IBM

Open by design™

Using Serverless Platform to Create Microservices

Ying Chun Guo(Daisy)
guoyingc@cn.ibm.com

Self Introduction

- Ying Chun Guo “Daisy”
  - IBM senior software engineer
  - 9 years experience in open source communities, i.e. OpenOffice and OpenStack
  - Initiator of OpenStack I18n team, the first PTL
  - Apache OpenWhisk committer

Contact with me with WeChat

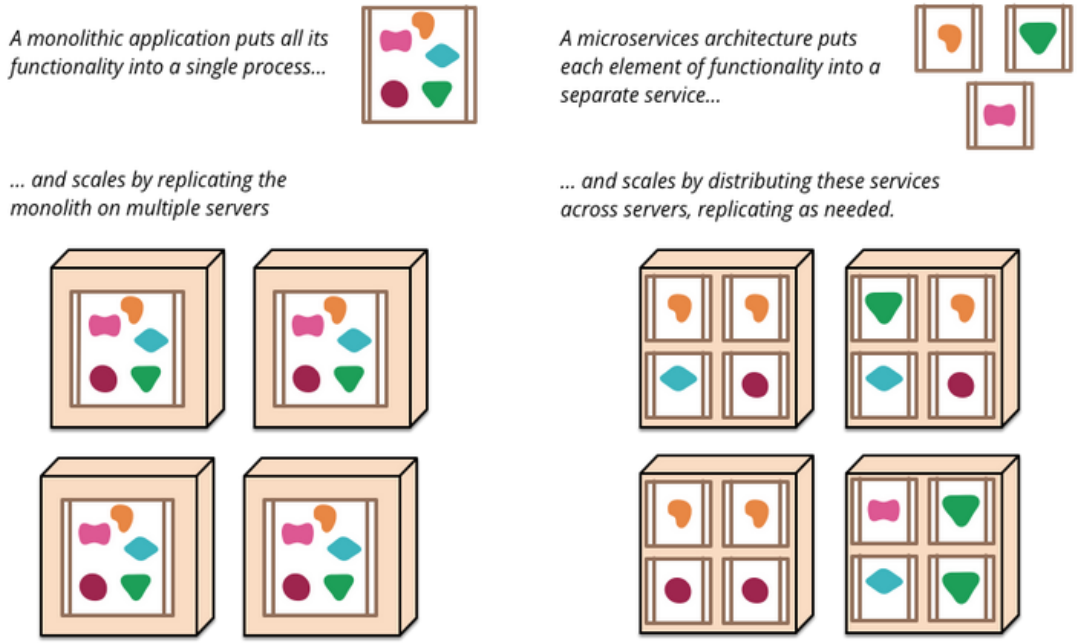
Agenda

- What is “Microservice Architecture”?
- Serverless VS. Microservice
- How Apache OpenWhisk support Microservices?
- Summary

What is “Microservice Architecture” ?

What is “Microservice Architecture”?

- An approach to developing a suite of small services each running in its own process and communicating with lightweight mechanisms, often an HTTP resource API.
- These services are built around business capabilities and independently deployable by fully automated deployment machinery.
- There is a bare minimum of centralized management of these services, which may be written in different programming languages and use different data storage technologies.



