Apache Dubbo (Incubating) 2.7.0新特性和Dubbo OPS

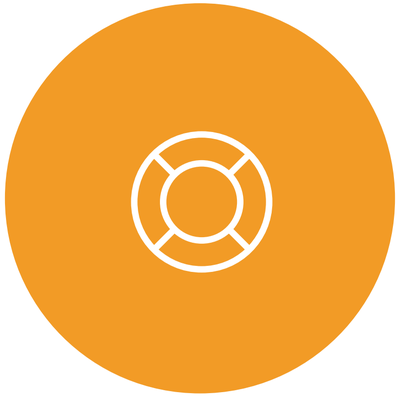
阿里巴巴-中间件技术部

庄旻轩

大纲

Dubbo现状

Dubbo2.7.0



Dubbo OPS

联系我们

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 16223，Fork 数增长 7869，Watch 数增加 1301，Contributor164人

在 Github Java 类项目 star 数排名 13 位，每天 2000 UV。

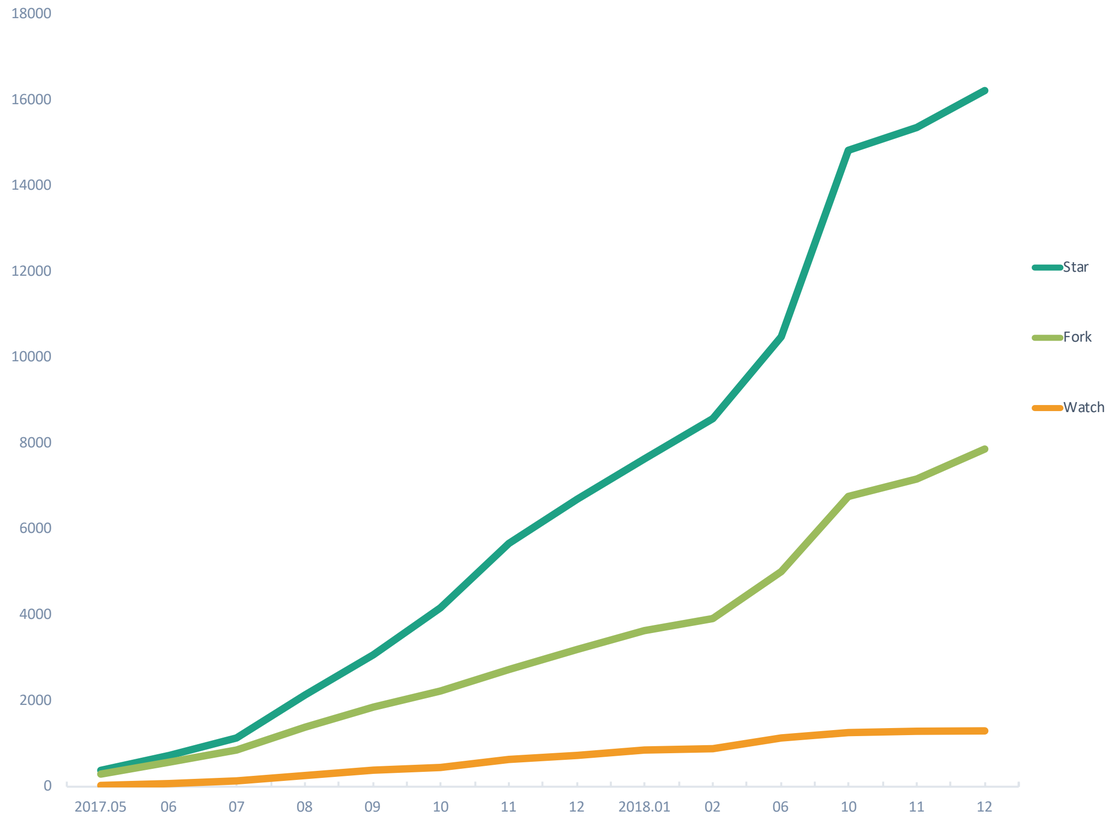
2018最受欢迎中国开源软件Java类第一

开源云联盟首届中国优秀开源项目一等奖

24K STAR

16K FORK

3K WATCH



工作原理

Dubbo 是高效的服务框架

1 **导出服务**
服务提供方通过指定端口对外暴露服务

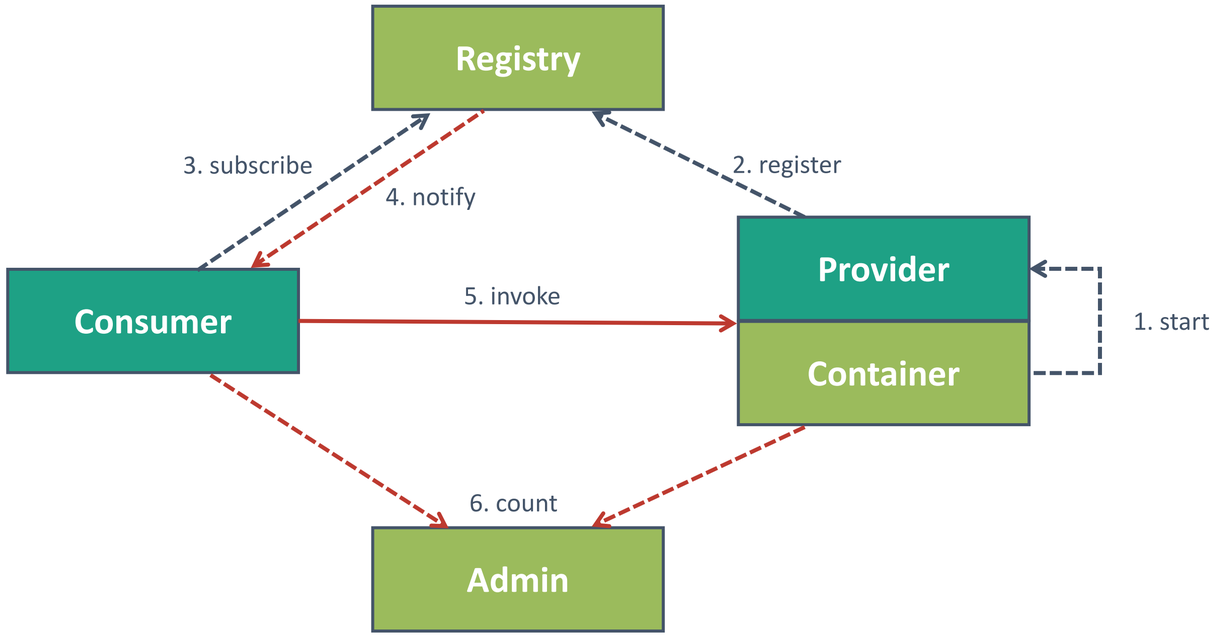
2 **注册服务**
提供方向注册中心注册自己的信息

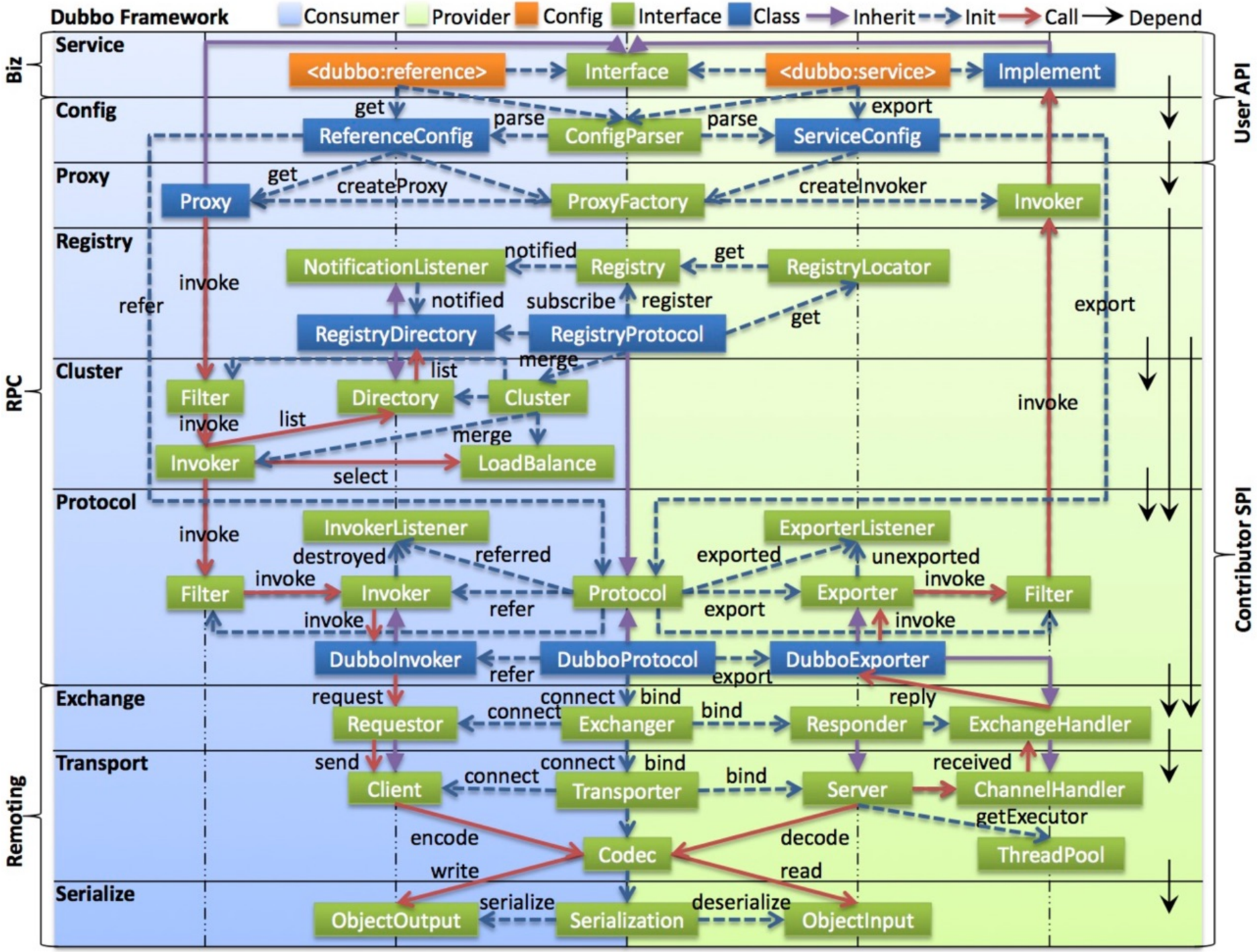
3 **订阅服务**
服务调用方通过注册中心订阅自己感兴趣的服务

4 **发现服务**
注册中心向调用方推送地址列表

5 **调用服务**
调用方选择一个地址发起 RPC 调用

6 **监控**
服务提供方和调用方的统计数据由监控模块收集展示




Dubbo Framework
Consumer
Provider
Config
Interface
Class
Inherit
Init
Call
Depend
Biz
RPC
Remoting
User API
Contributor SPI
Service
Config
Proxy
Registry
Cluster
Protocol
Exchange
Transport
