

.NET Microservices with Dapr

Marc Müller
Principal Consultant



marc.mueller@4tecture.ch
[@muellermarc](https://twitter.com/muellermarc)
www.4tecture.ch

4tecture[®]
empower your software solutions



About me:

Marc Müller
Principal Consultant
@muellermarc



dotNET lab

4tecture
empower your software solutions

Our Products:

Multi-Tenant OpenID
Connect Identity Provider



ProAuth

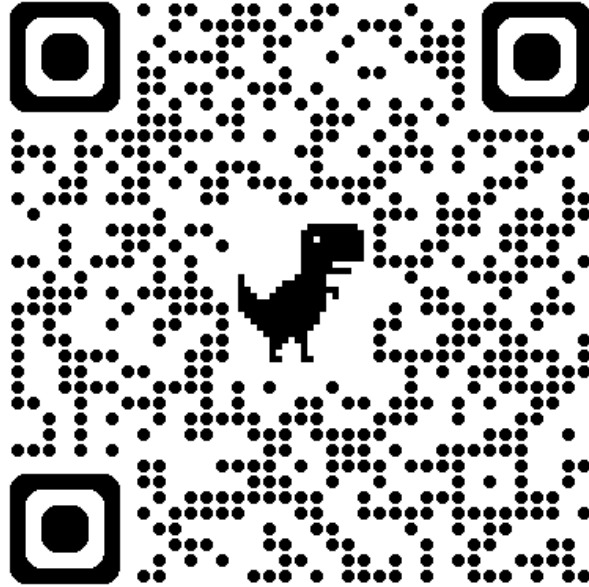
www.proauth.net

Enterprise Application
Framework for .NET

ReaFx

www.reafx.net

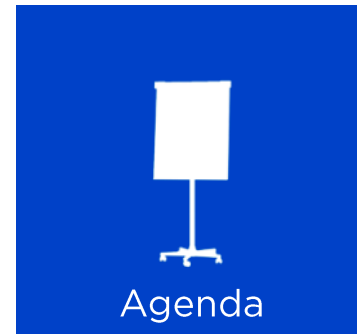
Slide Download



<https://www.4tecture.ch/events/dwx24dapr>

Agenda

- Intro
- Dapr Basics
 - BB: Service Invocation
 - BB: Pub/Sub
 - BB: State Management
 - BB: Virtual Actors
 - BB: Secret Management
 - BB: Observability
 - BB: Resiliency
- Conclusion



A close-up, low-angle shot of several rowers in a boat, wearing blue and red uniforms, pulling their oars. The oars are black with yellow handles and are connected to a complex mechanical system. The background is a bright, slightly blurred body of water.

Dapr

Intro

4tecture[®]
empower your software solutions

Cloud computing characteristics by NIST

- On-demand self-service
- Broad network access
- Resource pooling
- Rapid elasticity
- Measured service

CNCF cloud native definition

Cloud native technologies empower organizations to build and run scalable applications in modern, dynamic environments such as public, private, and hybrid clouds. Containers, service meshes, microservices, immutable infrastructure, and declarative APIs exemplify this approach.

These techniques enable loosely coupled systems that are resilient, manageable, and observable. Combined with robust automation, they allow engineers to make high-impact changes frequently and predictably with minimal toil.

The Cloud Native Computing Foundation seeks to drive adoption of this paradigm by fostering and sustaining an ecosystem of open source, vendor-neutral projects. We democratize state-of-the-art patterns to make these innovations accessible for everyone.

Cloud native applications

Cloud-native applications should be designed and implemented to use and take advantage of the cloud computing characteristics.

Cloud native applications characteristics

- Immutable Packaging & Execution
- Decoupled Configurations and Secrets
- Statelessness and Statefulness
- Modular Applications / Microservices
- Polyglot Paradigm (multi lang / tech)
- Centralized Logging and Monitoring
- DevOps and SRE processes
- Automation
- API-Centric

Commonly used with cloud native apps

- Microservices
- Cloud platforms
- Containers
- Kubernetes
- Immutable infrastructures
- Declarative APIs
- Continuous Delivery technologies

Challenges with Distributed Apps

- Lots of components / services
- Service-to-service communication
- Decoupling by using events
- Handling state across multiple instances
- Stateful services / actors
- Local dev environment vs. cloud environment

