

探索  
现代云开发

# 微服务和甲骨文无服务器计算

胡平

甲骨文云专家团队  
资深销售顾问

**QCon**  
全球软件开发大会

x

ORACLE  
甲骨文

# 现代化应用开发演变

最近15年的历史告诉我们一切都是围绕提高速度和敏捷性



# 什么是微服务?

可以独立部署，且相互之间可以通过交互实现更广使用场景的最小化功能服务

## 单体应用

- 单独、单体化应用
- 必须部署整套应用
- 每个应用对应单独数据存储
- 本地进程调用, 外部SOAP调用
- 以技术层来组织应用
- 开发者不做运维
- 每个运行实例来保存状态信息
- 整个应用采用同一个技术框架



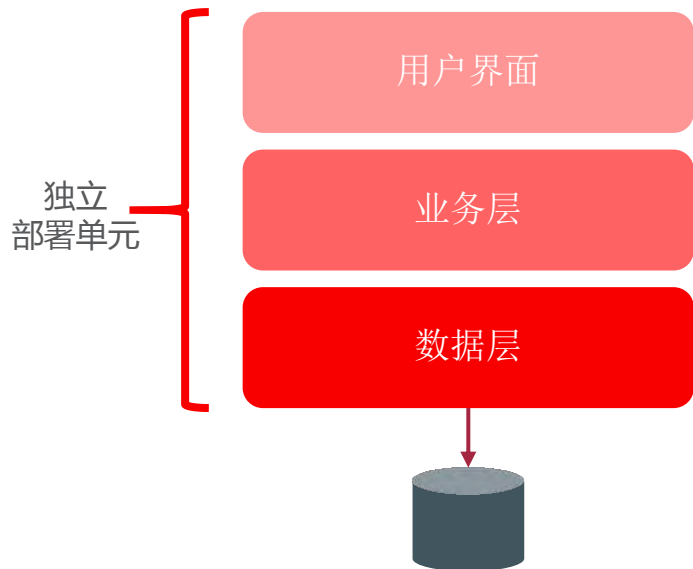
## 现代化微服务

- 更多、更小的最小化功能微服务
- 每个微服务可以单独部署
- 每个微服务都有自己的数据存储
- 基于REST的调用, 消息或者二进制数据
- 以业务能力来组织应用
- 开发和运营是结合在一起永恒地支撑生产
- 状态信息是在分布式数据网格- 实例是无状态的
- 每个微服务都可以选择不同的技术框架

