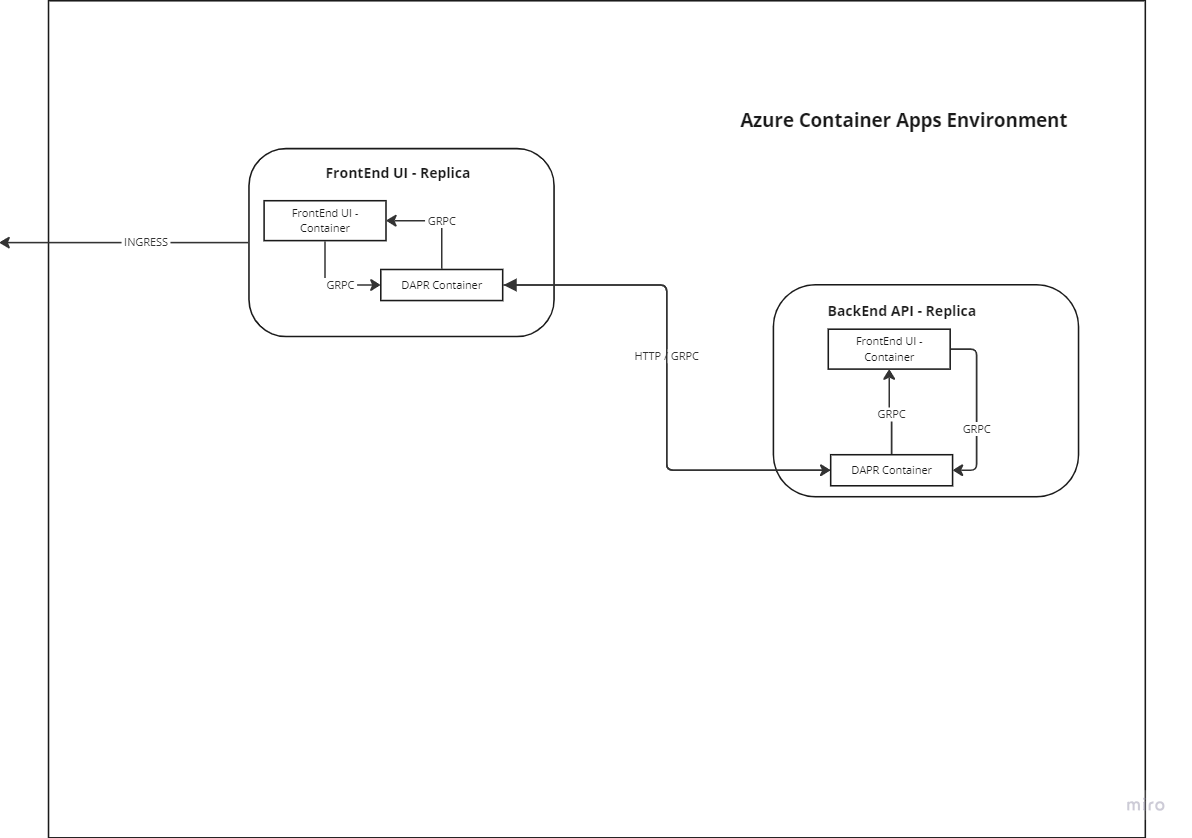
1. Search task
2. Create task
3. Edit task
4. Complete task
5. External task

****

# ACA

## Revisions

Revisions are a way to deploy multiple versions of an app where you have the option to send the traffic to a certain revision.

Can have up to 100 revisions.

2 types

Single revision mode

Multiple Mode - good for A/B testing . Canary deployment

## Pod

Composed of the application container or any required side car container

# Microservices – Challenges

But more often than not we face the same challenges:

* Recovering state after failures
* Services discovery and calling other microservices.
* Integration with external resources
* Asynchronous communications between different services
* Distributed tracing
* Measuring message calls and performance across components and networked services

# DAPR

Distributed application Runtime.