dapr.io



Portable, event-driven runtime



Build connected distributed applications faster

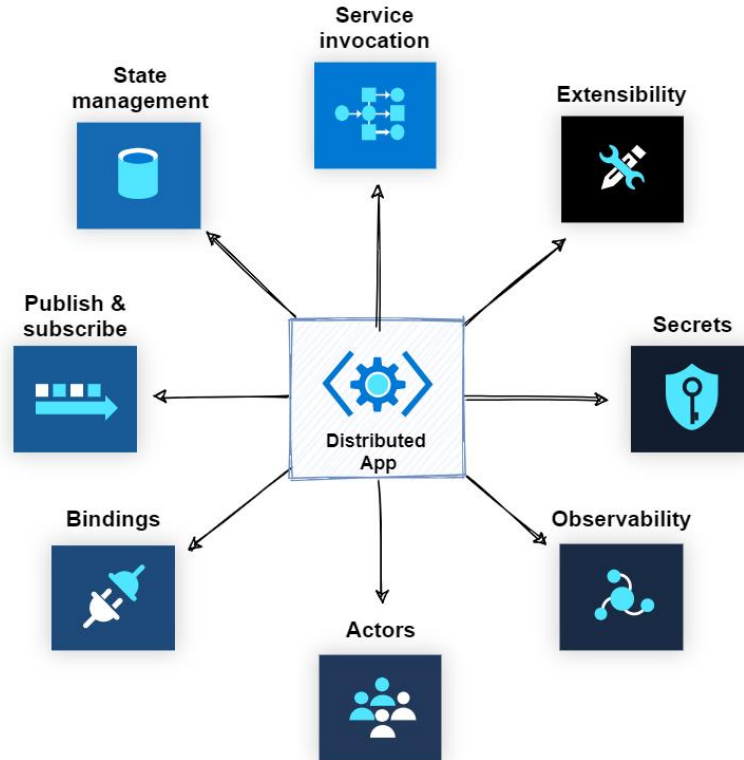


APIs for solving distributed application challenges

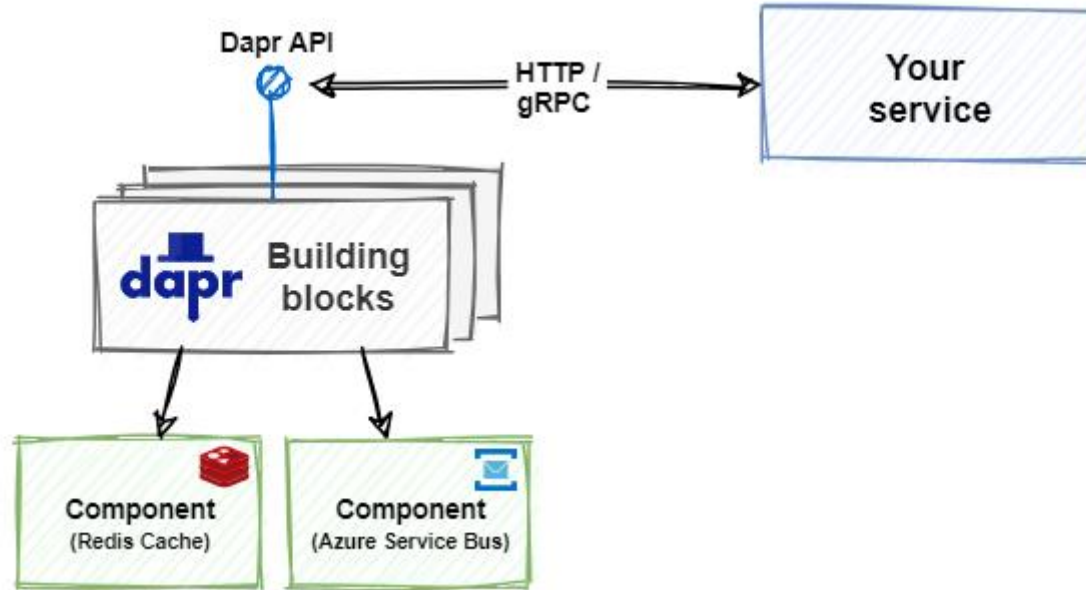


Cloud and Edge

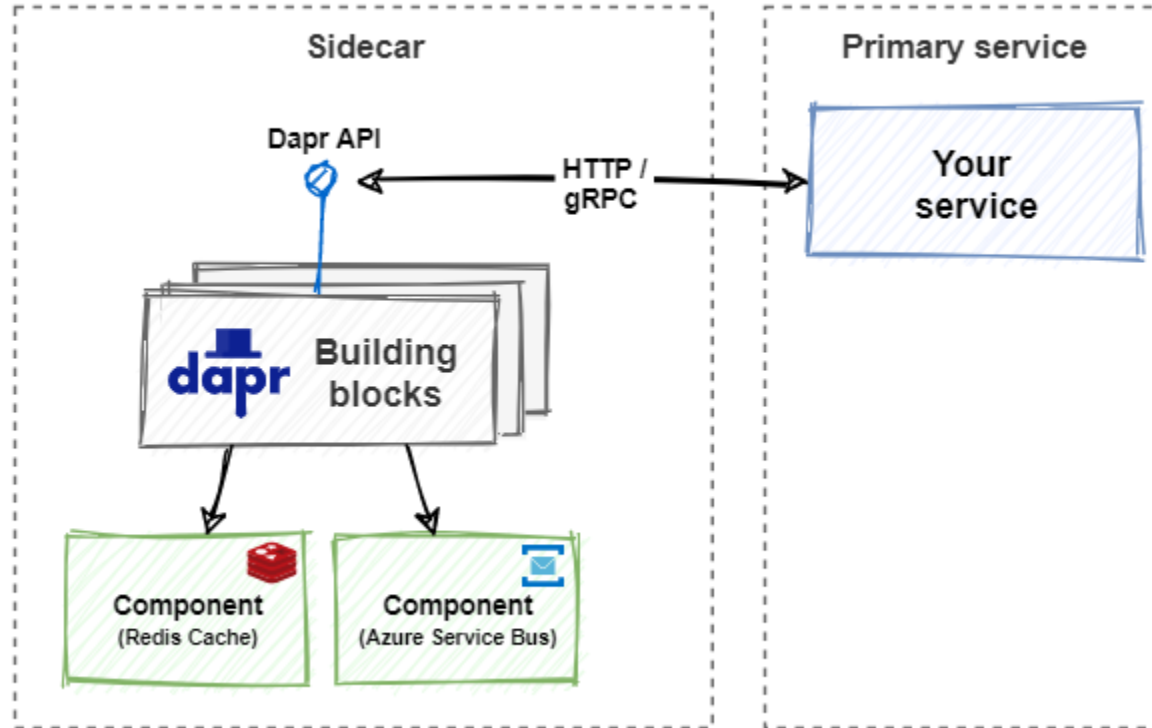
Dapr Building Blocks



Dapr Building Blocks Abstraction



Dapr Sidecar Architecture





Dapr Basics

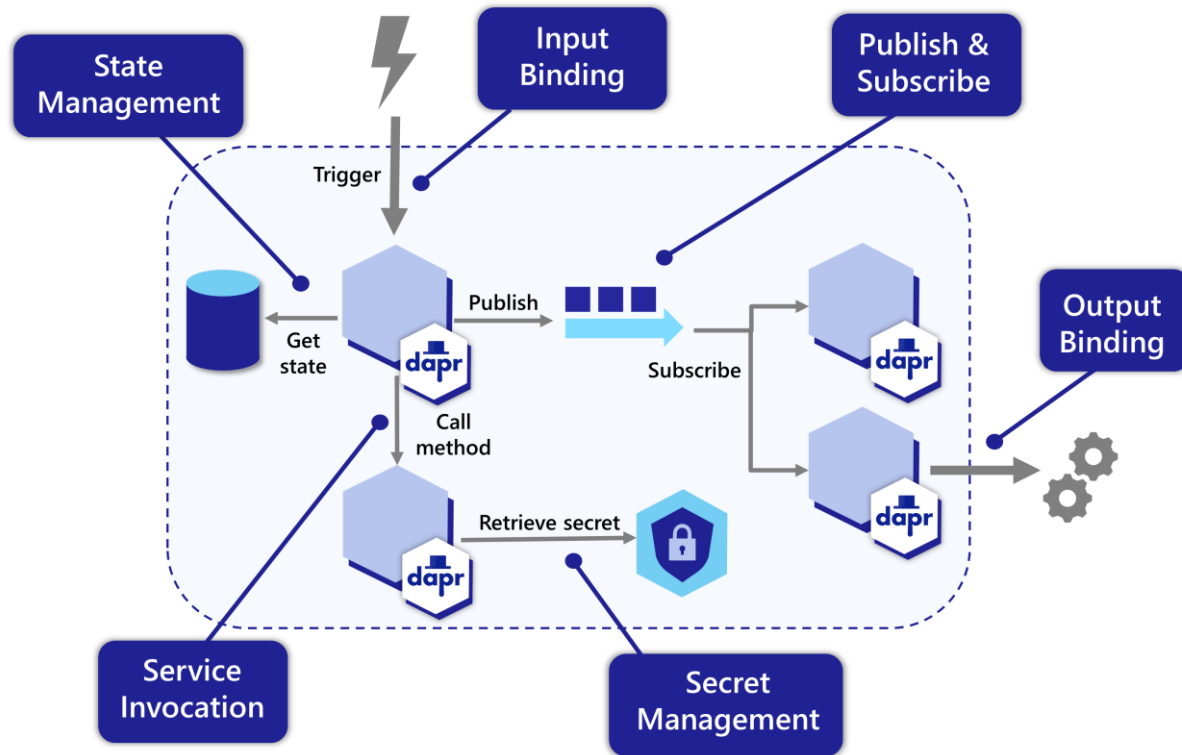
Dapr

- dapr.io
- Open source
- Originated at Microsoft
- Cloud Native Computing Foundation (CNCF)

Dapr – High Level Definition

- Any language or framework
- Portable APIs
- Building blocks applying best practices
 - Use the blocks you need
 - No big bang framework
- Platform agnostic
- Extensible and pluggable components

Dapr Building Blocks Overview



Single API - Multiple Components

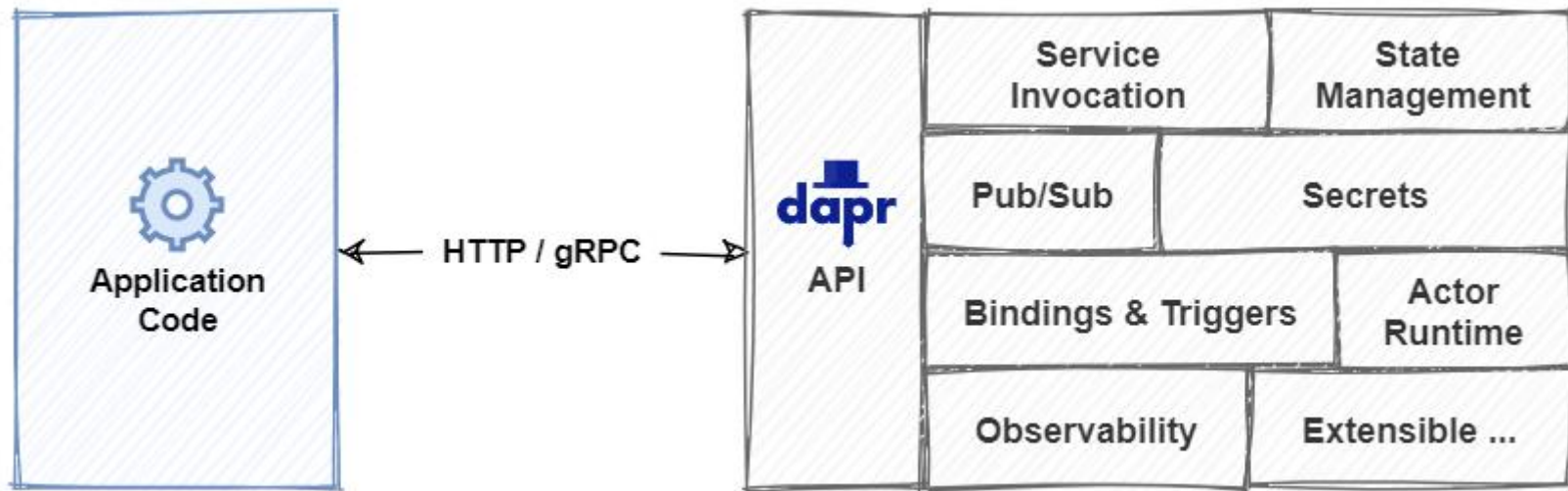
```
var weatherForecast =  
await daprClient.GetStateAsync<WeatherForecast>("statestore", "AMS");  
  
daprClient.SaveStateAsync("statestore", "AMS", weatherForecast);
```

- AWS DynamoDB
- Aerospike
- Azure Blob Storage
- Azure CosmosDB
- Azure Table Storage
- Cassandra
- Cloud Firestore (Datastore mode)
- CloudState
- Couchbase
- Etcd
- HashiCorp Consul
- Hazelcast
- Memcached
- MongoDB
- PostgreSQL
- Redis
- RethinkDB
- SQL Server
- Zookeeper