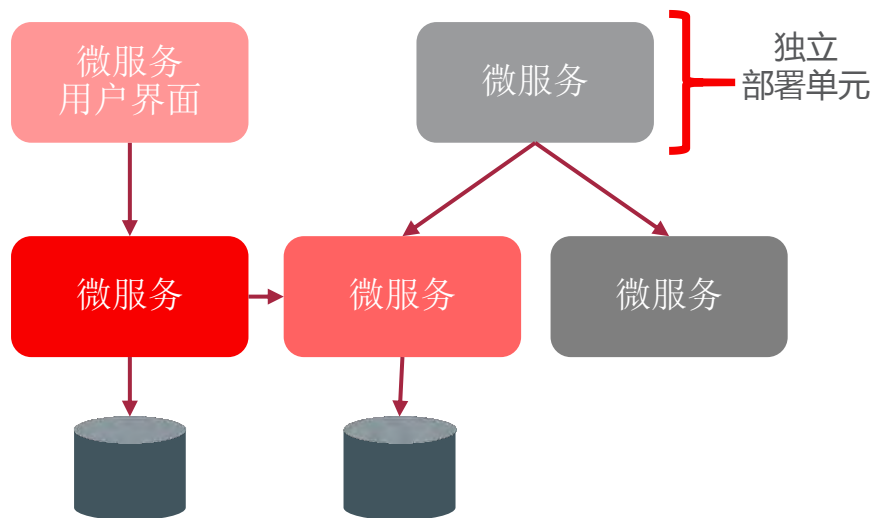


# 单体架构和微服务架构比较

## 单体架构



## 微服务架构

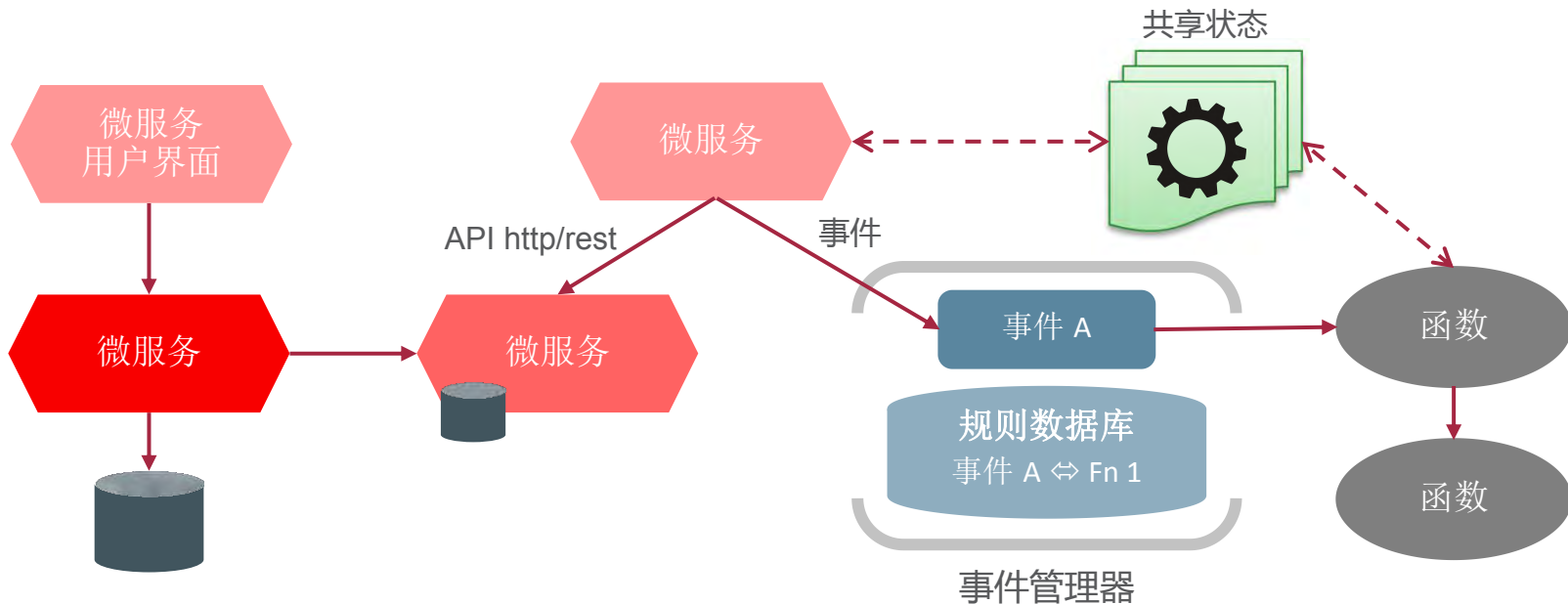


服务间可以采用同步API或基于事件异步的通讯方式



# 愿景：统一的“服务”和“函数”

使用服务和函数构建复合应用程序，共享状态和事件触发器





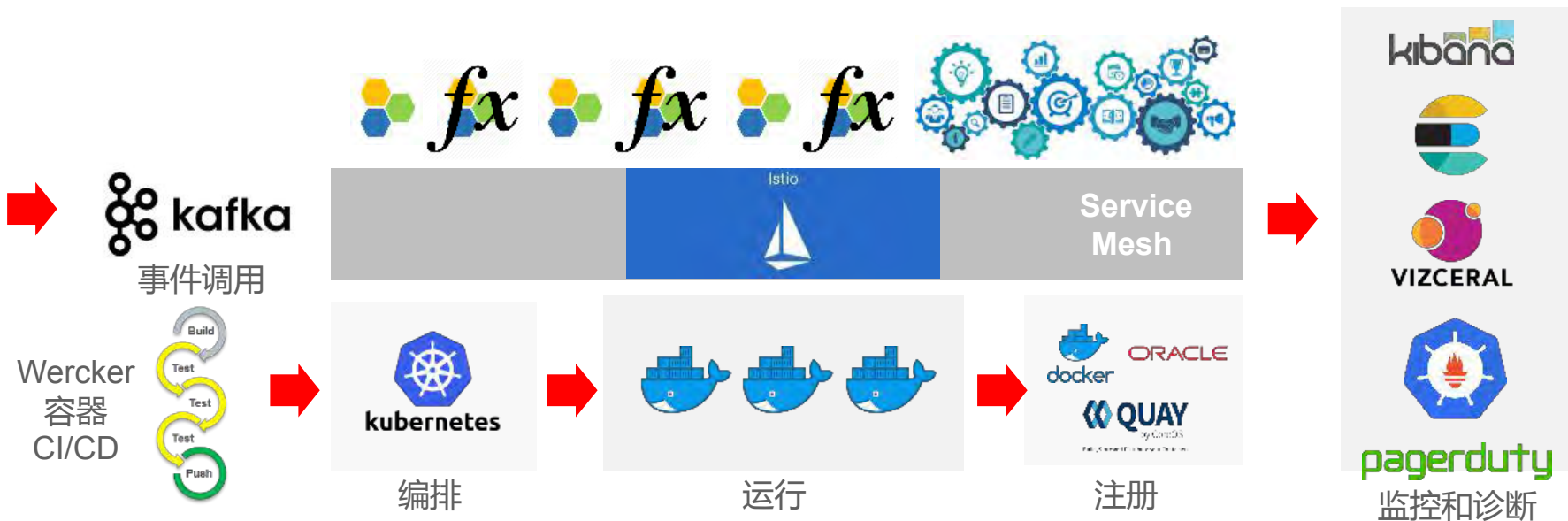
# 甲骨文云微服务和无服务器解决方案

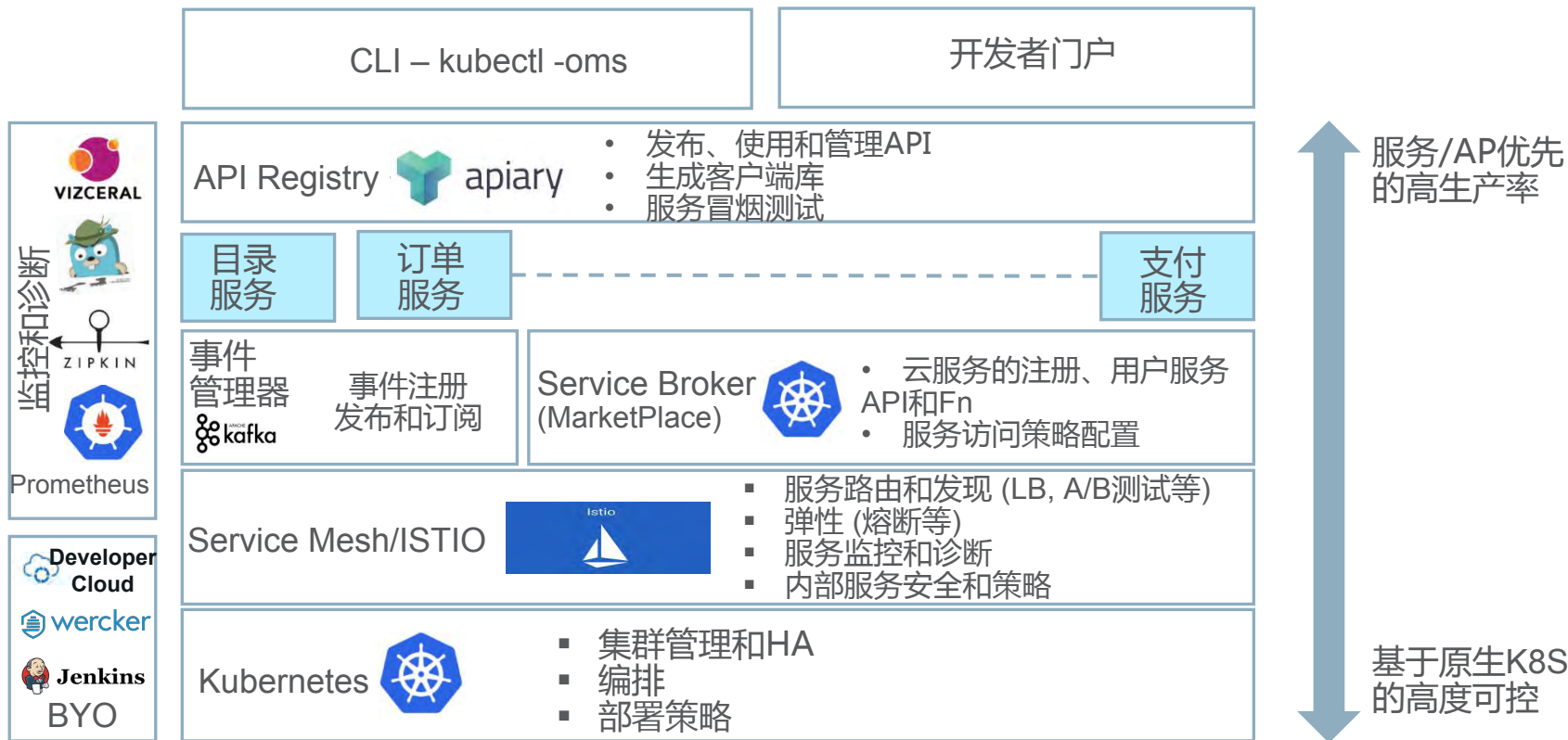
QCon  
全球软件开发大会

x

ORACLE  
甲骨文

完整现代化应用开发平台，同时支持微服务和函数的混合应用场景，从微服务应用和函数自动化CI/CD DevOps，到基于底层K8S的容器编排和调度，基于Istio微服务的实现，最后到集成开源监控框架实现应用端到端监控和诊断，整个容器原生应用方案全部基于PaaS方式实现，让开发者更关注于业务逻辑本身。





