OpenIoTSummit Europe

The OpenWhisk Platform

Cloud native • Serverless • Event driven • Microservices

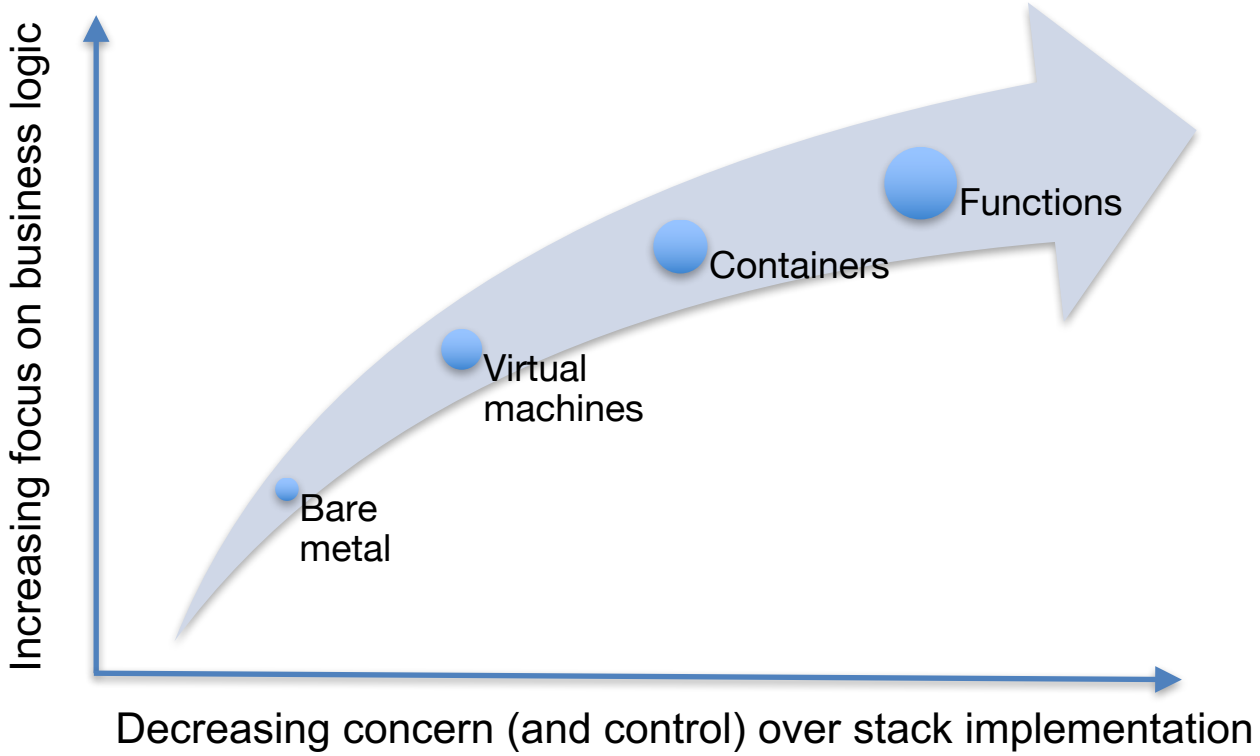
What you will learn today

- How cloud computing has recently evolved to enable developers to write cloud native applications better, faster, and cheaper using serverless technology.

- How OpenWhisk provides an open source platform to enable cloud native, serverless, event driven applications.

