Apache Dubbo (Incubating) 2.7.0新特性介绍及演示

阿里巴巴-中间件技术部

徐靖峰

自我介绍

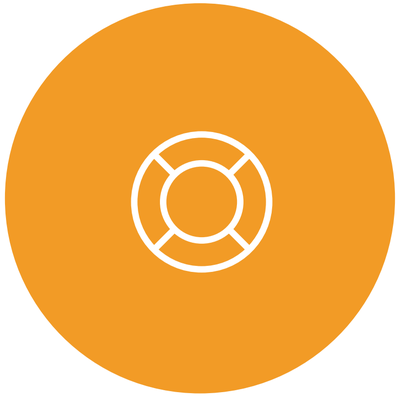
徐靖峰（kirito）

- 知识分享爱好者&公众号：Kirito的技术分享
- Apache Dubbo Committer
- 阿里中间件服务治理框架 HSF 研发相关工作
- 开源爱好者： https://github.com/lexburner

大纲

Dubbo介绍

Dubbo2.7.0



Dubbo OPS

联系我们

Copyright © 2019 The Apache Software Foundation

开源现状

自重新维护以来 github 数据显著增长

Star 数增长

160%

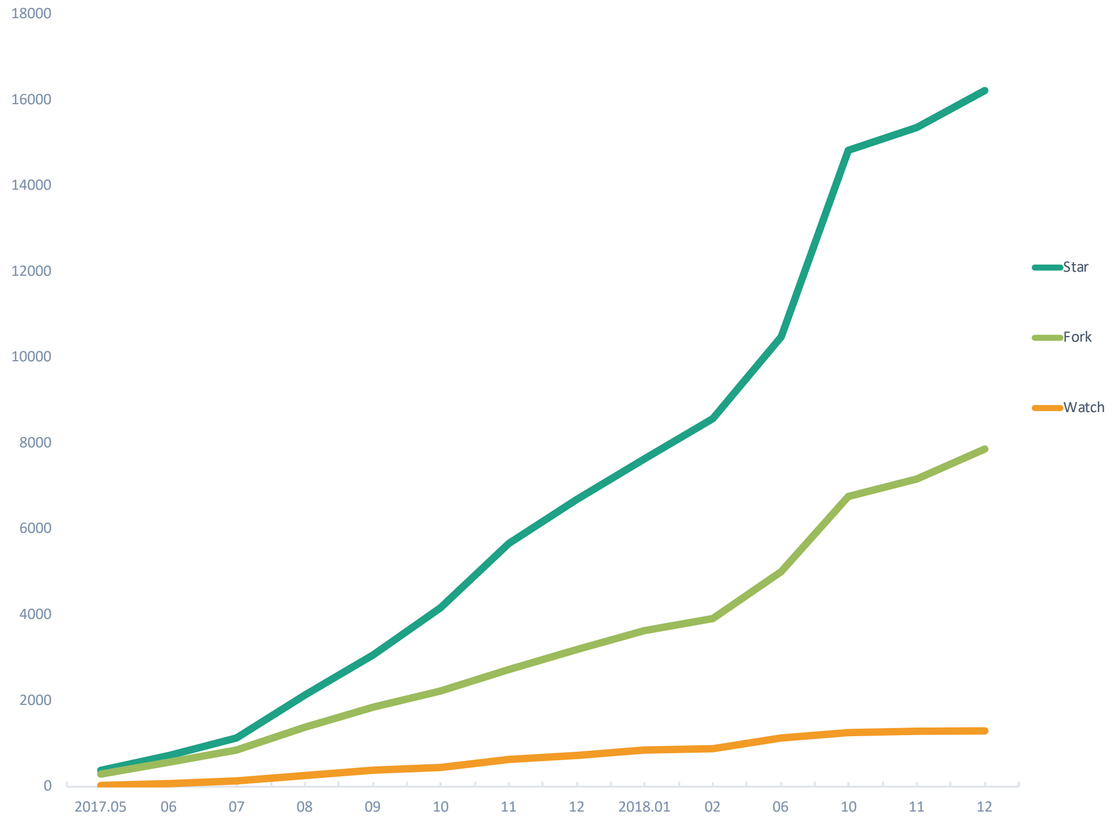
自 2017 年 7 月 重启 Dubbo 开源，到目前 Star 数增长 17000+，Fork 数增长 8000+，Watch 数增加 1300+, Contributor 164 人

在 Github Java 类项目 star 数排名 13 位，每天 2000 UV。
2018最受欢迎中国开源软件Java类第一
开源云联盟首届中国优秀开源项目一等奖

