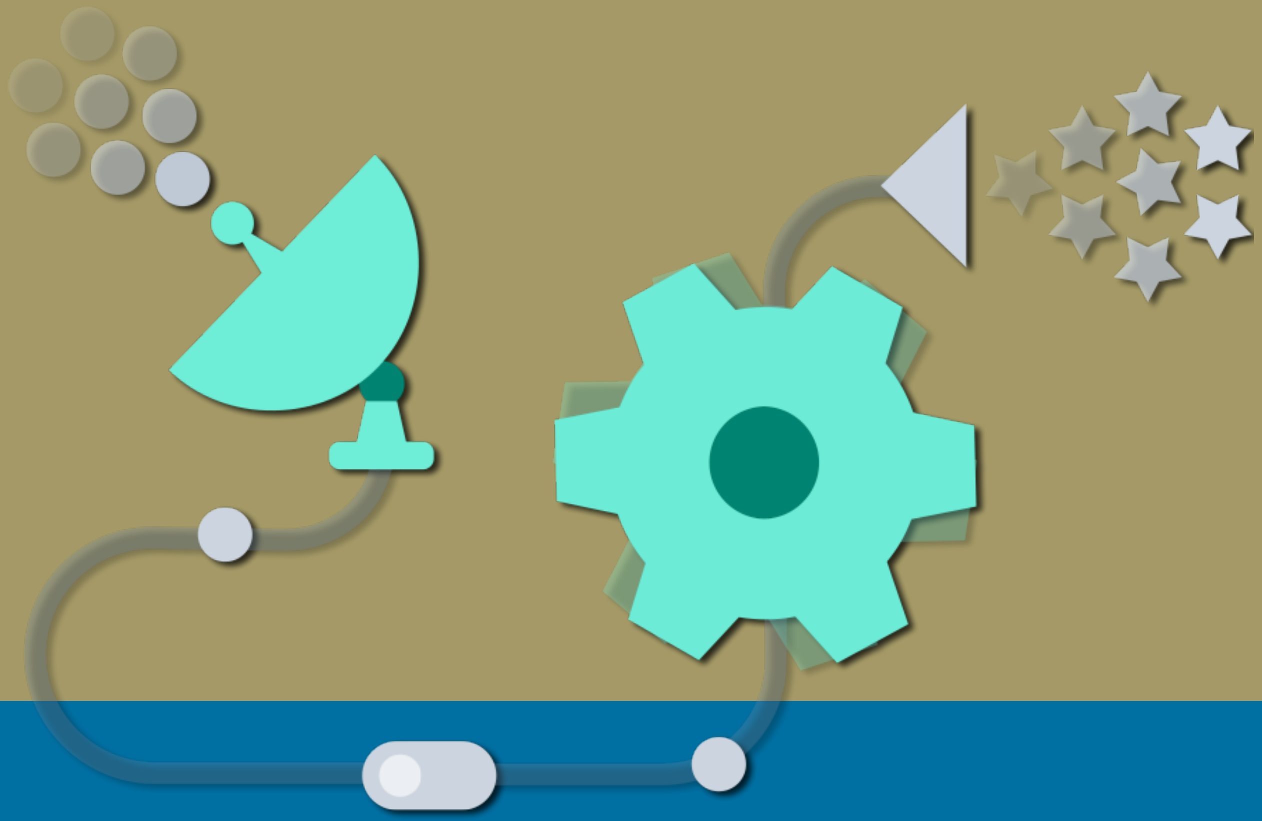
Deep Dive into OpenWhisk Core Services

http://openwhisk.org/

#openwhisk

https://openwhisk-team.slack.com/



Vincent Hou, Software Engineer, *IBM Cloud Open Technologies*