预集成的服务网络 ( Istio ) 和其他CNCF组件可以透明地处理微服务底层问题

自动的服务注册和发现

动态的负载均衡和健康检查

服务间的通讯安全

Service  
Mesh

基于熔断、超时和重试的弹性

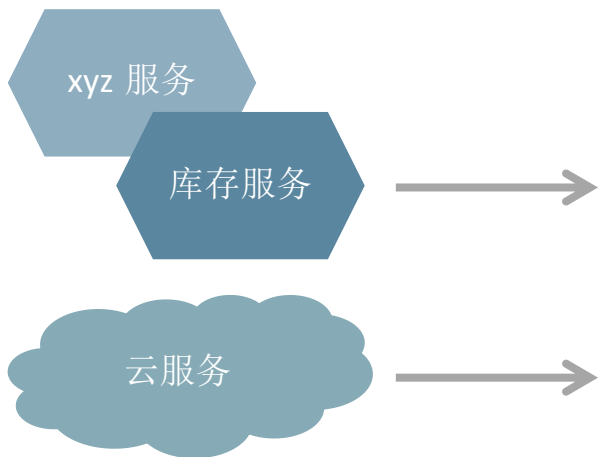
诊断数据自动发送到运维工具

金丝雀发布、A / B测试的动态路由



解耦的，灵活的、基于规则的异步通信基础架构

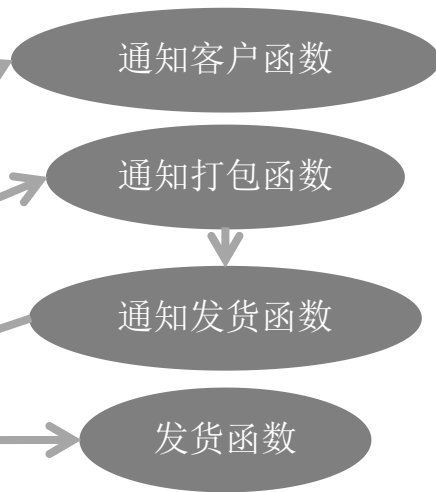
发布事件的  
微服务和云服务  
事件提供者



规则数据库将  
事件映射到消费者  
事件管理器



事件触发功能  
(单个/连锁)  
事件消费者

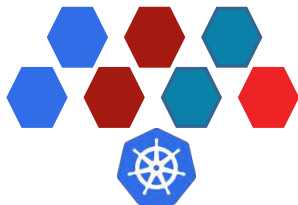
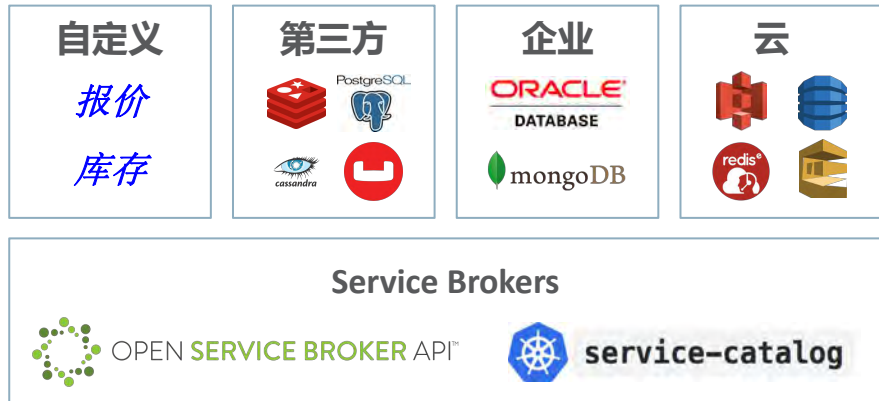


# Service Broker

一致的、透明的、安全的和开放的

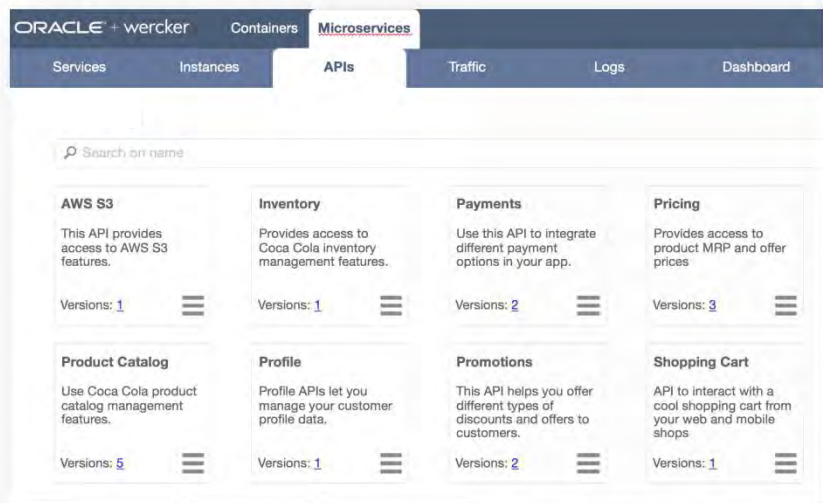
QCon  
全球软件开发大会

ORACLE  
甲骨文



- 一致的访问服务方式（自定义，第三方，企业，云）
- 位置透明 - 服务可以在任何地方运行，群集或单节点，企业或云，外部或本地
- 安全管理 - 与服务实例共享的凭证
- 开放 - 基于Open Service Broker API 和Kubernetes服务目录项目，甲骨文将自己的修改反馈给Kubernetes项目

## 所有资源的一站式服务 - API, 事件和流



## 集成的API开发和管理

- 快速设计, 构建和测试API
- 开放API和API蓝图格式
- 客户和服务端代码生成器
- 与所有微服务平台组件完全集成
- 安全与策略驱动的API
- 获取API使用情况洞察



# 更匹配应用开发者关注点的运维管理

QCon  
全球软件开发大会

ORACLE  
甲骨文

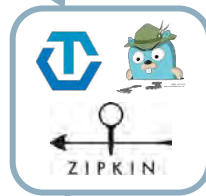
将应用作为逻辑服务/ API /数据流来操作，只有在需要时才深入到Pod/容器/基础设施

运维就绪 - 预先连接，只需部署就可以开始监控



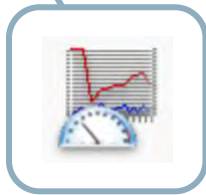
直观的体验 - 实时和历史监控

系统工具 - 深入到Pod/容器/基础设施



可追溯性 - 内置分布式跟踪支持

集成的JVM诊断 - 具有自我调节功能



日志 - 与跟踪和诊断相关的视图

# 多语言编程模型

支持多种语言和数据库，Java的高级体验

QCon  
全球软件开发大会

x

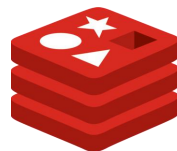
ORACLE  
甲骨文



+ more ...



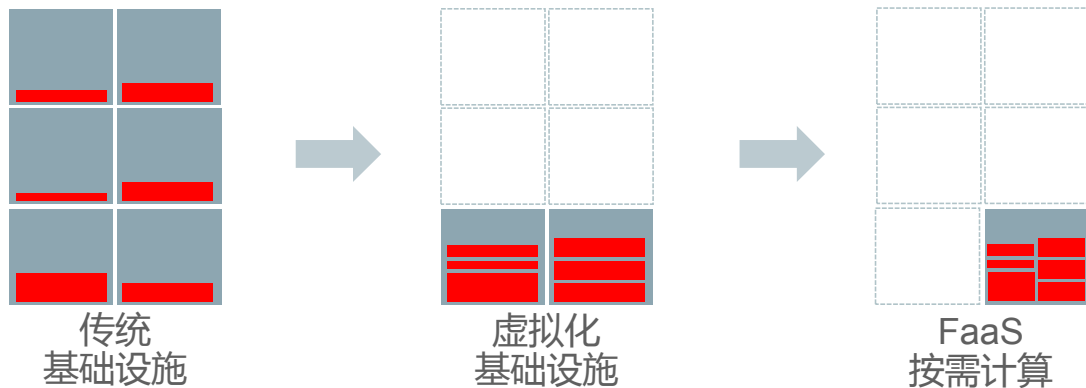
PostgreSQL



+ more ...