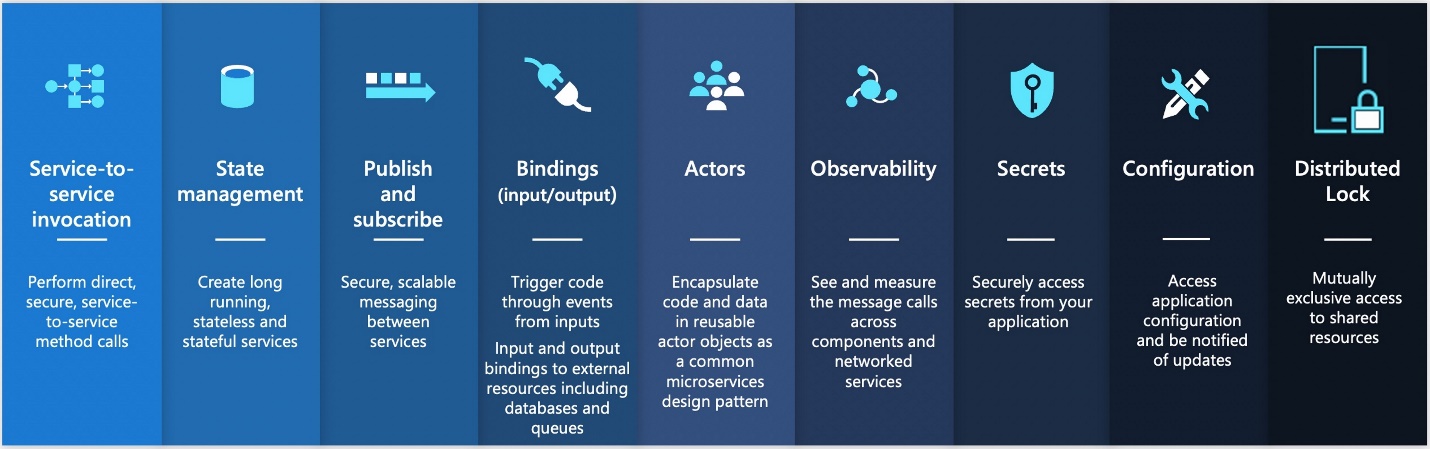
Offers a solution for the common challenges that are faced in any distributed microservice application.

Can be used with any language (Go, .NET python, Node, Java, C++) and run anywhere.

(On-premise, Kubernetes, and any public cloud (e.g. Azure)

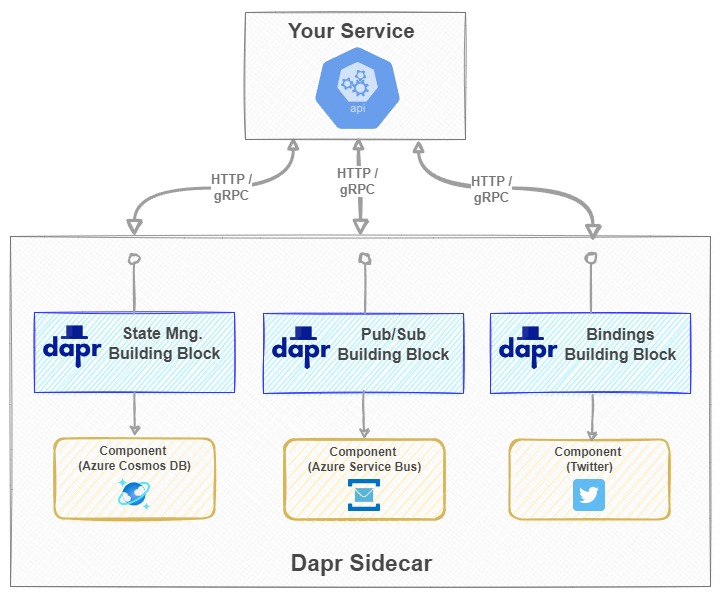
Dapr’s core component is its building blocks . So far it supports 9 building blocks

Simply put, a Building Block is a modular component which encapsulates best practices and can be accessed over standard HTTP or gRPC APIs.



## Dapr & Microservices

Dapr exposes its Building Blocks and components through a **sidecar architecture**. A sidecar enables Dapr to run in a separate memory process or separate container alongside your service. Sidecars provide isolation and encapsulation as they aren't part of the service but connected to it.



While Azure Container Apps features the building blocks for running microservices, using the Distributed Application Runtime (Dapr) provides an even richer microservices programming model.

<https://github.com/dotnet-architecture/eBooks/blob/1ed30275281b9060964fcb2a4c363fe7797fe3f3/current/dapr-for-net-developers/Dapr-for-NET-Developers.pdf>

Because the calls will flow through container sidecars, Dapr can inject some useful cross-cutting behaviors that are meaningfully abstracted from our application containers.

Some features include

* Automatically retry calls upon failure.
* Make calls between services secured with mutual authentication (mTLS), including automatic certificate rollover.
* Control what operations clients can perform using access control policies.
* Capture traces and metrics for all calls between services to provide insights and diagnostics.

# Demo

## Application Overview

1. Frontend  web app that accepts requests from public users to manage their tasks. It invokes the component " -Backend" endpoints via HTTP or gRPC.
2. Backend is a backend Web API which contains the business logic of tasks management service, data storage, and publishing messages to Azure Service Bus Topic.
3. Processor-Backend is an event-driven backend processor which is responsible for sending emails to task owners based on messages coming from Azure Service Bus Topic. Here there is a continuously running background processor, which is based on Dapr Cron timer configuration, to flag overdue tasks.