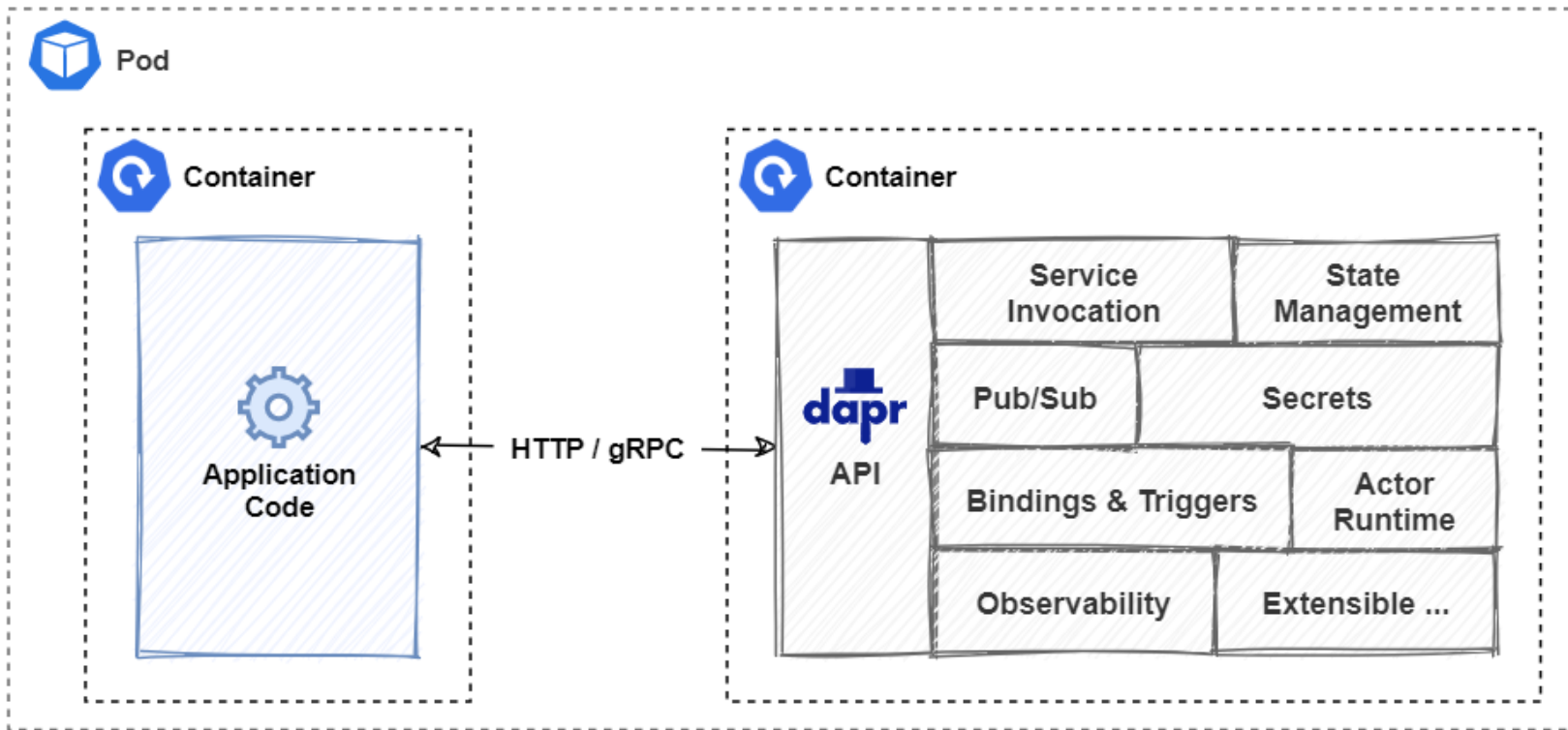
The Dapr sidecar provides built-in
security, **resiliency** and **observability**
capabilities.

Speeds up application development by providing an integrated set of APIs for communication, state, and workflow.

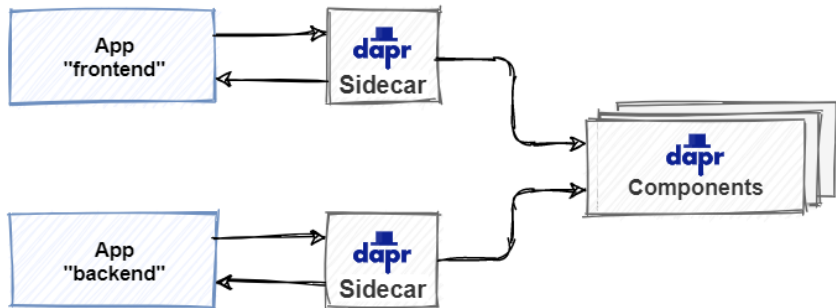
Self-hosted Sidecar



Kubernetes-hosted Sidecar



Sidecar Performance Considerations



- Dapr operation:
≥ 1 out-of-process network call
- Heavily optimized sidecar implementation
- gRPC with multiplexing, bidirectional full-duplex, streaming
- Overhead should be sub-millisecond

SDK Overview

Language	Status	Client	Server extensions	Actor	Workflow
.NET	Stable	✓	ASP.NET Core	✓	✓
Python	Stable	✓	gRPC FastAPI Flask	✓	✓
Java	Stable	✓	Spring Boot Quarkus	✓	✓
Go	Stable	✓	✓	✓	✓
PHP	Stable	✓	✓	✓	
Javascript	Stable	✓		✓	✓
C++	In development	✓			
Rust	In development	✓		✓	

Runtimes



Self-Hosted

- Dedicated process next to your application process
- Executable or Docker Image

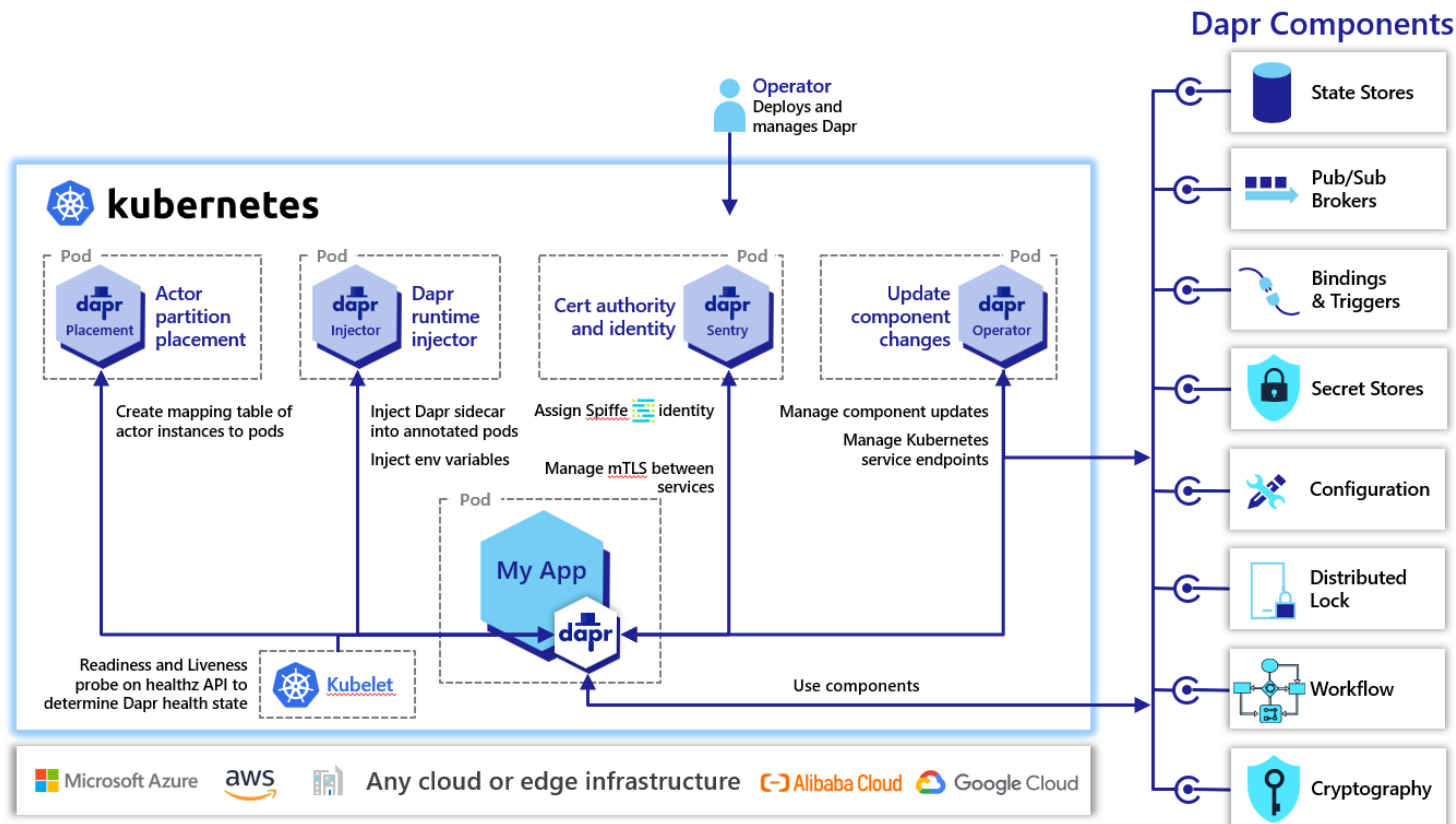
Kubernetes

- Sidecar container in your pod, uses localhost interface
- Usually injected based on attributes