25K STAR

16K FORK

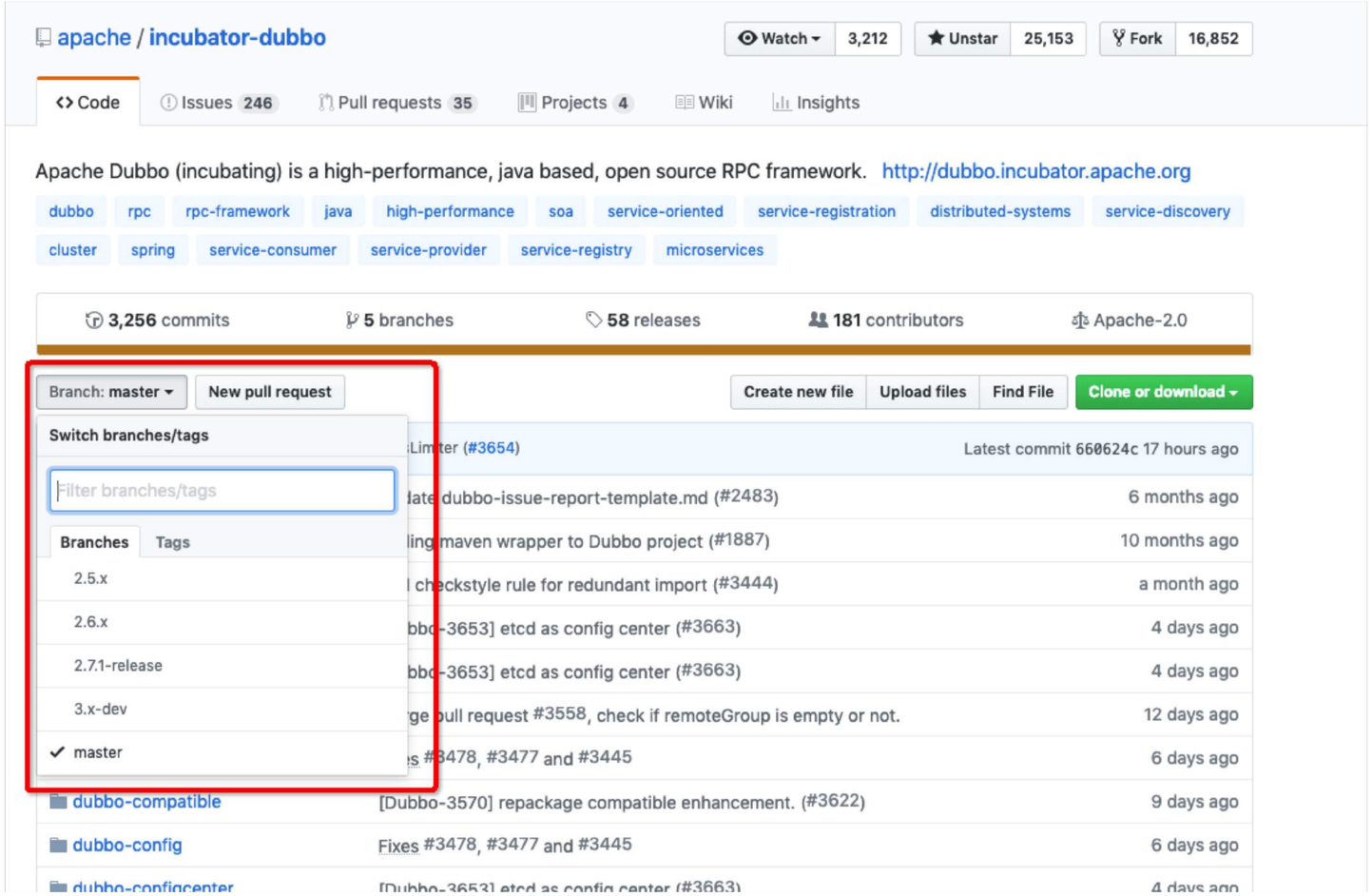
3K WATCH



Copyright © 2019 The Apache Software Foundation

分支介绍

Github 分支介绍



2.5.x	不维护
2.6.x	bugfix
master	稳定维护（推荐）
3.x-dev	前瞻性的版本

Copyright © 2019 The Apache Software Foundation

Dubbo 2.7

新版本特性

异步支持

CompletableFuture

三大中心改造

注册中心

元数据中心

配置中心

服务治理增强

标签路由

应用级别路由

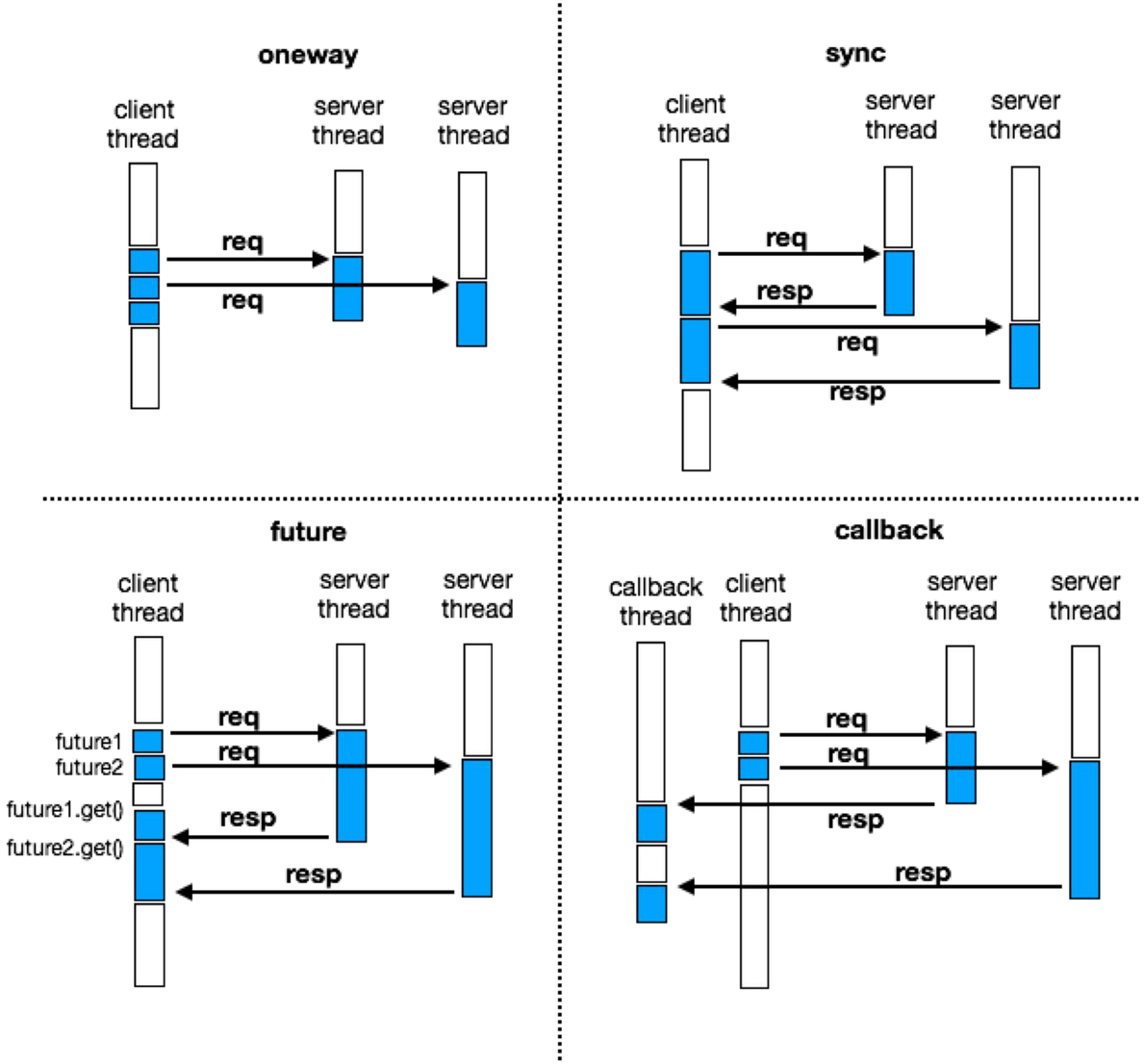
Copyright © 2019 The Apache Software Foundation

