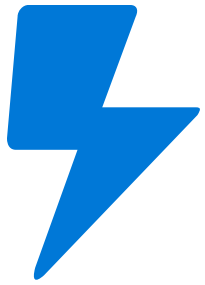
Introduction to Azure Functions

Seoul OSS Developer Forum – January 20, 2020

Anthony Chu
Program Manager – Azure Functions

What is Serverless Computing?

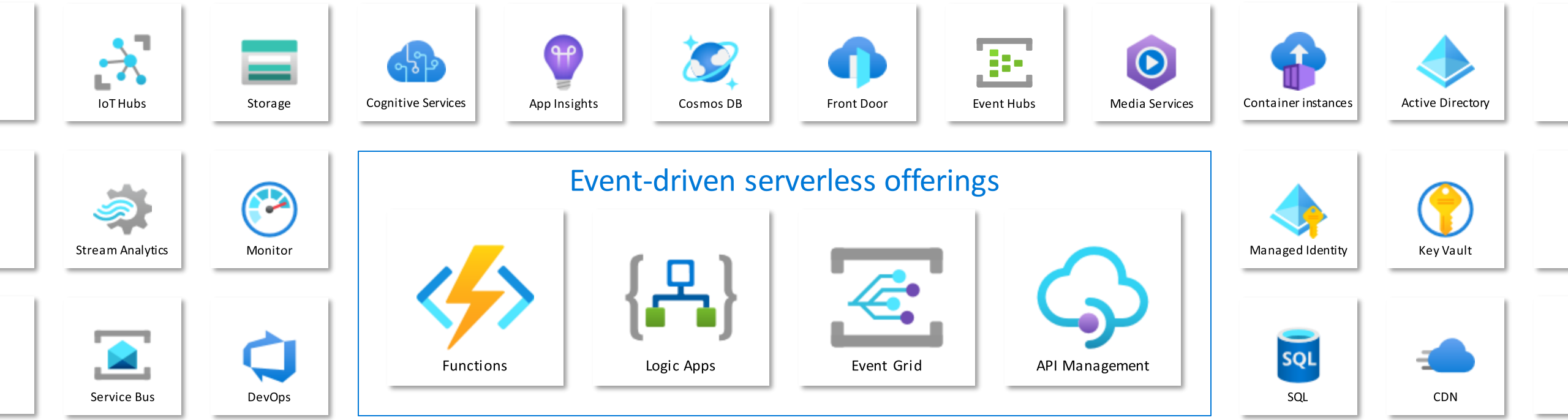
Full abstraction of servers



Instant, event-driven scalability

Pay per use

Azure serverless ecosystem



- IDE integration
- Local development
- Flexible deployment options
- Built-in security
- Rich monitoring
- Compliance and management

Azure Functions

Events

Trigger from HTTP, Timers, Azure Services (and more)

Code

C#, JavaScript/TypeScript, Java, Python, PowerShell (and more)

Inputs / Outputs

Retrieve and send data to many Azure and other services

Azure Functions is open source

github.com/azure/azurefunctions

- Azure Functions Host
- Azure Functions Core Tools (CLI)
- Language workers (Node, Python, Java, etc)
- Extensions (triggers, input and output bindings)
- VS Code extension
- … and more