# 什么是无服务器？

- 云服务中对底层基础设施抽象类别，以便开发人员不用考虑服务器，虚拟机和其他IaaS组件
- 函数即服务（FaaS）是无服务器体系结构应用程序编程的一种模型，一般与数据服务结合使用
- 提高开发人员的生产力和灵活性，同时降低成本
- 服务按功能扩展，客户不为空闲资源付费



- **开源**—无厂商锁定
- **平台独立**—笔记本, 服务器, 云
- **可用性**—初学者简单入门, 降低高阶者的学习成本
- **基于容器**—融入容器生态圈
- **编排独立**—可以部署到Kubernetes、Swarm、Mesos或其他编排框架

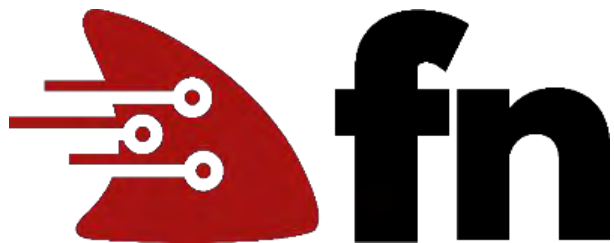


甲骨文开源Fn project

QCon  
全球软件开发大会

x

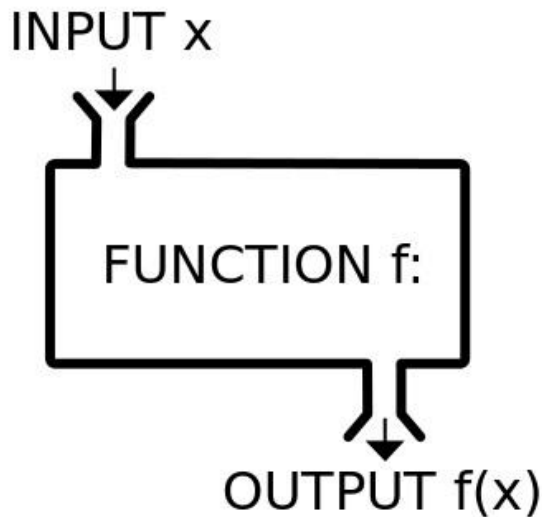
ORACLE  
甲骨文



[www.fnproject.io](http://www.fnproject.io)



- 将一小块代码包装到具有依赖关系的容器镜像中
  - 通过STDIN和环境变量获取输入
  - 生成输出到STDOUT
  - 错误记录到STDERR
- 
- Fn服务器处理其他所有事情，比如API网关，管道处理，存储日志等。