Introducing serverless, event driven computing

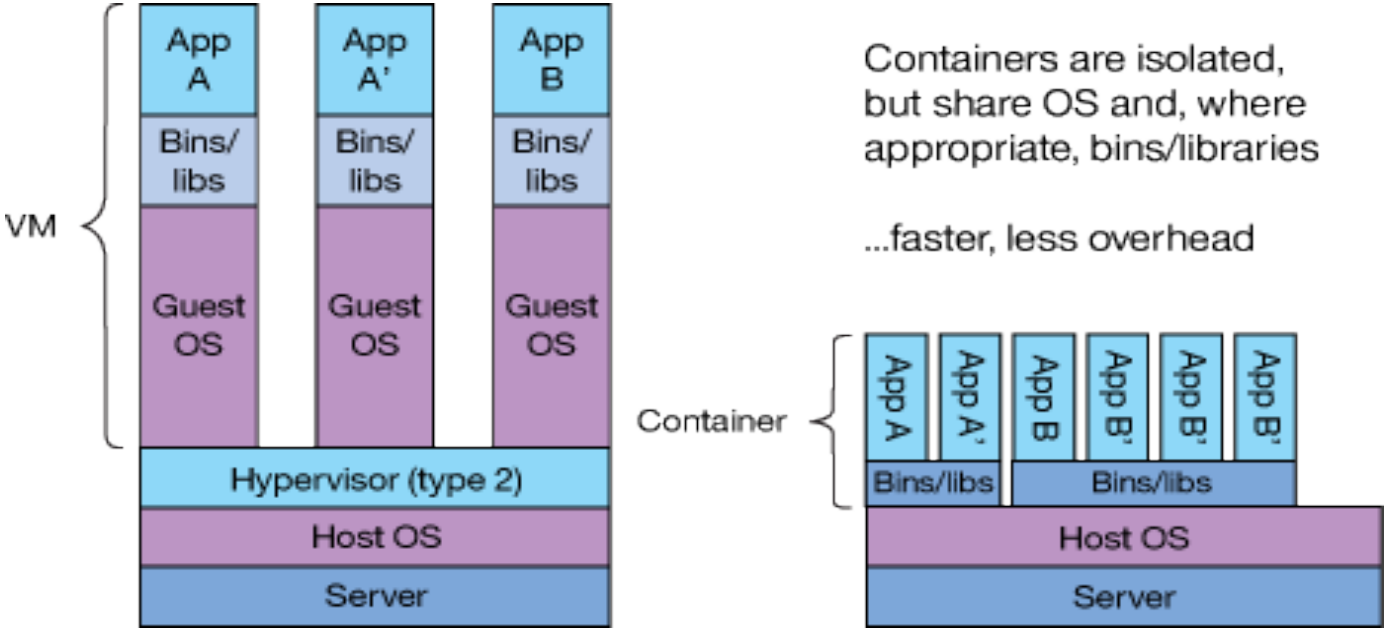
Cloud advances mean developers can write apps better, faster, and cheaper



What is “Serverless”?

- Allows Developers to offload operational tasks, such as hosting, scaling
- “Function as a service”, similar to “Platform as a Service”
- Treats compute resources as utilities
- Solutions being offered by Amazon, Microsoft, Google

VMs vs Containers



Many workloads match serverless, event driven programming

Execute app logic in response to database triggers

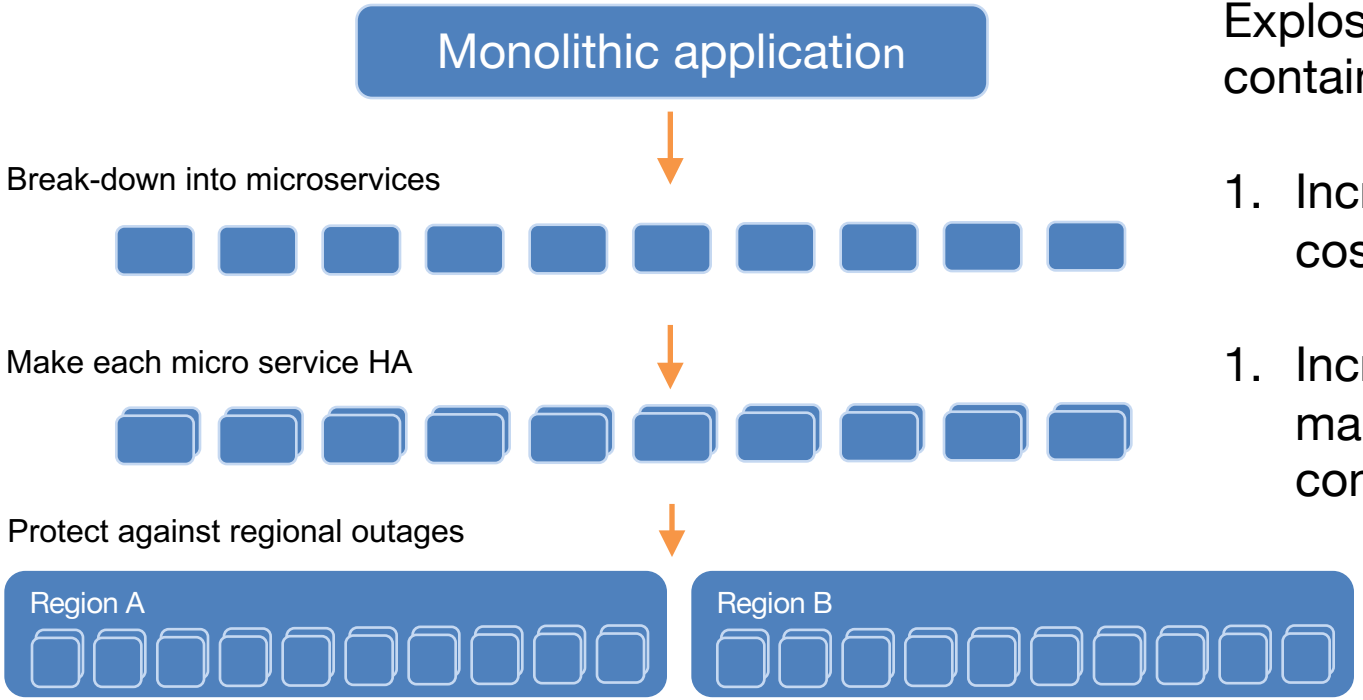
Execute app logic in response to sensor data