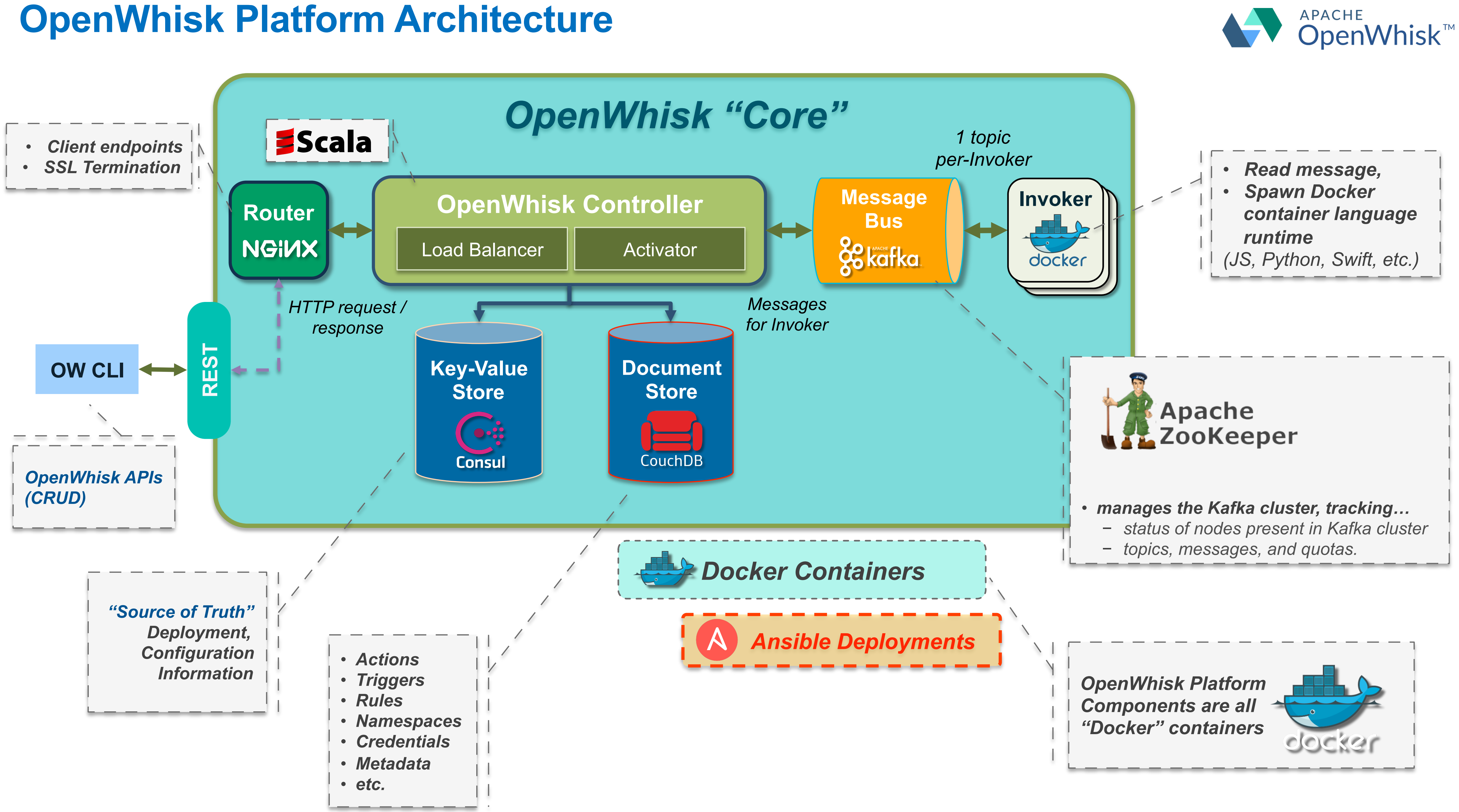
Session Topics

- Architecture of Apache OpenWhisk
- Core Services of Implementation
- How to Contribute to Apache OpenWhisk
- QA