Serverless

- Integrated in Azure
- i.e. Container Apps

Dapr in Kubernetes



Dapr from development to hosting

Use any language or runtime



HTTP/gRPC



Workflow



Publish /
Subscribe



Service
Invocation



State
Management



Actors



Observability



Security



External
Configuration



Secrets



Bindings



Cryptography



Distributed
lock

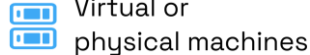


Middleware

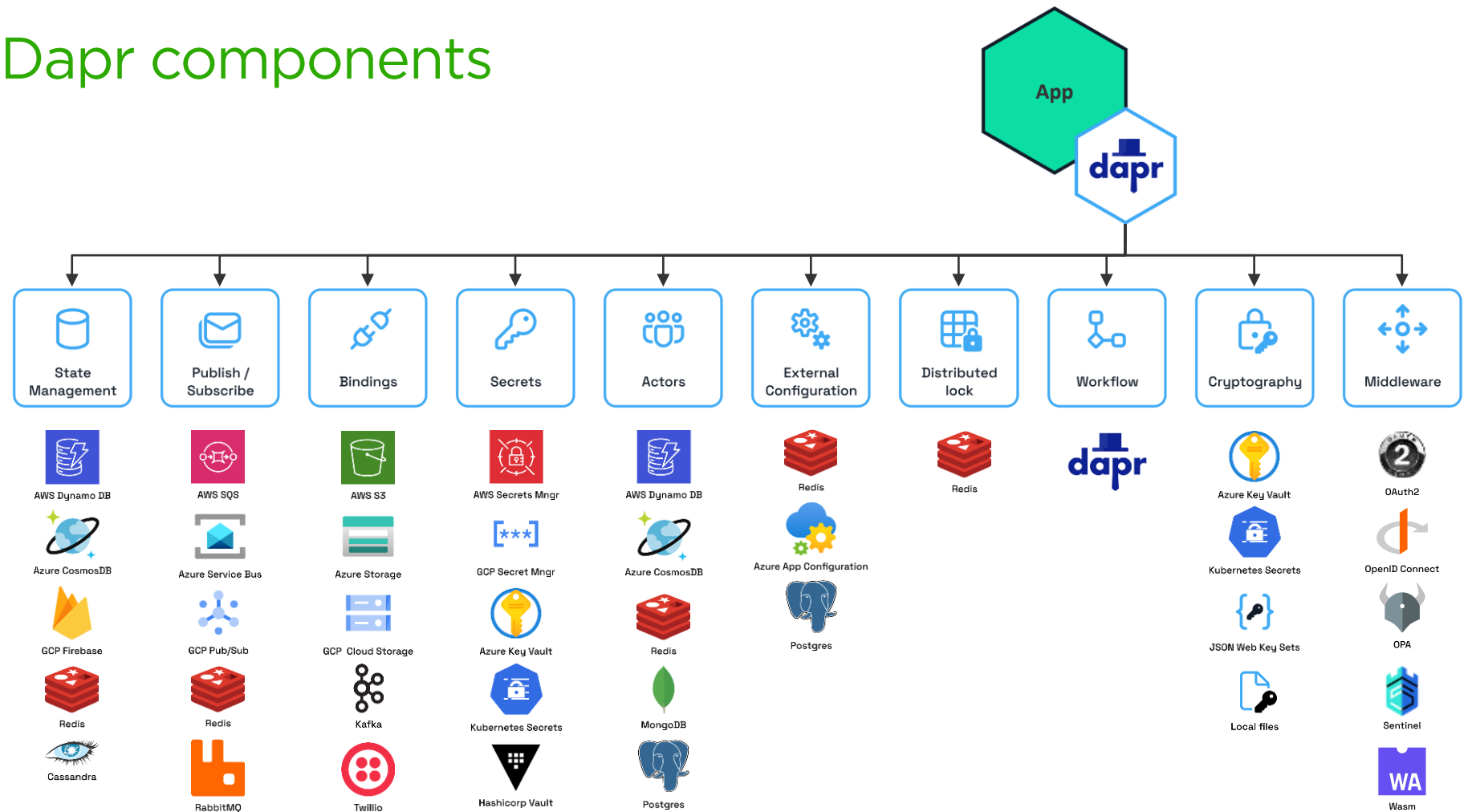


Resiliency

Host on any cloud or edge infrastructure



Dapr components

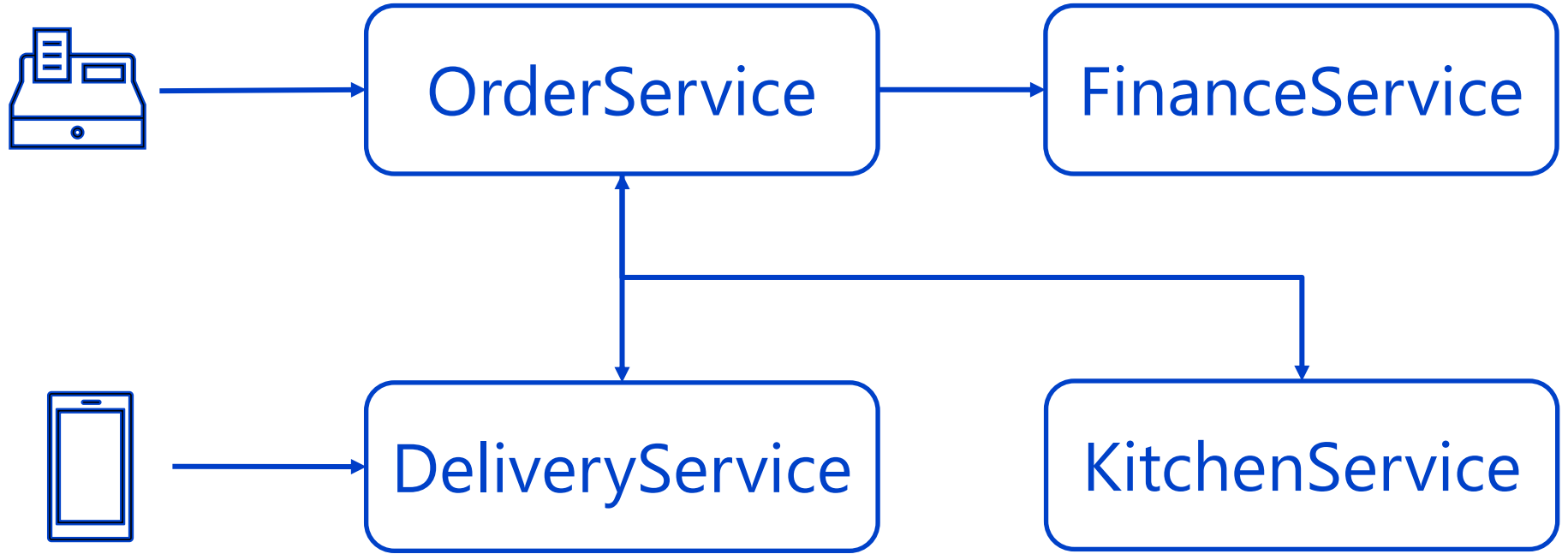


A close-up, low-angle shot of several rowers in a boat, wearing blue and red uniforms, pulling their oars. The oars are black with yellow and blue handles. The background is a bright, slightly blurred body of water.

Demo first

4tecture[®]
empower your software solutions

Fast Food Company





A person is seen from the side, sitting at a desk in a dark room. They are looking at two computer monitors. The left monitor displays a web application with a sidebar and a main content area. The right monitor displays a code editor with syntax-highlighted code. A red Coca-Cola can is on the desk between the monitors. The person's hands are on a keyboard. The word "DEMO" is overlaid in large white letters on the left side of the image.

DEMO

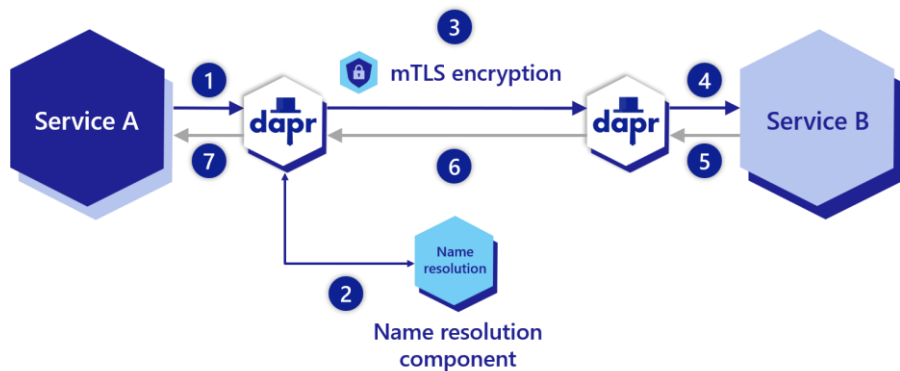
A background image showing a rowing team in a boat. The rowers are wearing blue and red uniforms and are captured in a synchronized rowing motion. The focus is on the oars and the rowers' hands, with the water visible in the foreground.

