CHINA OpenInfra Days

CHINA OpenInfra Days | IT大咖说 知识共享平台

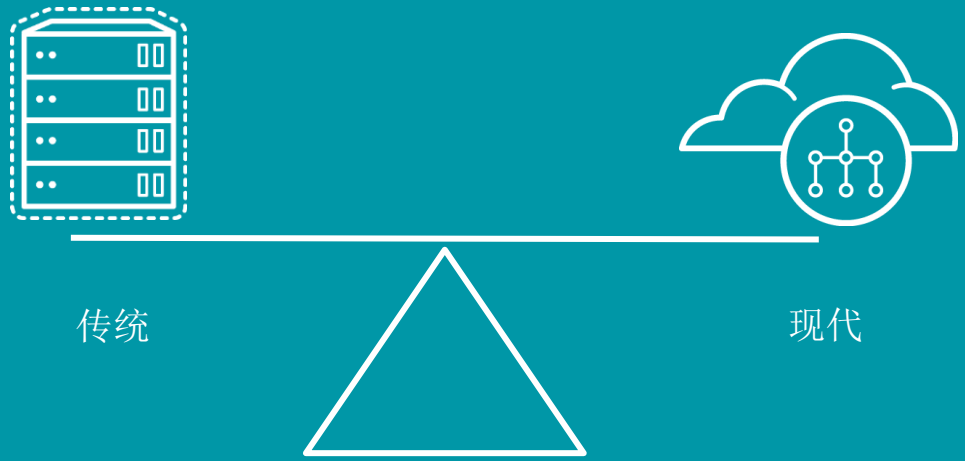
# 从云就绪到云原生再到无服务器架构

## ——看红帽云堆栈的演进之路

张家驹
红帽首席解决方案架构师

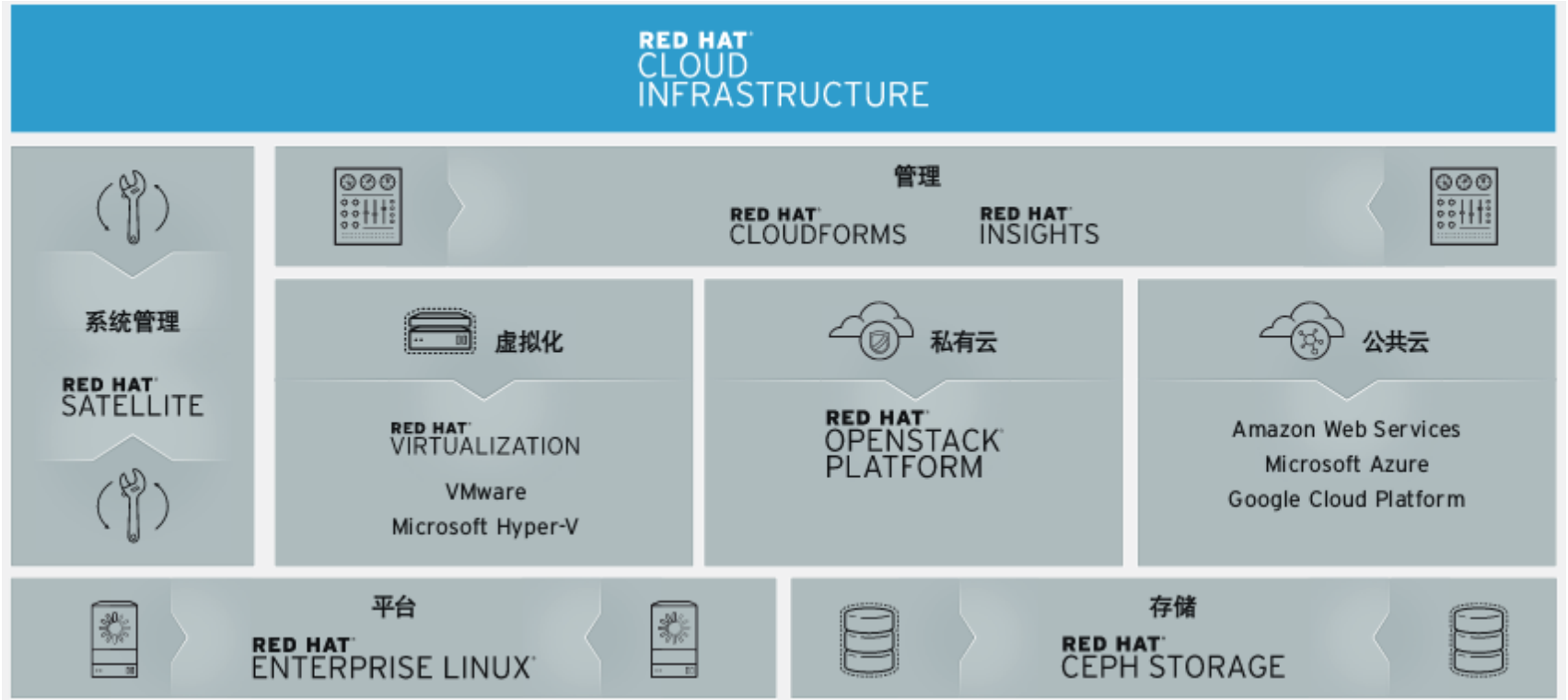
# 从虚拟化到云化

| 虚拟化 |
| --- |
| 大的、有状态的VM |
| 1个应用对应1～3个VM |
| VM生命周期以年来计算 |
| 纵向扩展 |
| 在基础架构层实现高可用 |

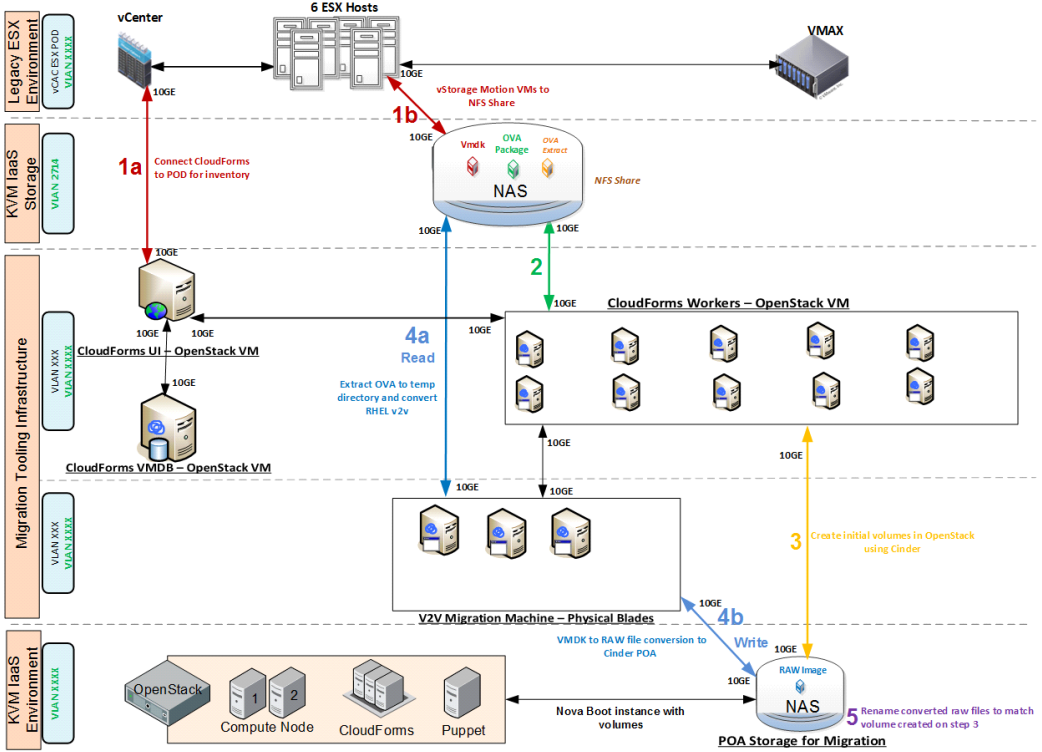



| 云化 |
| --- |
| 小的、无状态的实例 |
| 1个应用对应若干个实例 |
| 实例生命周期从几小时到几个月不等 |
| 横向扩展 |
| 在应用层实现高可用 |

