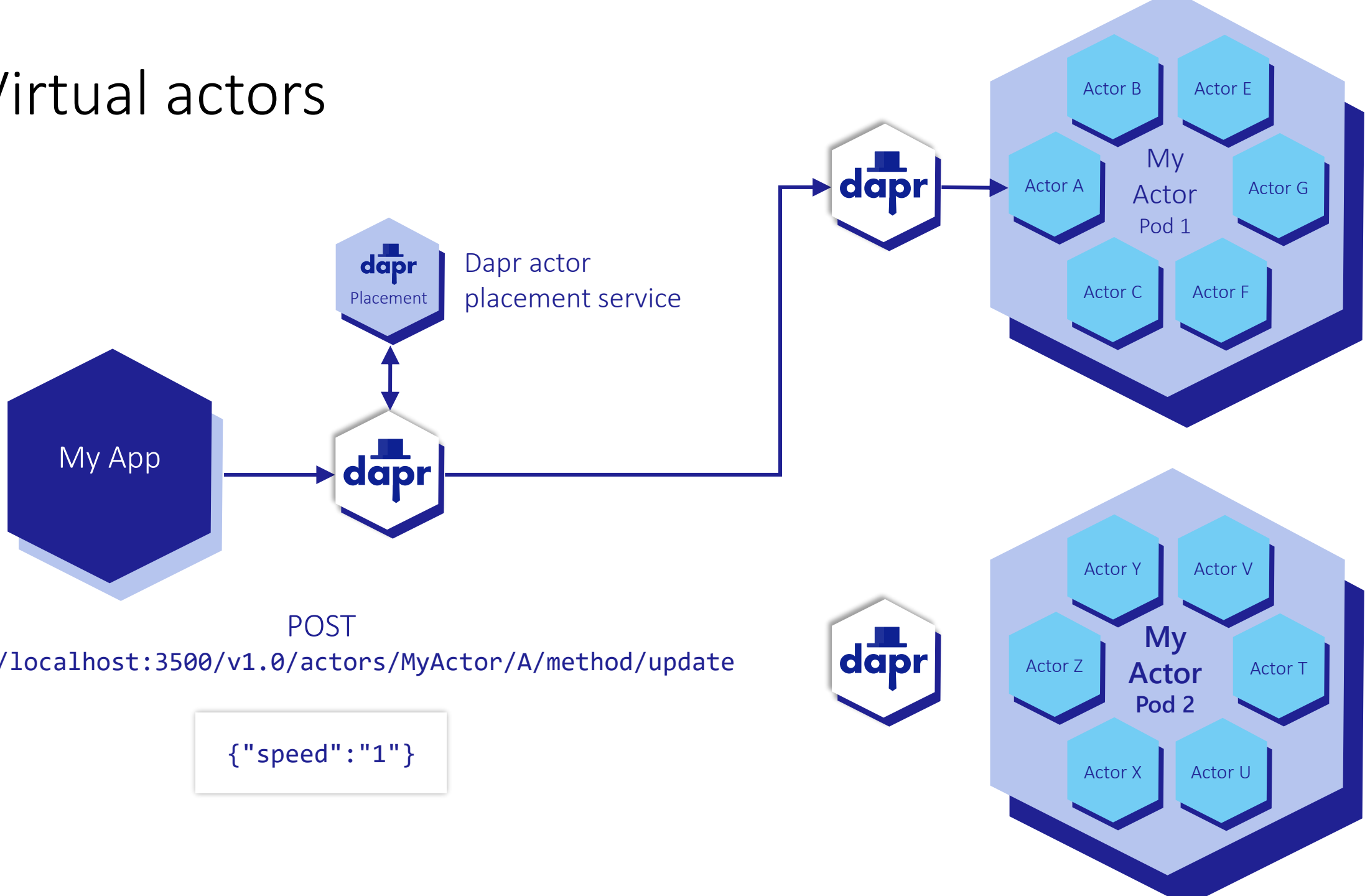
Dapr Actor features:

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



Virtually identical to Service Fabric Reliable Actors

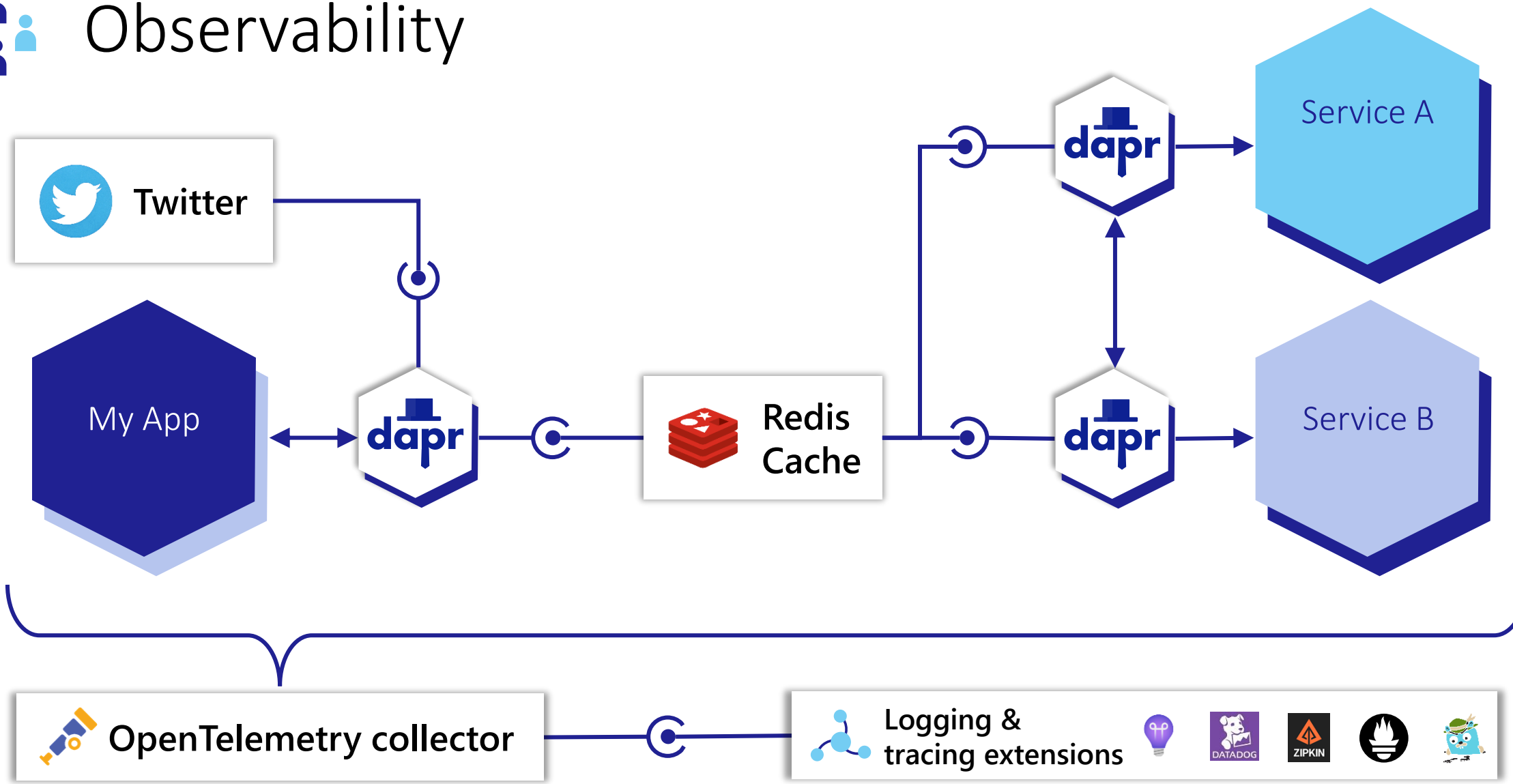
Virtual actors



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

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

Observability

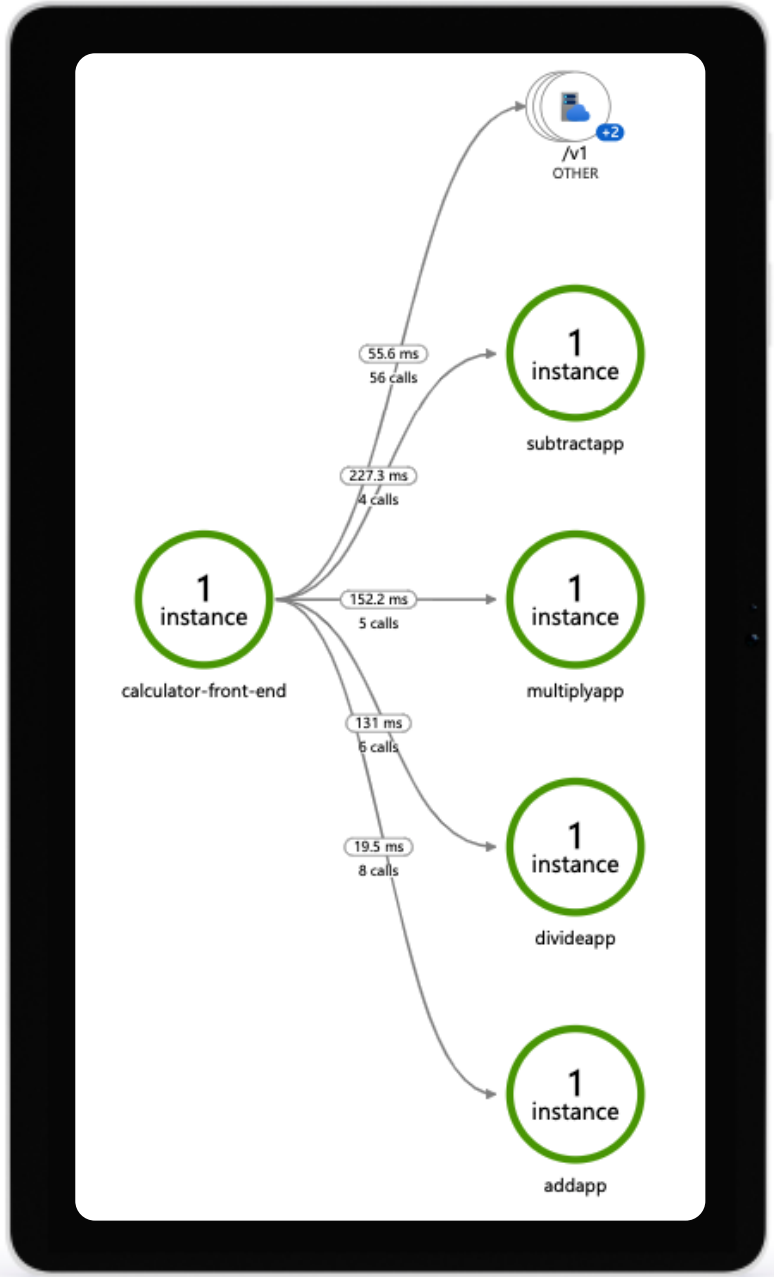


Distributed tracing

Emit tracing data from calls to/from Dapr sidecars and system services for easy application-level instrumentation

Dapr distributed tracing features:

- Built-in Zipkin collection and viewer
- Configurable sampling rates
- TBD
- TBD
- TBD



Metrics

Built-in monitoring capabilities to understand the behavior of the Dapr sidecar and system services

Dapr Metrics features:

- Call latency
- CPU/memory usage
- Error rates
- Sidecar injection failures
- System health