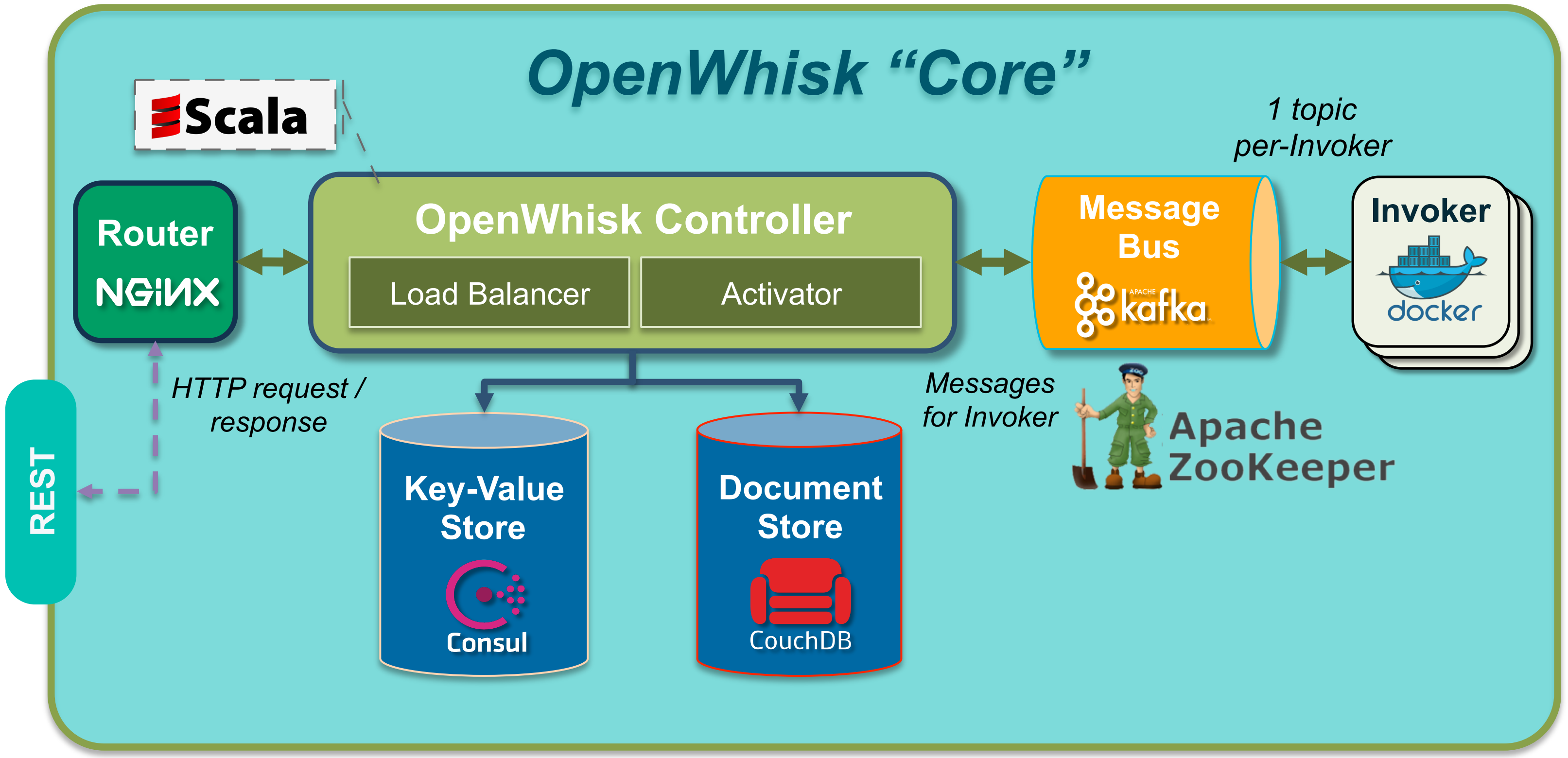
APACHE OpenWhisk™

Architecture of Apache OpenWhisk

OpenWhisk Platform Architecture



OpenWhisk Core Services



- Services containerized
- Nginx
- Controller
- CouchDB
- Consul
- Kafka & ZooKeeper
- Invoker

Docker Containers

Core services of Implementation

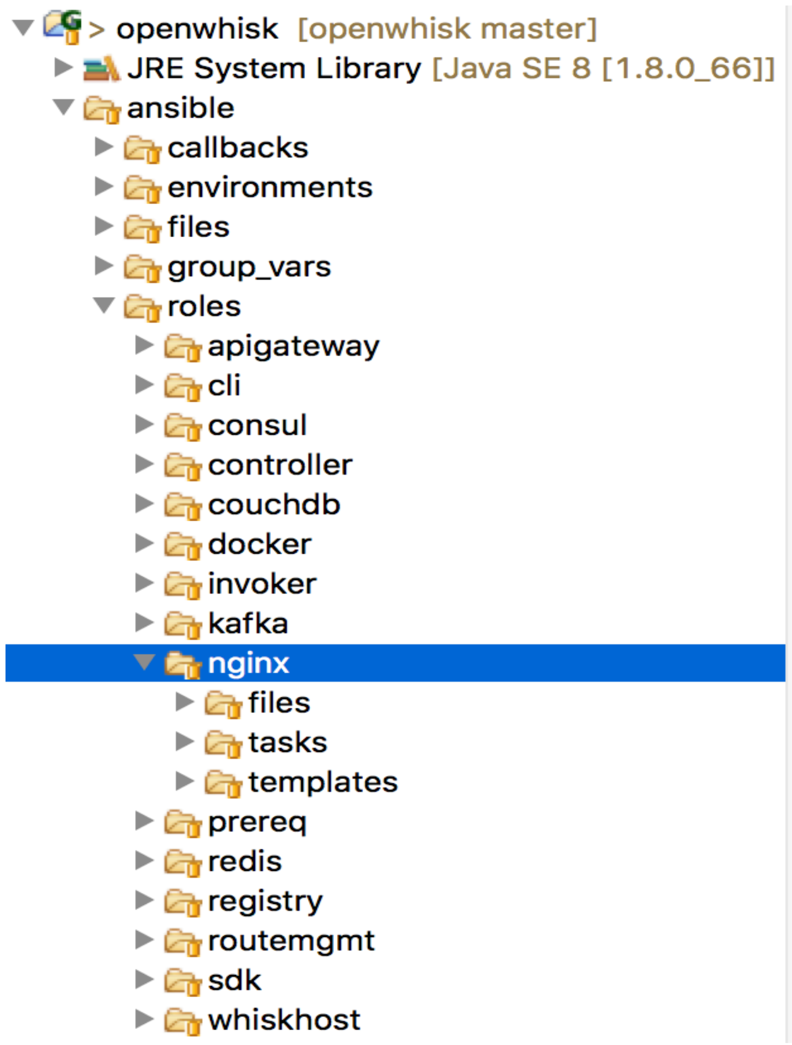
Nginx – Expose HTTP(S) Public Endpoint

OpenWhisk launches Nginx service in a container.