Serialize
<dubbo:reference>
Interface
<dubbo:service>
Implement
get
parse
parse
export
ReferenceConfig
ConfigParser
ServiceConfig
get
createProxy
createInvoker
Proxy
ProxyFactory
Invoker
notified
get
NotificationListener
Registry
RegistryLocator
invoke
refer
notified
subscribe
register
export
get
RegistryDirectory
RegistryProtocol
merge
list
Filter
Directory
Cluster
invoke
list
merge
invoke
Invoker
LoadBalance
select
InvokerListener
ExporterListener
invoke
destroyed
referred
exported
unexported
Filter
invoke
Invoker
Protocol
Exporter
invoke
Filter
refer
export
invoke
invoke
DubboInvoker
DubboProtocol
DubboExporter
refer
export
request
connect
bind
reply
connect
Requestor
Exchanger
bind
Responder
ExchangeHandler
send
connect
bind
bind
received
Client
connect
Transporter
Server
ChannelHandler
getExecutor
encode
decode
Codec
ThreadPool
write
read
serialize
deserialize
ObjectOutput
Serialization
ObjectInput

Dubbo新版本特性

元数据改造

注册中心

配置中心

元数据中心

异步支持

CompletableFuture

@DubboAsync

服务端异步

Dubbo OPS

全新UI设计

支持Dubo2.7中的新特性

服务测试

异步支持

- Future获取方式
- Future无法实现自动回调
- 不支持Provider端异步

异步支持

- Jdk8新特性：CompletableFuture方式
- 支持Provider端的异步
- 支持了Promise方式

异步支持

方式二改造：provider端异步

```
public class AsyncServiceImpl implements AsyncService {
    @Override
    public String sayHello(String name) {
        final AsyncContext asyncContext = RpcContext.startAsync();
        new Thread(() -> {
            asyncContext.signalContextSwitch();
            try {
                Thread.sleep(500);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
            asyncContext.write("Hello " + name + ", response from provider.");
        }).start();
        return "hello, " + name;
    }
}
```

Provider

```
<dubbo:service async="true" interface="AsyncService" ref="asyncService"/>
```