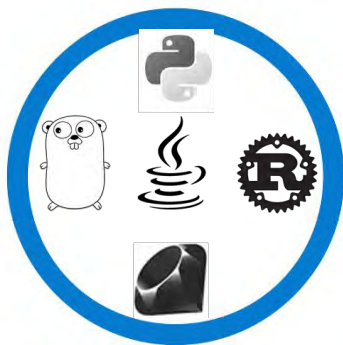




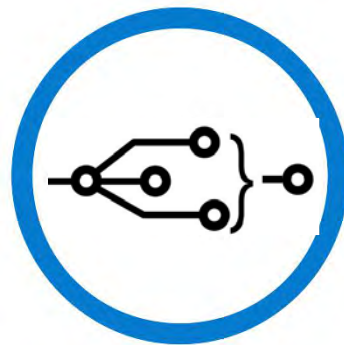
# 甲骨文Fn project核心组件



Fn Server



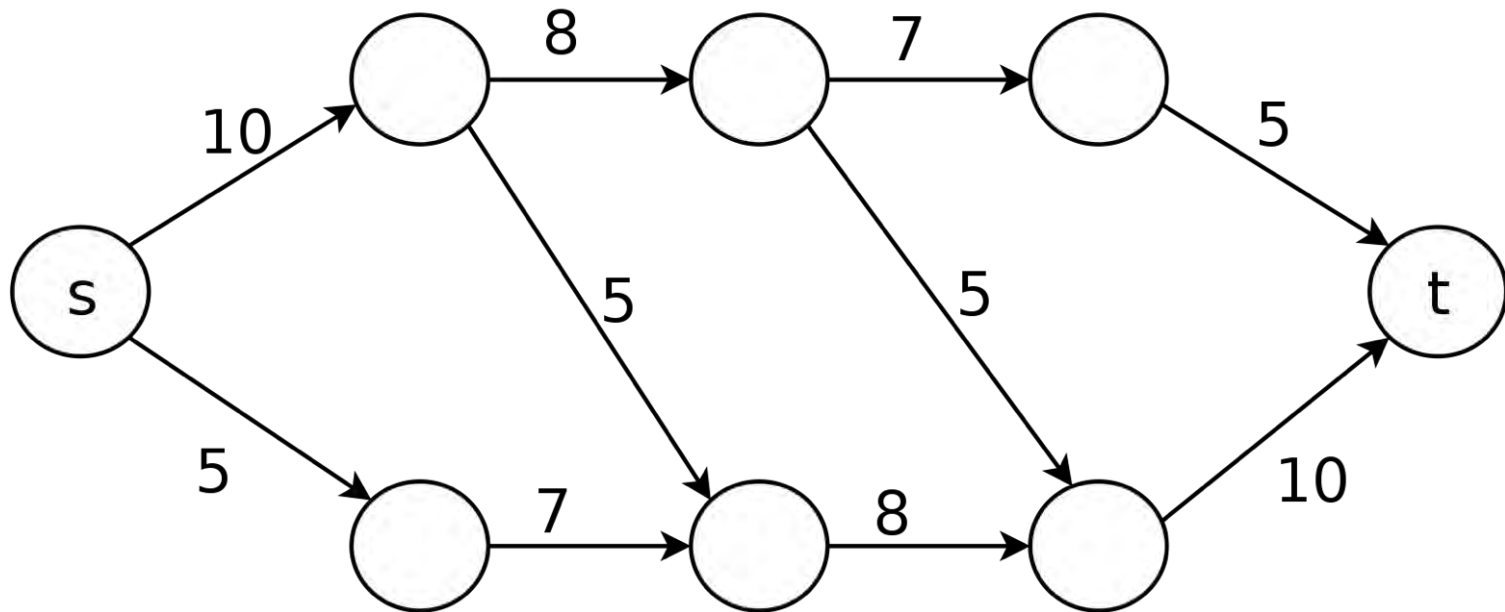
FDKs



Fn  
Flow

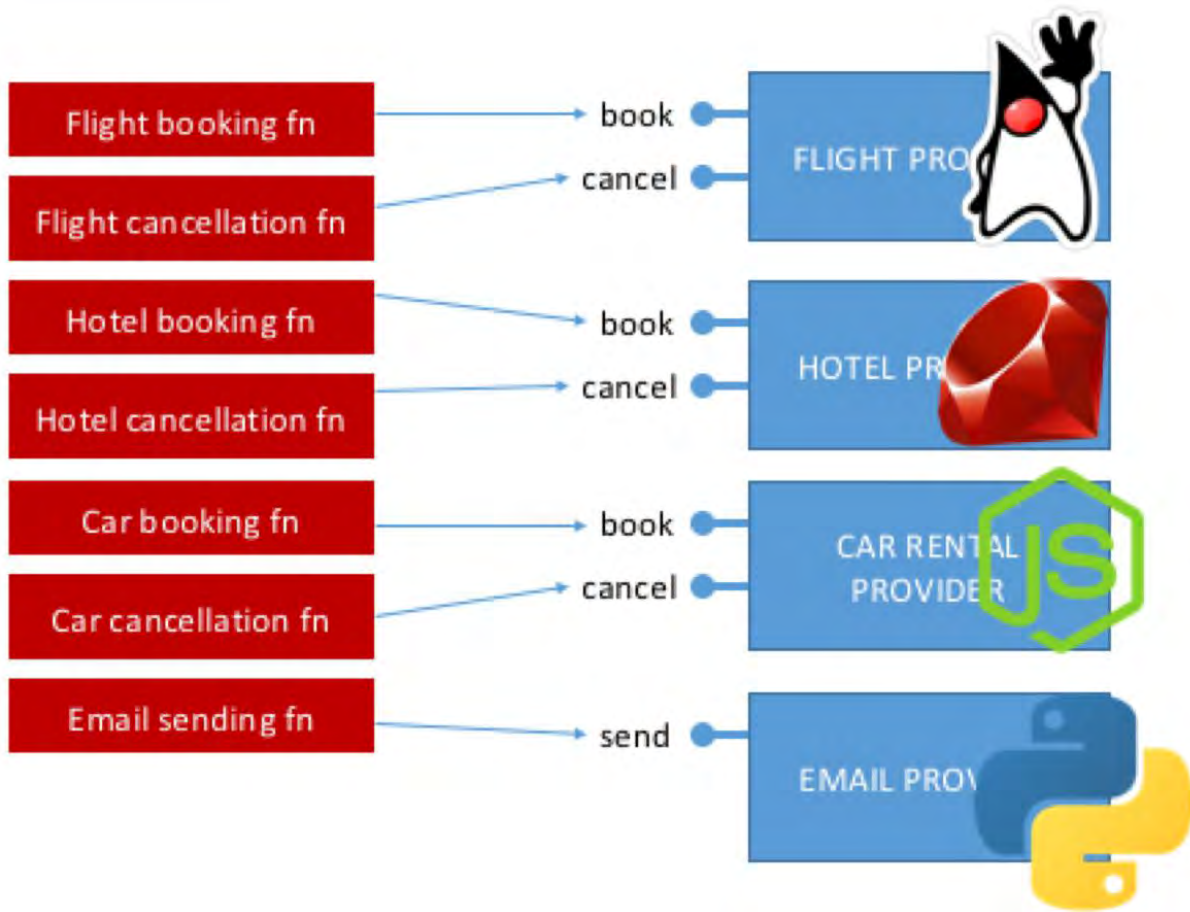


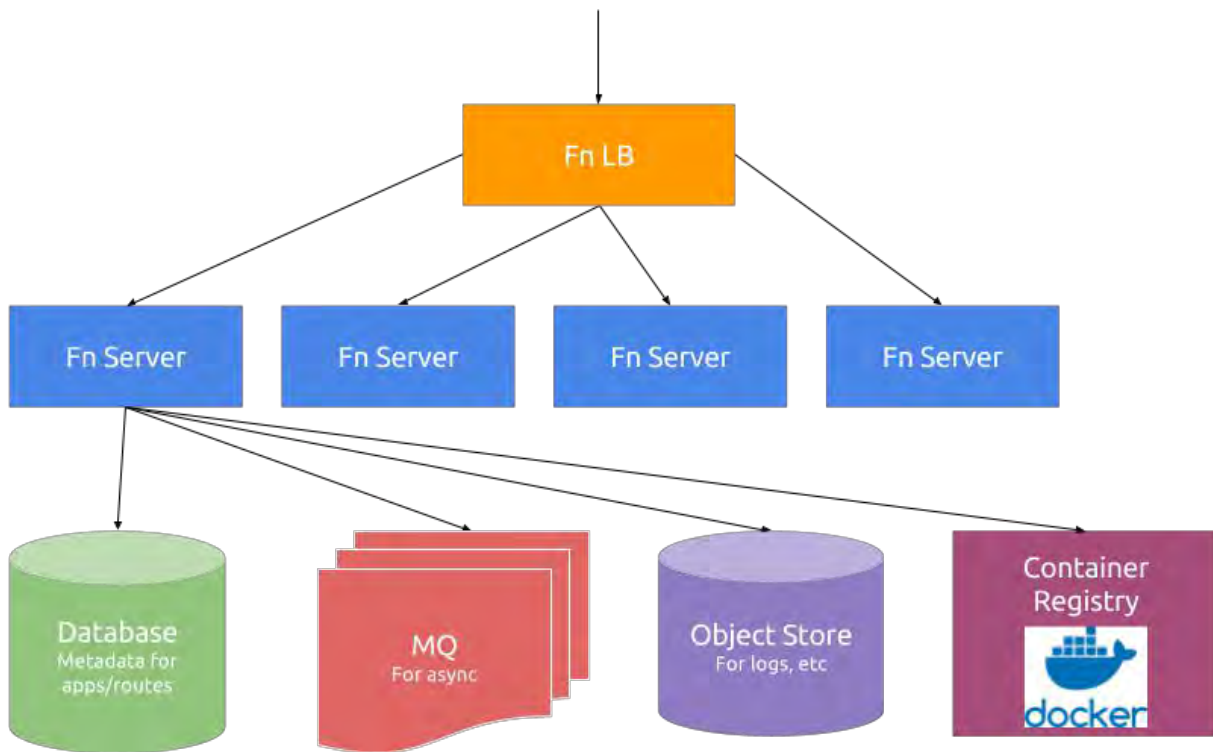
# Fn Flow – 函数组合





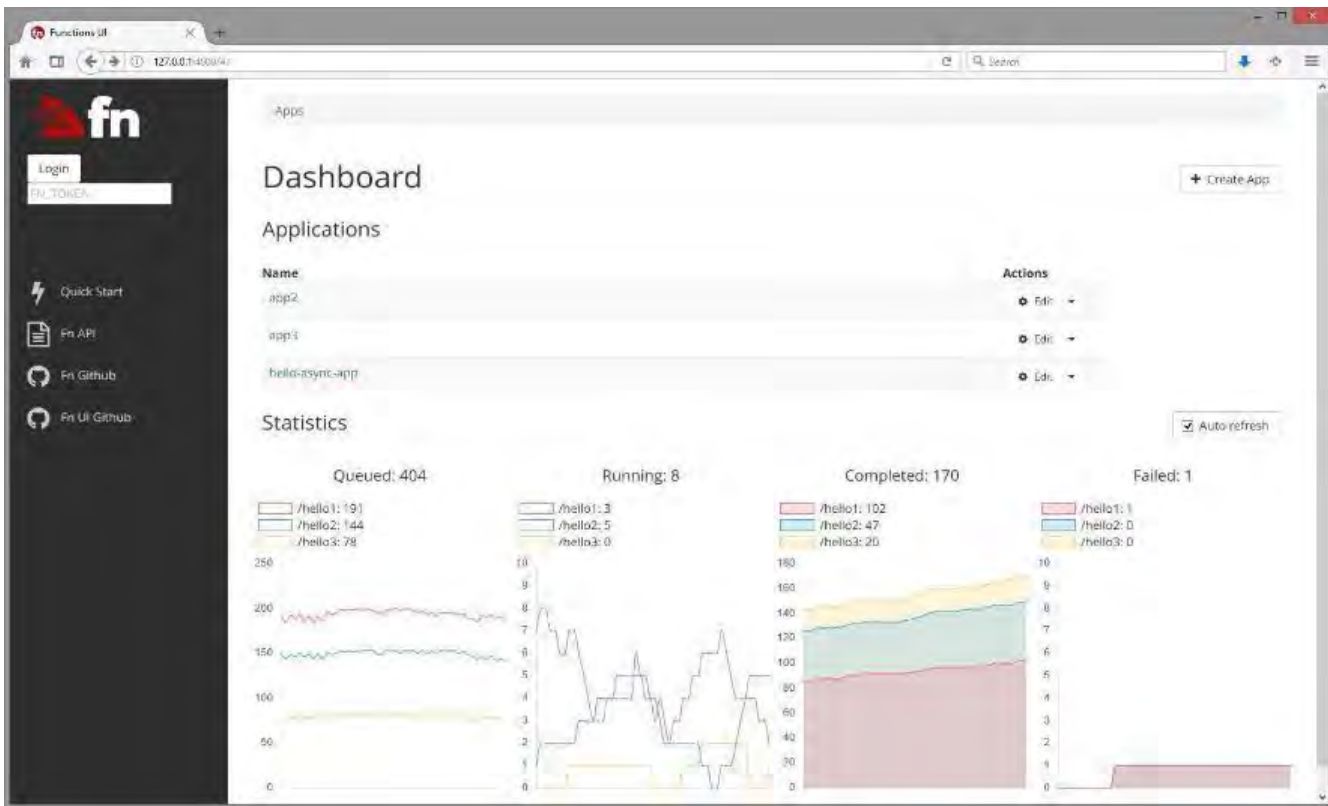
# Fn Flow 示例



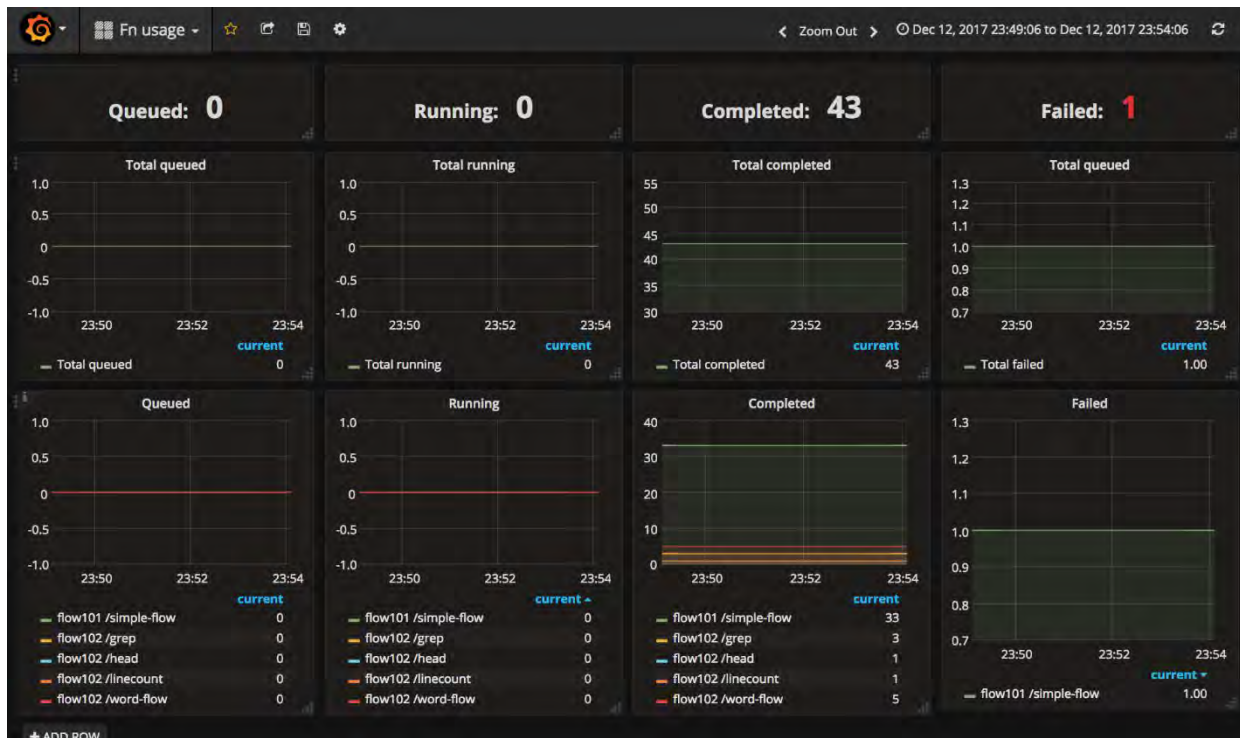




# Fn UI界面指标监控



由Fn服务器生成的普罗米修斯指标，在Grafana中可视化





# Java函数示例

```
1 package com.example.fn;
2
3 public class HelloFunction {
4
5     ... public String handleRequest(String input) {
6     ...     String name = (input == null || input.isEmpty()) ? "world" : input;