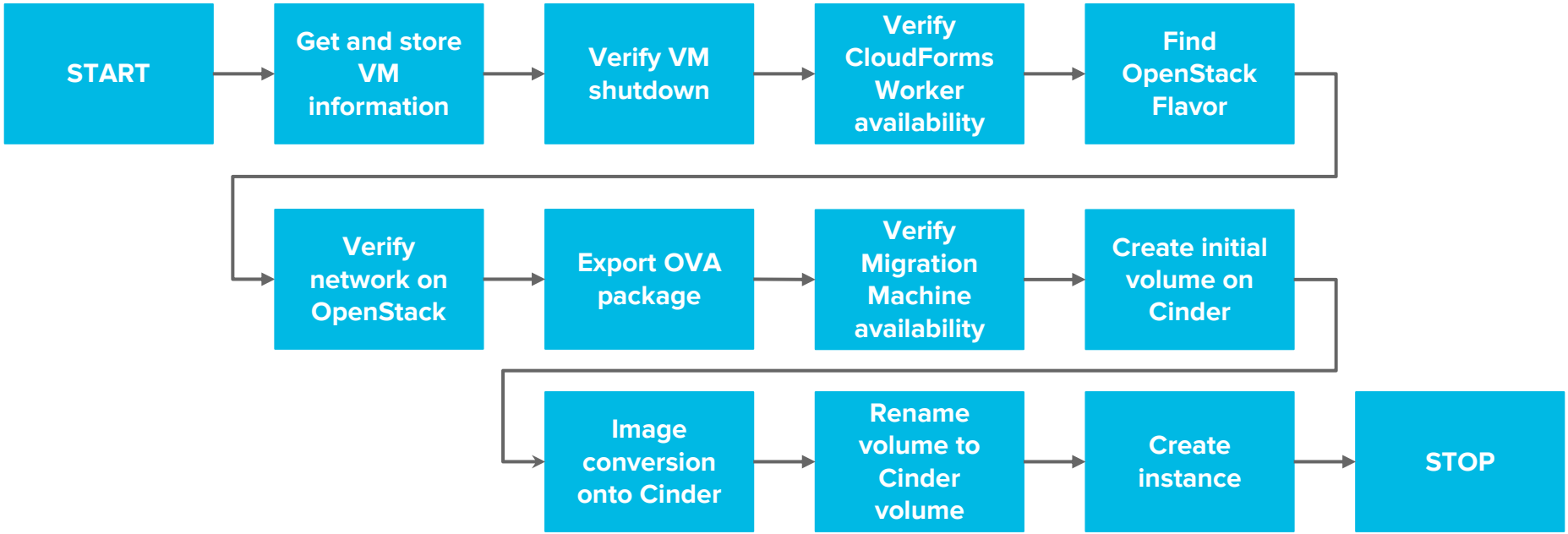
# 红帽IaaS云堆栈


RED HAT
CLOUD
INFRASTRUCTURE
管理
RED HAT
CLOUDFORMS
RED HAT
INSIGHTS
系统管理
RED HAT
SATELLITE
虚拟化
RED HAT
VIRTUALIZATION
VMware
Microsoft Hyper-V
私有云
RED HAT
OPENSTACK
PLATFORM
公共云
Amazon Web Services
Microsoft Azure
Google Cloud Platform
平台
RED HAT
ENTERPRISE LINUX
存储
RED HAT
CEPH STORAGE

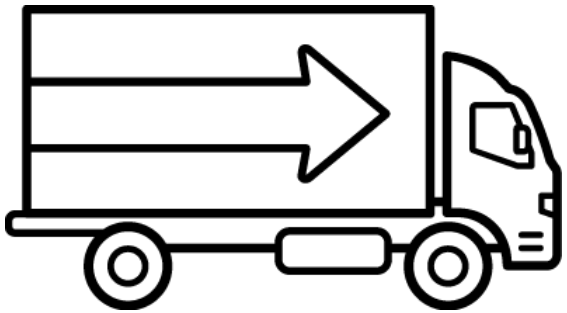
# 虚拟机的迁移

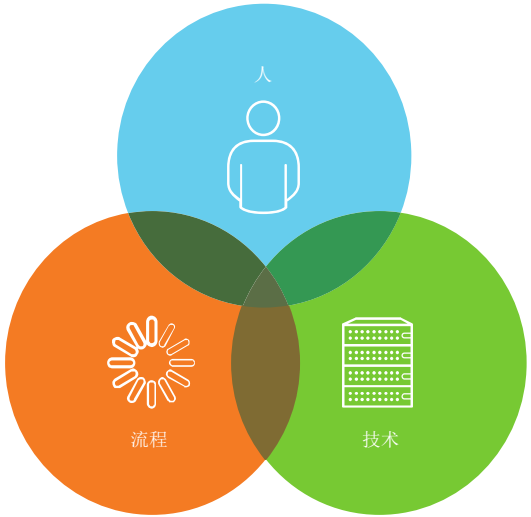

Legacy ESX Environment
vCAC ESX POD
VLAN XXXX
vCenter
6 ESX Hosts
VMAX
10GE
vStorage Motion VMs to NFS Share
1b
KVM IaaS Storage
VLAN 2714
1a
Connect CloudForms to POD for inventory
Vmdk
OVA Package
OVA Extract
NAS
NFS Share
2
Migration Tooling Infrastructure
VLAN XXX
VLAN XXXX
CloudForms UI – OpenStack VM
CloudForms Workers – OpenStack VM
4a
Read
Extract OVA to temp directory and convert RHEL v2v
CloudForms VMDB – OpenStack VM
3
Create initial volumes in OpenStack using Cinder
V2V Migration Machine – Physical Blades
4b
Write
VMDK to RAW file conversion to Cinder POA
KVM IaaS Environment
VLAN XXXX
OpenStack
Compute Node
CloudForms
Puppet
Nova Boot instance with volumes
RAW Image
NAS
5
Rename converted raw files to match volume created on step 3
POA Storage for Migration