- *Reverse proxy: first entry point into OpenWhisk*
- *Security: SSL termination*

Ansible Role for Nginx

- *Install certificates.*
- *Install configuration file.*
- *Pull the image of Nginx.*
- *Launch the Nginx.*



```
▼ > openwhisk  [openwhisk master]
  ▶ JRE System Library [Java SE 8 [1.8.0_66]]
  ▼ ansible
    ▶ callbacks
    ▶ environments
    ▶ files
    ▶ group_vars
    ▼ roles
      ▶ apigateway
      ▶ cli
      ▶ consul
      ▶ controller
      ▶ couchdb
      ▶ docker
      ▶ invoker
      ▶ kafka
      ▼ nginx
        ▶ files
        ▶ tasks
        ▶ templates
      ▶ prereq
      ▶ redis
      ▶ registry
      ▶ routemgmt
      ▶ sdk
      ▶ whiskhost
```

APACHE OpenWhisk™

Controller – Implement Scala-based REST API

Controller uses three major packages: Spray, Akka and Swagger.

- *Spray: server-side REST/HTTP support*
  - *spray-can: HTTP server, package common/scala/src/main/scala/whisk/http*
  - *spray-json: json implementation in scala, almost all scala files in controller*
  - *spray-http: model of requests, responses and headers.*
  - *spray-httpx: datastore access, marshalling, unmarshalling*
  - *spray-routing: routing DSL for web services, request context, authentication, etc, almost all actors in controller*
  - *spray-client: client-side HTTP support*

- *Akka: describing OpenWhisk modeling entities for API in actors and futures*

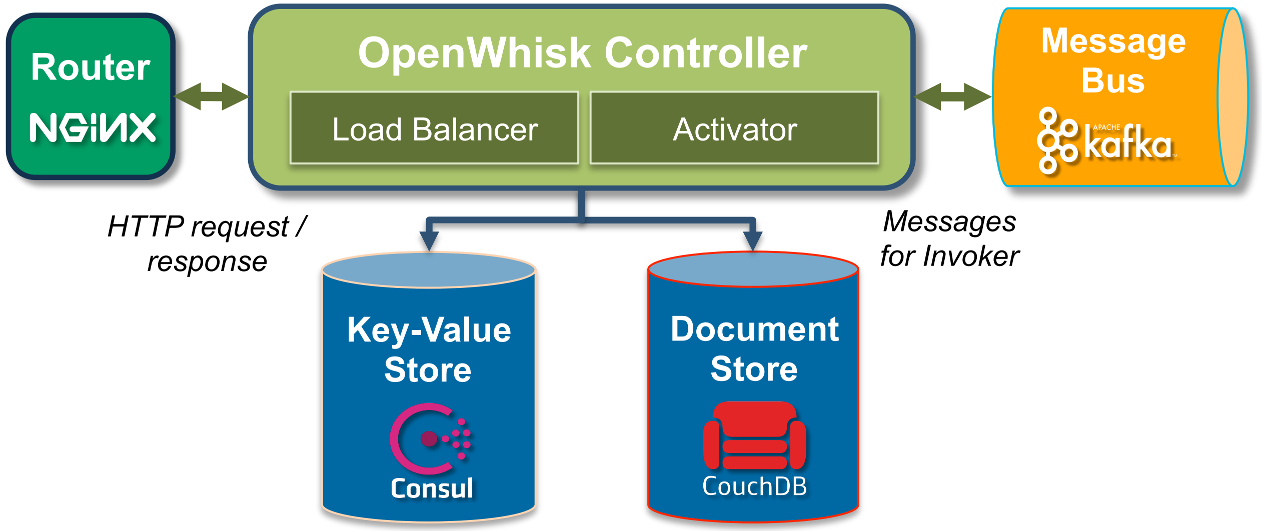
- *Swagger: describing OpenWhisk API structure, able to build interactive and neat docs, and generate client code. Standardization.*

Controller – Implement Scala-based REST API

Controller contains two sub-services: activator and load balancer, in terms of functionalities.

- *Activator: processing events produced by triggers, calling the action bound by a rule to a particular trigger. (POST to load balancer) core/controller/src/main/scala/whisk/core/controller/Triggers.scala*

- *Load balancer: selecting a proper invoker to run the action, publishing messages to message service. whisk.core.loadBalancer*