Execute app logic in response to cognitive trends

Execute app logic in response to scheduled tasks

Provide easy server-side backend for mobile app

It’s expensive to scale microservices



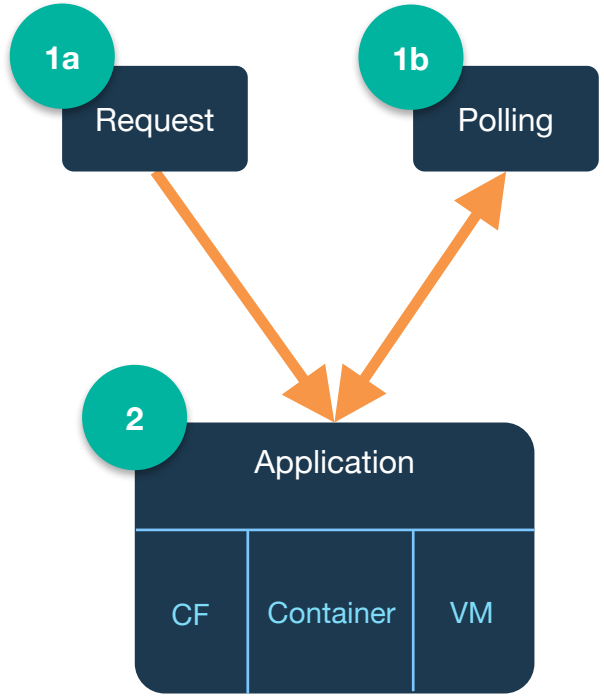
Explosion in number of containers / processes:

1. Increase of infrastructure cost footprint

1. Increase of operational management cost and complexity

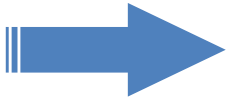
Programming and pricing models aren't efficient

- Continuous polling needed in the absence of an event driven programming model.
- Charged for resources, even when idle.
- Worries persist about capacity management.



Billing model offers a better match between app and resources

Applications charged by compute time (millisecond) rather than reserved memory (GB/hour).