# 迁移流程概览

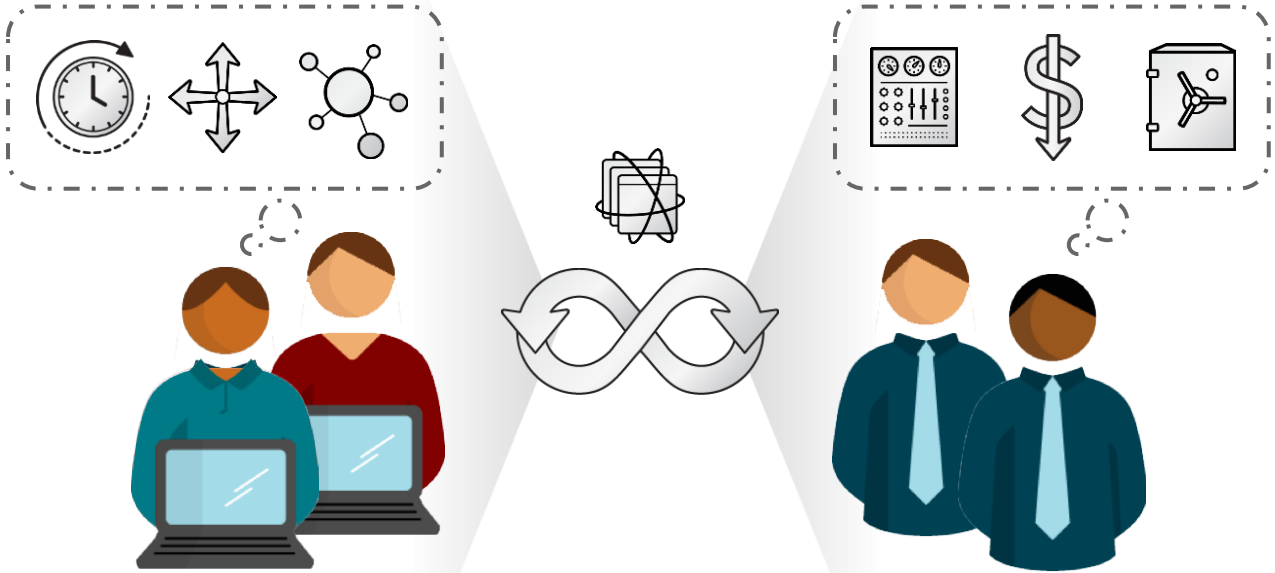

START
Get and store VM information
Verify VM shutdown
Verify CloudForms Worker availability
Find OpenStack Flavor
Verify network on OpenStack
Export OVA package
Verify Migration Machine availability
Create initial volume on Cinder
Image conversion onto Cinder
Rename volume to Cinder volume
Create instance
STOP