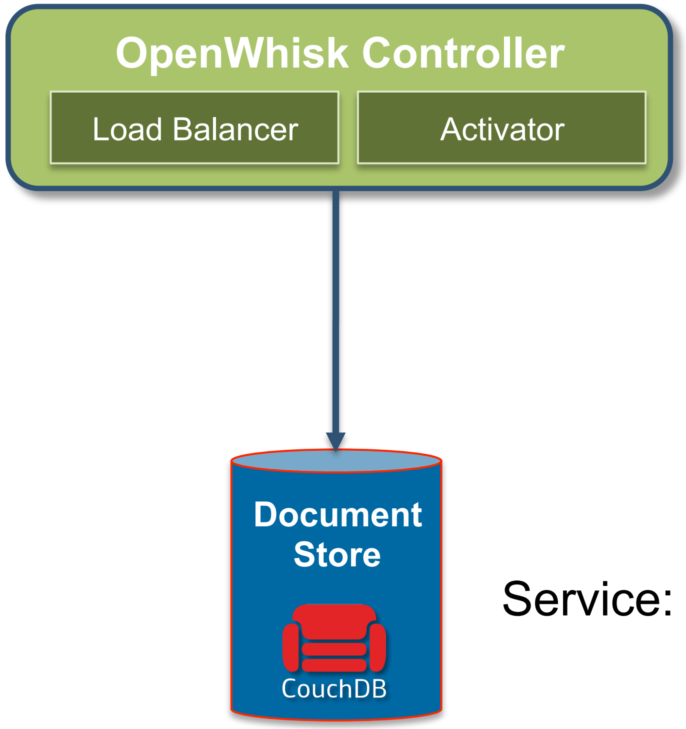


APACHE OpenWhisk™

CouchDB – maintain the system state

CouchDB saves three major documents: subjects, and whisks and activations.

- *Subjects: credentials, used to authenticate and authorize by controller.*
- *Whisks: namespaces, and the definitions of actions, triggers, and rules, etc, used to query by controller.*
- *Activations: activations, used to query by controller.*



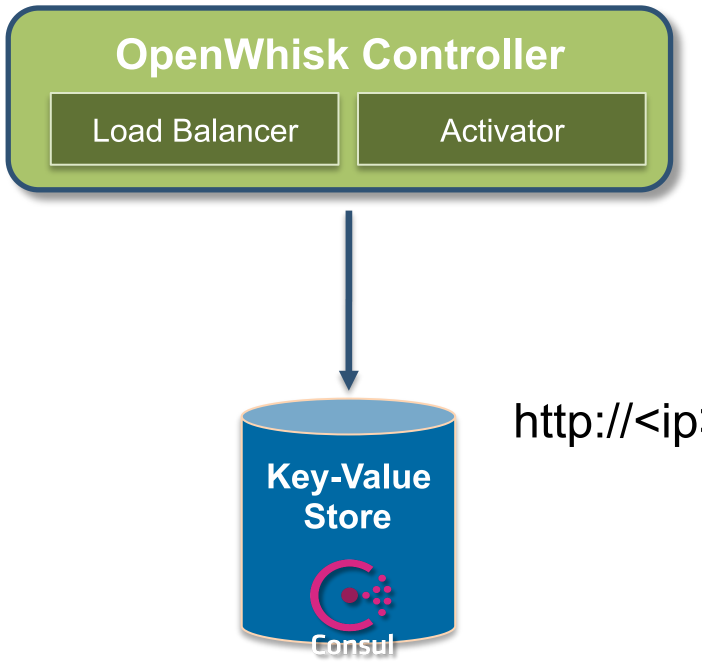
Service: http://<ip>:5984/_utils/

APACHE OpenWhisk™

Consul – maintain service state

Consul manages the state information of all the containerized services in OpenWhisk.

- *Load balancer accesses Consul service to decide which invoker to pick for the execution of the action.*
- *Every OpenWhisk service self-register itself in Consul by Registrator service.*

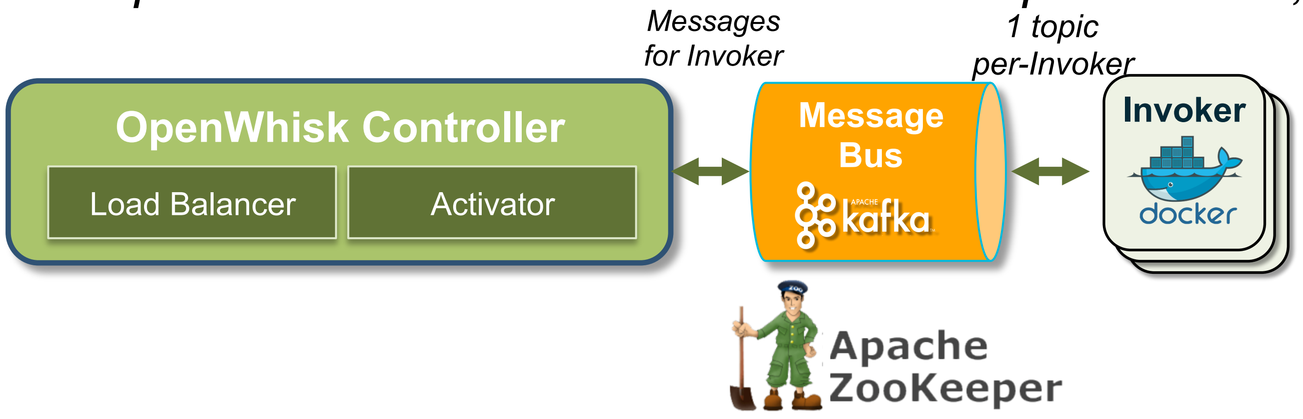


http://<ip>:8500/ui/#/dc1/services

Kafka – Messaging service of OpenWhisk

Kafka buffers the messages sent by the Controller before delivering them to the Invokers.