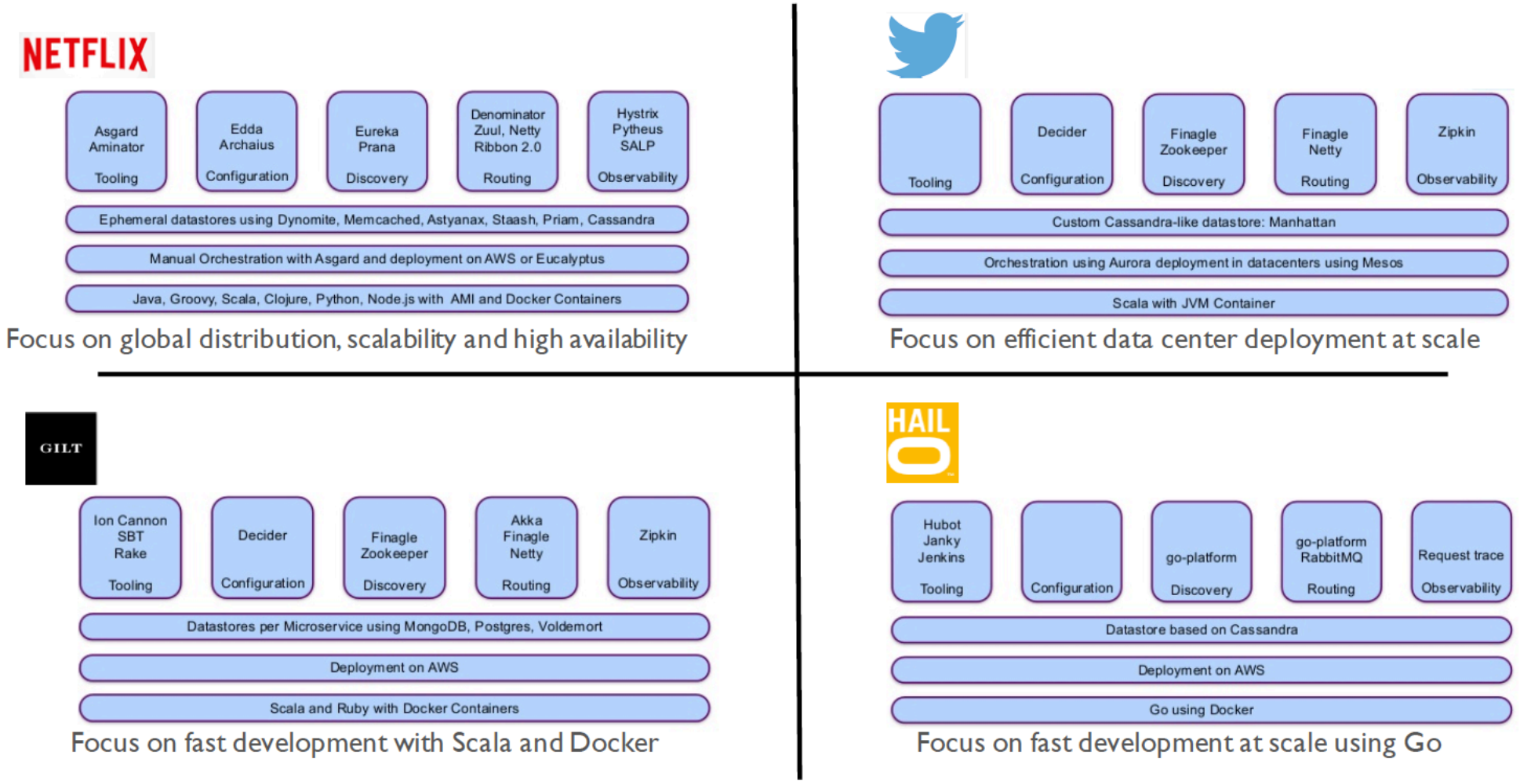
https://www.martinfowler.com/articles/microservices.html

Characteristics of “Microservices Architecture”

All “true” microservices architectures share the same common characteristics in that they –

- Are componentized
- Are organized around business capabilities
- Have decentralized governance
- Have decentralized data management
- Communicate via APIs only
- Are designed for failure
- Enable continuous delivery
- Embrace evolutionary design
- Are owned by cross-functional teams embracing DevOps practices