7
8     ...     return "Hello, " + name + "!";
9     ... }
10
11 }
```



```
1 version: 0.0.1
2 runtime: java
3 cmd: com.example.fn.HelloFunction::handleRequest
4 build_image: fnproject/fn-java-fdk-build:jdk9-1.0.55
5 run_image: fnproject/fn-java-fdk:jdk9-1.0.55
6 format: http
```



```
$ fn deploy --app myapp
```

HelloFunction



func.yaml

```
version: 0.0.2  
runtime: java  
cmd: com.example.faas.HelloFunction::handleRequest
```

# fn deploy — 构建函数容器

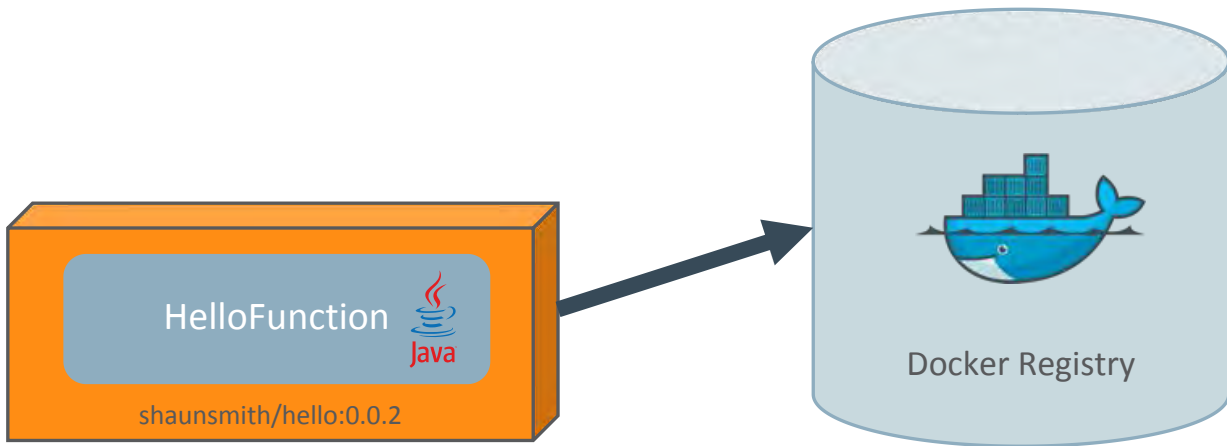


func.yaml

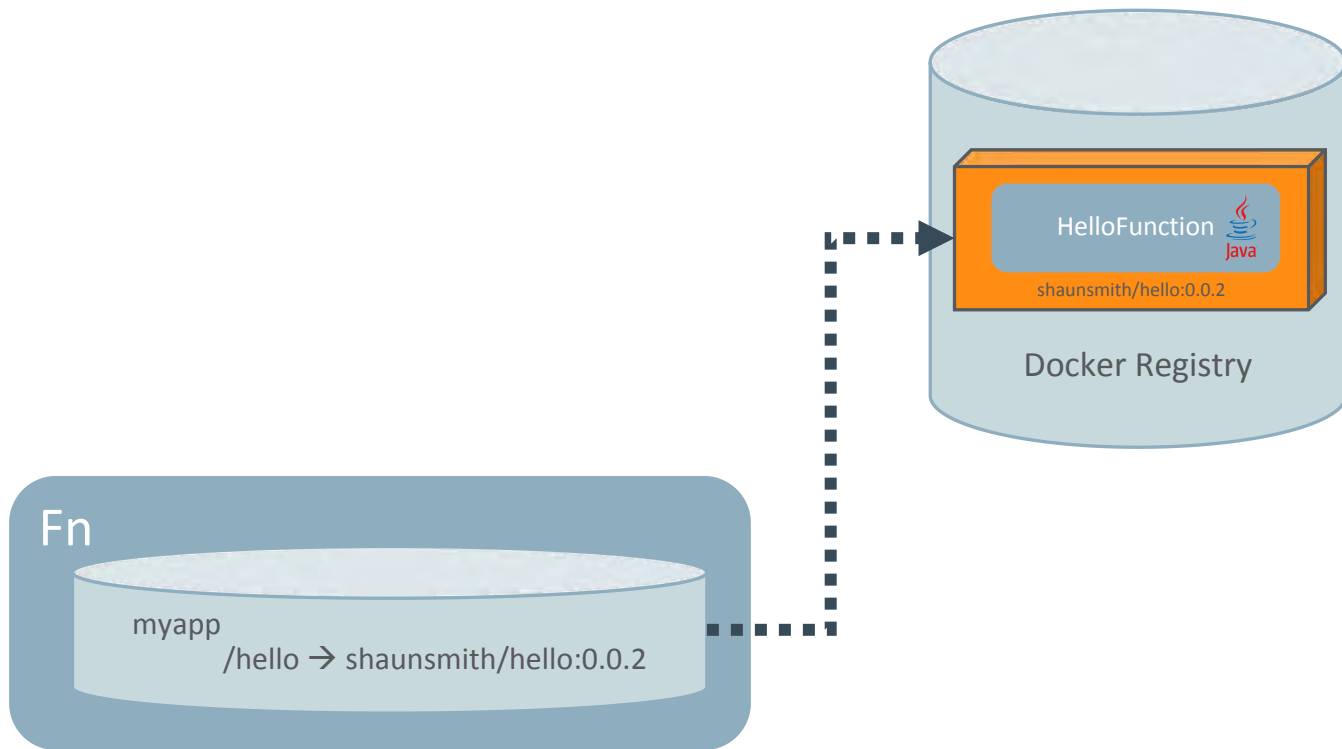
```
version: 0.0.2  
runtime: java  
cmd: com.example.faas.HelloFunction::handleRequest
```