Dapr hosting environments

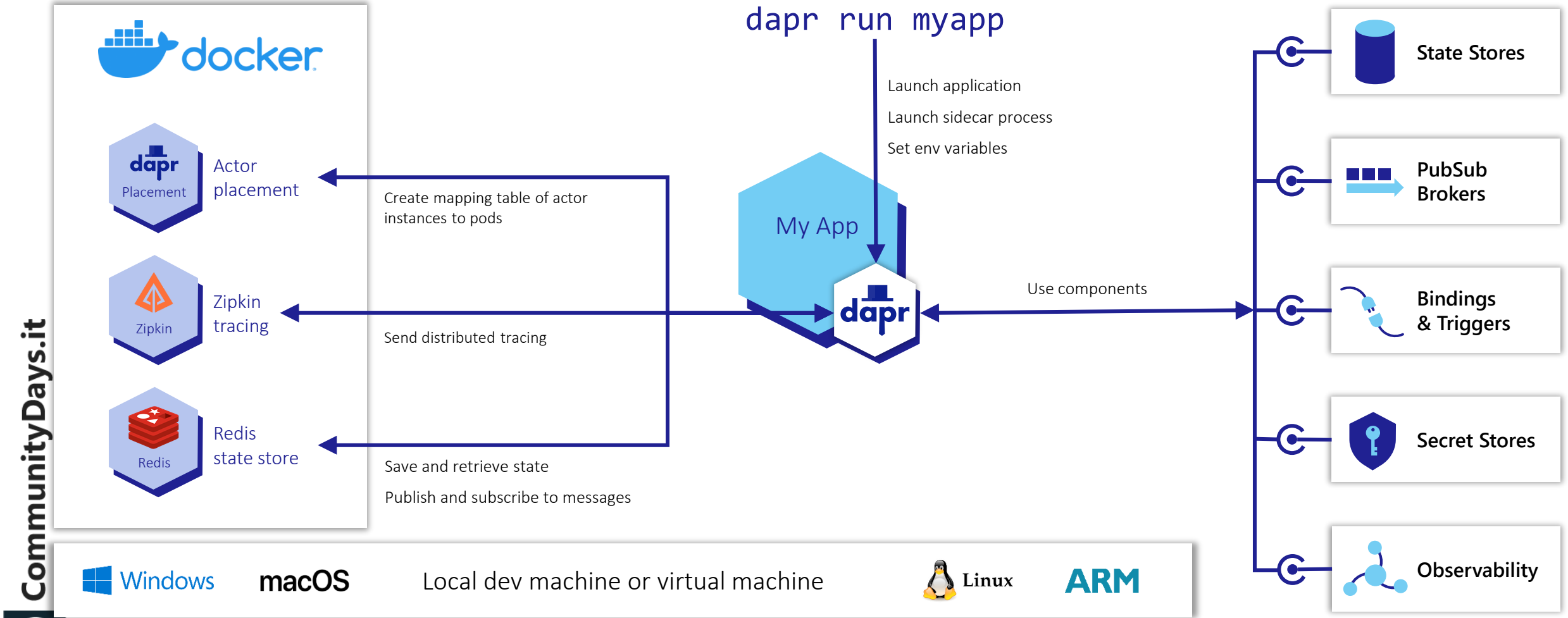
Self-hosted

- Get started with `dapr init`
- Easy setup with Docker images
  - Sets up placement, Zipkin, Redis
  - `slim-init` available without Docker
- Run any application with Dapr sidecar using `dapr run`