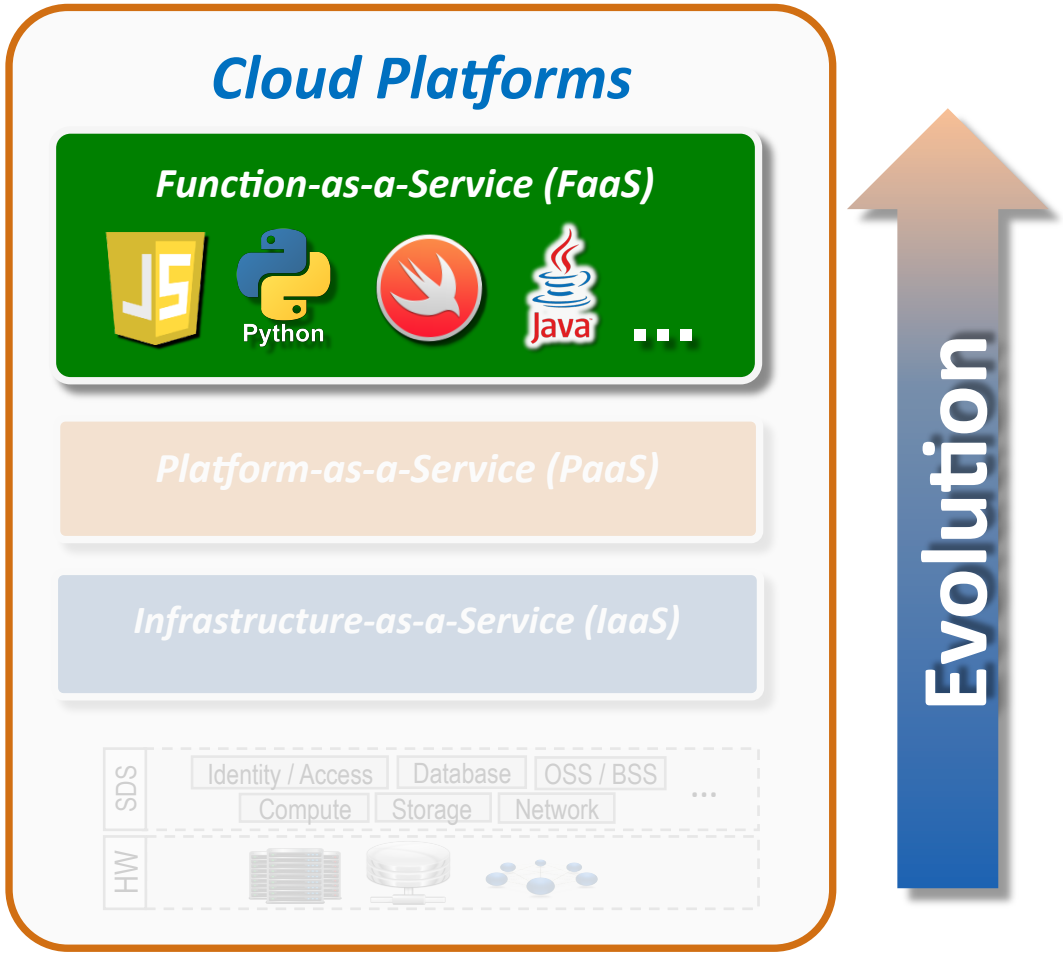
Sample of Microservices Architecture



Adrian Cockcroft, "State of the Art in Microservices," December 4, 2014

Serverless VS. Microservices

Review: What is Serverless ?



Evolution

Serverless = Functions

- **Workloads**: Simple, single-tasked Functions
- No "Back-end" Servers Configuration
  - ***Automatic scaling,*** *based upon load*
  - ***Driven by events,*** *and their data*
- Majority of Functions & Orchestration are "Front-end"
  - around workflows and tasks around the applications data … where
  - ***Developers Focus on ONLY writing Application and Business logic!***

There are still Servers!

- ***But they are a 'No-Op' for you!***
  - *Provider's DevOps teams configure, manage and assure Functions scale and run efficiently.*

No Configuration *of Servers, Only Pay for Compute time functions actually use*

APACHE OpenWhisk™

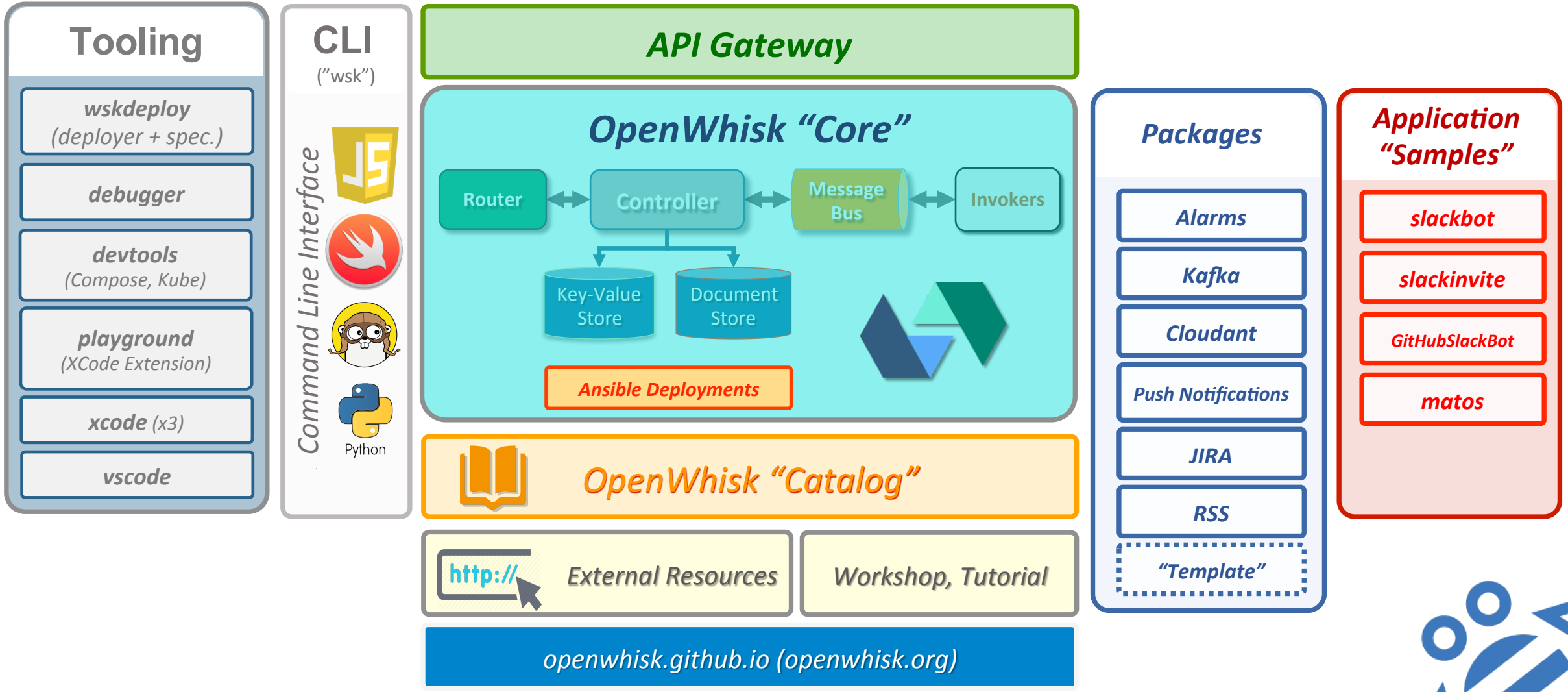
Review: Characteristics of Serverless Computing