思考题

同步和异步的区别？

Copyright © 2019 The Apache Software Foundation

异步支持



Dubbo 线程模型

Copyright © 2019 The Apache Software Foundation

异步支持

Dubbo 2.6 异步支持

```
<dubbo:reference id="asyncService"
interface="org.apache.dubbo.demo.api.AsyncService"
async="true”/>
```

服务配置

```
public interface AsyncService {
    String sayHello(String name);
}
```

接口定义

```
AsyncService.sayHello("Han Meimei");
Future<String> fooFuture = RpcContext.getContext().getFuture();
fooFuture.get();
```

异步调用

不需要改造接口

改变了编程模型

不支持 Callback

Copyright © 2019 The Apache Software Foundation

异步支持

Dubbo 2.7 异步支持

```
public interface AsyncService {
  String sayHello(String name);
  default CompletableFuture<String> sayHiAsync(String name) {
    return CompletableFuture.completedFuture(sayHello(name));
  }
}
```

接口定义

需要改造接口

符合异步编程模型

支持 Callback

```
CompletableFuture<String> future = asyncService.sayHiAsync("Han MeiMei");
future.whenComplete((retValue, exception) -> {
    System.out.println(retValue);
});
```

异步调用

Copyright © 2019 The Apache Software Foundation

异步支持

Q：如果 RPC 接口只定义了同步接口，有办法使用异步调用吗？

A：2.6 中的异步调用唯一的优势在于，不需要在接口层面做改造，又可以进行异步调用，这种方式仍然在 2.7 中保留；使用 Dubbo 官方提供的 compiler hacker，编译期自动重写同步方法，请在此讨论和跟进具体进展。

Q：关于异步接口的设计问题，为何不提供编译插件，根据原接口，自动编译出一个 XxxAsync 接口？

A：Dubbo 2.7 采用过这种设计，但接口的膨胀会导致服务类的增量发布，而且接口名的变化会影响服务治理的一些相关逻辑，改为方法添加 Async 后缀相对影响范围较小。

Q：Dubbo 分为了客户端异步和服务端异步，刚刚你介绍的是客户端异步，为什么不提服务端异步呢？

A：Dubbo 2.7 新增了服务端异步的支持，但实际上，Dubbo 的业务线程池模型，本身就可以理解为异步调用，个人认为服务端异步的特性较为鸡肋。

Copyright © 2019 The Apache Software Foundation

思考题

什么是元数据？

Copyright © 2019 The Apache Software Foundation

元数据改造–背景

Analysis

性能

推送量大 -> 存储数据量大 -> 网络传输量大 -> 延迟严重

参数分析

生产者端注册30+参数，有接近一半是不需要作为注册中心进行传递
消费者端注册25+参数，只有个别需要传递给注册中心

新需求

OPS-服务测试需要元数据信息

Copyright © 2019 The Apache Software Foundation

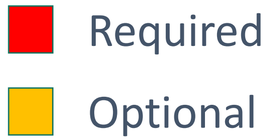
元数据改造–Dubbo2.6原始方案

服务

```
<dubbo:service interface="com.alibaba.dubbo.demo.DemoService" ref="demoService"
executes="4500" retries="7" owner="kirito"/>
```

服务治理信息+元数据(zookeeper)

```
dubbo://30.5.120.185:20880/com.alibaba.dubbo.demo.DemoService?
anyhost=true&
application=demo-provider&
interface=com.alibaba.dubbo.demo.DemoService&
methods=sayHello&
bean.name=com.alibaba.dubbo.demo.DemoService&
dubbo=2.0.2&executes=4500&
generic=false&owner=kirito&
pid=84228&retries=7&side=provider&timestamp=1552965771067
```



Copyright © 2019 The Apache Software Foundation

元数据改造–Dubbo2.7过度方案

服务

```
<dubbo:service interface="com.alibaba.dubbo.demo.DemoService" ref="demoService"
executes="4500" retries="7" owner="kirito"/>
```

服务治理信息(zookeeper)不变

元数据(zookeeper)


```
{
  "parameters": {
    "owner": "kirito",
    "side": "provider",
    "release": "2.7.0",
    "methods": "sayHello",
    "dubbo": "2.0.2",
    "interface": "org.apache.dubbo.demo.api.DemoService",
    "generic": "false",
    "retries": "7",
    "application": "meetup-demo-provider",
    "executes": "4500",
    "bean.name": "org.apache.dubbo.demo.api.DemoService",
    "anyhost": "true"
  },
  "canonicalName": "org.apache.dubbo.demo.api.DemoService",
  "codeSource": "file:/Users/xujingfeng/IdeaProjects/dubbo2.7-demo/dubbo-basic-api/target/classes/",
  "methods": [
    {"name": "sayHello"...},
    {"name": "sayHello"...},
    {"name": "sayHello"...},
    {"name": "sayHello"...}
  ],
  "types": [
    {"type": "char"...},
    {"type": "int"...},
    {"type": "java.lang.Long"...},
    {"type": "java.lang.String"...},
    {"type": "long"...},
    {"type": "org.apache.dubbo.demo.model.User"...},
    {"type": "org.apache.dubbo.demo.model.Result"...}
  ]
}
```