Functions everywhere

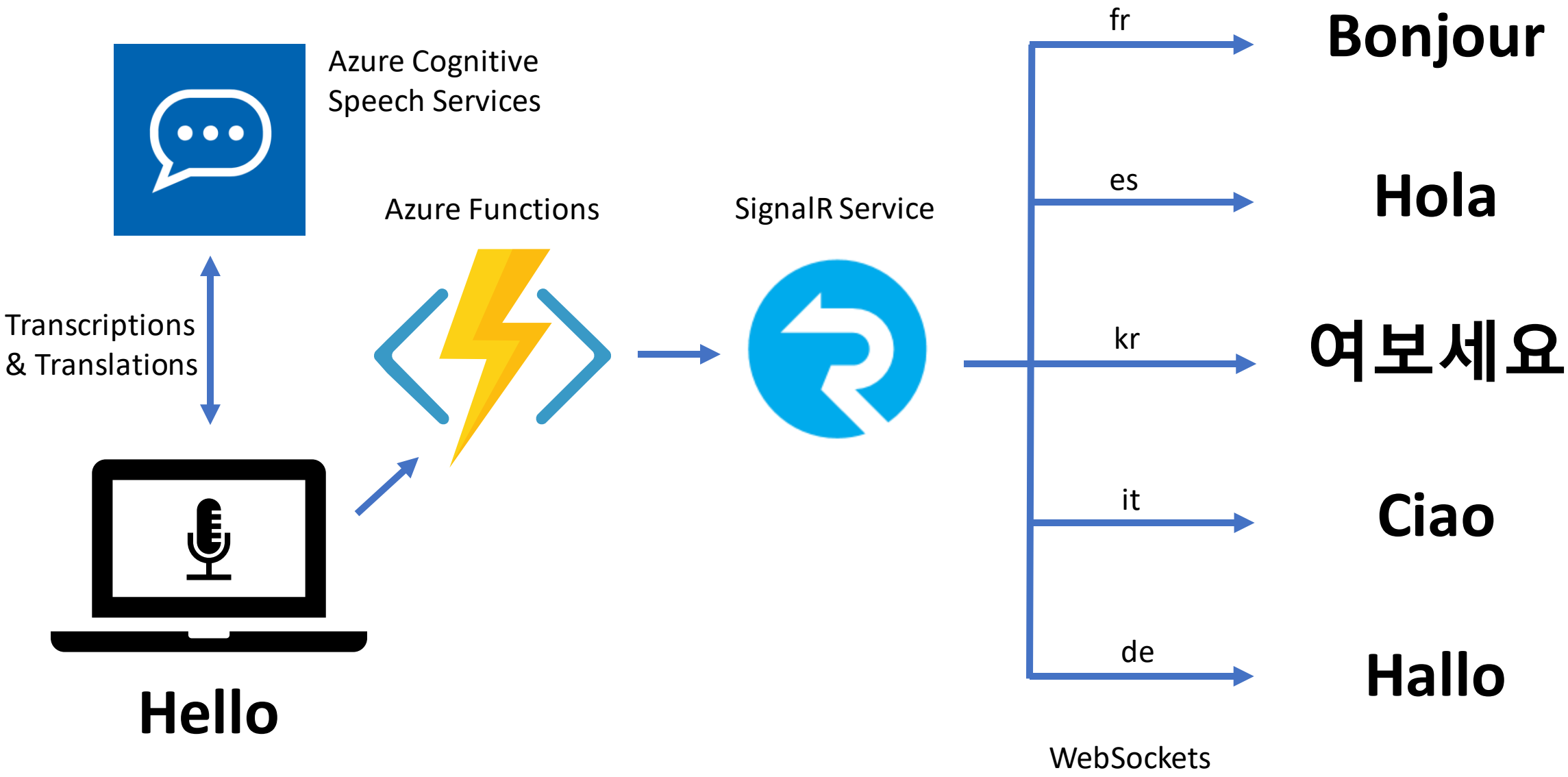
https://github.com/azure/azure-functions

Azure Functions host runtime | Azure Functions Core Tools | Azure Functions base Docker image | •••

	Development	Hosting						
	Local dev machine	Azure Functions service	Azure Functions service	Azure Functions service	IoT devices	Additional Azure hosts	Non-Azure hosts / on-premises	Non-Azure hosts / on-premises
Platform	Core Tools + favorite editor	Consumption plan	App Service plan	Premium plan	Azure IoT Edge	AKS, Service Fabric, ...	K8s, raw VMs, & more	App Service on Azure Stack
Application delivery	Code or container	Code	Code or container	Code or container	Container	Container	Container	Code
Operating system	Windows, macOS, or Linux	Windows or Linux	Windows or Linux	Windows or Linux	Linux	Linux	Linux	Windows