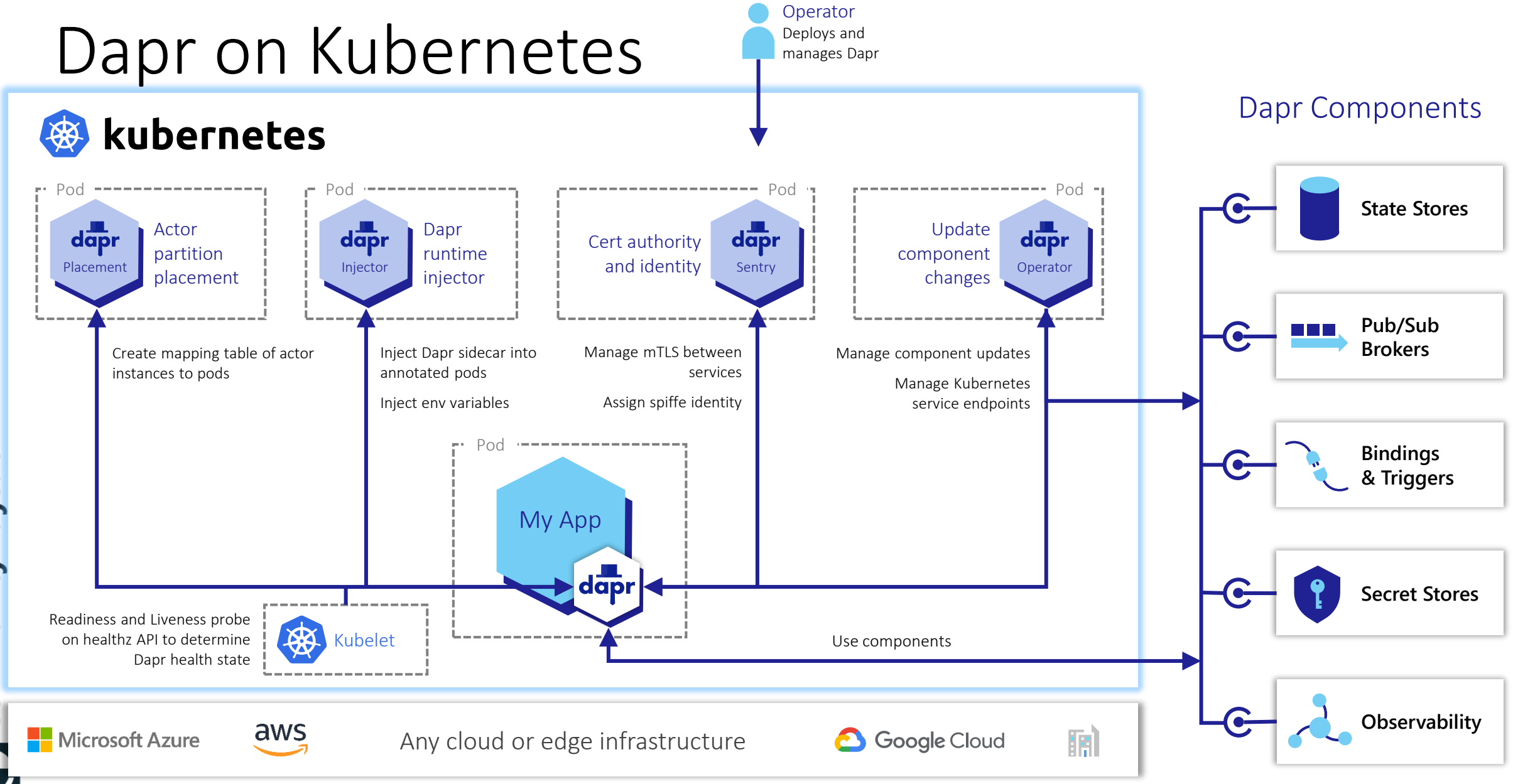
kubernetes

- Get started with `dapr init -k`
- Fully managed Dapr control plane
  - Deploys dashboard, placement, operator, sentry, and injector pods
- Automatically inject Dapr sidecar into all annotated pods
- Upgrade with `dapr upgrade` or Helm