Dapr Building Blocks

Service Invocation and Configuration

4tecture[®]
empower your software solutions

Service Invocation

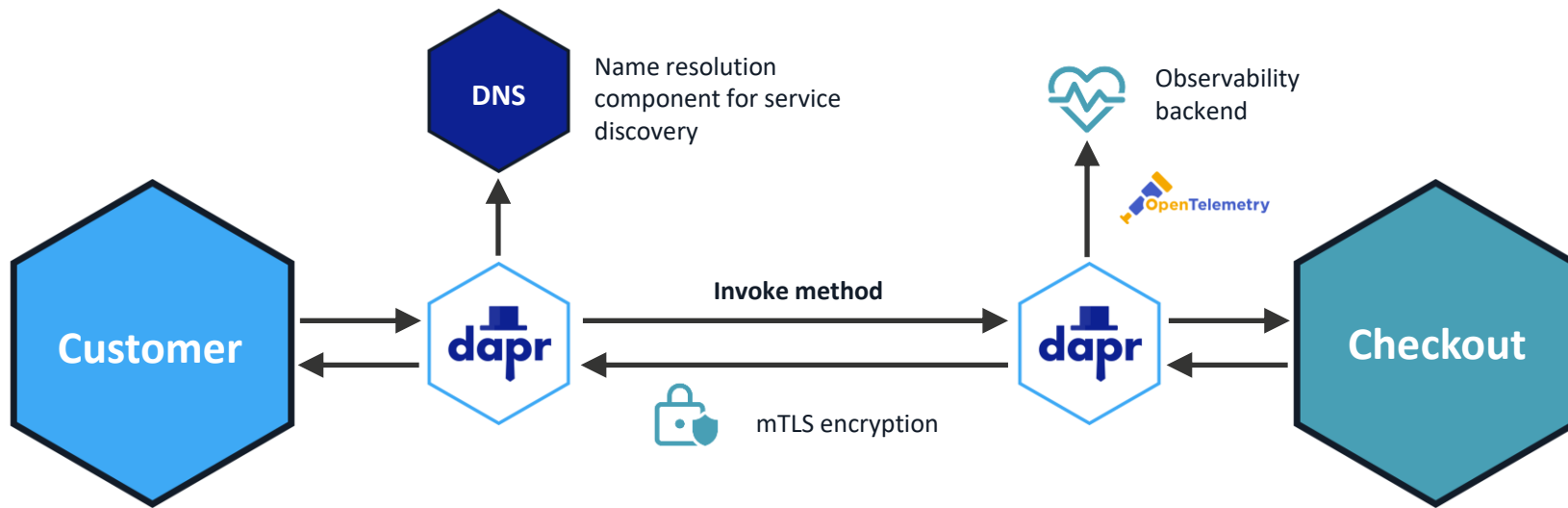


- HTTP and gRPC
- mTLS (with Dapr Sentry)
- Resiliency including retries
- Tracing and metrics with observability
- Access control (policies)
- Namespace scoping
- Load balancing (round robin with mDNS)
- Pluggable service discovery

Service Invocation



Service
Invocation



POST

<http://localhost:3500/v1.0/invoke/checkout/method/order>

POST

<http://localhost:5100/order>

A person is shown in profile, sitting at a desk and working on two computer monitors. The room is dark, with the primary light source being the screens. The left monitor displays a web application with a sidebar and a main content area. The right monitor shows a code editor with syntax-highlighted text. On the desk, there is a red Coca-Cola can, a white computer mouse, and a small container with pens. The person's hands are on a keyboard, which is partially visible.

DEMO

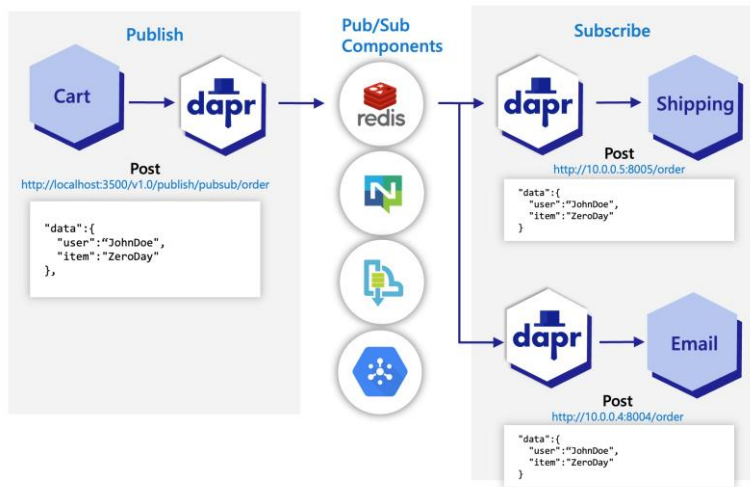
A close-up, low-angle shot of rowers in a boat, showing their legs and hands gripping oars. The rowers are wearing blue and red uniforms. The oars are black with yellow and blue handles. The background is a bright, slightly blurred body of water.

Dapr Building Blocks

Publish & Subscribe

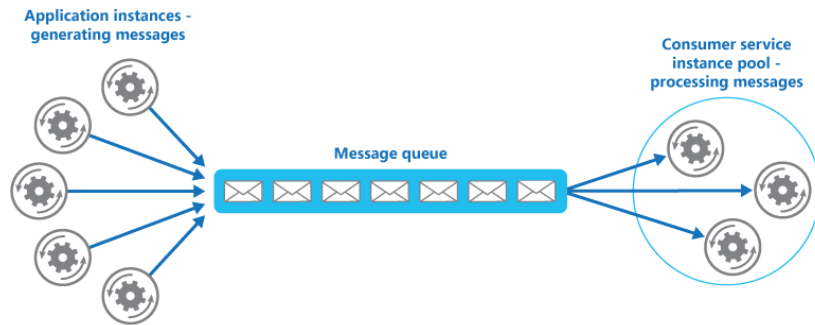
4tecture[®]
empower your software solutions

Publish & Subscribe



- Platform-agnostic API to send and receive messages
- At-least-once message delivery guarantee
- Integration with various message brokers
- CloudEvents 1.0 specification
- Message content type
- Content-based Routing
- Dead letter topics
- Namespace consumer groups
- Scoping topics

Competing consumers pattern

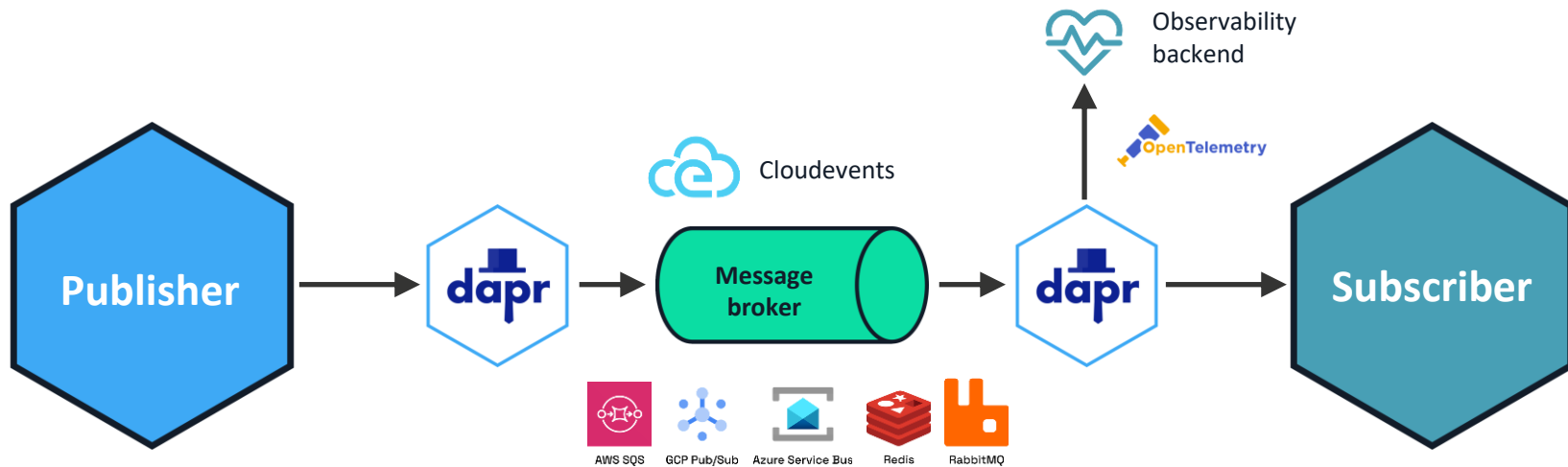


- Multiple application instances using a single consumer group
- Same app id = same consumer group
- Dapr delivers each message to only one instance of that application

Publish / Subscribe



Publish /
Subscribe



POST

<http://localhost:3500/v1.0/publish/mybroker/order-messages>

POST

<http://localhost:5100/orders>

Pub/Sub Brokers

Generic

Component	Status	Component version	Since runtime version
Apache Kafka	Stable	v1	1.5
In-memory	Stable	v1	1.7
JetStream	Beta	v1	1.10
KubeMQ	Beta	v1	1.10
MQTT3	Stable	v1	1.7
Pulsar	Stable	v1	1.10
RabbitMQ	Stable	v1	1.7
Redis Streams	Stable	v1	1.0
RocketMQ	Alpha	v1	1.8
Solace-AMQP	Beta	v1	1.10

Amazon Web Services (AWS)

Component	Status	Component version	Since runtime version
AWS SNS/SQS	Stable	v1	1.10

Google Cloud Platform (GCP)

Component	Status	Component version	Since runtime version
GCP Pub/Sub	Stable	v1	1.11

Microsoft Azure

Component	Status	Component version	Since runtime version
Azure Event Hubs	Stable	v1	1.8
Azure Service Bus Queues	Beta	v1	1.10
Azure Service Bus Topics	Stable	v1	1.0