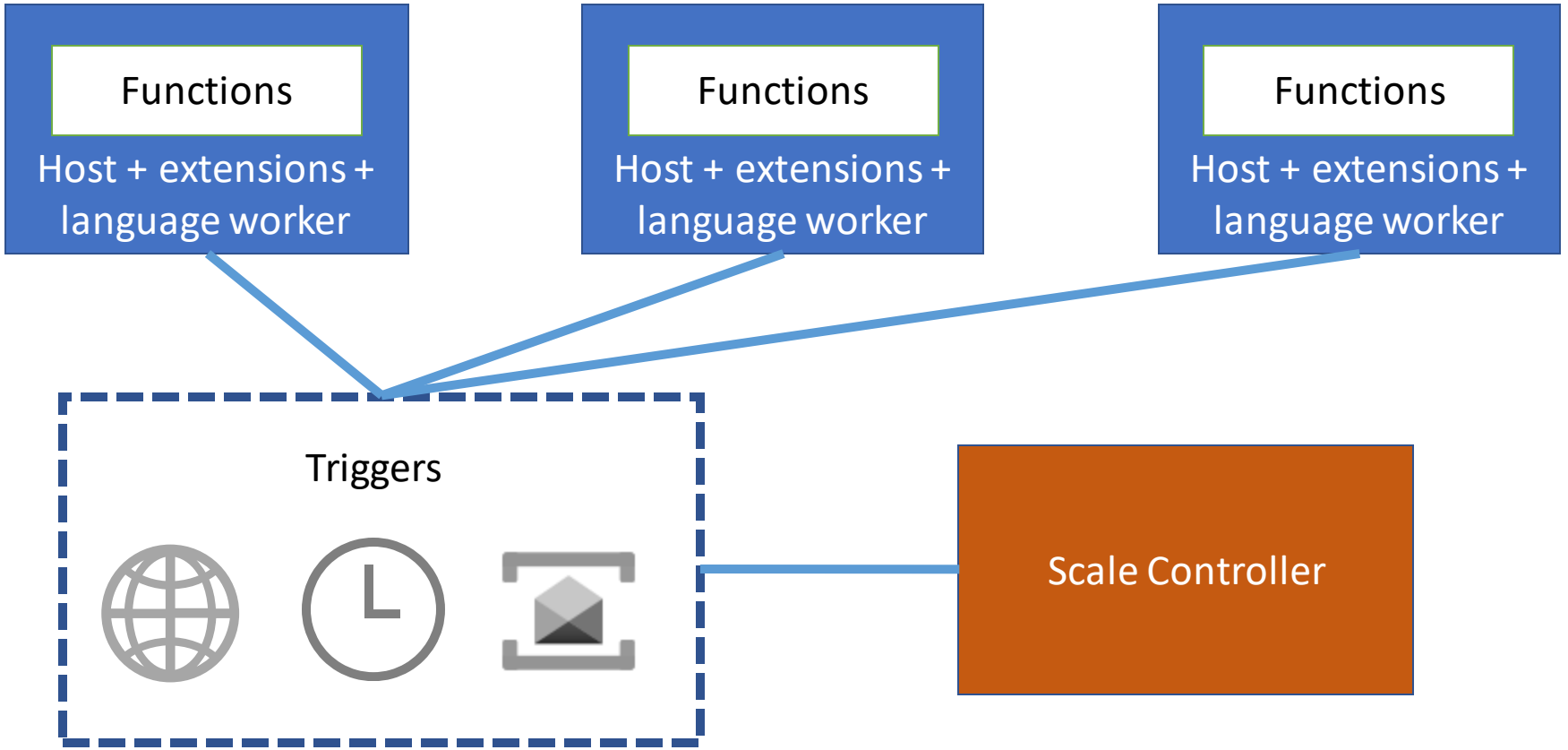
Build and deploy a function

A more complicated example



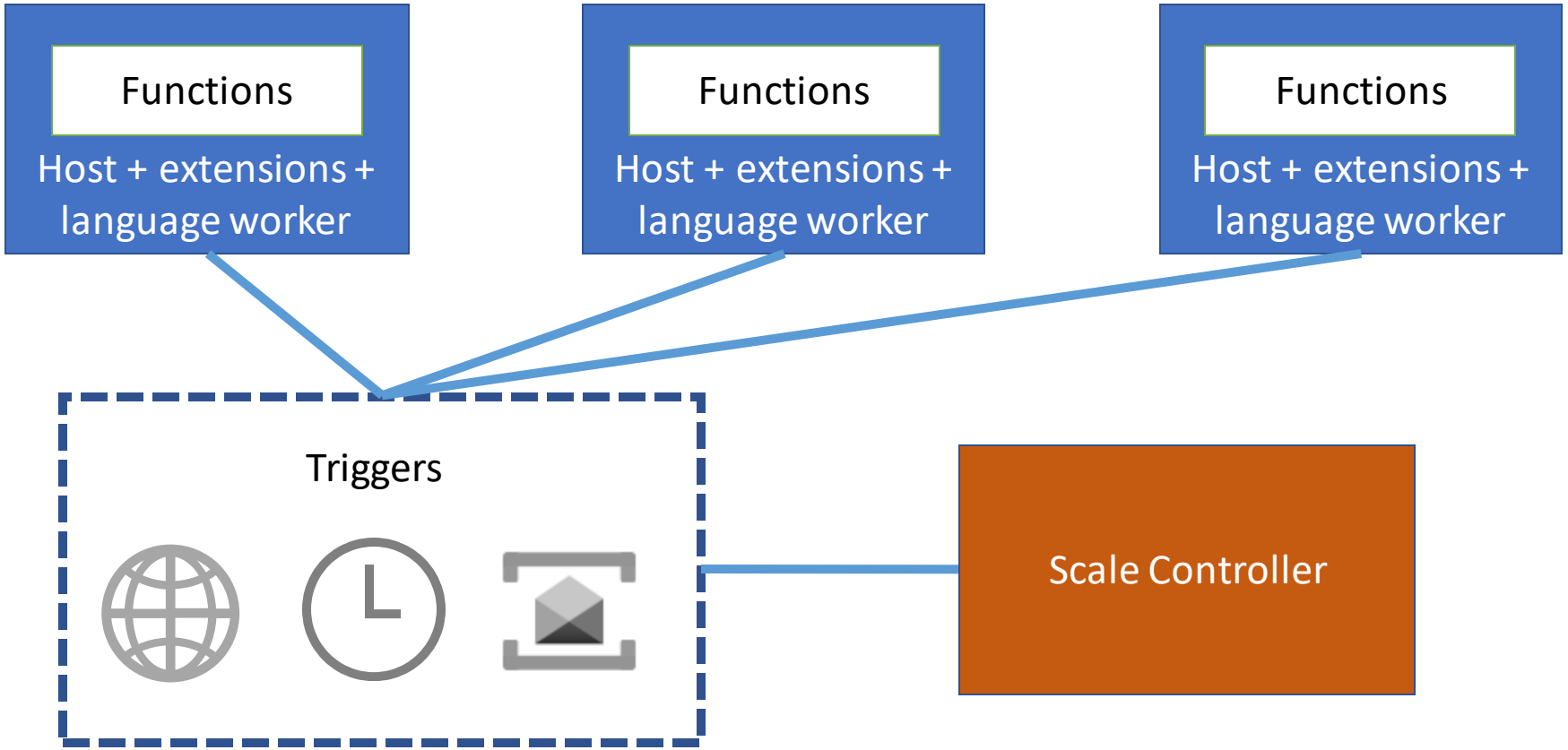
Live serverless speech translation

Scaling Azure Functions



Azure Functions Scaling

Deploying Azure Functions to Kubernetes



KEDA

- Kubernetes Event-Driven Autoscaling
- Scales Kubernetes deployments based on events
  - Azure Service Bus, Event Hubs, Queues, Blobs
  - Apache Kafka, RabbitMQ, Prometheus
  - AWS SQS, GCP Pub/Sub
- Works on any Kubernetes deployment (including Azure Functions)
- Azure Functions tools can deploy to Kubernetes with KEDA