Serverless can also mean applications where some amount of server-side logic is still written by the application developer but unlike traditional architectures is run in stateless compute containers that are event-triggered, ephemeral (may only last for one invocation), and fully managed by a 3rd party.

—— Mike Roberts

- Running back end code without managing your own server systems or your own server applications.
- Not require coding to a specific language, framework or library.
- Functions are running in a separate context of environment.
- Horizontal scaling is completely automatic, elastic, and managed by the provider.
- Functions can be triggered by event types defined by the provider.
- Functions can be also triggered as a response to inbound http requests, typically in some kind of API gateway.

Review: Apache OpenWhisk "Eco-System"



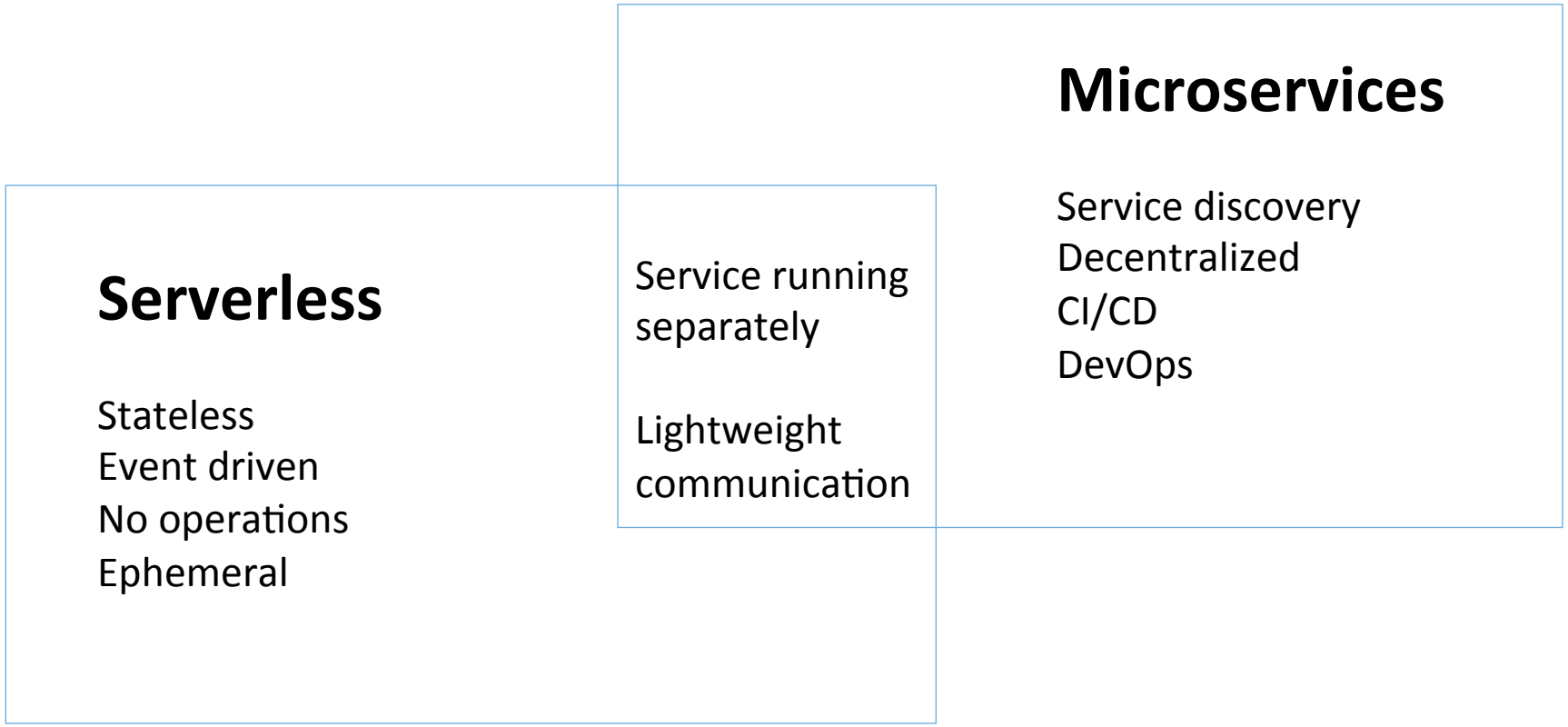
APACHE OpenWhisk™

How “Serverless Computing” fit into “Microservices Architecture”?