# 云就绪需要考虑多种因素





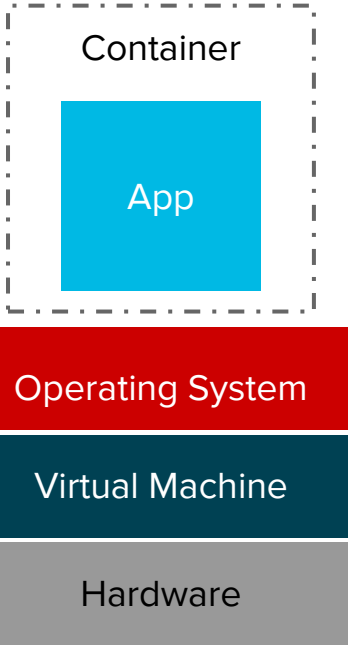
# 人，流程，技术 -> DevOps

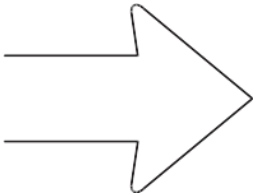


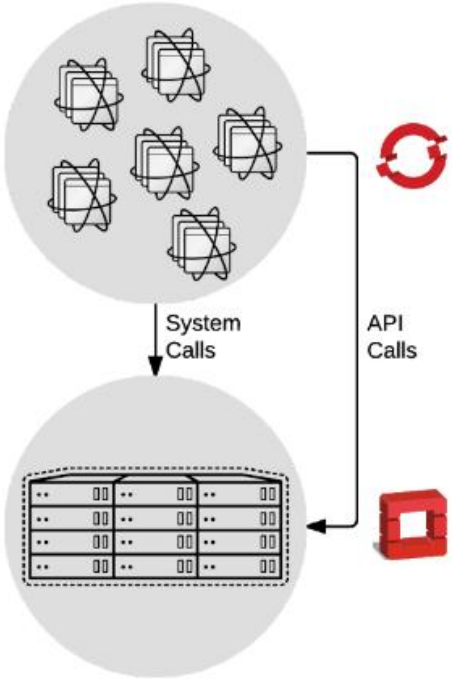
# 技术堆栈的演进

Controlled by Developers

Controlled by IT Operations





# 云原生应用的特点

## Service-based

- 架构模块化，松散耦合，高度内聚
- 提高开发的速度，
- 提高升级的速度，
- 面对不同的消费者的灵活性
- 可扩展性

## API

- 降低耦合性，
- 服务发布规约
- 服务规划灵活性
- 服务调度标准化
- 访问控制，QoS，计费
- 消费者分析

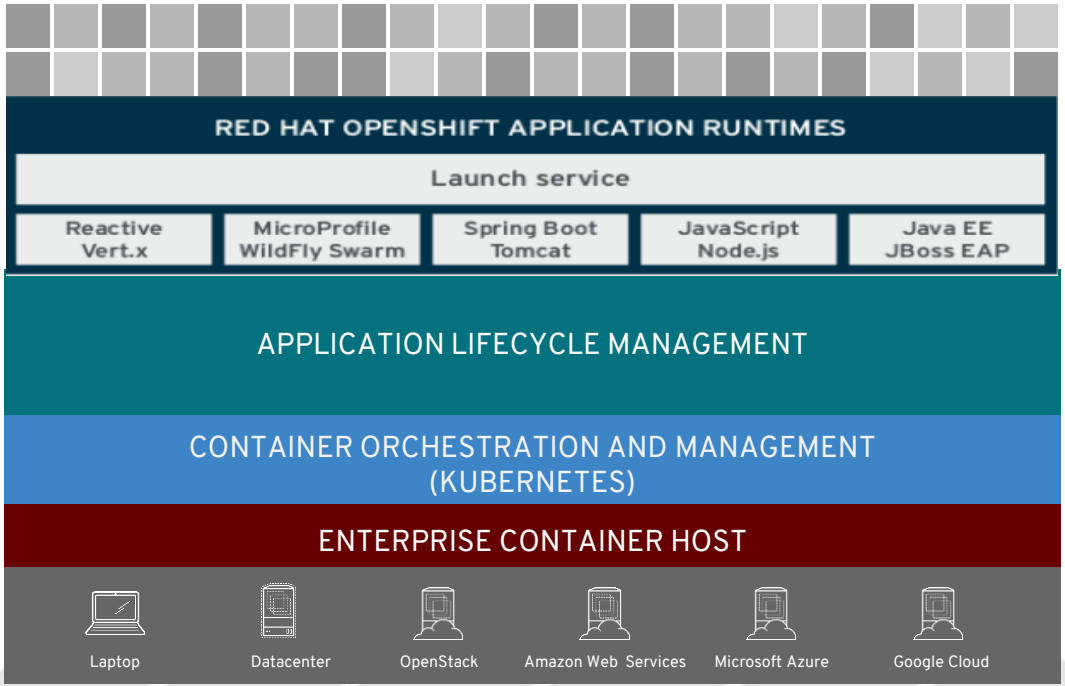
## Containers

- 标准化的容器调度
- 端到端 SDLC
- 加速 SDLC
- 一致性
- 更新和补丁
- 易回滚
- 有效的资源使用

## DevOps

- 自助服务
- 持续交付
- 自动化
- 敏捷
- 高效部署
- 提高质量
- 降低风险

# 红帽云原生应用开发堆栈

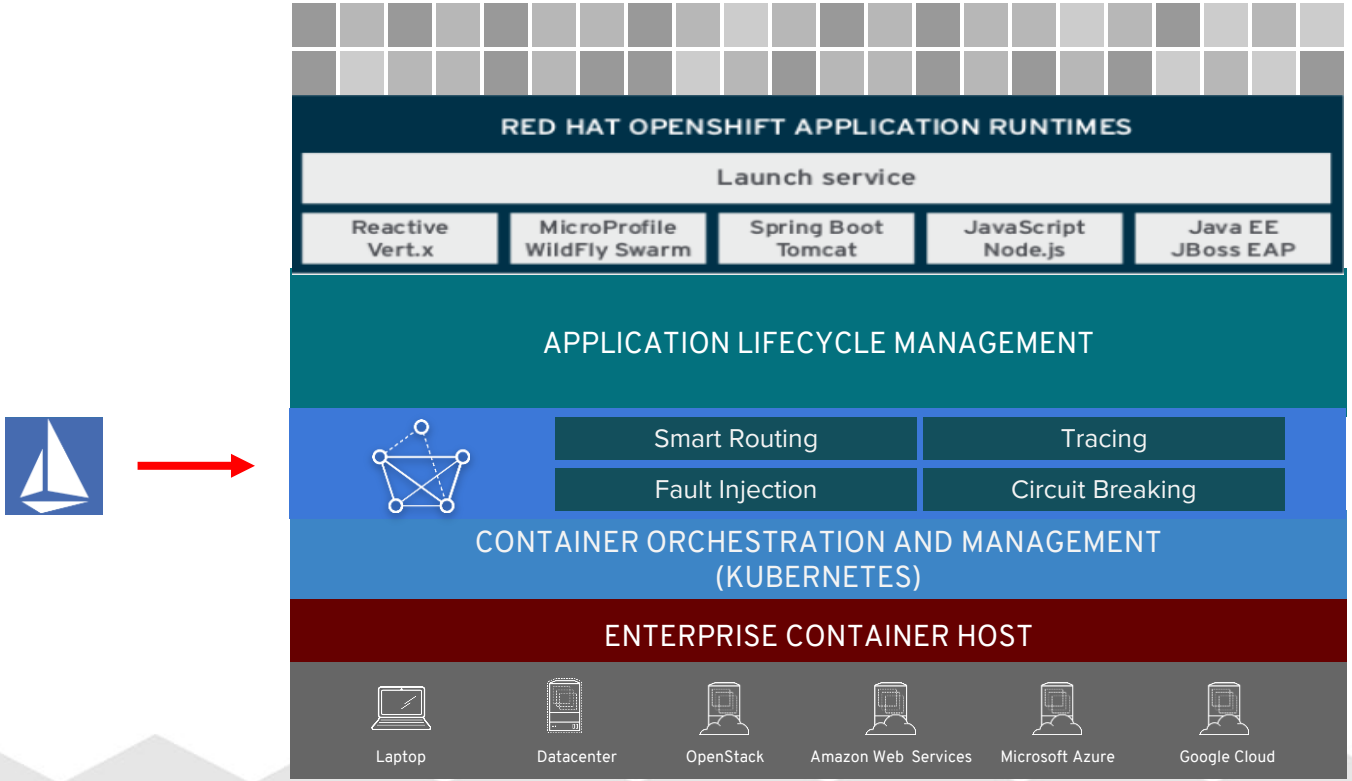



ANY APPLICATION

RED HAT® OPENSHIFT

ANY INFRASTRUCTURE

# 红帽云原生应用开发堆栈的演进




ANY APPLICATION

RED HAT® OPENSHIFT

ANY INFRASTRUCTURE

# 应用开发的简化

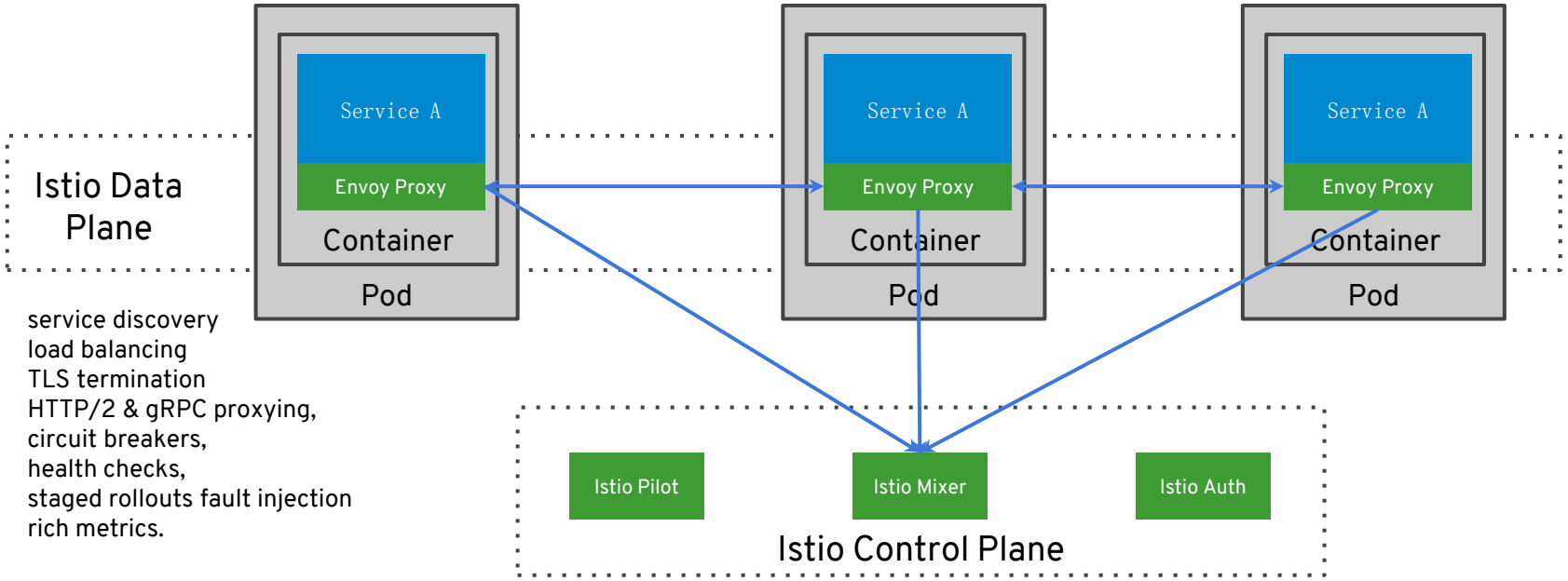

spring-cloud-netflix-hystrix
spring-cloud-netflix-zuul
spring-cloud-netflix-eureka-client
spring-cloud-netflix-ribbon
spring-cloud-netflix-atlas
spring-cloud-netflix-spectator
spring-cloud-netflix-hystrix-stream
...
@Enable....150MagicThings


org.wildfly.swarm.hystrix
org.wildfly.swarm.ribbon
org.wildfly.swarm.topology
org.wildfly.swarm.camel-zookeeper
org.wildfly.swarm.hystrix
org.wildfly.swarm.hystrix
...


vertx-circuit-breaker
vertx-service-discovery
vertx-dropwizard-metrics
Vertx-zipkin
...

+ Node.js
+ Go
+ Python
+ Ruby
+ Perl
+ ....

# ISTIO




- service discovery
- load balancing
- TLS termination
- HTTP/2 & gRPC proxying,
- circuit breakers,
- health checks,
- staged rollouts fault injection
- rich metrics.