Serverless functions limitations

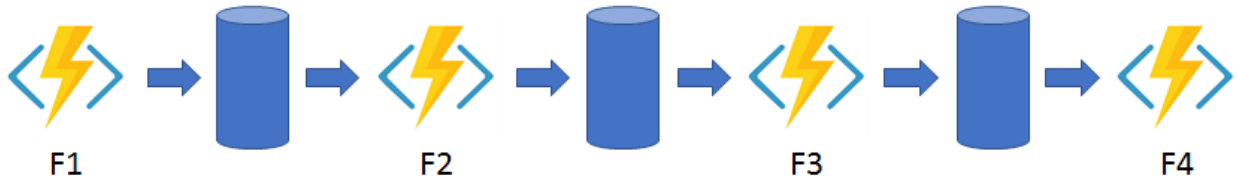
- Each invocation has a maximum duration
- Difficult to model patterns such as fan out / fan in
- Hard to maintain state between function calls

Solution: Durable Functions

Durable Functions

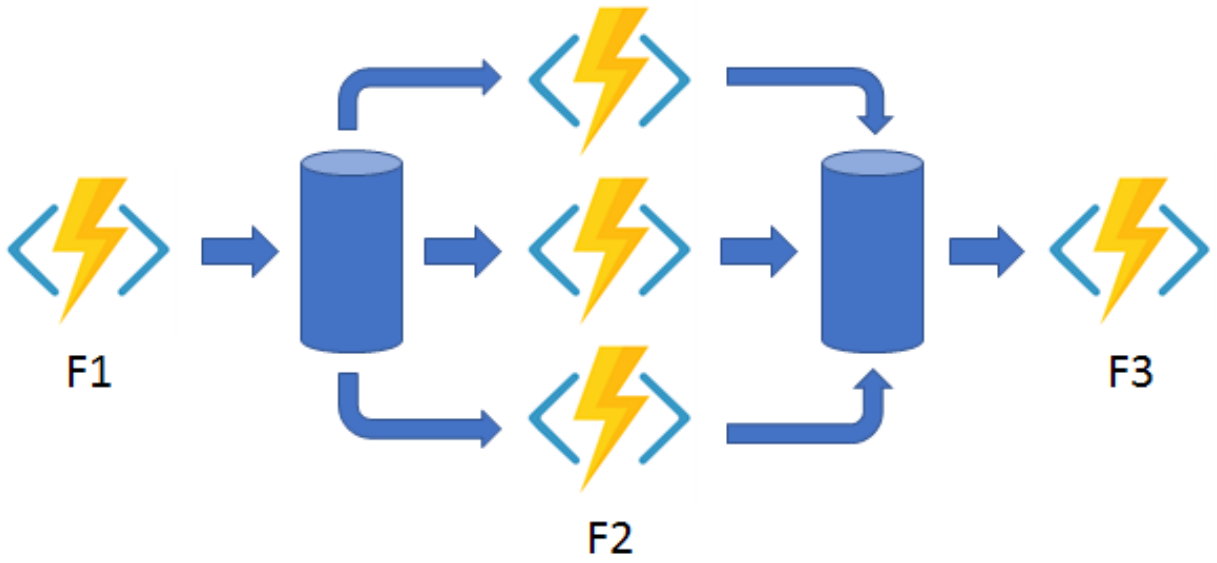
- Write stateful serverless workflows
- Patterns:
  - Function chaining
  - Fan out, fan in
  - Asynchronous HTTP APIs
  - Monitor
  - Human interaction

Function chaining




```
const df = require("durable-functions");

module.exports = df.orchestrator(function*(context) {
    try {
        const x = yield context.df.callActivity("F1");
        const y = yield context.df.callActivity("F2", x);
        const z = yield context.df.callActivity("F3", y);
        return    yield context.df.callActivity("F4", z);
    } catch (error) {
        // Error handling or compensation goes here.
    }
});
```