元数据中心分离

Copyright © 2019 The Apache Software Foundation

元数据改造–Dubbo2.7最终方案

服务

```
<dubbo:service interface="com.alibaba.dubbo.demo.DemoService" ref="demoService"
executes="4500" retries="7" owner="kirito"/>
<dubbo:registry address="zookeeper://127.0.0.1:2181" simplified="true" />
```

服务治理信息(zookeeper)

```
dubbo://30.5.120.185:20880/org.apache.dubbo.demo.api.DemoService?
application=demo-provider&
dubbo=2.0.2&
release=2.7.0&
timestamp=1552975501873
```

注册中心数据格式简化

Copyright © 2019 The Apache Software Foundation

思考题

配置中心在分布式系统的作用

Copyright © 2019 The Apache Software Foundation

配置中心

Analysis

职责

- 外部化配置。启动配置的集中式存储
- 服务治理。服务治理规则的存储与通知

配置中心支持

- Zookeeper
- Apollo
- Nacos（ 2.7.1-release ）

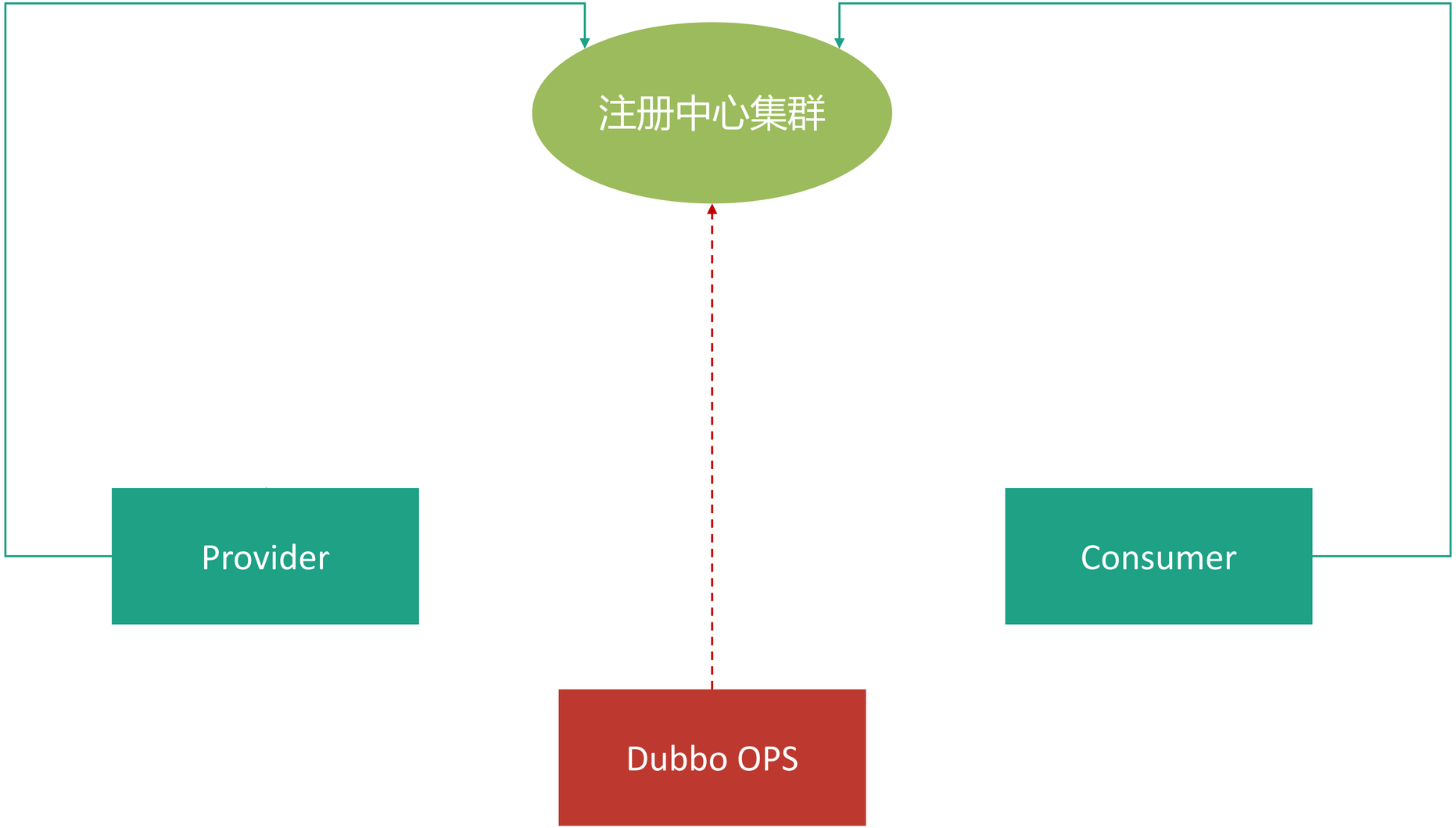
Copyright © 2019 The Apache Software Foundation