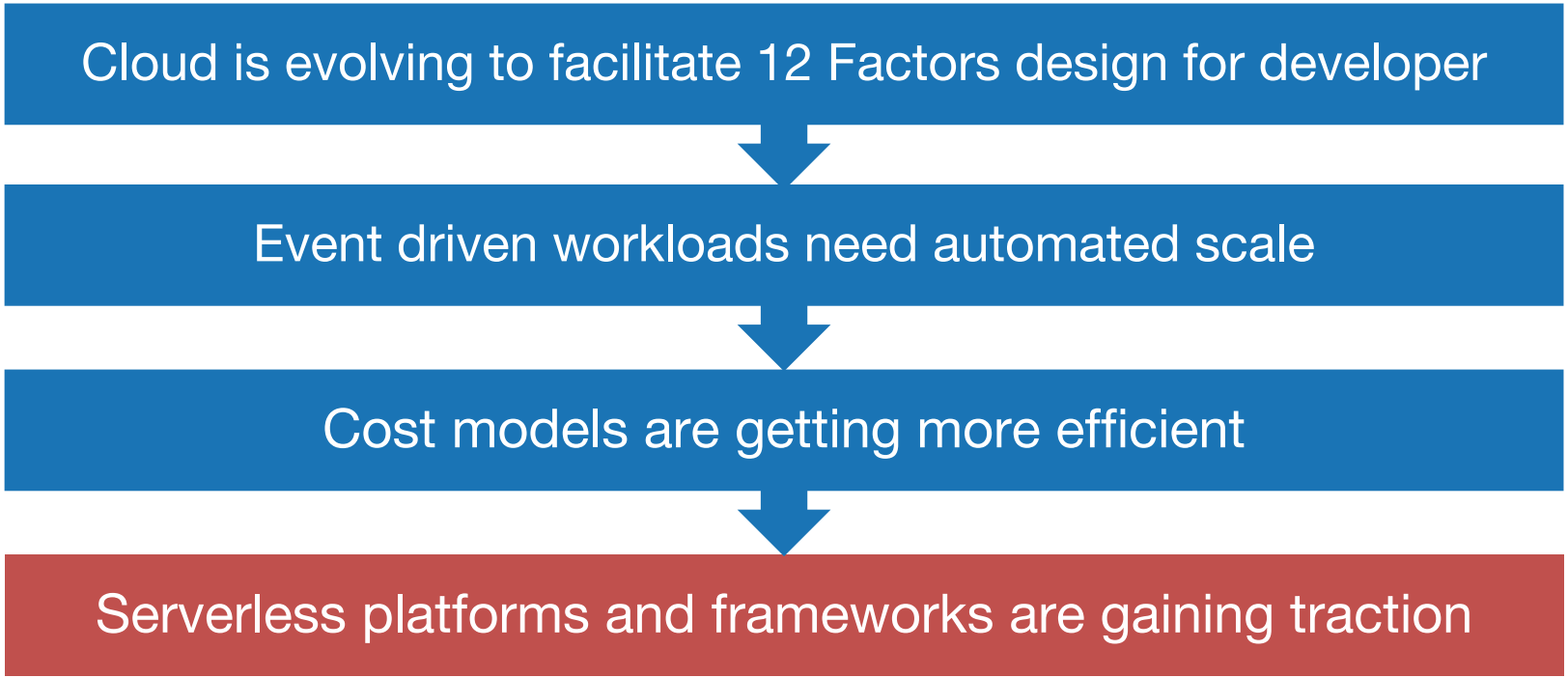
Greater linkage between cloud resources used and business operations executed.

While many applications must still be deployed in a daemon model, serverless provides an alternative that can mean substantial cost savings for a variety of event driven workloads.

Serverless offloads most cloud native app 12 Factor concerns

I	**Codebase**	**Handled by developer** (Manage versioning of functions on their own)
II	**Dependencies**	**Handled by developer**, facilitated by serverless platform (Runtimes and packages)
III	**Config**	**Handled by platform** (Environment variables or injected parameters)
IV	**Backing services**	**Handled by platform** (Connection information injected as parameters)
V	**Build, release, run**	**Handled by platform** (Deployed resources are immutable and internally versioned)
VI	**Processes**	**Handled by platform** (Single stateless containers often used)
VII	**Port binding**	**Handled by platform** (Actions or functions are automatically discovered)
VIII	**Concurrency**	**Handled by platform** (Process model is hidden and scales in response to demand)
IX	**Disposability**	**Handled by platform** (Lifecycle is hidden from the user, fast startup and elastic scale is prioritized)
X	**Dev/prod parity**	**Handled by developer** (The developer is the deployer)
XI	**Logs**	**Handled by platform** (Developer writes to console.log, platform handles log streaming)
XII	**Admin processes**	**Handled by developer** (No distinction between one off processes and long running)

Technological and business factors make serverless compelling



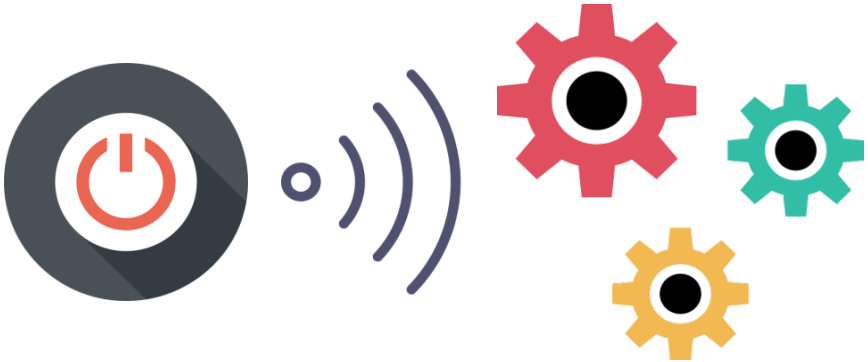
Enter OpenWhisk, a fabric and platform for the serverless, event driven programming model