Fan out, fan in

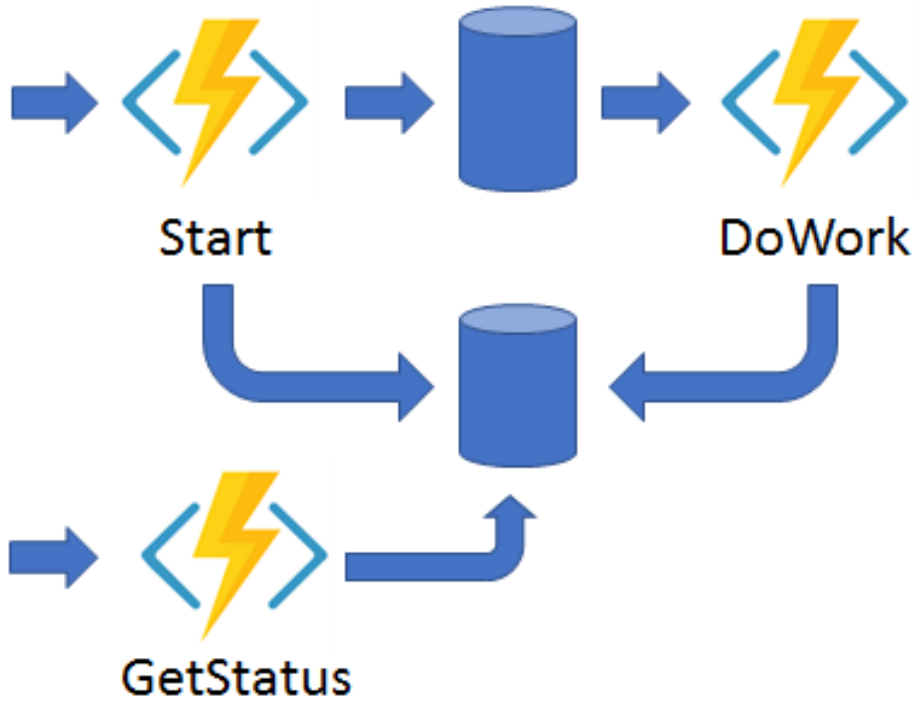


Fan out, fan in

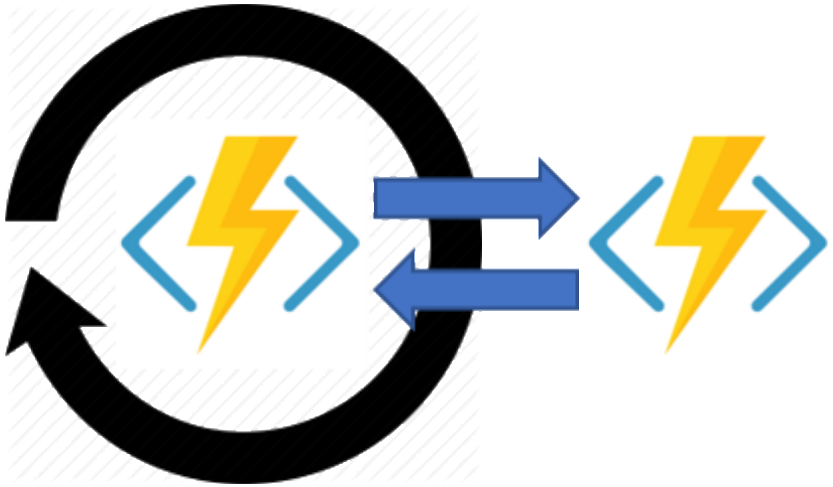
```
const df = require("durable-functions");

module.exports = df.orchestrator(function*(context) {
    const parallelTasks = [];

    // Get a list of N work items to process in parallel.
    const workBatch = yield context.df.callActivity("F1");
    for (let i = 0; i < workBatch.length; i++) {
        parallelTasks.push(context.df.callActivity("F2", workBatch[i]));
    }

    yield context.df.Task.all(parallelTasks);

    // Aggregate all N outputs and send the result to F3.
    const sum = parallelTasks.reduce((prev, curr) => prev + curr, 0);
    yield context.df.callActivity("F3", sum);
});
```