思考题

Dubbo 里面有哪几种节点角色？

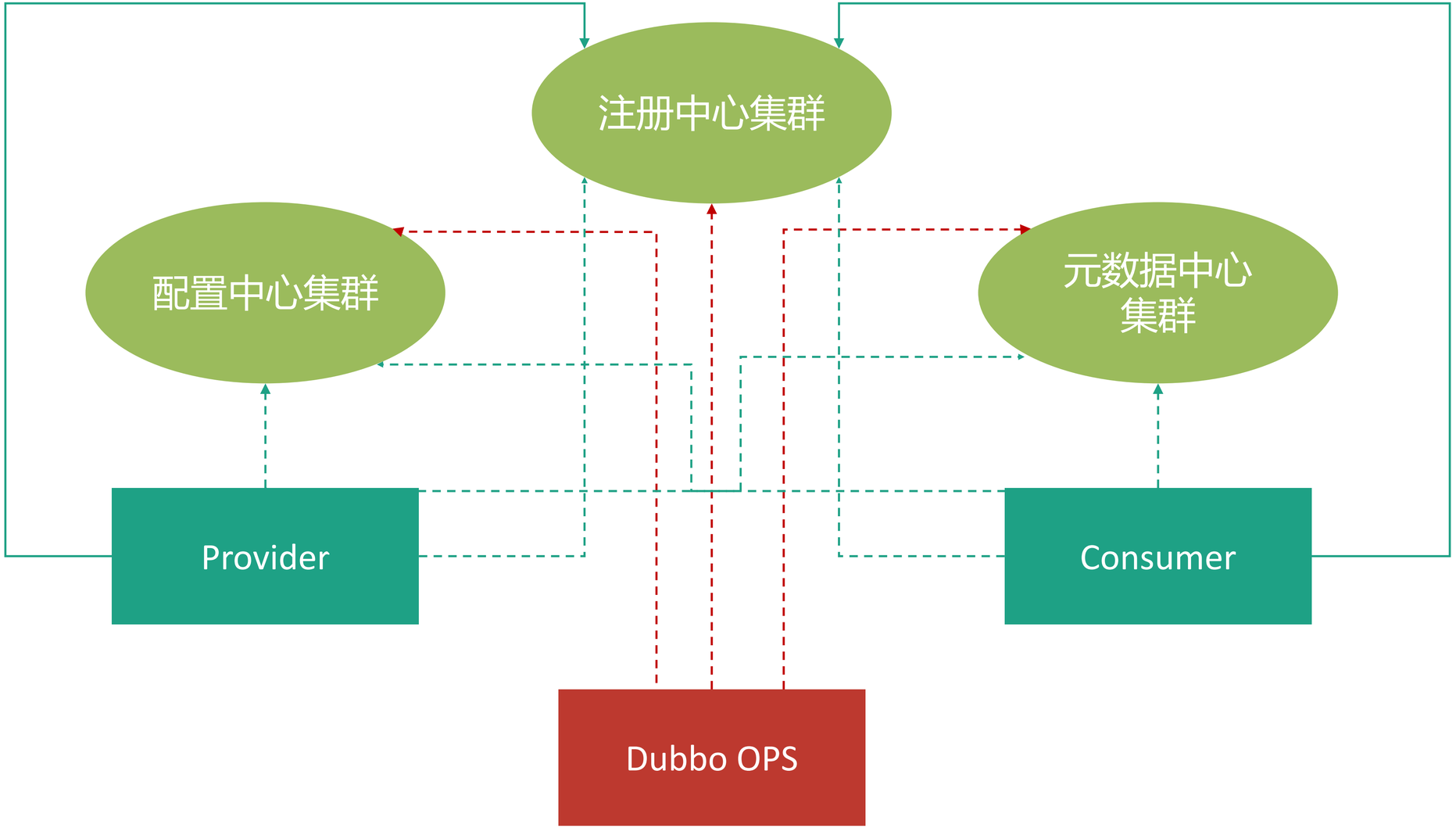
Copyright © 2019 The Apache Software Foundation

一个中心



Copyright © 2019 The Apache Software Foundation

三个中心



Copyright © 2019 The Apache Software Foundation

思考题

Dubbo 中路由层和负载均衡层的区别

Copyright © 2019 The Apache Software Foundation

路由实现

2.7 之前

- ScriptRouter
- MockRouter
- ConditionRouter
- FileRouter
- TagRouter(伪)

2.7 新增

- TagRouter
- AppRouter
- ServiceRouter

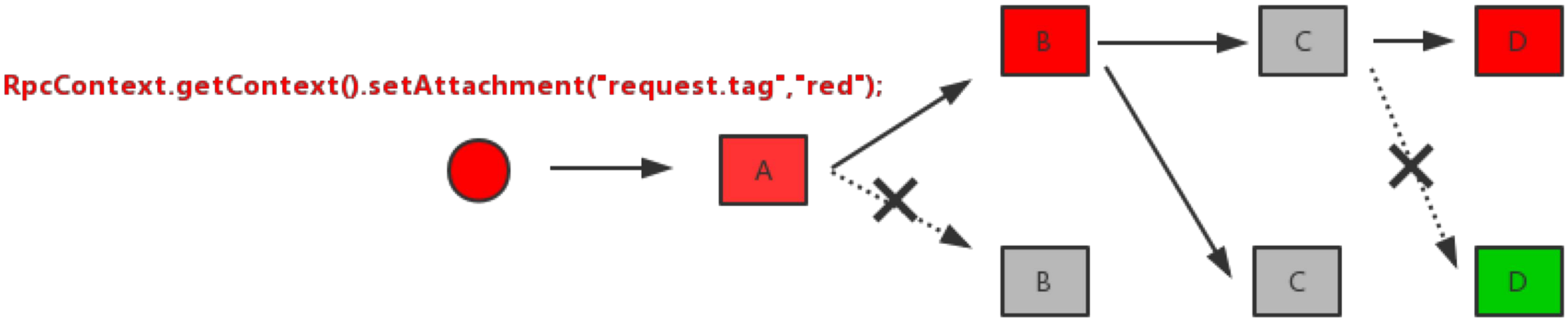
Copyright © 2019 The Apache Software Foundation