Characteristics of Microservice Architecture:

- ✔ Are componentized
- ➢ Are organized around business capabilities
- ✔ Have decentralized governance
- ➢ Have decentralized data management
- ✔ Communicate via APIs only
- ➢ Are designed for failure
- ➢ Enable continuous delivery
- ✔ Embrace evolutionary design
- ➢ Are owned by cross-functional teams embracing DevOps practices

Serverless VS. Microservices



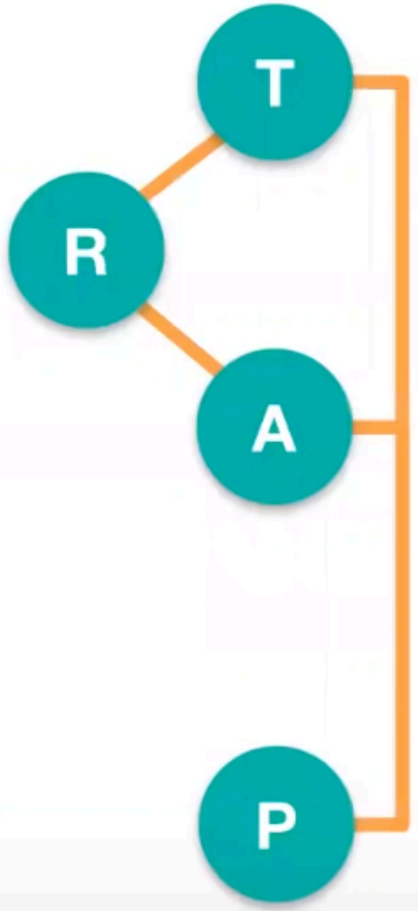
How Apache OpenWhisk supports microservices ?

Review： OpenWhisk Programming Model

Data sources define events they emit as **Triggers**.

Developers map **Actions** to **Triggers** via **Rules**.

Packages provide integration with external services.