- traffic mgmt
- discovery

- access control
- usage policies
- telemetry collection

- authentication
- policy enforcement
- Id & credentials

# 云原生应用开发的生态系统



## Red Hat portfolio
*Container-optimized*

| | | | |
|---|---|---|---|
| Web server | Caching | Business process | Node.js |
| API management | Single sign-on | Java EE | Mobile |
| Integration | Messaging | Storage | Real-time decision |

## Third-party ISVs
*Red Hat Container Catalog*

*15 YEARS AGO*
Certifying on RHEL

ORACLE
SAP
IBM

*TODAY*
Certifying on OpenShift+RHEL

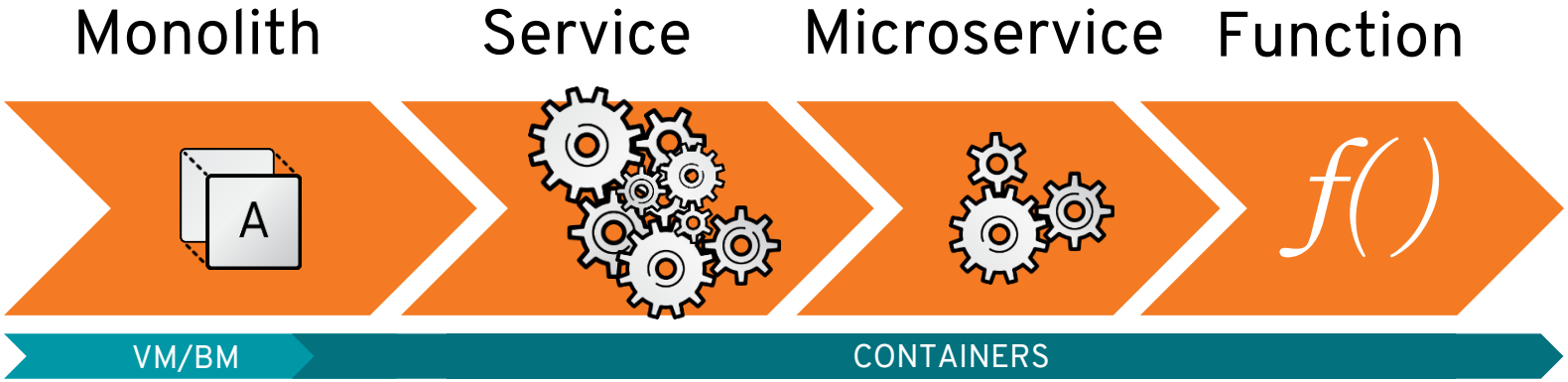
Microsoft*
SAP*
VERITAS
f5
New Relic.

## Cloud services
*Service Broker*

amazon web services
Microsoft Azure
Google Cloud Platform

RED HAT® OPENSHIFT

# 架构的持续演进




- 长时间运行
- 紧密耦合
- 有状态

- 自治性
- 松散耦合

- 单一目的
- 无状态
- 独立可扩展
- 自动化
- 易扩充的开发团队

- 单一功能
- 事件驱动

# 无服务器架构

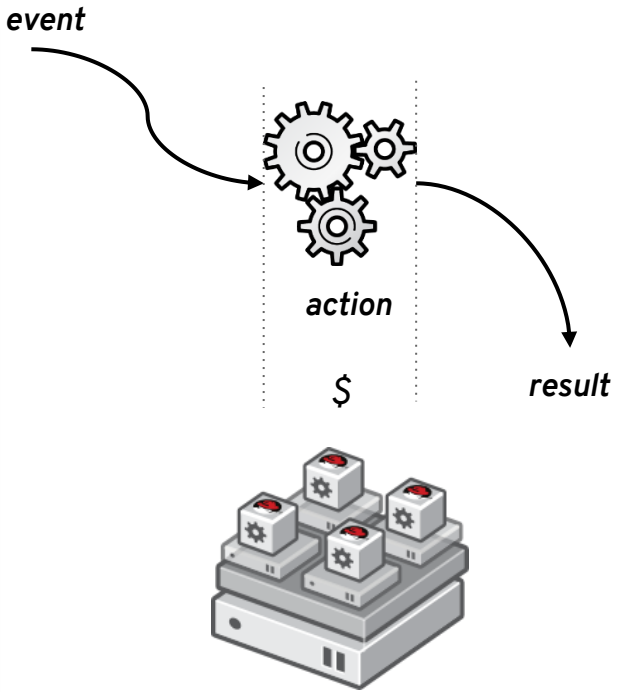
**Wikipedia 的定义:**

*“Serverless 是一种云计算执行模型，云计算提供商动态管理所有机器资源的分配。 定价基于应用程序消耗的实际资源量，而不是预先购买的资源容量。 这是一种更高效的计算形式。”*

**MartinFowler.com 的定义:**

*“应用大部分或完全依赖第三方云中运行的应用程序或服务来处理服务器端的状态或逻辑，这些第三方的应用程序或服务构成了一个庞大的生态系统，主要包括可通过云接入的数据库，服务等（后端即服务，Backend as a Service，BaaS）。”*

*“一部分服务器端逻辑仍由应用程序开发人员编写，但与传统体系架构不同的是，这些应用程序是在无状态计算容器中运行，基于事件触发，短生命周期的（可能仅仅被调用一次），并由第三方完全管理（函数即服务，Function as a Service，FaaS）。”*



# 无服务器架构的技术选型

| 项目 | 是否开源 | 是否支持 Kubernetes | 社区规模 | 评分 | 启动时间 |
|---|---|---|---|---|---|
| Apache OpenWhisk | Yes | Yes | Large | ★★★★ | 2015 |
| Fission | Yes | Yes | Small | ★★ | 2016 |
| Funktion | Yes | Yes | Tiny | ★★ | 2017 |
| Project Riff | Yes | Yes | Tiny | ★★ | Late 2017 |
| Amazon Lambda | No | No | Large | ★★★★ | 2014 |
| Azure Functions | No | No | Small | ★★★ | Late 2016 |
| Google Cloud Functions *(beta)* | No | No | Small | ★★★ | 2016 |