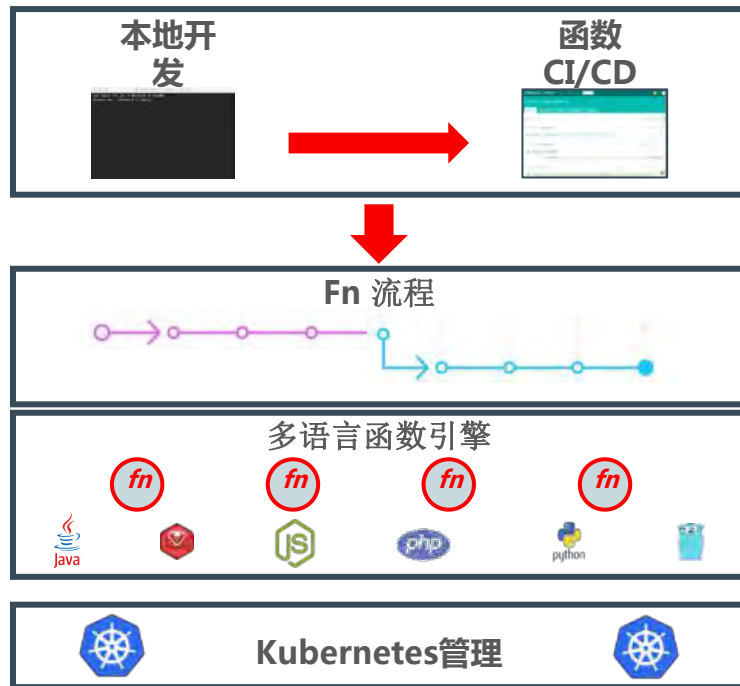
# fn deploy — 推送到容器注册



# fn deploy — 定义函数路由



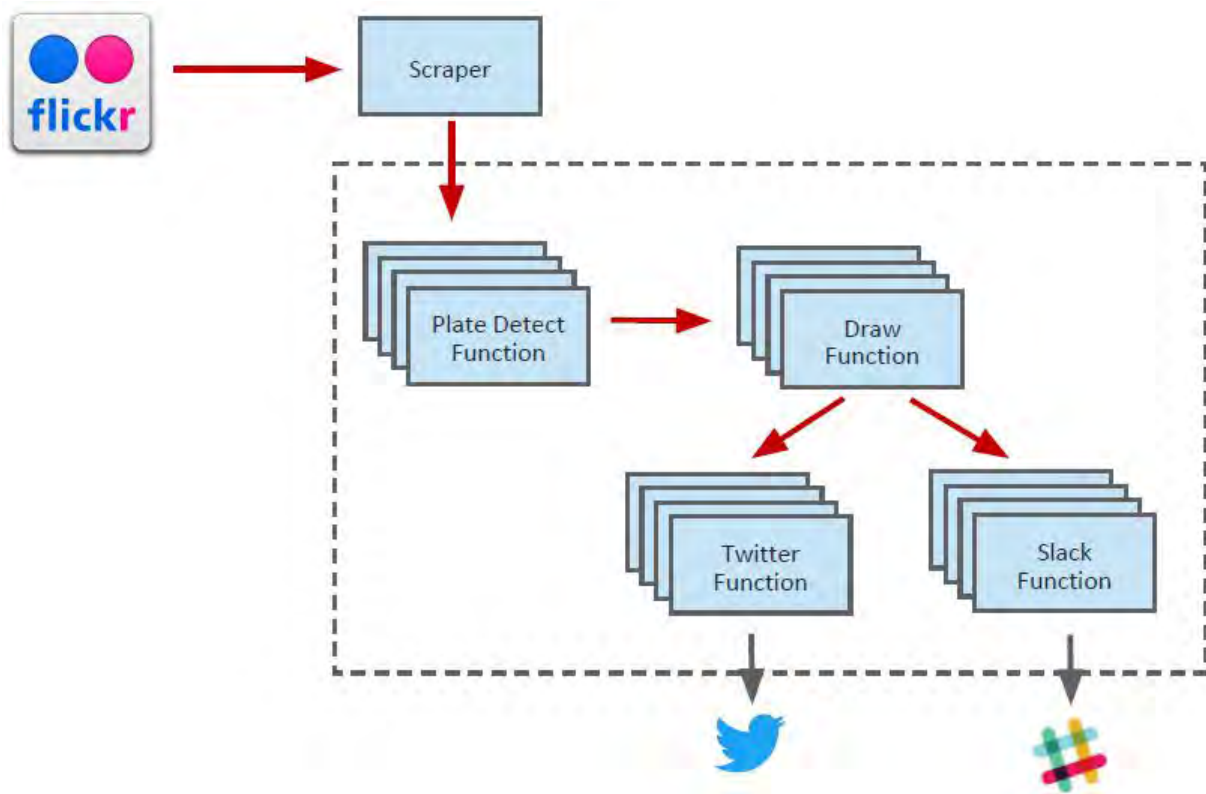
- 开源
- 云中立
- 支持各种语言，对Java进一步优化
- 容器原生，本地开发体验
- 无痛迁移Amazon Lamda程序