Dapr in self-hosted Docker mode



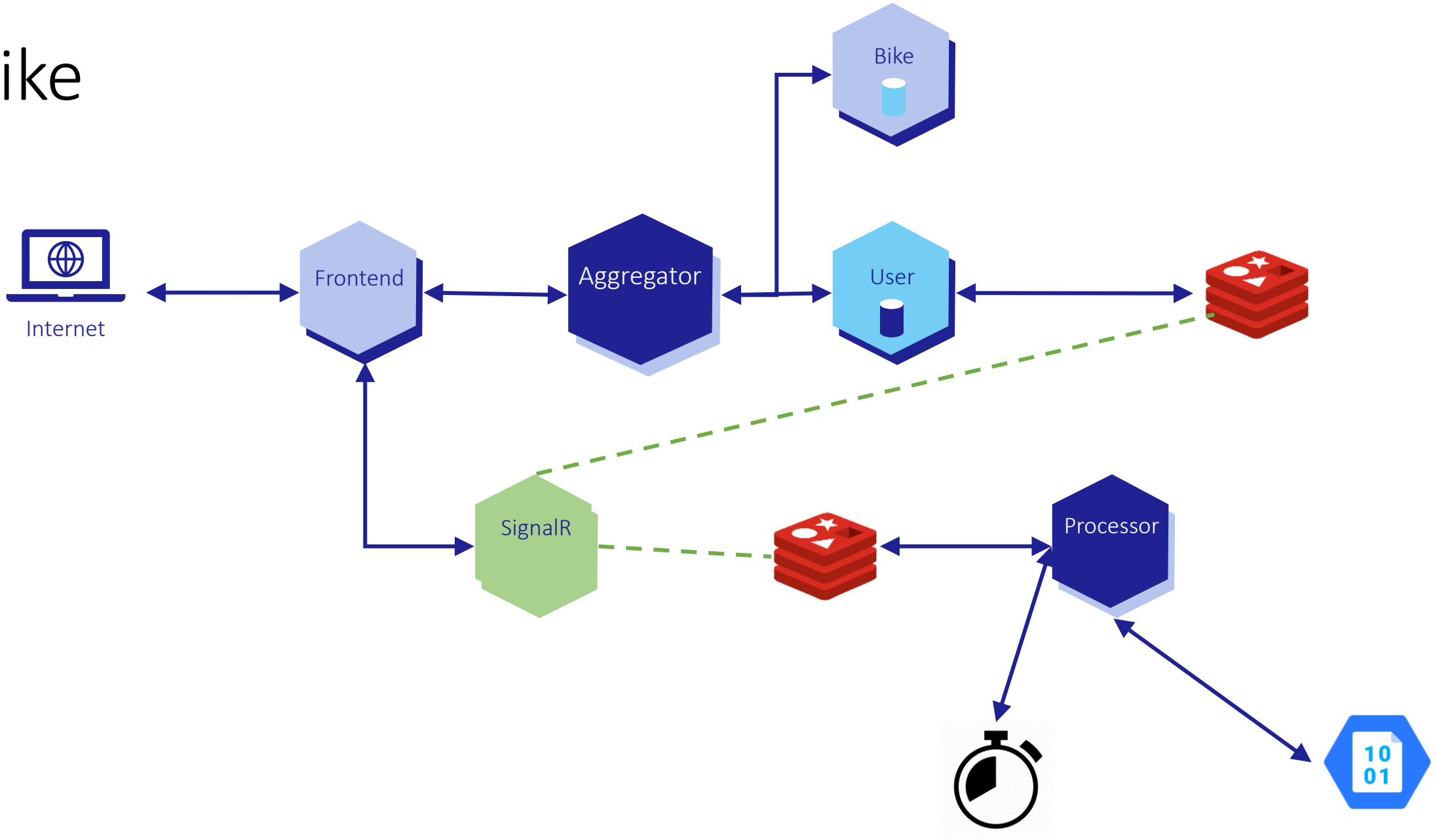
Dapr on Kubernetes



Demo

Bike Service

eBike

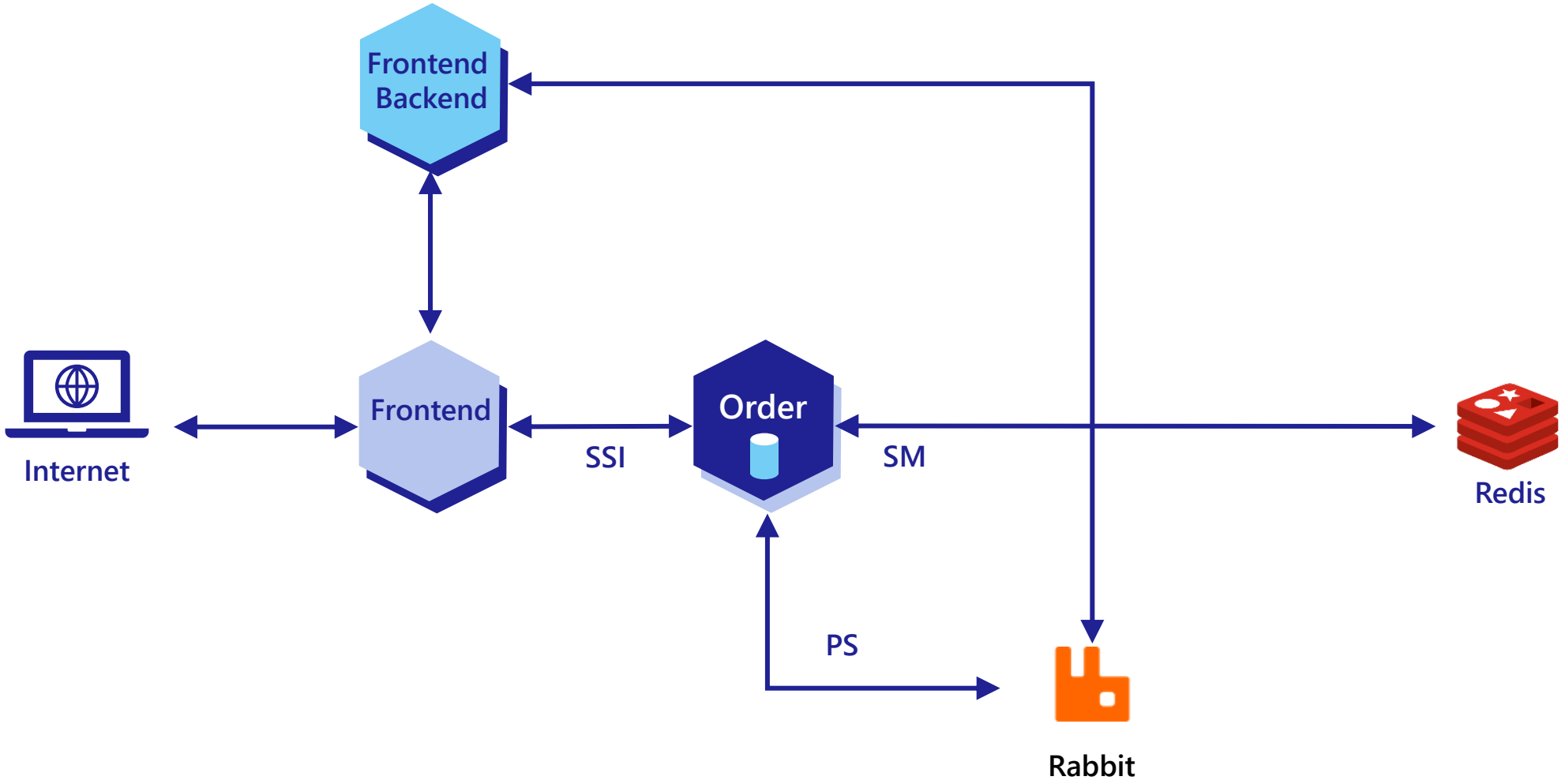


Demo

Order Service

Sample App – Order

SSI = Servcie To Service Invocation
PS = Publish & Subscribe
SM = State Managment



CommunityDays.it

Grazie!

- Il materiale sarà online nei prossimi giorni su http://www.communitydays.it