# 红帽无服务器架构解决方案

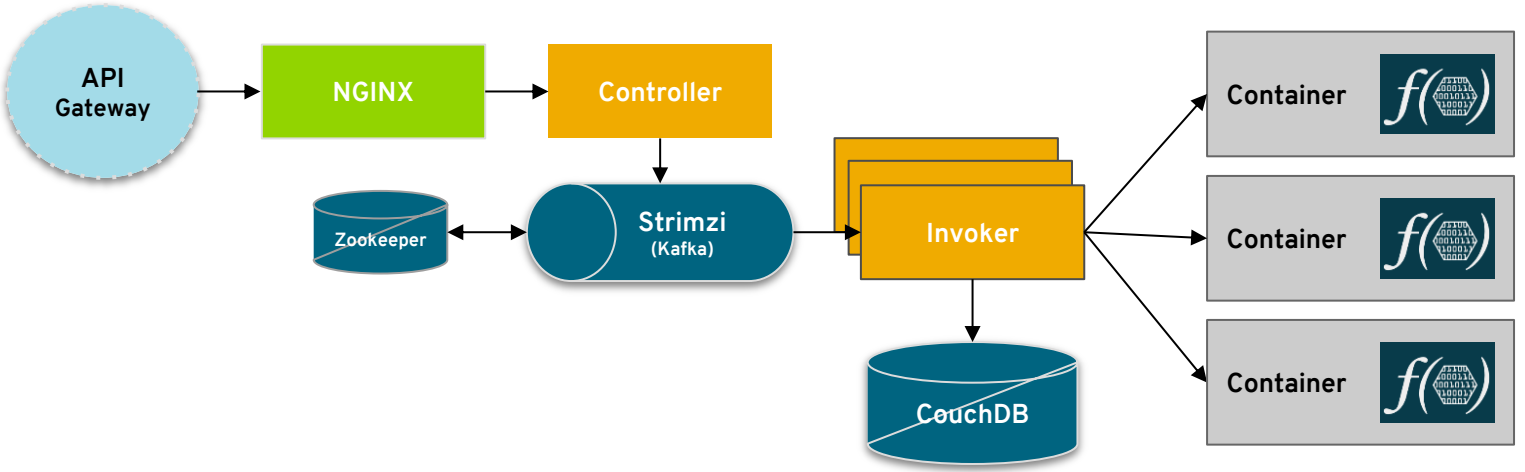
APACHE OpenWhisk™

RED HAT® OPENSHIFT CLOUD FUNCTIONS

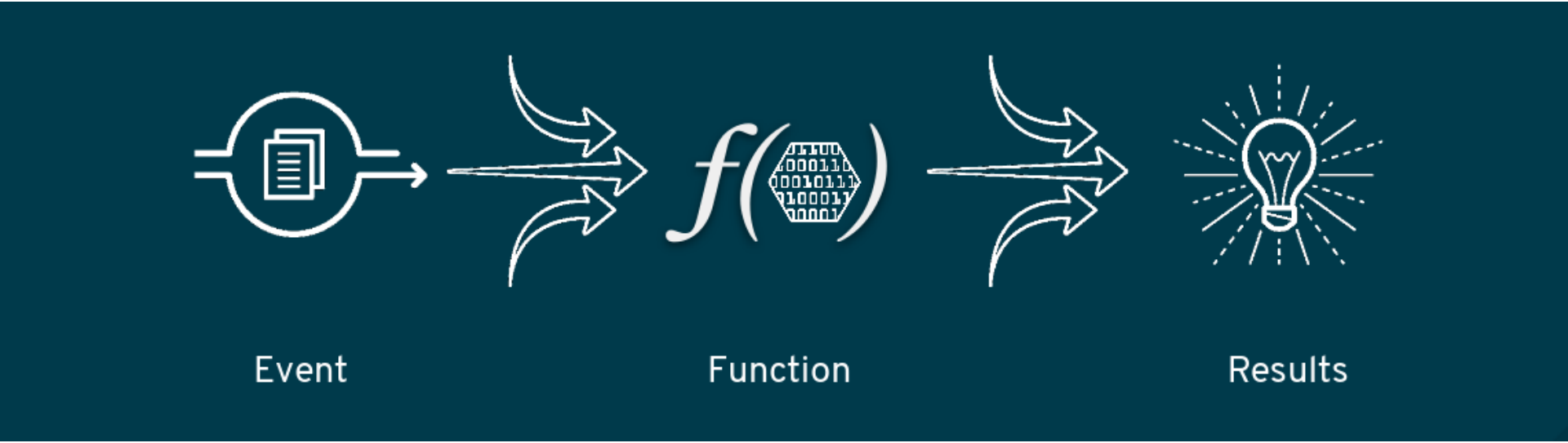
- IBM 最早发起
  - 红帽, IBM, Adobe
- 在 Apache 软件组织下孵化
- 可在任何地方运行
- 无规定的平台
- 活跃的社区

- 企业级就绪
- 基于 OpenShift 优化定制
- 整合到云红帽产品线
- 红帽提供完整的支持
- 在 OpenShift Online 和 OCP 中使用
- 与在线开发工具集成