- *One invoker is subscribed to one topic. Communication between controller and invoker is buffered and persisted by Kafka.(System crash and heavy load covered)*
- *Kafka cluster is managed by ZooKeeper: track status of nodes and also to keep track of the topics, messages, and quotas.*
- *Non-blocking & blocking: asynchronous and synchronous supported.*
- *ActivationID is used to trace the status of running an action.*
- *Input to the Kafka: action to be run and parameters, sent by controller.*



APACHE OpenWhisk™

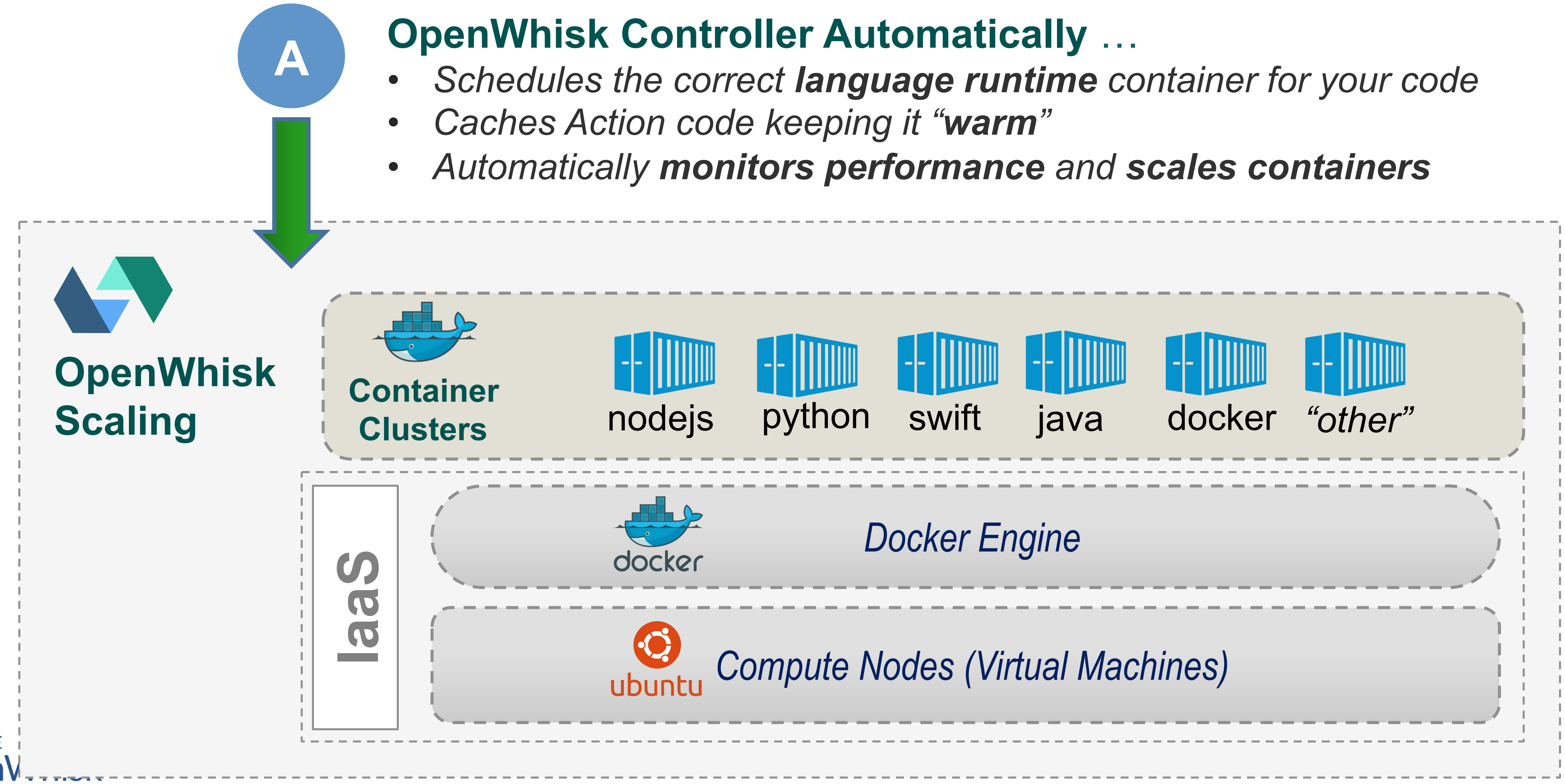
Invokers – Run Language-Specific Docker Containers for Actions