OpenWhisk provides an elegant solution

OpenWhisk is a cloud platform

*that **executes code***

*in **response to events***



Triggers, actions, rules (and packages)

Services define the events they emit as **triggers**.
Developers associate **actions** to handle the events via **rules**.
Packages are used to bundle and distribute sets of actions

Triggers

T *A class of events that can happen*

Social events

Data changes

Device readings

User input

Location updates

Actions

A *Code that runs in response to an event (that is, an event-handler)*

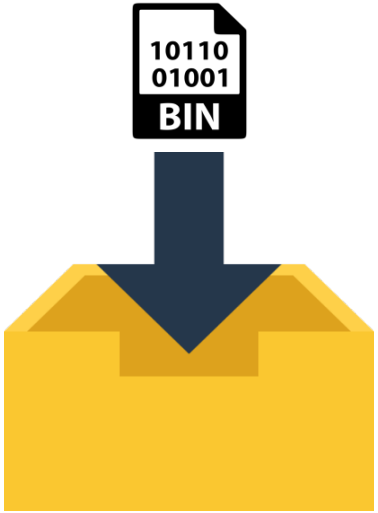
Actions

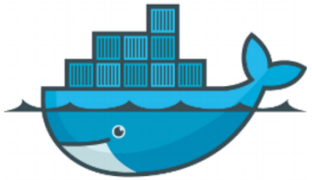
- Can be written in a variety of languages, such as JavaScript, Python, Java, and Swift

```
function main(msg) {
        return { message: 'Hello, ' + msg.name + ' from ' + msg.place };
};
```

Actions

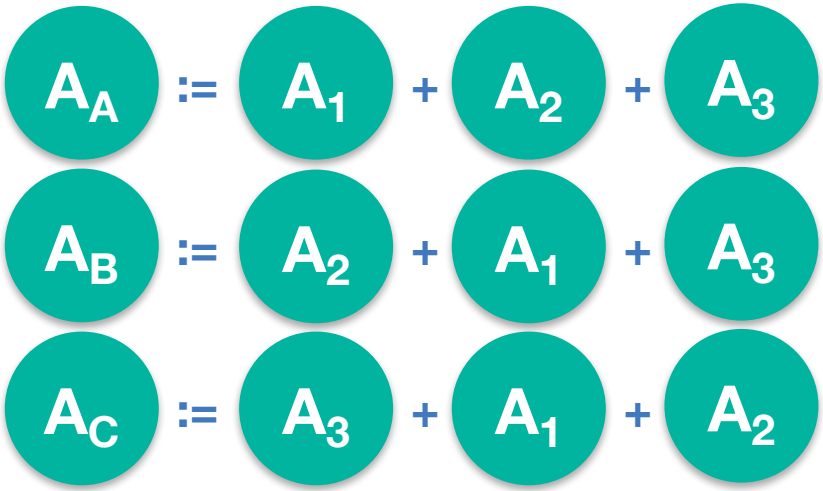
A Or any other arbitrary binary with Docker