DEMO

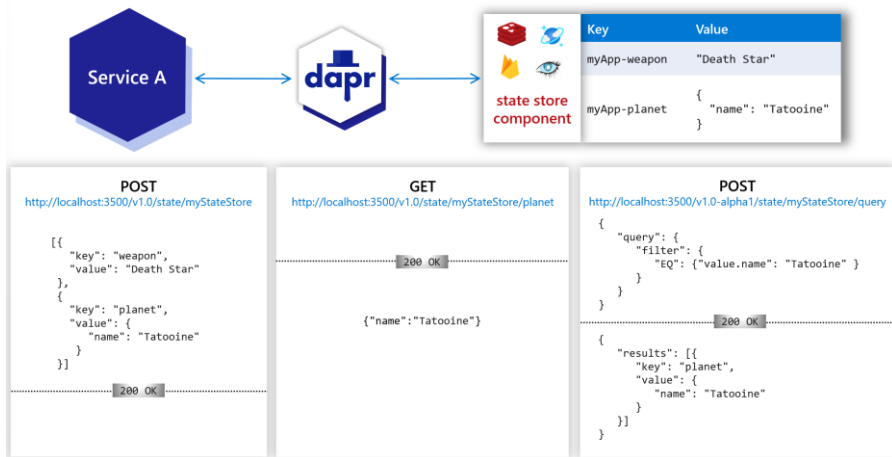
A background image showing a rowing team in a boat. The rowers are wearing blue and red uniforms and are captured in a synchronized rowing motion. The focus is on the oars and the rowers' hands, with the water visible in the background.

Dapr Building Blocks

State Management

4tecture[®]
empower your software solutions

State Management



- Configurable state store behavior (default eventually consistent, last-write-wins concurrency pattern)
- Optimistic concurrency with ETag
- Time to live (TTL)
- State encryption
- Querying state

State Stores

Generic

Component	CRUD	Transactional	ETag	TTL	Actors	Query	Status	Component version	Since runtime version
Aerospike	✓	□	✓	□	□	□	Alpha	v1	1.0
Apache Cassandra	✓	□	□	✓	□	□	Stable	v1	1.9
CockroachDB	✓	✓	✓	✓	✓	✓	Stable	v1	1.10
Couchbase	✓	□	✓	□	□	□	Alpha	v1	1.0
etcd	✓	✓	✓	✓	✓	□	Beta	v2	1.12
Hashicorp Consul	✓	□	□	□	□	□	Alpha	v1	1.0
Hazelcast	✓	□	□	□	□	□	Alpha	v1	1.0
In-memory	✓	✓	✓	✓	✓	□	Stable	v1	1.9
JetStream KV	✓	□	□	□	□	□	Alpha	v1	1.7
Memcached	✓	□	□	✓	□	□	Stable	v1	1.9
MongoDB	✓	✓	✓	✓	✓	✓	Stable	v1	1.0
MySQL & MariaDB	✓	✓	✓	✓	✓	□	Stable	v1	1.10
Oracle Database	✓	✓	✓	✓	✓	□	Beta	v1	1.7
PostgreSQL v1	✓	✓	✓	✓	✓	✓	Stable	v1	1.0
PostgreSQL v2	✓	✓	✓	✓	✓	□	Stable	v2	1.13
Redis	✓	✓	✓	✓	✓	✓	Stable	v1	1.0
RethinkDB	✓	□	□	□	□	□	Beta	v1	1.9
SQLite	✓	✓	✓	✓	✓	□	Stable	v1	1.11
Zookeeper	✓	□	✓	□	□	□	Alpha	v1	1.0

Amazon Web Services (AWS)

Component	CRUD	Transactional	ETag	TTL	Actors	Query	Status	Component version	Since runtime version
AWS DynamoDB	✓	✓	✓	✓	✓	□	Stable	v1	1.10

Cloudflare

Component	CRUD	Transactional	ETag	TTL	Actors	Query	Status	Component version	Since runtime version
Cloudflare Workers KV	✓	□	□	✓	□	□	Beta	v1	1.10

Google Cloud Platform (GCP)

Component	CRUD	Transactional	ETag	TTL	Actors	Query	Status	Component version	Since runtime version
GCP Firestore	✓	□	□	□	□	□	Stable	v1	1.11

Microsoft Azure

Component	CRUD	Transactional	ETag	TTL	Actors	Query	Status	Component version	Since runtime version
Azure Blob Storage	✓	□	✓	□	□	□	Stable	v2	1.13
Azure Cosmos DB	✓	✓	✓	✓	✓	✓	Stable	v1	1.0
Azure Table Storage	✓	□	✓	□	□	□	Stable	v1	1.9
Microsoft SQL Server	✓	✓	✓	✓	✓	□	Stable	v1	1.5

Oracle Cloud

Component	CRUD	Transactional	ETag	TTL	Actors	Query	Status	Component version	Since runtime version
Autonomous Database (ATP and ADW)	✓	✓	✓	✓	✓	□	Alpha	v1	1.7
Object Storage	✓	□	✓	✓	□	□	Alpha	v1	1.6



A person is seated at a desk in a dark room, illuminated by the blue light of two computer monitors. The person is viewed from the side, focused on the screens. The left monitor displays a web application with a sidebar and a main content area. The right monitor shows a code editor with syntax-highlighted code. On the desk, there is a red Coca-Cola can, a white mouse, and a pen holder with several pens. The overall atmosphere is one of quiet concentration and productivity.

DEMO

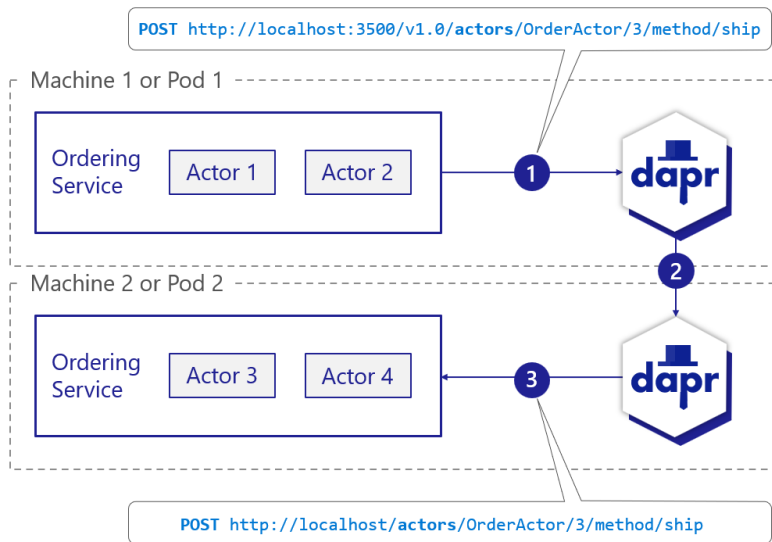
A close-up, low-angle shot of rowers in a boat, showing their legs and hands gripping the oars. The rowers are wearing blue and red uniforms. The oars are black with yellow and blue handles. The background is a bright, overexposed sky and water.

Dapr Building Blocks

Virtual Actors

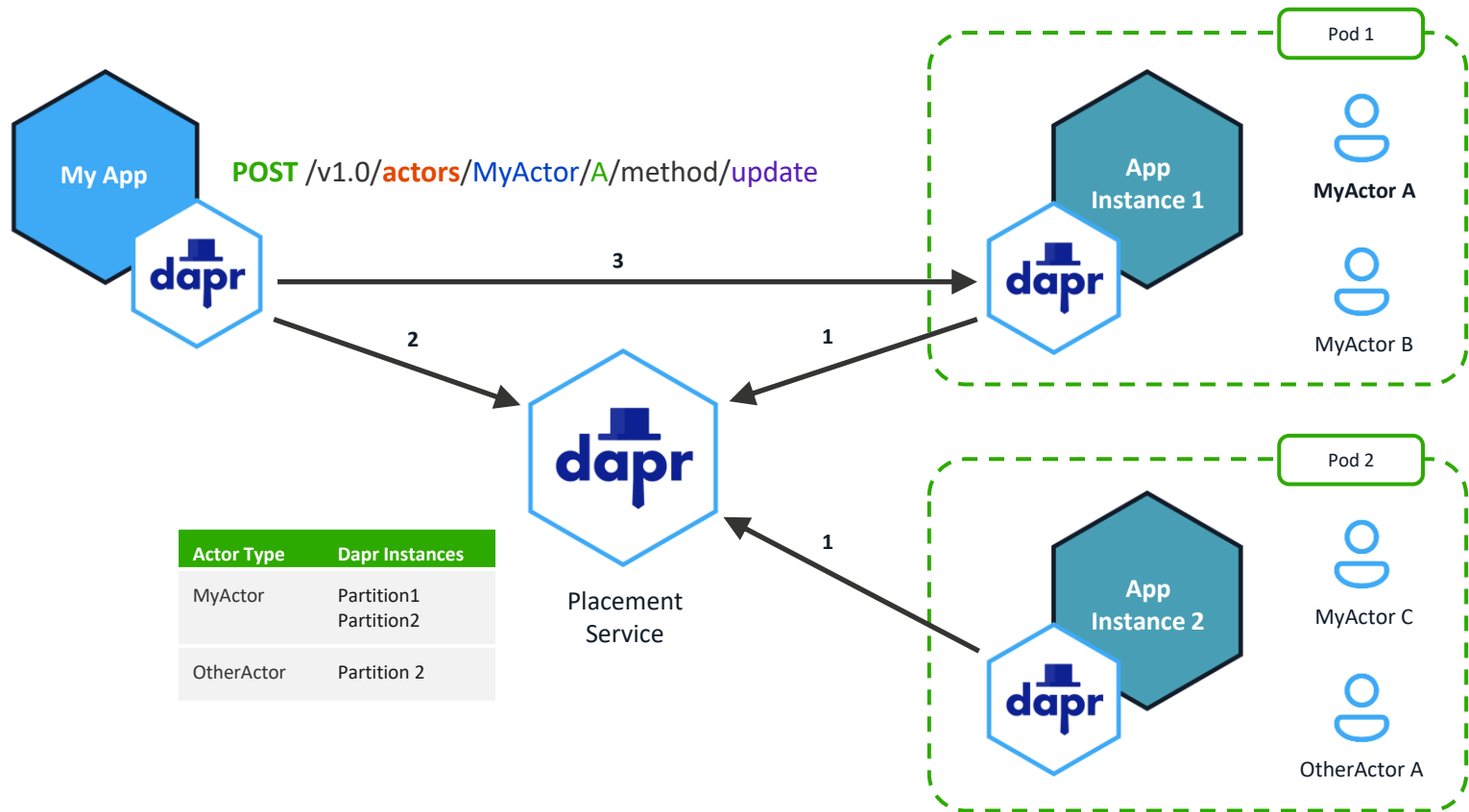
4tecture[®]
empower your software solutions