OpenWhisk supports many languages (runtimes) for Actions

- *JavaScript, Swift, Java, Python and more in the future…*

OpenWhisk Controller Automatically …

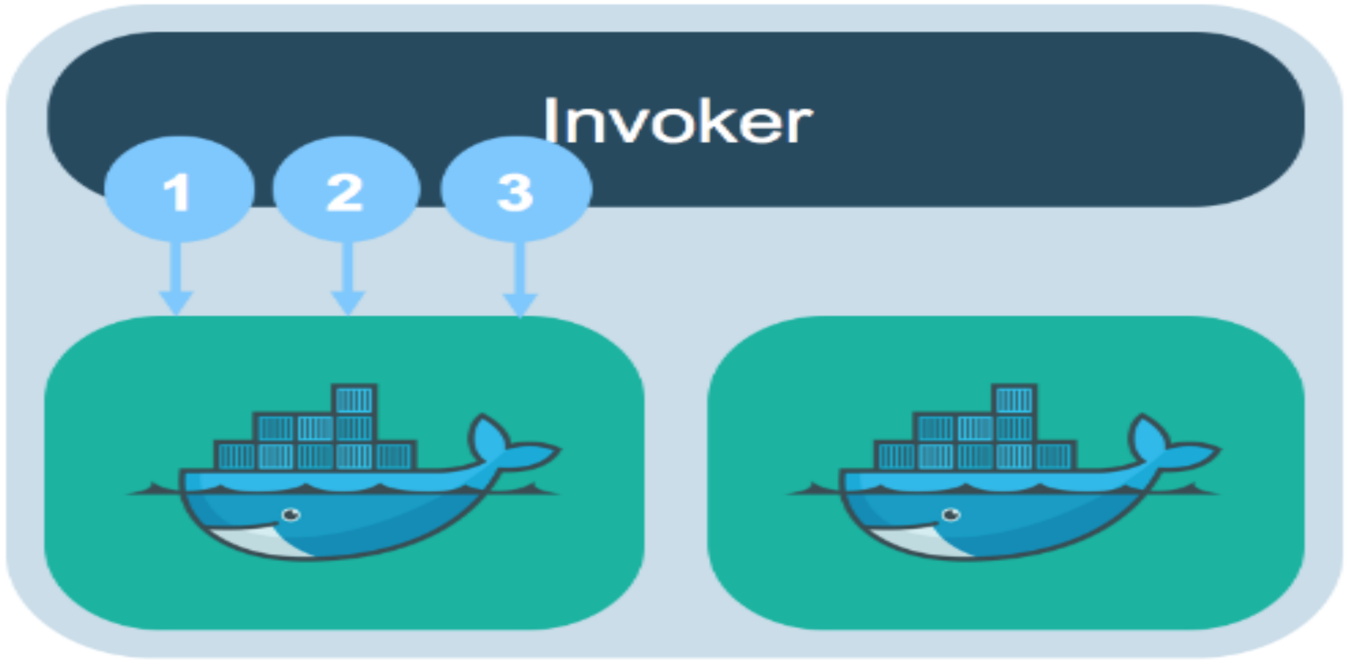
- *Schedules the correct **language runtime** container for your code*
- *Caches Action code keeping it "**warm**"*
- *Automatically **monitors performance** and **scales containers***



Invokers – Most Scalable service in OpenWhisk

One invoker is picked to run the action.

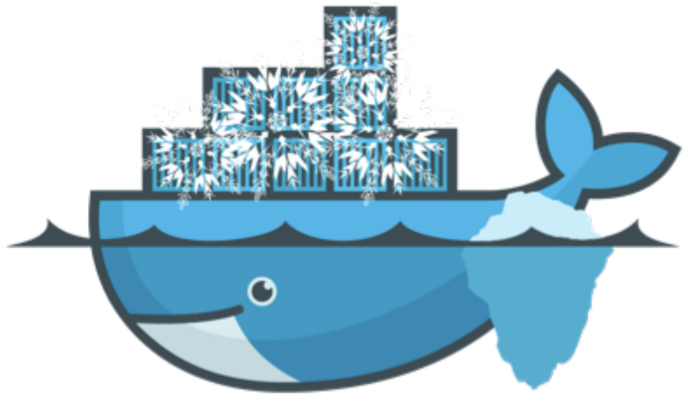
1. Starting the container via docker run and get the container's IP address via docker inspect.
2. Initializing the container with the action via POST /init.
3. Run the action via POST /runDocker run: launch a new container to run the action.