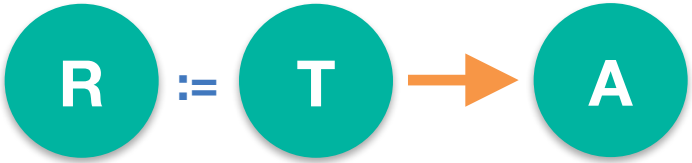
Actions

- Can be composed to create sequences that increase flexibility and foster reuse



Rules

R *An association of a trigger to an action in a many to many mapping.*



Packages

P *A shared collection of triggers and actions*

IBM

IBM Cloudant®
- A read
- A write
- T changes

IBM Watson
- A translate

The Weather Company
- A forecast

Open Source

kafka
- A post
- T topic

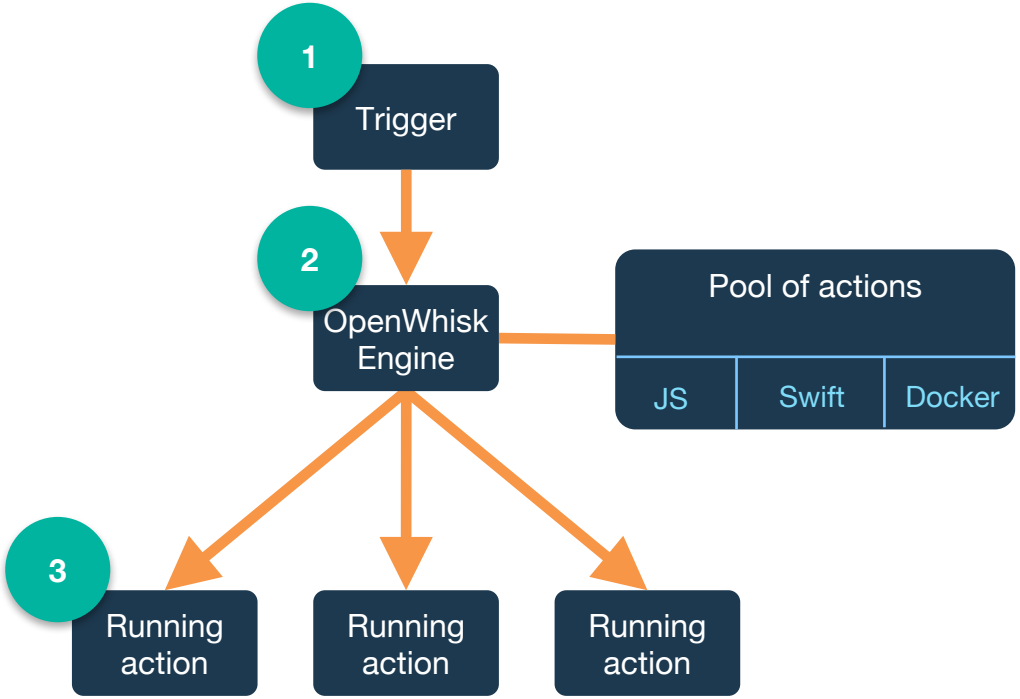
Third Party

Git
- T commit