Actors



- Virtual Actor pattern
- Stateful, long running objects with identity
- Encapsulate state and behavior within a distributed system
- Actor state store
- Actor timers and reminders

Actor placement



A person is seen from the side, sitting at a desk in a dark room, illuminated by the blue light of two computer monitors. The person is typing on a keyboard. The left monitor displays a web application with a sidebar and a main content area. The right monitor displays a code editor with syntax-highlighted code. A red Coca-Cola can is on the desk between the monitors. A mouse and a pen holder are also visible on the desk. The word "DEMO" is overlaid in large white letters on the left side of the image.

DEMO

A background image showing a rowing team in a boat. The rowers are wearing blue and red uniforms and are pulling oars with yellow handles. The boat is on water, and the background is slightly blurred.

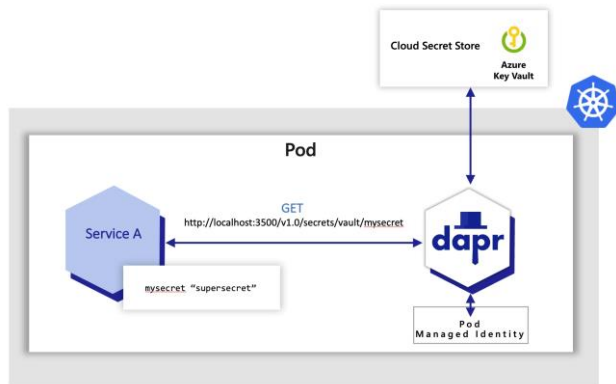
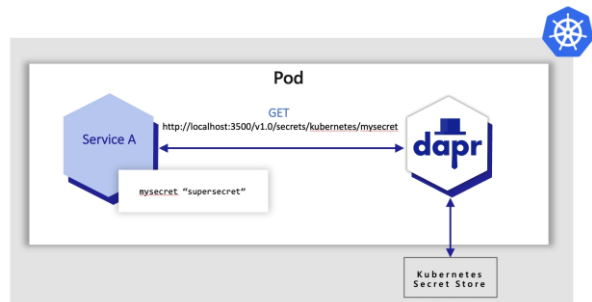
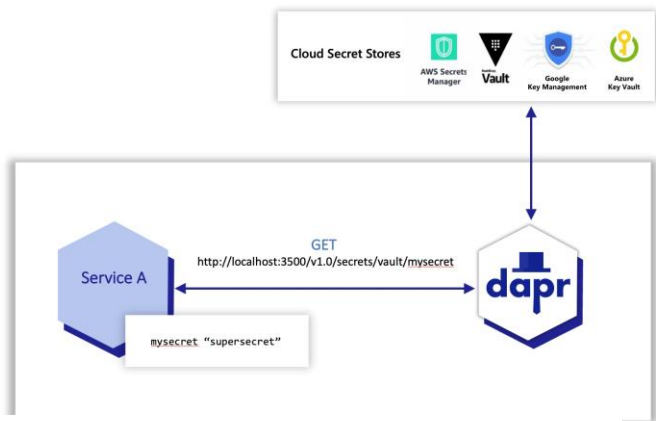
Dapr Building Blocks

Secret Management

4tecture[®]
empower your software solutions

Secret Management

- Access secret stores through generic Dapr API
- Secret scoping (limit access)



A person is seen from the side, sitting at a desk in a dark room. They are looking at two computer monitors. The left monitor displays a web application with a sidebar and a main content area. The right monitor displays a code editor with syntax-highlighted code. A red Coca-Cola can is on the desk between the monitors. The person's hands are on a keyboard. The word "DEMO" is overlaid in large white letters on the left side of the image.

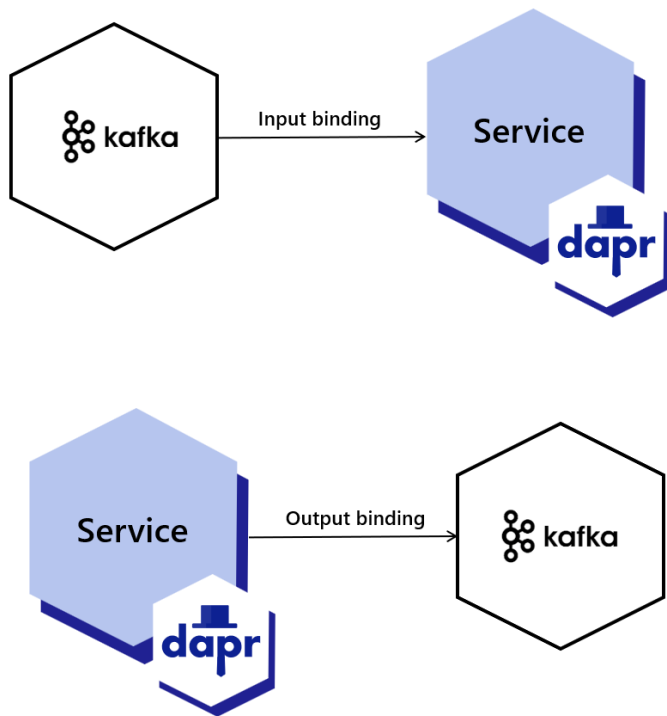
DEMO

A background image showing a rowing team in a boat. The rowers are wearing blue and red uniforms. The focus is on the oars and the rowing mechanism, which includes yellow and blue components. The text is overlaid on this image.

Dapr Building Blocks

Input/Output Bindings

Input/Output Binding



- Trigger your app with events coming from external systems
- Handle retries and failure recovery
- Portable app with environment-specific bindings

Bindings

Generic

Component	Input Binding	Output Binding	Status	Component version	Since runtime version
Apple Push Notifications (APN)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Alpha	v1	1.0
commercetools GraphQL	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Alpha	v1	1.8
Cron (Scheduler)	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Stable	v1	1.10
GraphQL	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Alpha	v1	1.0
HTTP	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Stable	v1	1.0
Huawei OBS	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Alpha	v1	1.8
InfluxDB	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Beta	v1	1.7
Kafka	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Stable	v1	1.8
Kitex	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Alpha	v1	1.11
KubeMQ	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Beta	v1	1.10
Kubernetes Events	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Alpha	v1	1.0
Local Storage	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Stable	v1	1.9
MQTT3	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Beta	v1	1.7
MySQL & MariaDB	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Alpha	v1	1.0
PostgreSQL	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Stable	v1	1.9
Postmark	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Alpha	v1	1.0
RabbitMQ	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Stable	v1	1.9
Redis	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Stable	v1	1.9
RethinkDB	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Beta	v1	1.9
SendGrid	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Alpha	v1	1.0
SMTP	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Alpha	v1	1.0
Twilio	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Alpha	v1	1.0
Wasm	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Alpha	v1	1.11

Microsoft Azure

Component	Input Binding	Output Binding	Status	Component version	Since runtime version
Azure Blob Storage	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Stable	v1	1.0
Azure Cosmos DB (Gremlin API)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Alpha	v1	1.5
Azure CosmosDB	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Stable	v1	1.7
Azure Event Grid	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Beta	v1	1.7
Azure Event Hubs	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Stable	v1	1.8
Azure OpenAI	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Alpha	v1	1.11
Azure Service Bus Queues	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Stable	v1	1.7
Azure SignalR	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Alpha	v1	1.0
Azure Storage Queues	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Stable	v1	1.0