服务配置

元数据改造

Analysis

性能

推送量大 -> 存储数据量大 -> 网络传输量大 -> 延迟严重

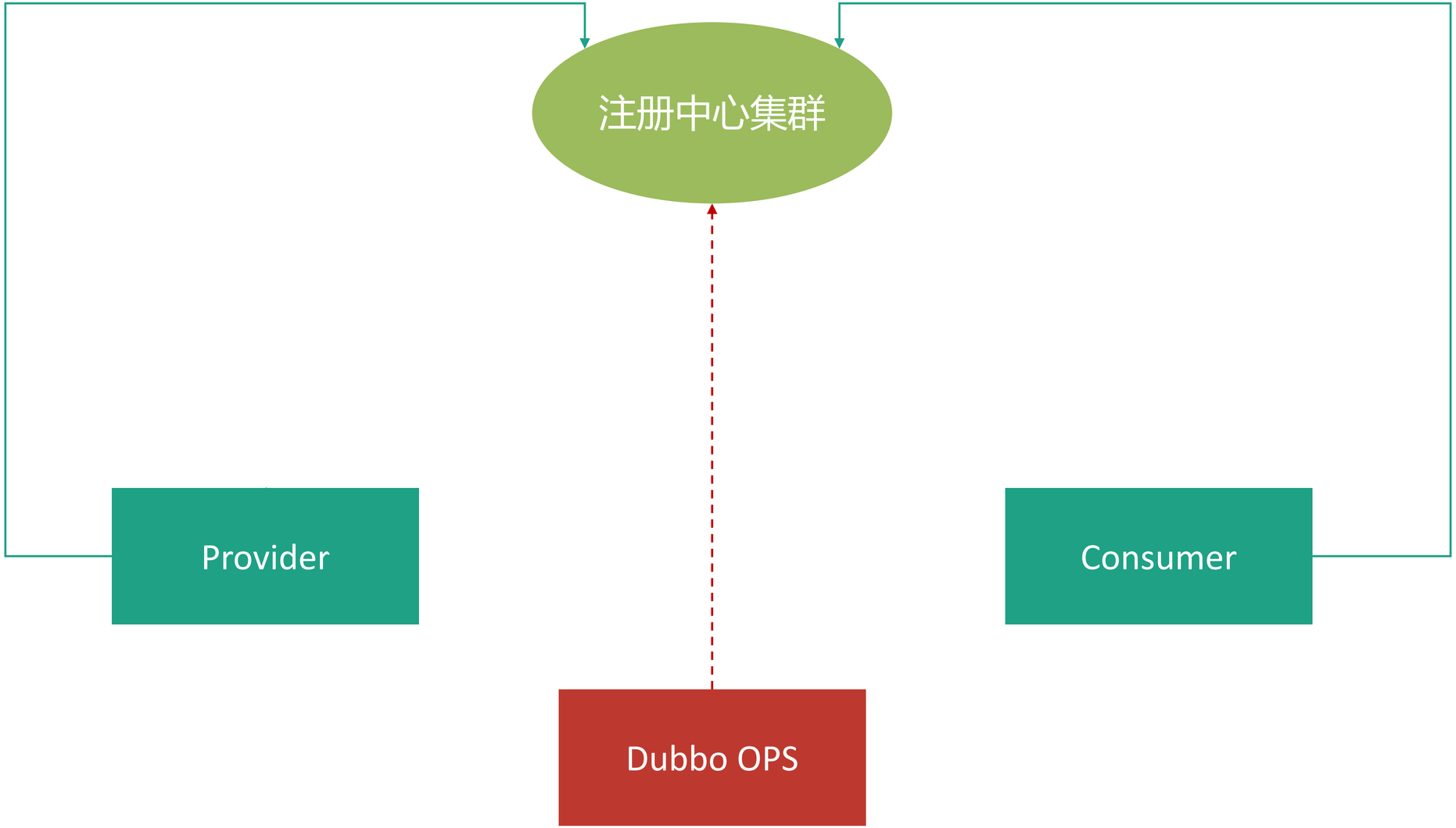
参数分析

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

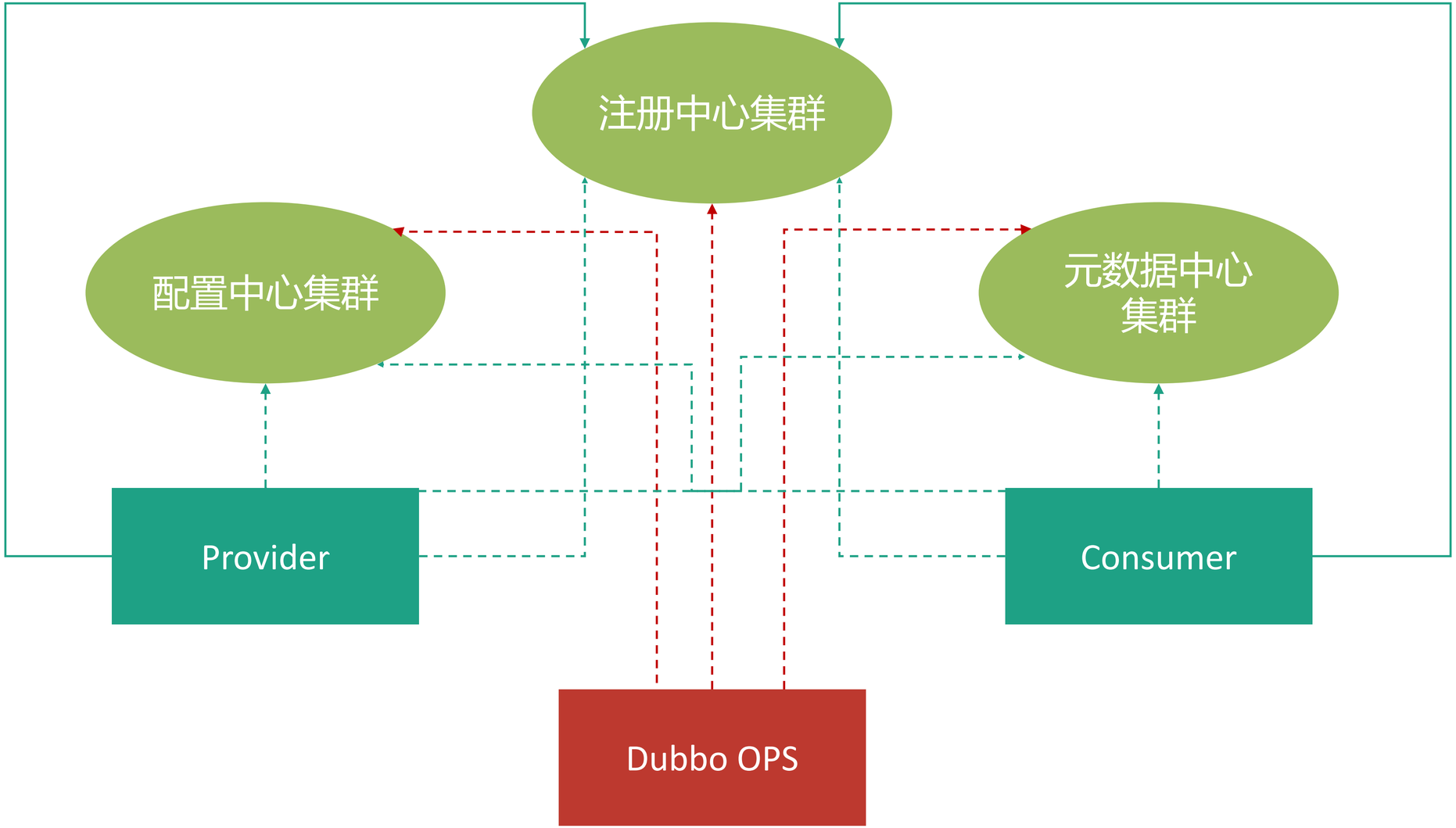
新需求

OPS-服务测试需要元数据信息