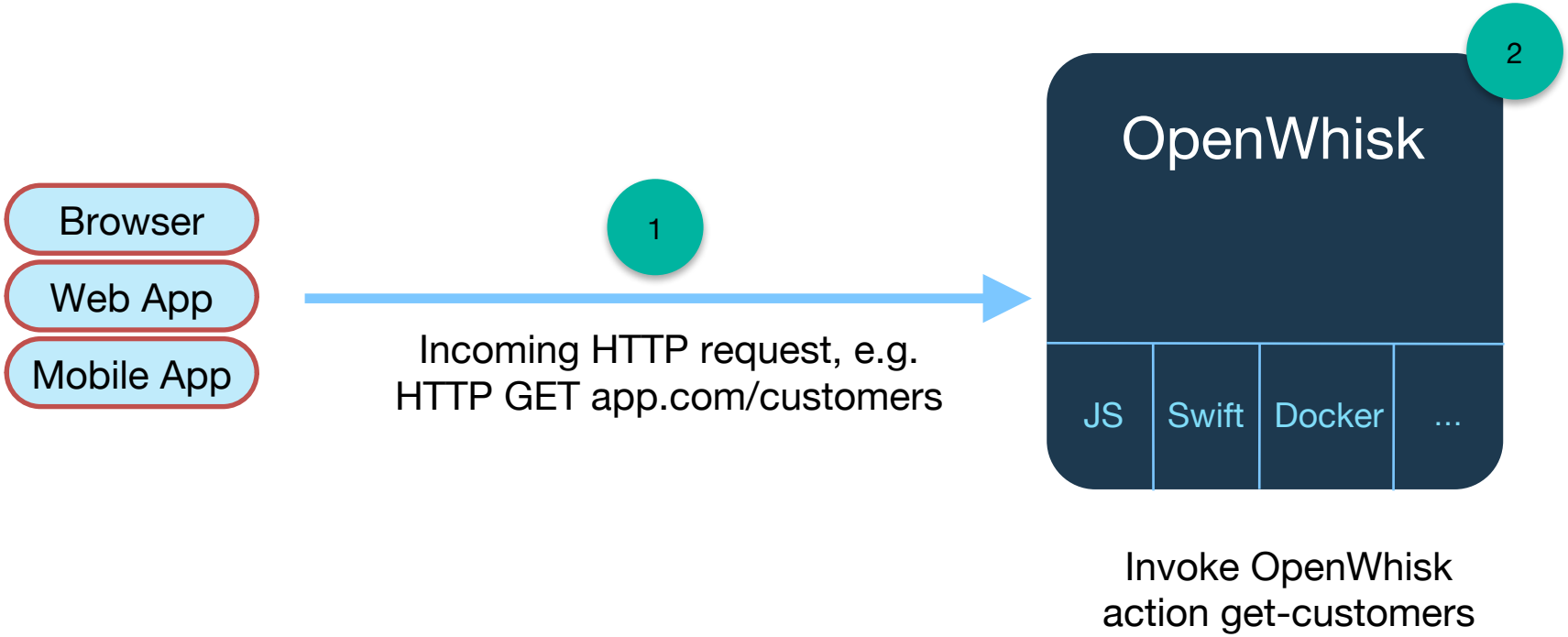
Yours


- A myAction
- T myFeed

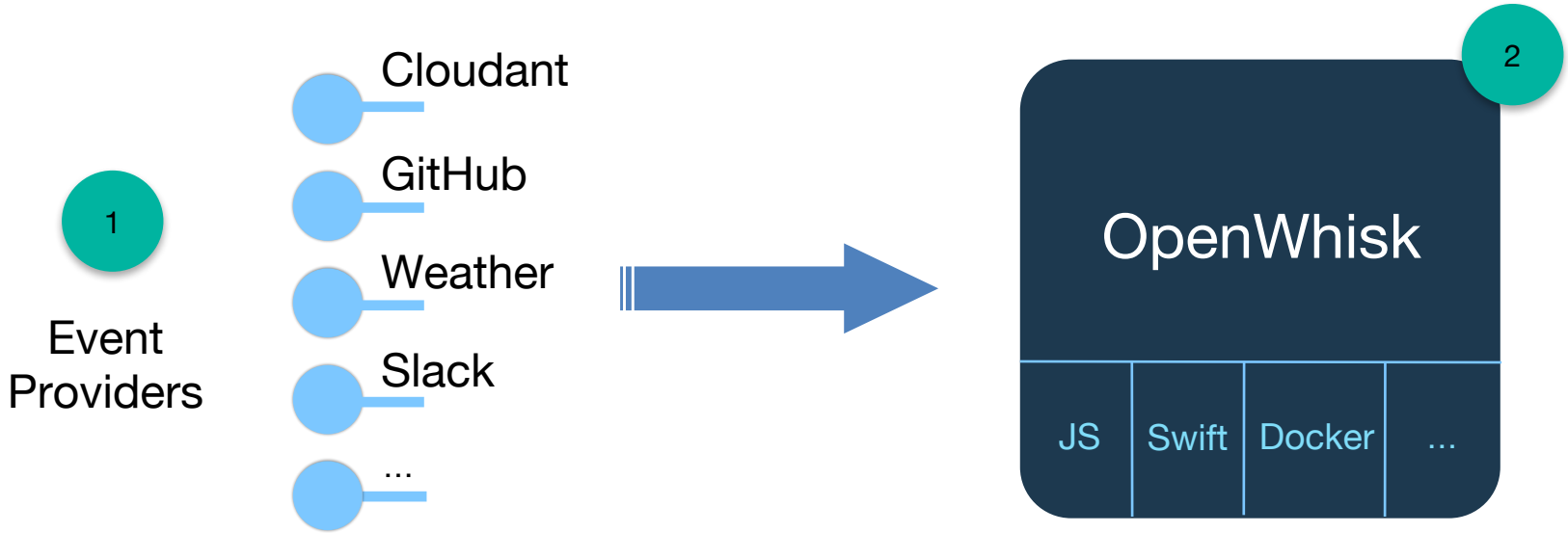
OpenWhisk execution model



OpenWhisk can implement REST microservices



OpenWhisk enables event driven applications



An event occurs, for example

- Commit pushed to GitHub repository
- Data entered in Cloudant

Which triggers execution of associated OpenWhisk action

LOB, SoR systems & databases

Service reports

Customer registry

Shipping system

SendGrid

Email: Filter on its way!