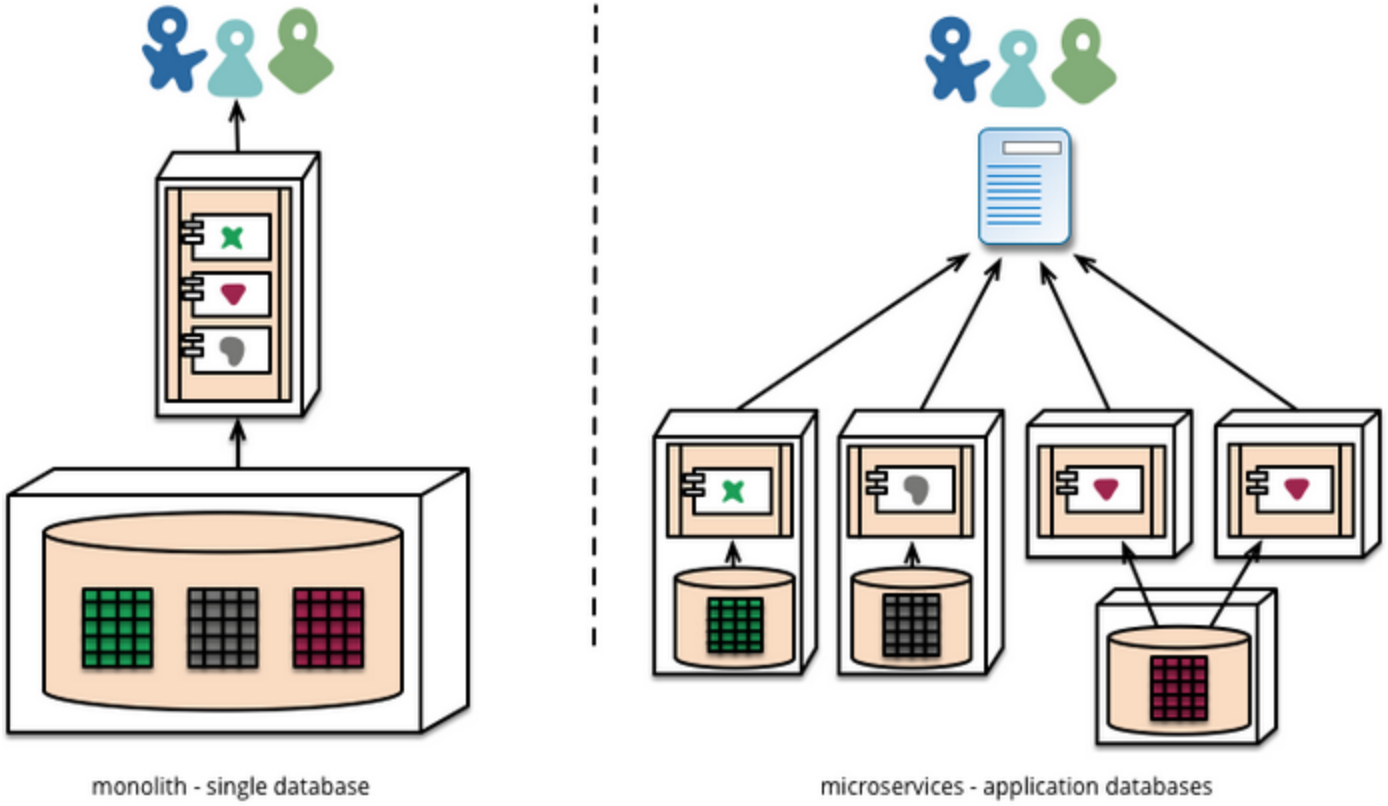


An action as a microservice

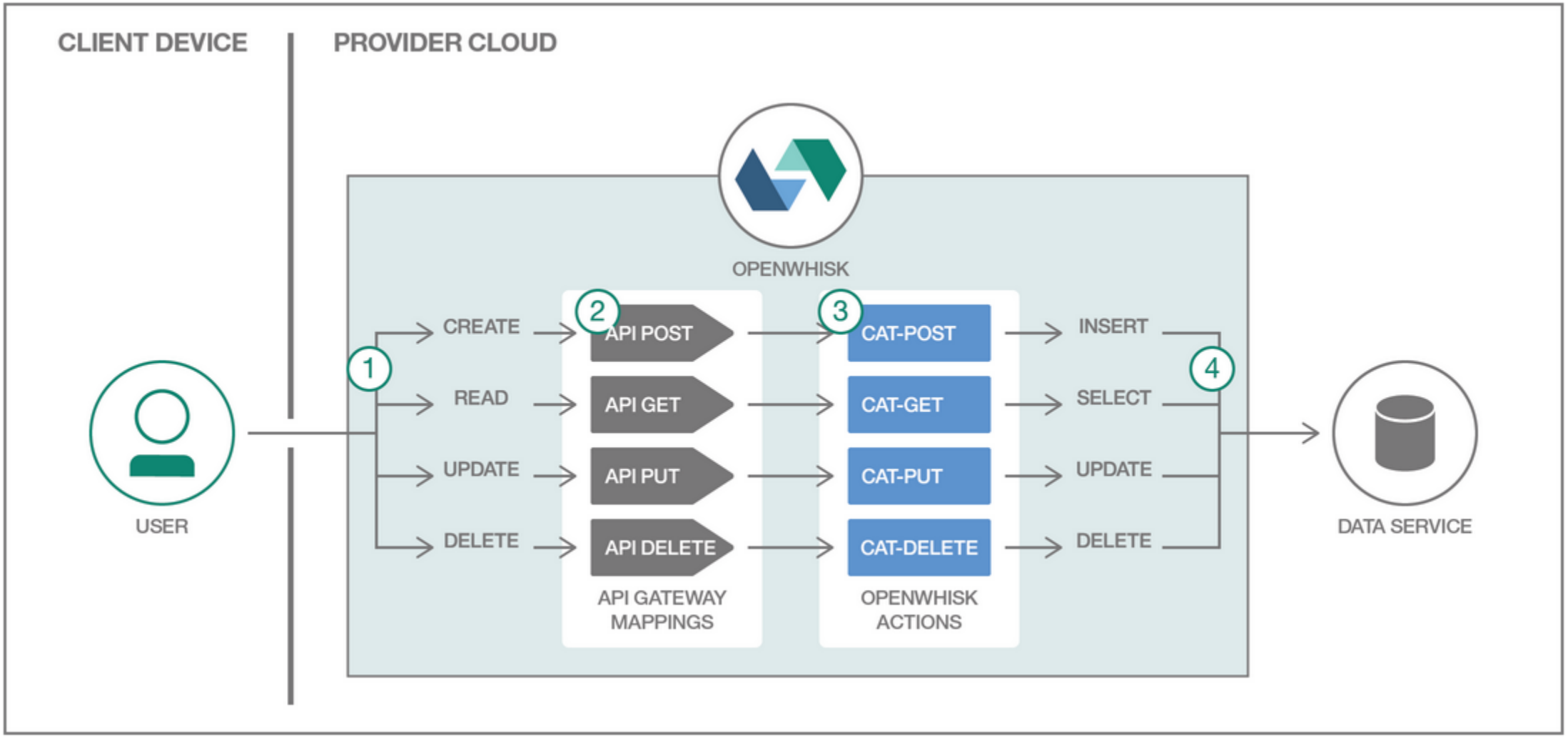
- OpenWhisk actions are independent of each other
- OpenWhisk actions can be implemented using variety of different languages supported by OpenWhisk and access various backend systems.
- Each action can be independently deployed and managed
- Each action is scaled independently of other actions.
- Interconnectivity between actions is provided by OpenWhisk in the form of rules, sequences, naming conventions, and even HTTP REST API.