思考题

Dubbo 中如果做灰度发布和流量隔离？

Copyright © 2019 The Apache Software Foundation

标签路由



Copyright © 2019 The Apache Software Foundation

Dubbo OPS

全新UI

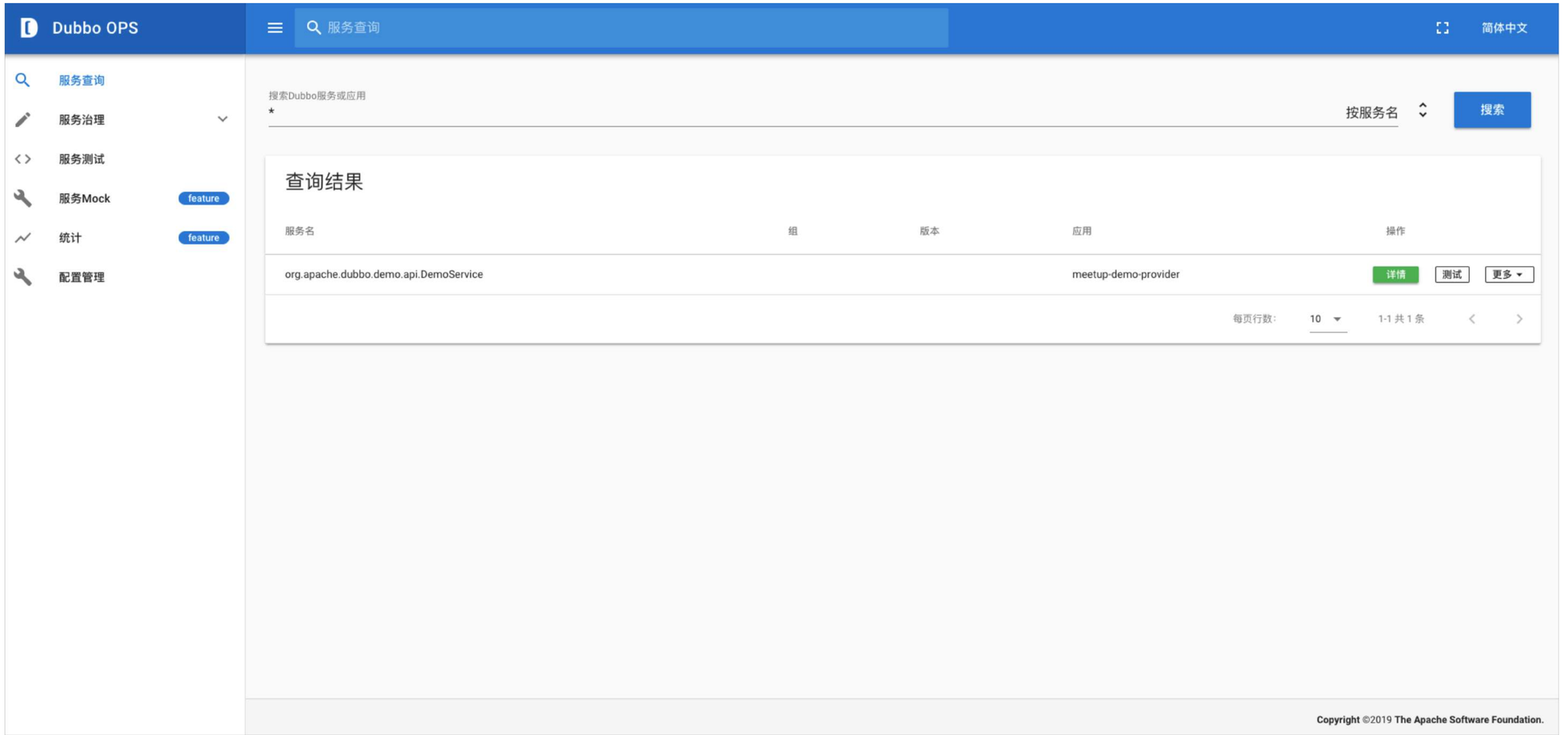
配置管理

服务测试

服务治理

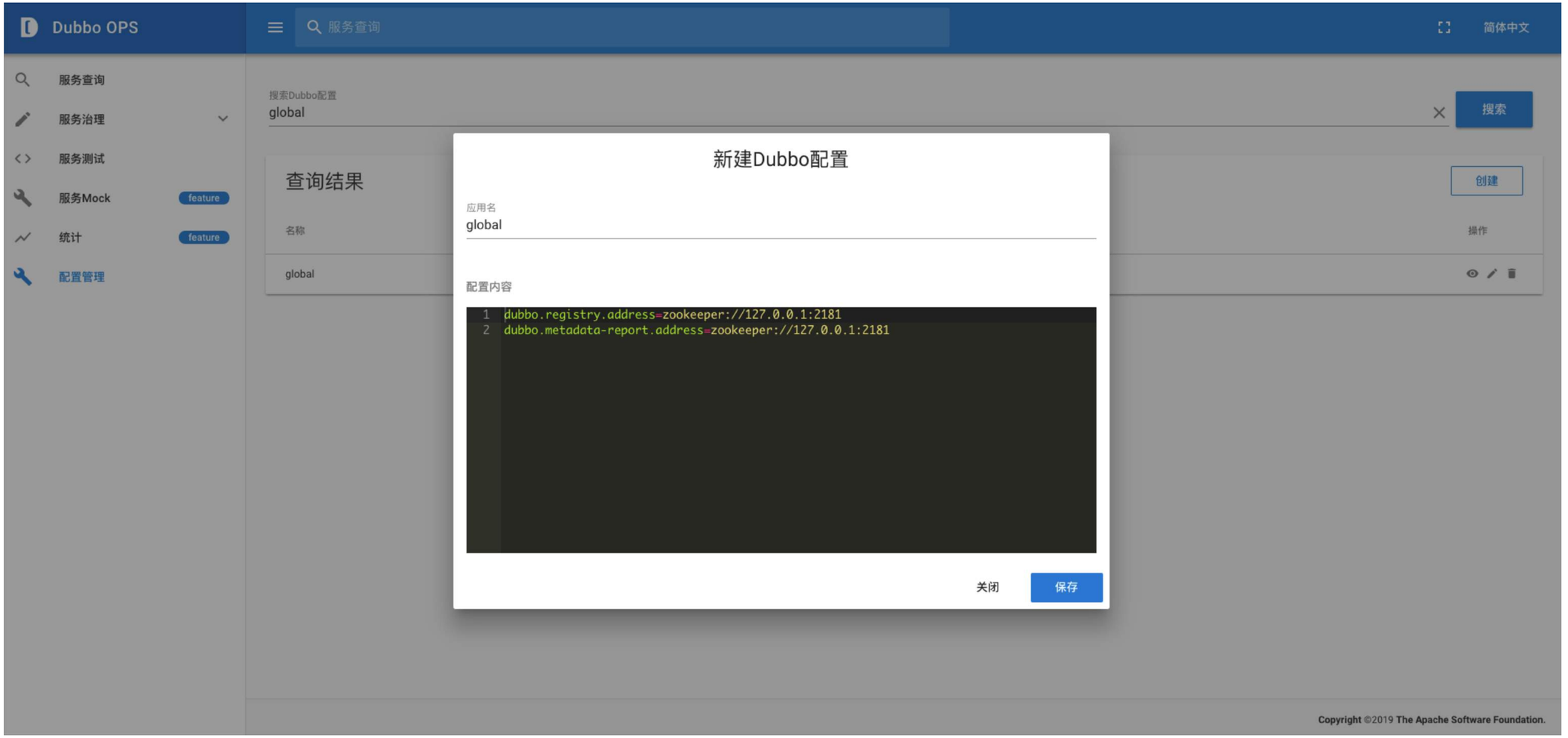
Copyright © 2019 The Apache Software Foundation