Asynchronous HTTP API

HTTP 202



Monitor

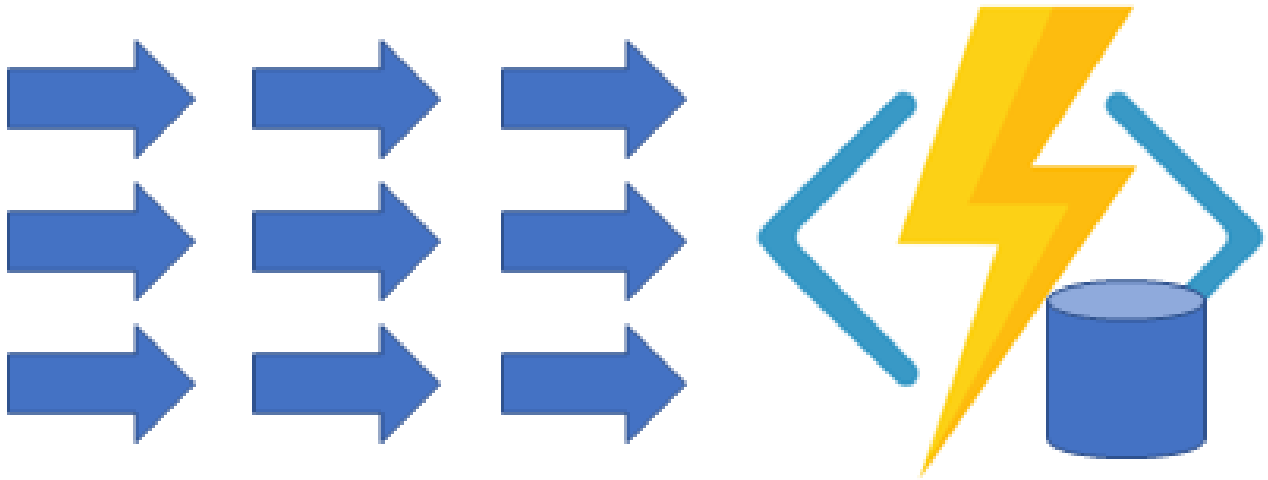


Human interaction

Human interaction

```
const df = require("durable-functions");
const moment = require('moment');

module.exports = df.orchestrator(function*(context) {
    yield context.df.callActivity("RequestApproval");

    const dueTime = moment.utc(context.df.currentUtcDateTime).add(72, 'h');
    const durableTimeout = context.df.createTimer(dueTime.toDate());

    const approvalEvent = context.df.waitForExternalEvent("ApprovalEvent");
    if (approvalEvent === yield context.df.Task.any([approvalEvent, durableTimeout])) {
        durableTimeout.cancel();
        yield context.df.callActivity("ProcessApproval", approvalEvent.result);
    } else {
        yield context.df.callActivity("Escalate");
    }
});
```

Actors (Durable Entities)



Actors (Durable Entities)

```
const df = require("durable-functions");

module.exports = df.entity(function(context) {
    const currentValue = context.df.getState(() => 0);
    switch (context.df.operationName) {
        case "add":
            const amount = context.df.getInput();
            context.df.setState(currentValue + amount);
            break;
        case "reset":
            context.df.setState(0);
            break;
        case "get":
            context.df.return(currentValue);
            break;
    }
});
```

```
[JsonObject(MemberSerialization.OptIn)]
public class Counter
{
    [JsonProperty("value")]
    public int Value { get; set; }

    public void Add(int amount)
    {
        this.Value += amount;
    }

    public Task Reset()
    {
        this.Value = 0;
        return Task.CompletedTask;
    }

    public Task<int> Get()
    {
        return Task.FromResult(this.Value);
    }

    public void Delete()
    {
        Entity.Current.DeleteState();
    }

    [FunctionName(nameof(Counter))]
    public static Task Run([EntityTrigger] IDurableEntityContext ctx)
        => ctx.DispatchAsync<Counter>();
}
```

Microsoft

IGNITE THE TOUR

서울
2020년 1월 21일-22일

COEX
서울특별시 강남구 영동대로 513

https://aka.ms/ignite-the-tour-seoul

개발자이신가요?

개발자 커뮤니티에 등록하시고
월간 뉴스레터를 받아보세요!

https://aka.ms/devKR