Sample 1: Database CRUD services

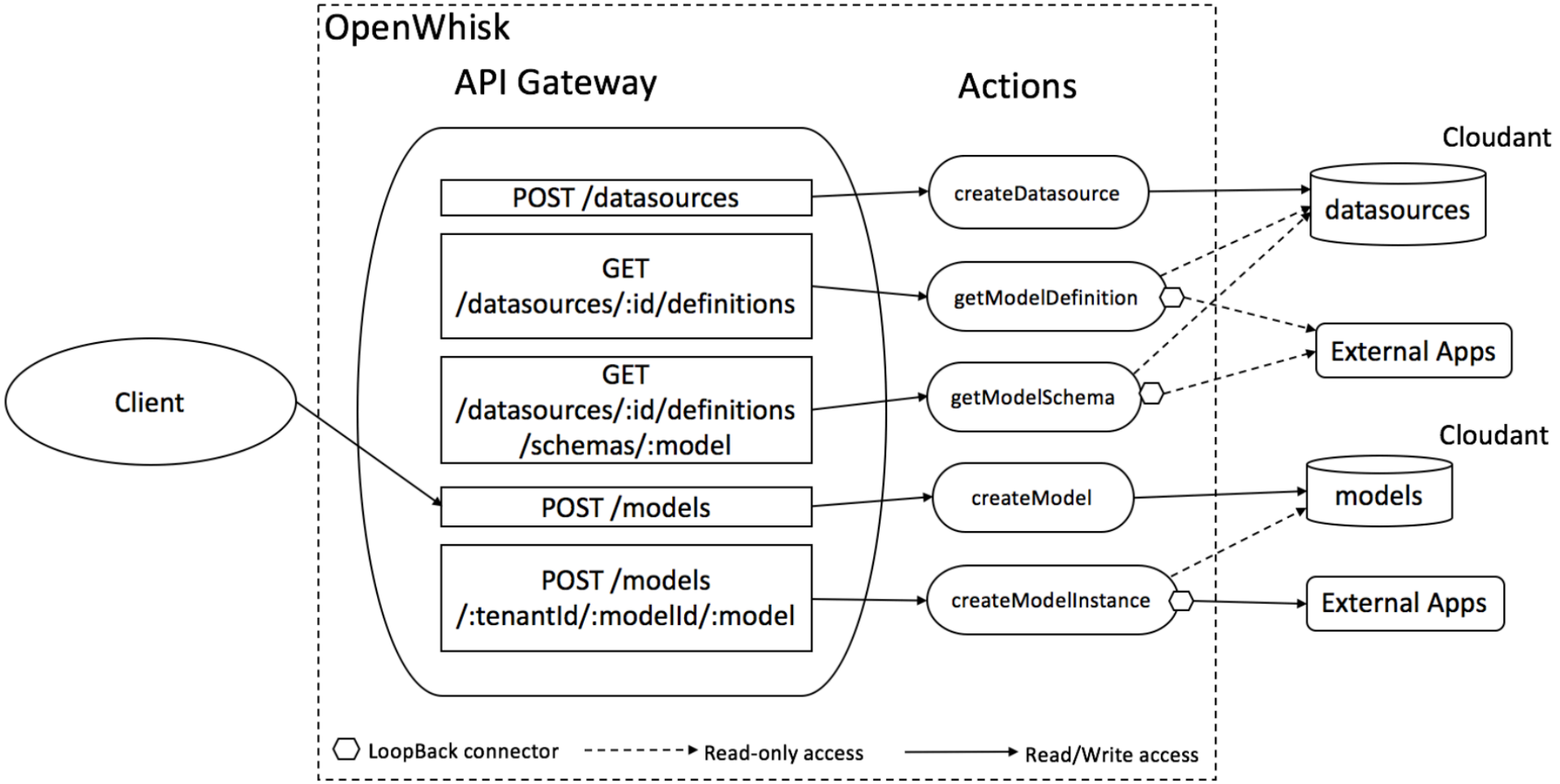
Using Action and API Gateway to implement database CRUD services.