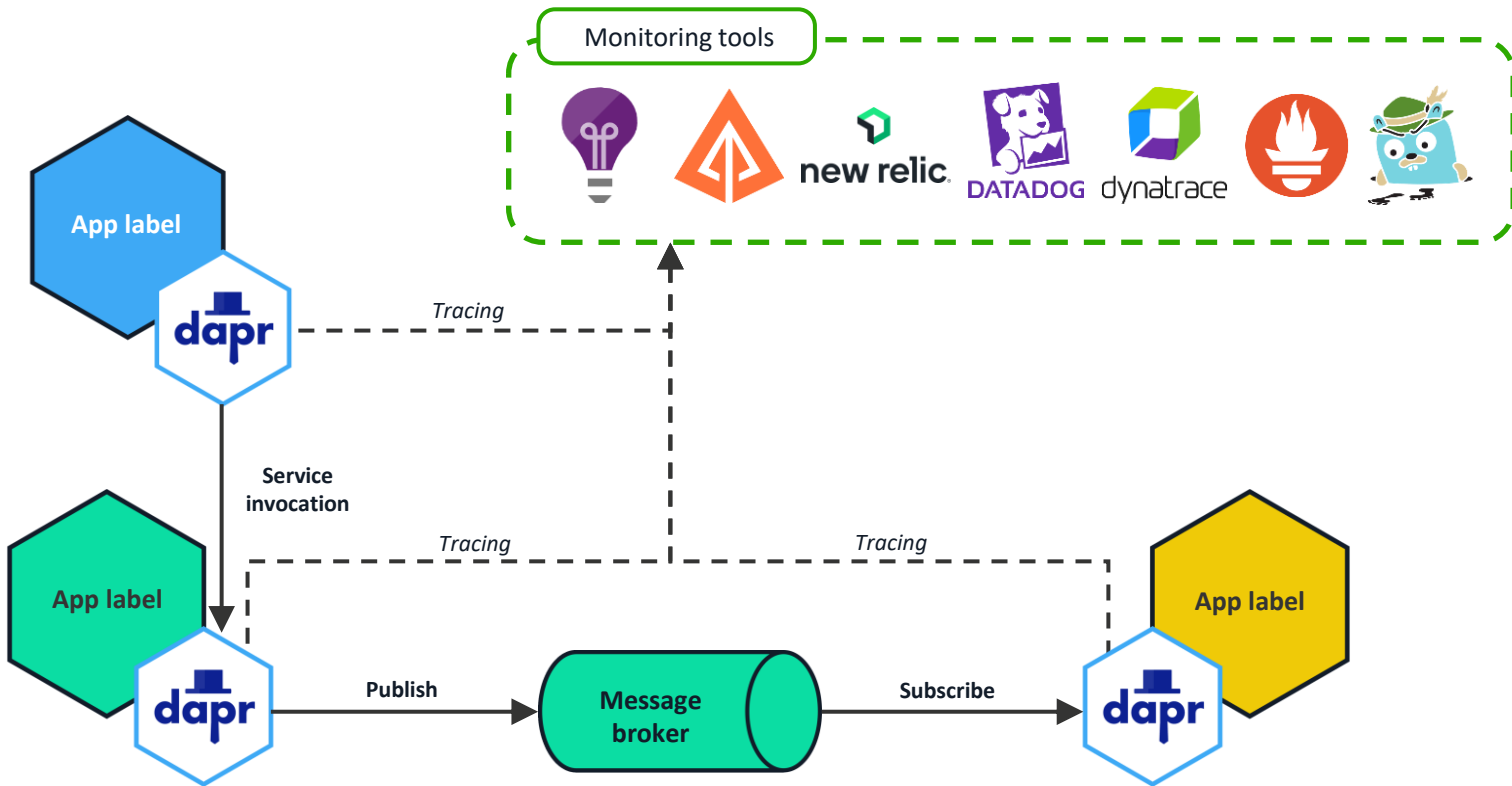
A close-up, low-angle shot of rowers in a boat, focusing on their hands and the oars. The rowers are wearing blue and red uniforms. The oars are black with yellow and blue handles. The background is a bright, slightly blurred body of water.

Dapr Building Blocks

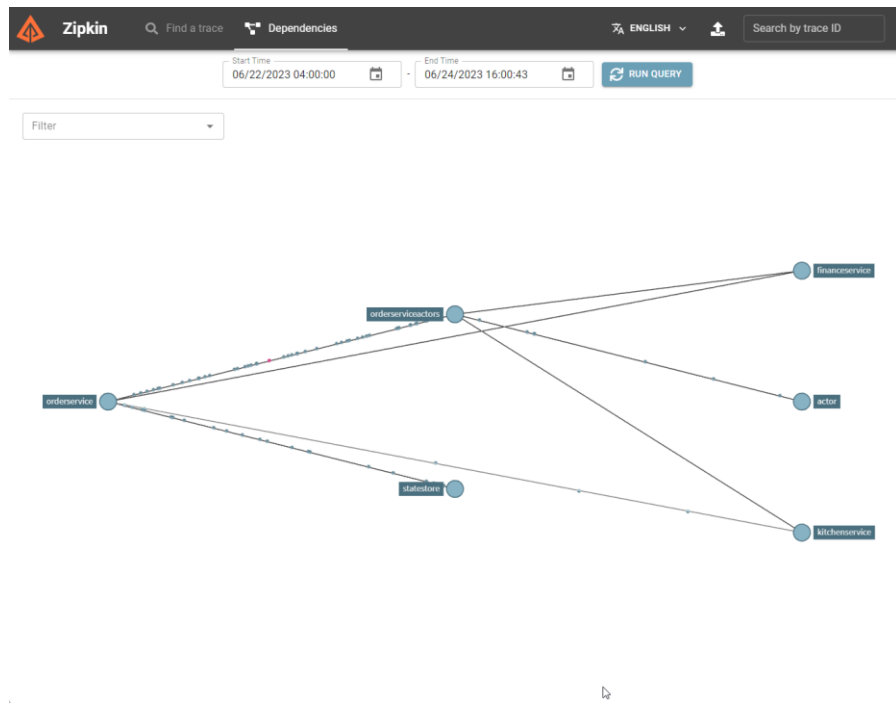
Observability

4tecture[®]
empower your software solutions

Distributed tracing



Observability



- Distributed tracing
- Open Telemetry (OTEL) and Zipkin protocols
- Used with service invocation and pub/sub APIs
- Sidecar health
- App health checks
 - Unsubscribing Pub/Sub
 - Stop input bindings
 - Short-circuiting all service-invocation requests

A close-up, low-angle shot of several rowers in a boat, wearing blue and red uniforms, pulling their oars. The oars are black with yellow and blue handles. The background is a bright, slightly blurred body of water.

Dapr Building Blocks

Resiliency

4tecture[®]
empower your software solutions

Resiliency

Resiliency patterns can be applied across Dapr APIs:

- Retries
- Timeouts
- Circuit breakers

Declarative and decoupled from application code.

Available across all component types, service invocation, and actors.

```
apiVersion: dapr.io/v1alpha1
```

```
kind: Resiliency
```

```
metadata:
```

```
  name: myresiliency
```

```
scopes:
```

```
  - order-processor
```

```
spec:
```

```
  policies:
```

```
    retries:
```

```
      retryForever:
```

```
        policy: constant
```

```
        duration: 5s
```

```
        maxRetries: -1
```

```
  circuitBreakers:
```

```
    simpleCB:
```

```
      maxRequests: 1
```

```
      timeout: 5s
```

```
      trip: consecutiveFailures >= 5
```

```
targets:
```

```
  components:
```

```
    statestore:
```

```
      outbound:
```

```
        retry: retryForever
```

```
        circuitBreaker: simpleCB
```


A person is seen from the side, sitting at a desk in a dark room, illuminated by the blue light of two computer monitors. The person is typing on a keyboard. The left monitor displays a web application with a sidebar and a main content area. The right monitor displays a code editor with syntax-highlighted code. A red Coca-Cola can is on the desk between the monitors. A mouse is visible on the right side of the desk. A pen holder with several pens is on the far right. The word "DEMO" is overlaid in large white letters on the left side of the image.

DEMO

A background image showing a rowing team in a boat. The rowers are wearing blue and red uniforms and are captured in a synchronized rowing motion. The focus is on the oars and the rowers' hands, with the water visible in the background.

.NET Microservices with Dapr

Conclusion

Pros / Cons

Advantages

- Develop faster
- Best practices
- Portability
- Focus on your logic

Disadvantages

- Additional hop / network overhead
- Common API – less features

Conclusion

- Suitable for most teams and applications
- Base your development on proven best practices
- Ideal, if portability is key (different environments / clouds, local, etc.)

A close-up, low-angle shot of rowers in a boat, showing their hands on the oars and the oarlocks. The rowers are wearing blue and red uniforms. The background is a bright, slightly blurred body of water.

.NET Microservices with Dapr

Q & A

4tecture[®]
empower your software solutions

Thank you for your attention!

If you have any questions do not hesitate to contact us:

4tecture GmbH
Industriestrasse 25
CH-8604 Volketswil
www.4tecture.ch

Marc Müller
Principal Consultant
www.powerofdevops.com



A background image showing several hands of different skin tones reaching towards the center, where they are assembling four interlocking wooden puzzle pieces. The pieces are colored light brown, white, red, and green. The overall scene is softly blurred, focusing attention on the hands and the puzzle.

4tecture[©]

empower your software solutions