全新 UI—演示



Copyright © 2019 The Apache Software Foundation

配置管理—演示



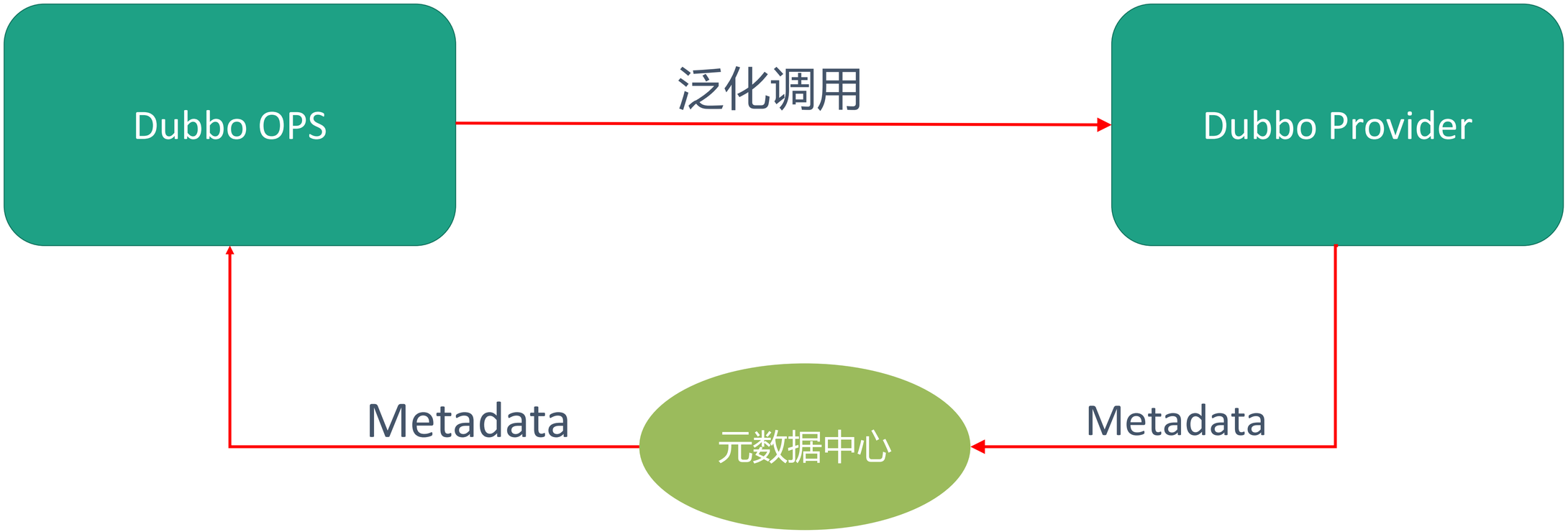
Copyright © 2019 The Apache Software Foundation