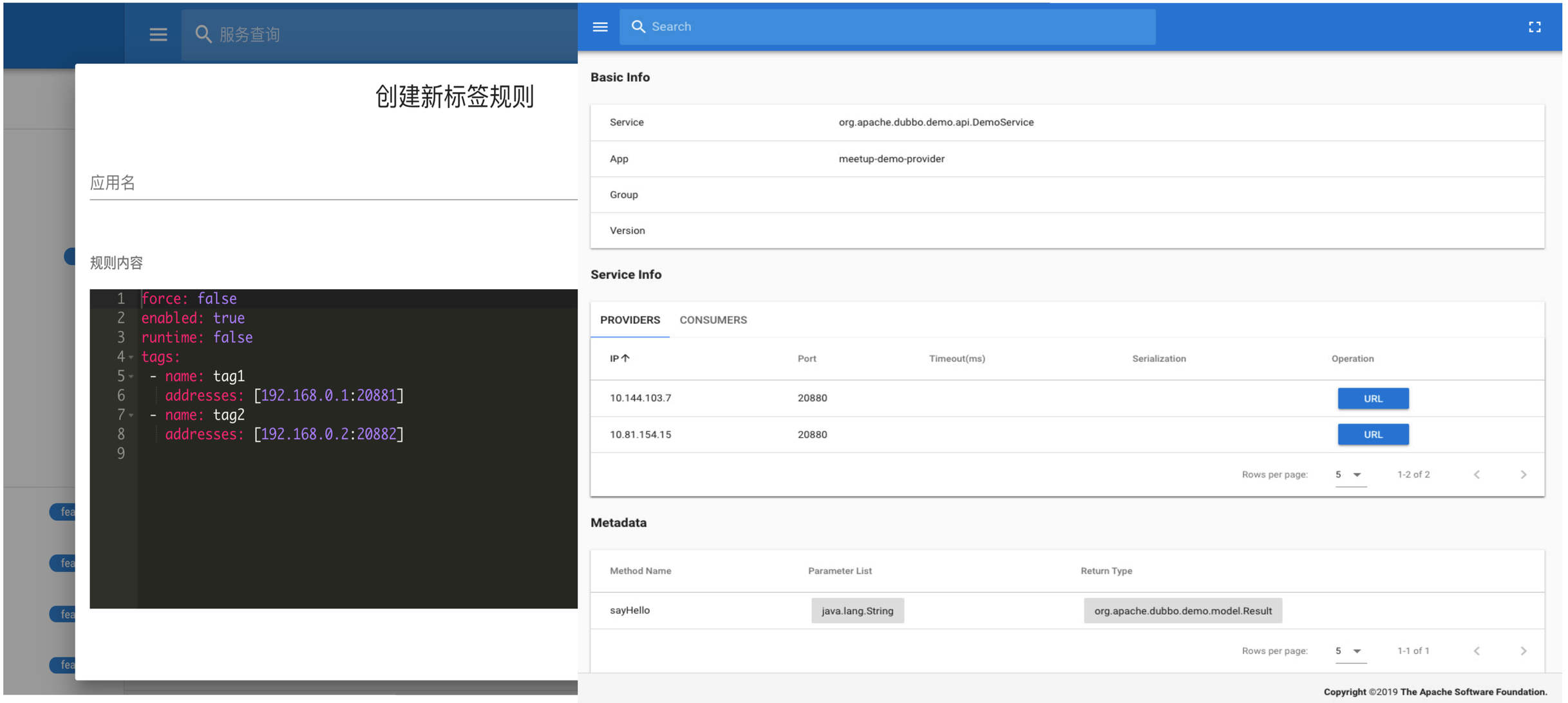
一个中心



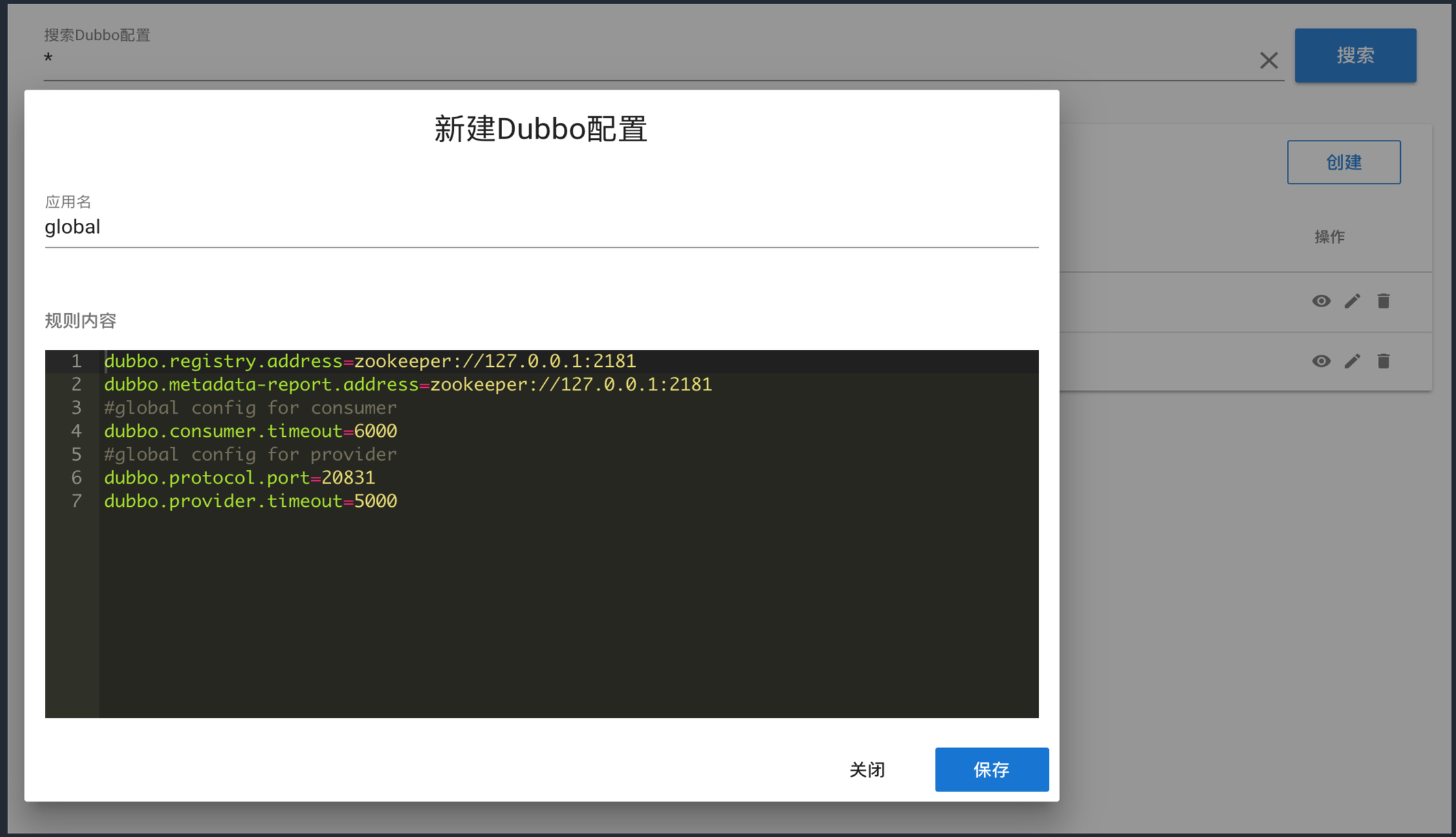
三个中心



Dubbo OPS

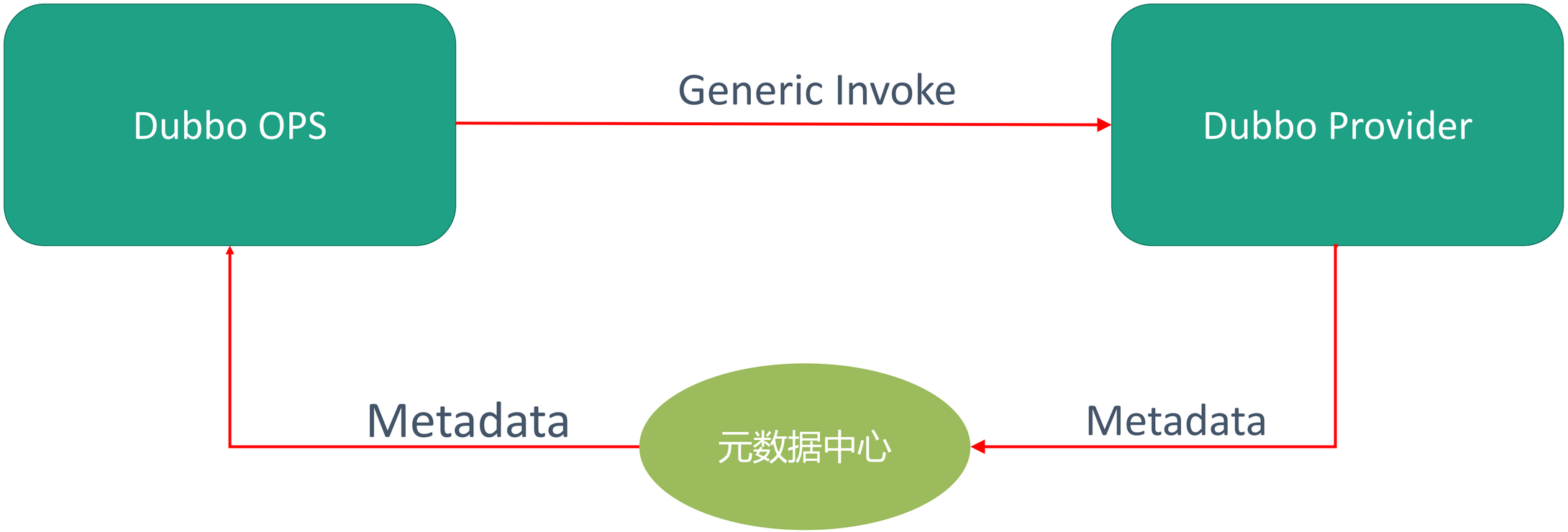


配置管理

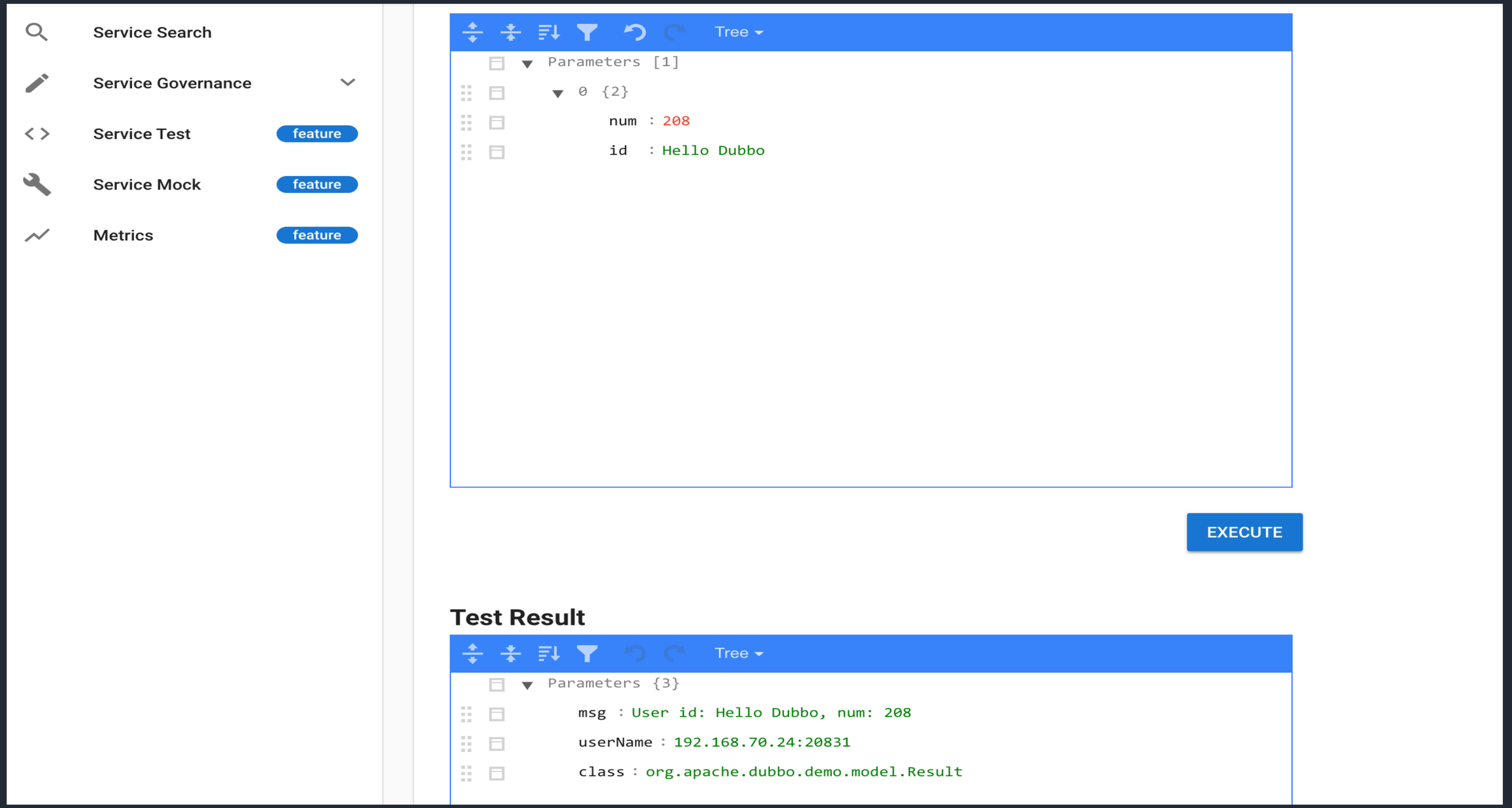

搜索Dubbo配置
*
搜索
创建
操作
新建Dubbo配置
应用名
global
规则内容
1 dubbo.registry.address=zookeeper://127.0.0.1:2181
2 dubbo.metadata-report.address=zookeeper://127.0.0.1:2181
3 #global config for consumer
4 dubbo.consumer.timeout=6000
5 #global config for provider
6 dubbo.protocol.port=20831
7 dubbo.provider.timeout=5000
关闭
保存

服务测试

Dubbo OPS



服务测试


Service Search
Service Governance
Service Test
feature
Service Mock
feature
Metrics
feature
Tree
Parameters [1]
0 {2}
num : 208
id : Hello Dubbo
EXECUTE
Test Result
Tree
Parameters {3}
msg : User id: Hello Dubbo, num: 208
userName : 192.168.70.24:20831
class : org.apache.dubbo.demo.model.Result

联系我们

Apache Dubbo开源讨论群

1157 Members

Scan the QR Code to Join the Group

Thank you !