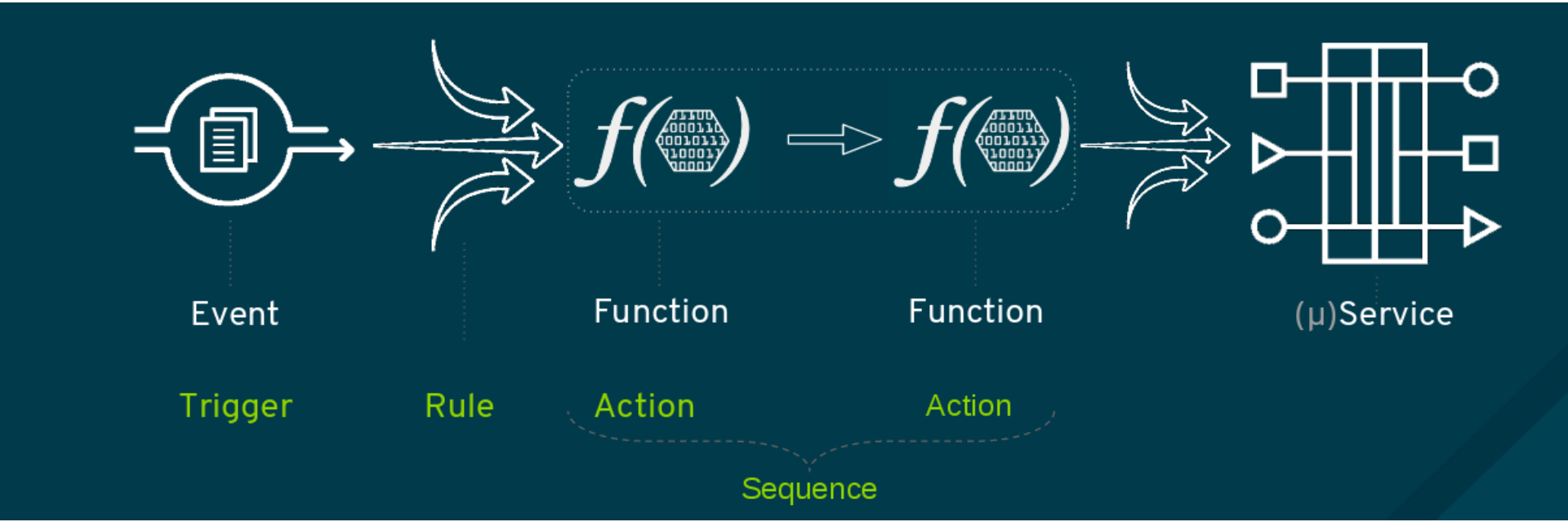
# OpenWhisk概览



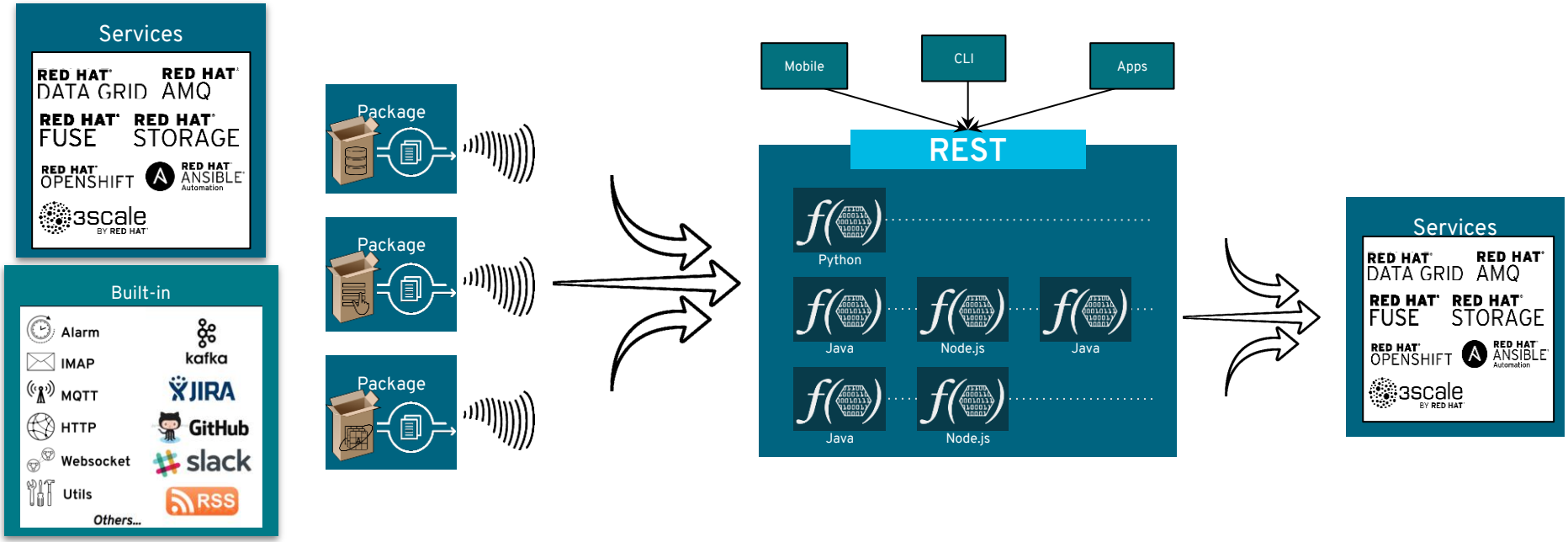
# 无服务器架构的基本原理



# OpenShift Cloud Functions的工作流性

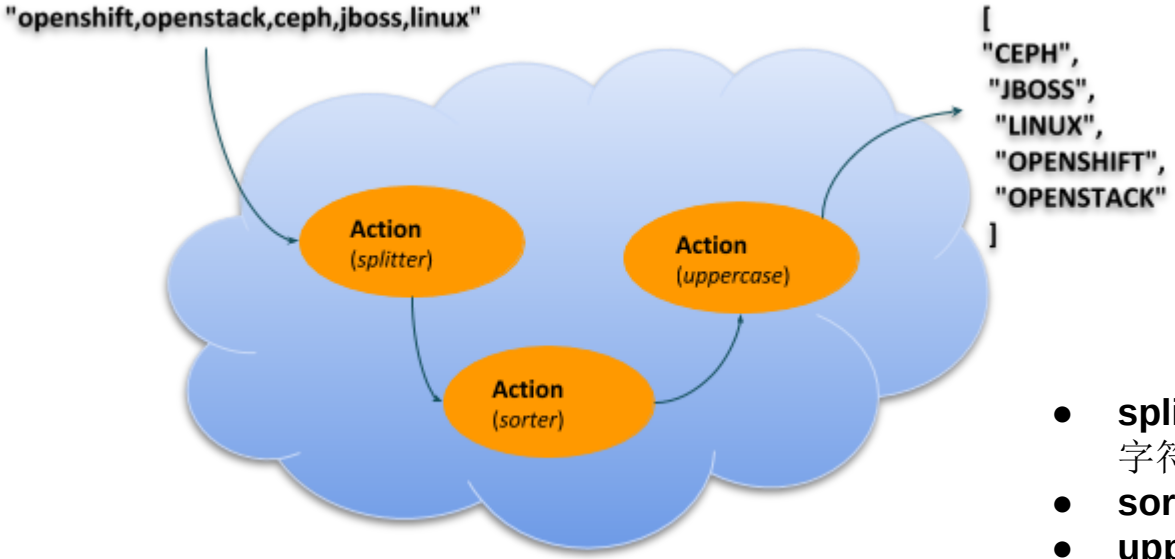


# OpenShift Cloud Functions的使用方法



# OpenShift Cloud Functions示例（1）

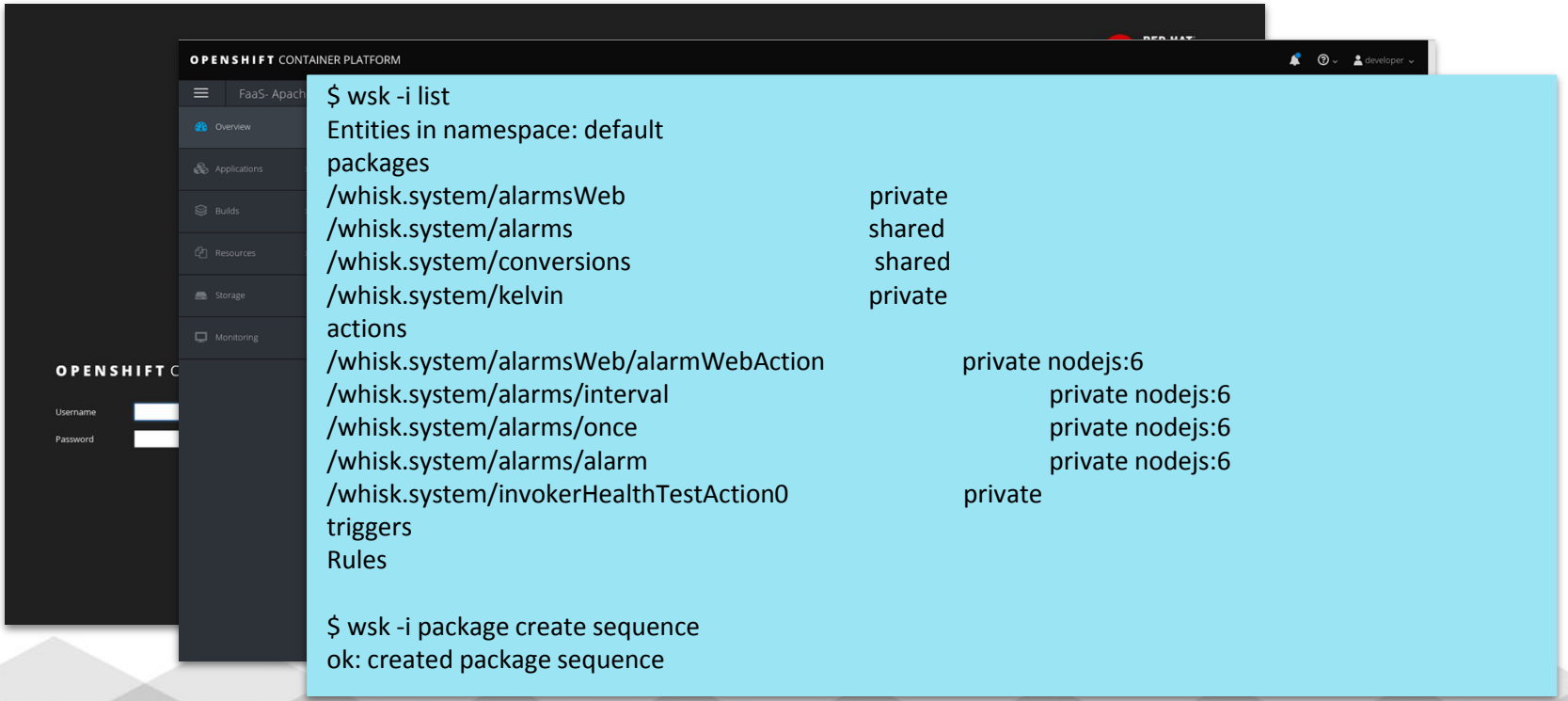
示例说明




- **splitter** - java, 根据规则拆分字符串成数组
- **sorter** - python，数组排序
- **uppercase** - nodejs，数组中的字符转化为大写字母