Sample 1: Database CRUD services



Sample 1: Database CRUD services

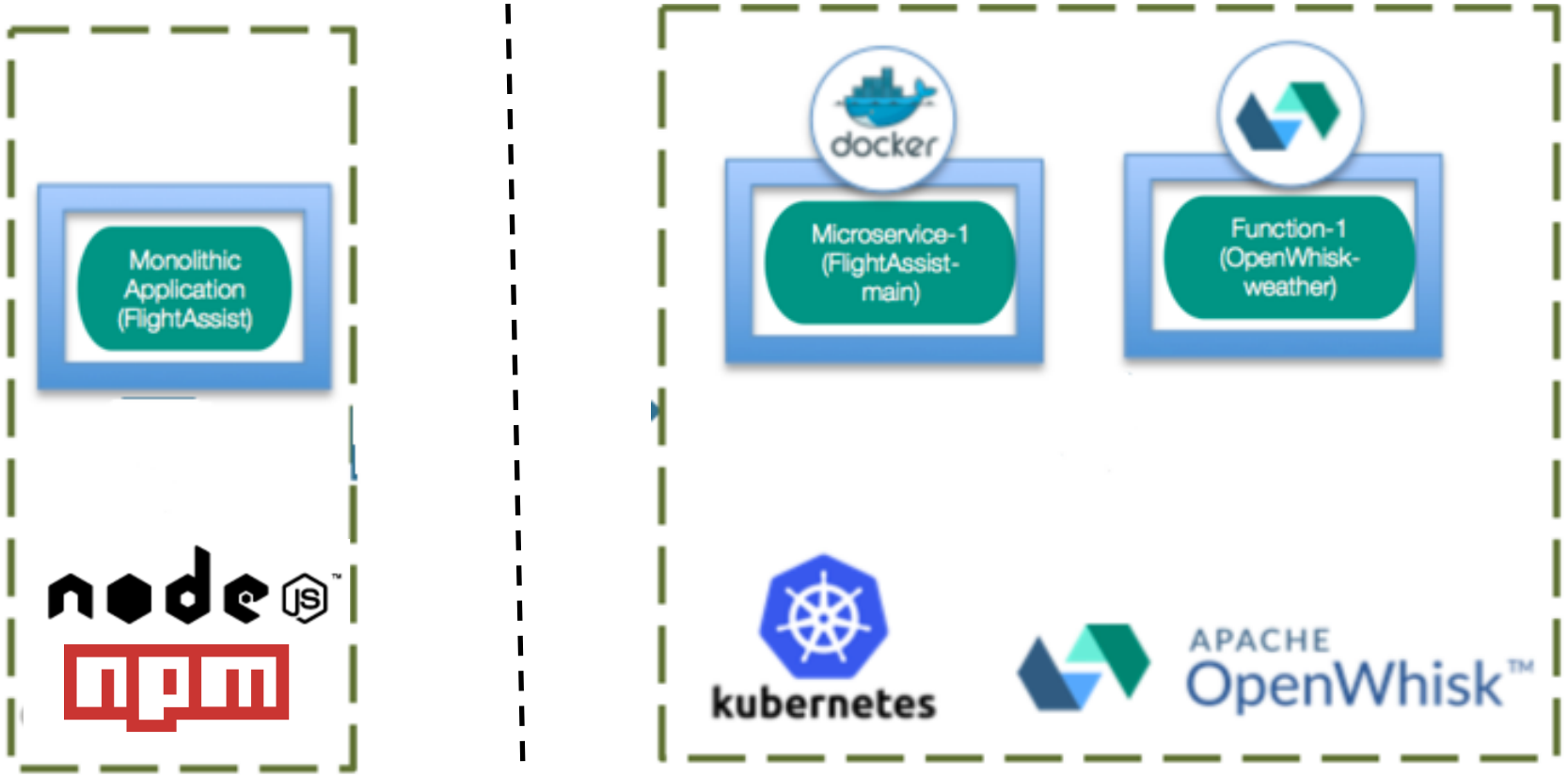


https://strongloop.com/strongblog/loopback-as-a-service-using-openwhisk/

Sample 2: A sample Node.js monolithic application: Flightasistant

Available 3rd services

Sample 2: A sample Node.js monolithic application: Flightasistant



https://github.com/IBM/Microservices-deployment-with-PaaS-Containers-and-Serverless-Platforms

Summary

- Apache OpenWhisk can be used to develop microservices.
- Apache OpenWhisk is not a full stack ‘Microservices Architecture’ platform, but can be part of it.

Reference

1. Serverless APIs with OpenWhisk

https://github.com/IBM/openwhisk-serverless-apis#2-create-openwhisk-actions-and-mappings

2. Navigate application deployment options with Cloud Foundry, Kubernetes, OpenWhisk and Istio

https://github.com/IBM/Microservices-deployment-with-PaaS-Containers-and-Serverless-Platforms

3. Microservices

https://www.martinfowler.com/articles/microservices.html

4. Serverless Architectures

https://www.martinfowler.com/articles/serverless.html

添加 IBMOpenTech
请求入群
与讲师同学互动

扫码填写
本课程调查问卷

下讲预告
6月15日晚8点
《使用Serverless平台实现IoT》

如需体验Apache OpenWhisk
请到bluemix.net注册并体验
任何问题，请微信咨询
IBMOpenTech