APACHE OpenWhisk™

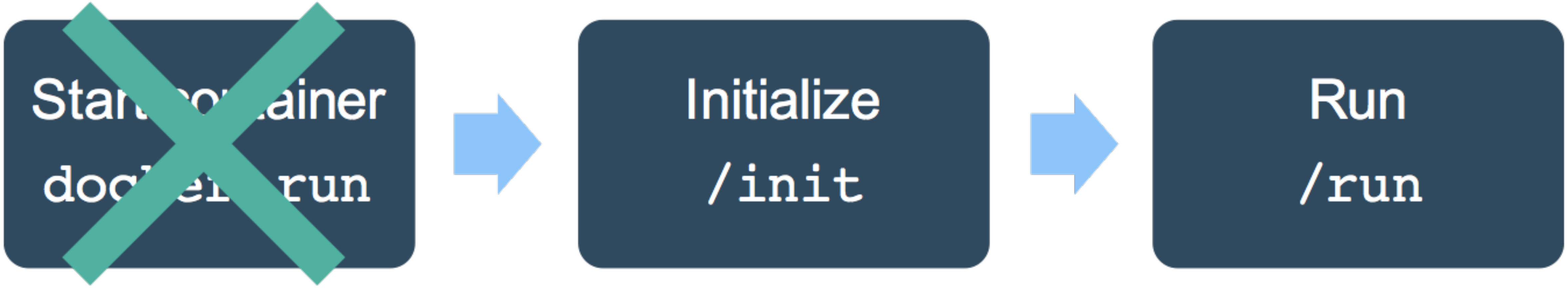
Invoker and container

$$\begin{tikzcd}\boxed{\begin{array}{c}\text{Start container}\\ \texttt{docker run}\end{array}} \arrow[r] & \boxed{\begin{array}{c}\text{Initialize}\\ \texttt{/init}\end{array}} \arrow[r] & \boxed{\begin{array}{c}\text{Run}\\ \texttt{/run}\end{array}}\end{tikzcd}$$


cold container

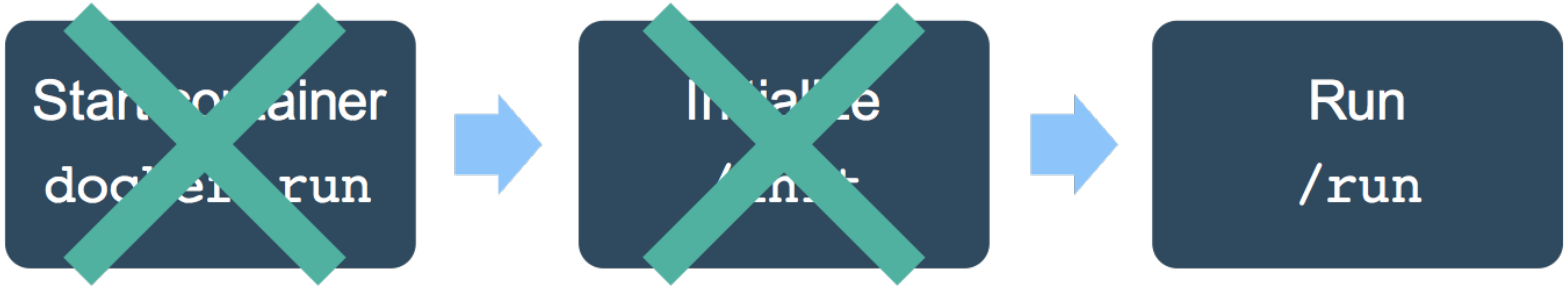
When an action is launched for the first time, its container is added to the pool for future use.

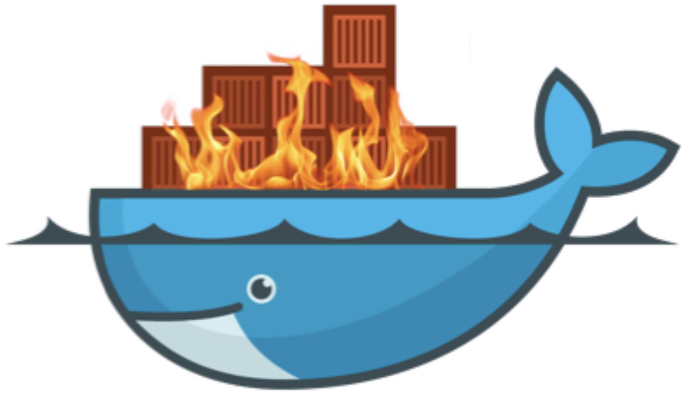
Invoker and container



pre-warmed container

Invoker and container




warm container

How to contribute to OpenWhisk

How to Contribute to OpenWhisk

- Join the OpenWhisk community: http://www.apache.org/licenses/#clas.
- Communication with the OpenWhisk community: email, slack, etc.
- Create github account and configure.
- Set up local development environment: Eclipse, IntelliJ, etc.

Reference:
https://medium.com/openwhisk/how-to-contribute-to-openwhisk-6164c54134a6, How to contribute to OpenWhisk.

APACHE
OpenWhisk™

APACHE INCUBATOR PODLING

Thank you!

Questions?

扫码关注
IBM开源技术微讲堂
获取最新课程信息

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