actions

Watson IOT

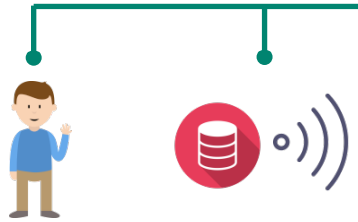
OpenWhisk

Need a new filter

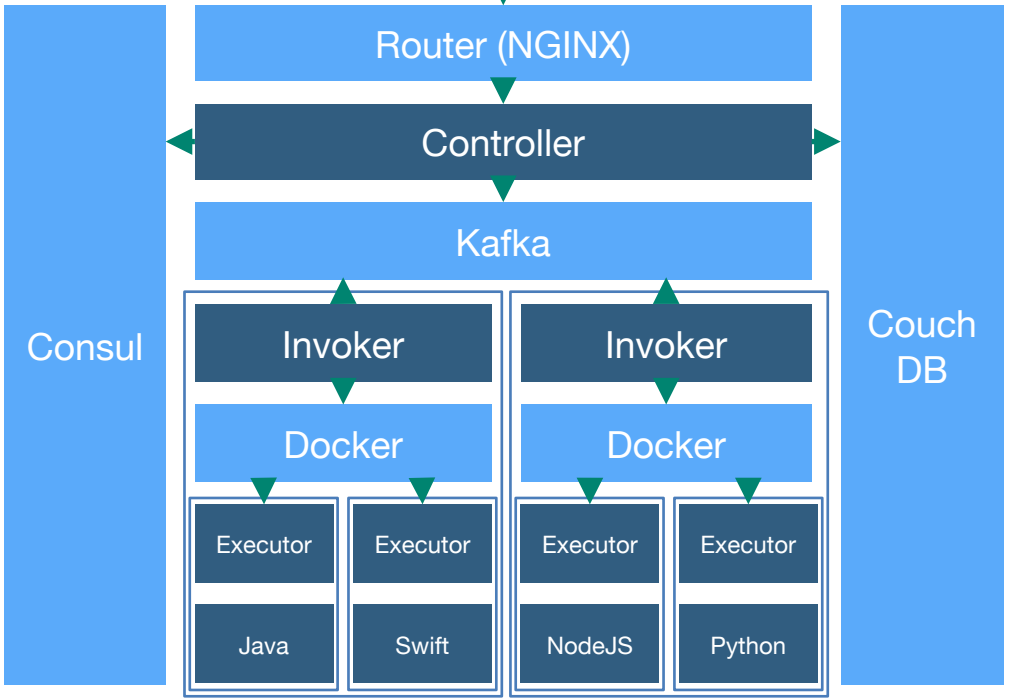
IBM Cloud

OpenWhisk
high level implementation architecture

OpenWhisk under the hood



1. Router receives request to API via CLI or UI
2. Controller checks entitlement and dispatches requests to Kafka
3. Invokers pull requests and start execution of the action



Summary

Serverless Benefits

- ✓ A flexible programming environment
- ✓ An open ecosystem of building blocks
- ✓ Compute task outsourcing to the cloud
- ✓ No servers to manage or maintain
- ✓ Automatic scaling to match workload
- ✓ Built-in fault tolerance
- ✓ A pay-as-you-go model

Join us to build a serverless platform for the future!

OpenWhisk.org

dwopen.slack.com
#openwhisk

What you learned today

- We’re in the early days of an evolution that is empowering developers to write cloud native applications better, faster, and cheaper

- OpenWhisk provides an open source platform to enable cloud native, serverless, event driven applications

But, this is still early in the evolution of serverless

- There are still rough areas to be addressed
  - Monitoring, debugging, developer tooling, workflows, and visibility require more work. The Bluemix hosted tools such as web IDE and console are value added services.
  - Best practices and common message formats need to be distilled.
  - The flexible polyglot nature of OpenWhisk must be balanced with developer responsibility (e.g., Docker image, npm build step)

OpenWhisk addresses many common workloads

Digital app workloads

OpenWhisk can help power various mobile, web and IoT app use cases by simplifying the programming model of orchestrating various services using events without a dedicated backend.

Big Data/Analytics pipeline

Complex data pipelines for Big Data/Analytics tasks can be scripted using changes in data services or streams for near real-time analytics and feedback.

DevOps and infrastructure as code

OpenWhisk can be used to automate DevOps pipelines based on events triggered from successful builds, or completed staging, or a go-live event.

Microservices builder

OpenWhisk can be used to easily build microservices given the footprint and programming model desired by microservices

OpenWhisk design principles

1. Provide an open interface for event providers
2. Offer polyglot support and simple extensibility for new runtimes
3. Support higher level constructs as appropriate (e.g. action sequences)
4. Scale dynamically on a per request basis
5. Enable sharing of actions and event providers
6. Leverage best of breed open source software (Docker, Kafka, Consul, …)
7. Use the Apache 2 License