**Fn Demo**



# 甲骨文 Vista 驾驶牌照 Demo

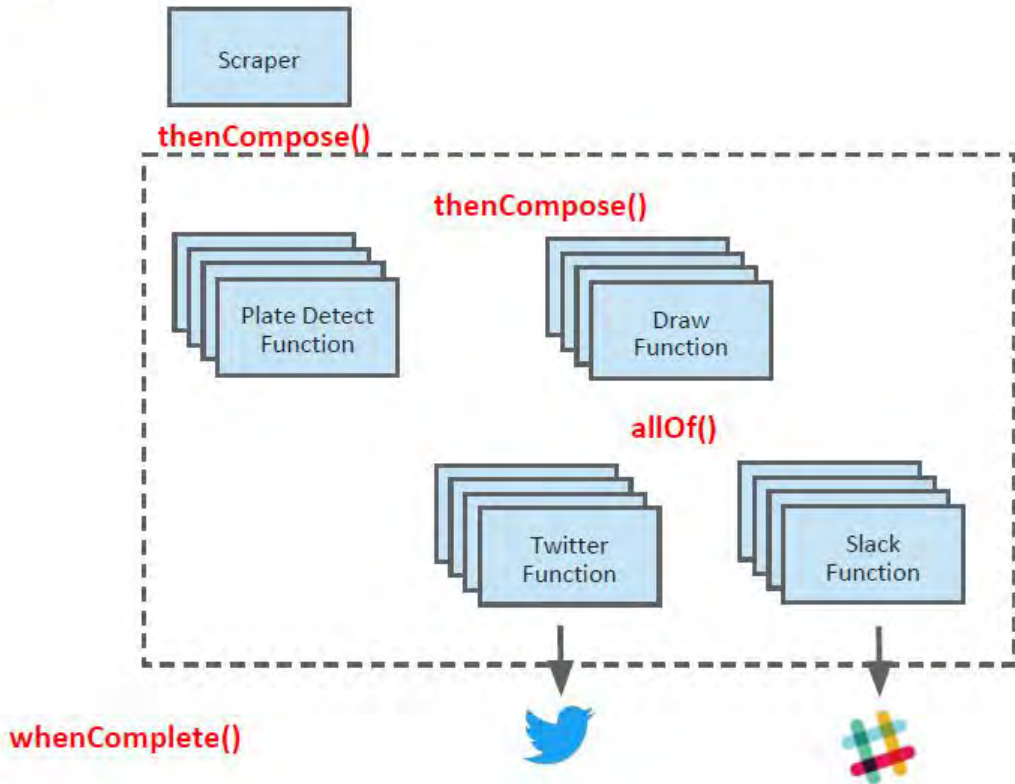






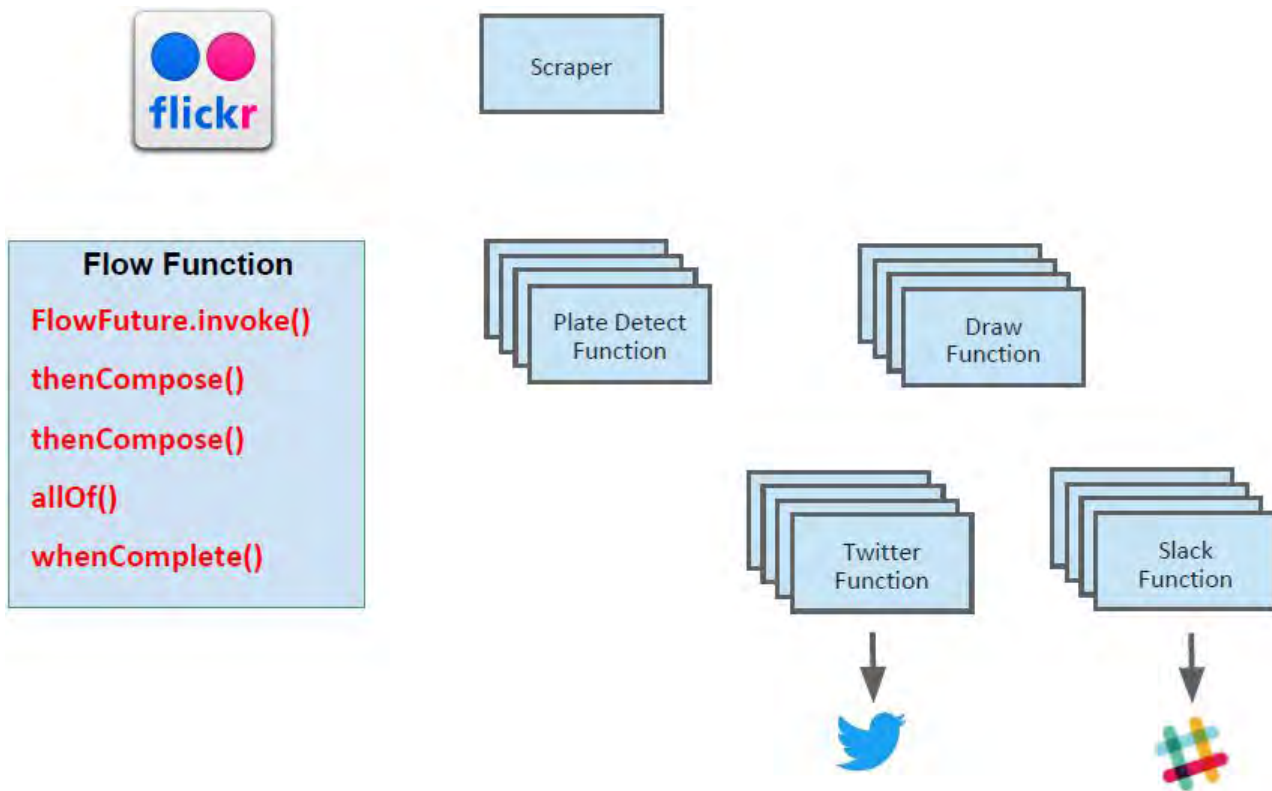
# 甲骨文 Vista 驾驶牌照 Demo

FlowFuture.invoke()





# 甲骨文 Vista 驾驶牌照 Demo

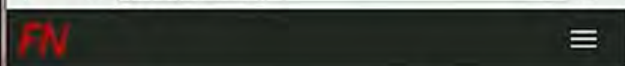


```
Flow Function  
FlowFuture.invoke()  
thenCompose()  
thenCompose()  
allOf()  
whenComplete()
```



# 甲骨文 Vista 驾驶牌照 Demo

```
1 public class VistaFlow {
2     public void handleRequest(ScrapeReq input) throws Exception {
3
4         FlowFuture<ScrapeResp> scrapes = currentFlow().invokeFunction("scraper");
5
6         scrapes.thenCompose(resp -> {
7             List<FlowFuture<?>> pendingTasks = results.stream().map(scrapeResult -> {
8
9                 String id = scrapeResult.id;
10
11                 return currentFlow().invokeFunction("detect-plates").thenCompose((plateResp) -> {
12                     return currentFlow().invokeFunction("draw").thenCompose((drawResp) -> {
13
14                         return currentFlow().allOf(
15                             currentFlow().invokeFunction("twitter"),
16                             currentFlow().invokeFunction("slack"));
17                     });
18                 });
19             });
20         });
21
22     }).whenComplete((v, throwable) -> {
23         if (throwable != null) {
24             postMessageToSlack("An Error Occurred.");
25         } else {
26             postMessageToSlack("Finished Scraping.");
27         }
28     });
29 }
30 }
```



# Dashboard

+ Create App

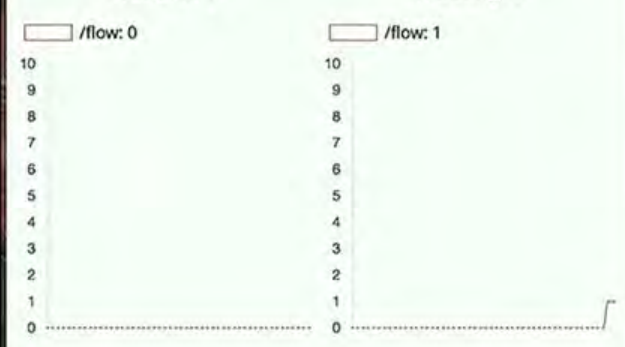
## Applications

Name	Actions
myapp	<a href="#">Edit</a>

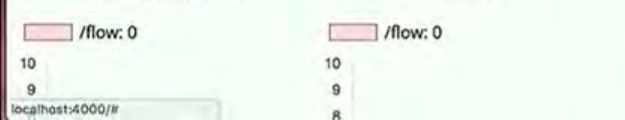
## Statistics

Auto refresh

Queued: 0      Running: 1



Completed: 0      Failed: 0



main

0: m

Pending Events:

## ostream

ed this channel today. This is the very beginn  
ostream channel.

urpose + Add an app & Invite others to this chan

---

Today

**rimura** 12:35 PM  
ned #demostream along with fnbot.

essage #demostream

# THANKS

主办方：



x

ORACLE  
甲骨文