# OpenShift Cloud Functions示例（2）

环境准备




```
$ wsk -i list
Entities in namespace: default
packages
/whisk.system/alarmsWeb                          private
/whisk.system/alarms                             shared
/whisk.system/conversions                         shared
/whisk.system/kelvin                             private
actions
/whisk.system/alarmsWeb/alarmWebAction                private nodejs:6
/whisk.system/alarms/interval                                 private nodejs:6
/whisk.system/alarms/once                                     private nodejs:6
/whisk.system/alarms/alarm                                    private nodejs:6
/whisk.system/invokerHealthTestAction0                private
triggers
Rules

$ wsk -i package create sequence
ok: created package sequence
```

# OpenShift Cloud Functions示例（3）

部署 splitter

```
import com.google.gson.JsonArray;
import com.google.gson.JsonObject;

public class App {

    public static JsonObject main(JsonObject args) {
        JsonObject response = new JsonObject();
        String text = null;
        if (args.has("text")) {
            text = args.getAsJsonPrimitive("text").getAsString();
        }
        String[] results = new String[] { text };
        if (text != null && text.indexOf(",") != -1) {
            results = text.split(",");
        }
        JsonArray splitStrings = new JsonArray();
        for (String var : results) {
            splitStrings.add(var);
        }
        response.add("result", splitStrings);
        return response;
    }

}
```


```
$ mvn clean install

$ wsk -i action create --web=true sequence/splitter target/splitter-1.0.jar --main
com.sample.App

$ wsk -i action get sequence/splitter --url
https://openwhisk-
faas.192.168.42.102.nip.io/api/v1/web/whisk.system/sequence/splitter

$ wsk -i action invoke sequence/splitter --result --param text
"openshift,openstack,ceph,jboss,linux"
{
                    "result": ["openshift", "openstack", "ceph","jboss","linux"]
}

$ curl -k https://openwhisk-
faas.192.168.42.102.nip.io/api/v1/web/whisk.system/sequence/splitter.json?text="ope
nshift,openstack,ceph,jboss,linux"
{
  "result": ["openshift", "openstack", "ceph", "jboss", "linux"]
}
```

# OpenShift Cloud Functions示例（4）

部署 sorter

```
def main(args):
    return {"result": sorted(args["result"]) }
```


```
$ wsk -i action create --web=true sequence/sorter sorter.py
ok: created action sequence/sorter

$ wsk -i action list | grep sequence
/whisk.system/sequence/sorter                                   private python:2
/whisk.system/sequence/splitter                                 private java

$ wsk -i action invoke sequence/sorter --result --param-file ./split.json
{
                "result": [
                "ceph",
                "jboss",
                "linux",
                "openshift",
                "openstack"
                ]
}
```

# OpenShift Cloud Functions示例（5）

部署 uppercase

```
function main(args) {
    return {"result": args["result"].map(s => s.toUpperCase()) }
 }
```

```
$ wsk -i action create sequence/uppercase uppercase.js
ok: created action sequence/uppercase

$ wsk -i action list | grep sequence
/whisk.system/sequence/uppercase                    private nodejs:6
/whisk.system/sequence/sorter                       private python:2
/whisk.system/sequence/splitter                     private java

$ wsk -i action invoke sequence/uppercase --result --param-file ./sorted.json
{
                "result": [
                "CEPH",
                "JBOSS",
                "LINUX",
                "OPENSHIFT",
                "OPENSTACK"
                ]
}
```

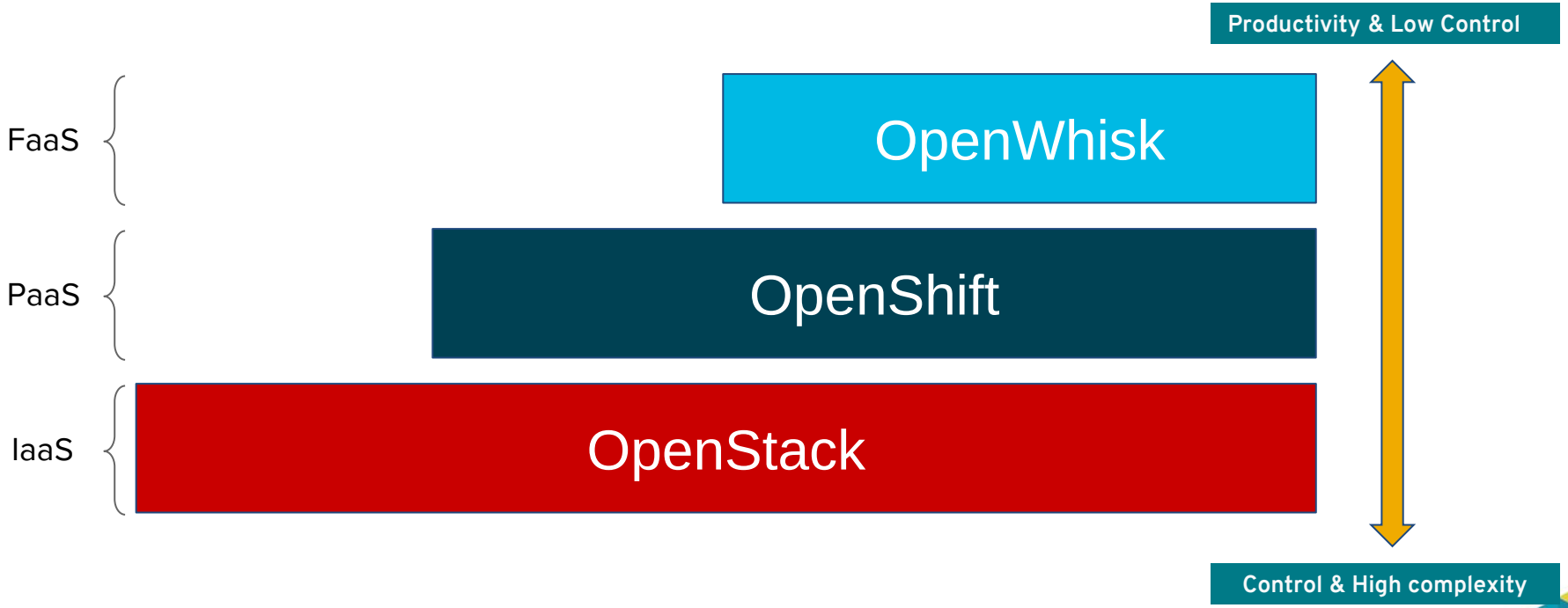
# OpenShift Cloud Functions示例（6）

创建 sequence, 并测试

```
$ wsk -i action create --web=true strings --sequence sequence/splitter,sequence/sorter,sequence/uppercase
$ wsk -i action list | grep strings
/whisk.system/strings                                                    private sequence
$ wsk -i action invoke strings --result --param text "openshift,openstack,ceph,jboss,linux"
{
                "result": [
                "CEPH",
                "JBOSS",
                "LINUX",
                "OPENSHIFT",
                "OPENSTACK"
                ]
}

$ wsk -i action get strings --url
https://openwhisk-faas.192.168.42.102.nip.io/api/v1/web/whisk.system/default/strings
$ curl -k https://openwhisk-
faas.192.168.42.102.nip.io/api/v1/web/whisk.system/default/strings.json?text="openshift,openstack,ceph,jbos
s,linux"
{
  "result": ["CEPH", "JBOSS", "LINUX", "OPENSHIFT", "OPENSTACK"]
}
```

# IaaS -> PaaS -> FaaS



CHINA
OpenInfra Days


# Thank You