思考题

什么是泛化调用，它有什么使用场景？

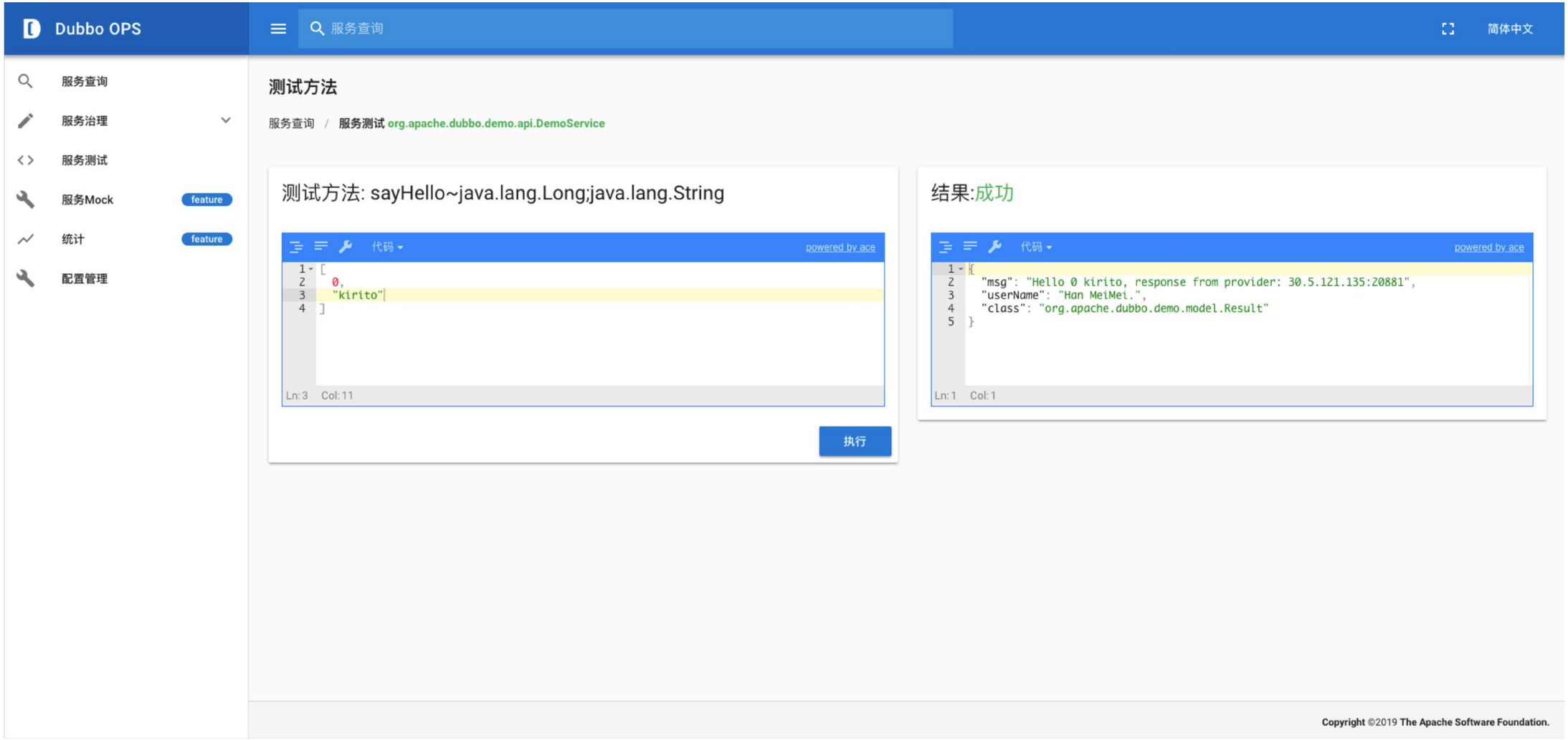
Copyright © 2019 The Apache Software Foundation

服务测试—原理



Copyright © 2019 The Apache Software Foundation

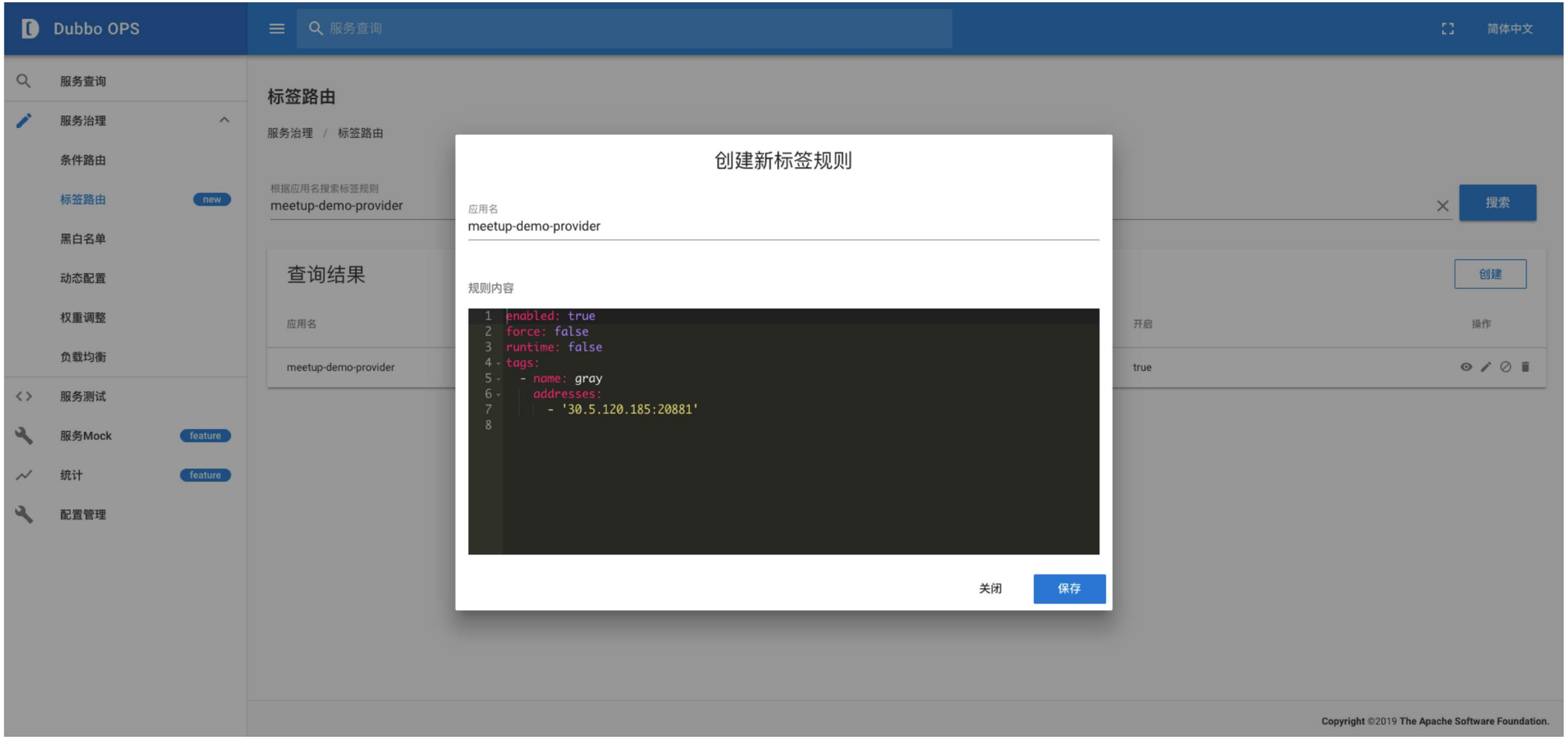
服务测试—演示



Copyright © 2019 The Apache Software Foundation

服务治理—演示

标签路由



Copyright © 2019 The Apache Software Foundation

Thank you !