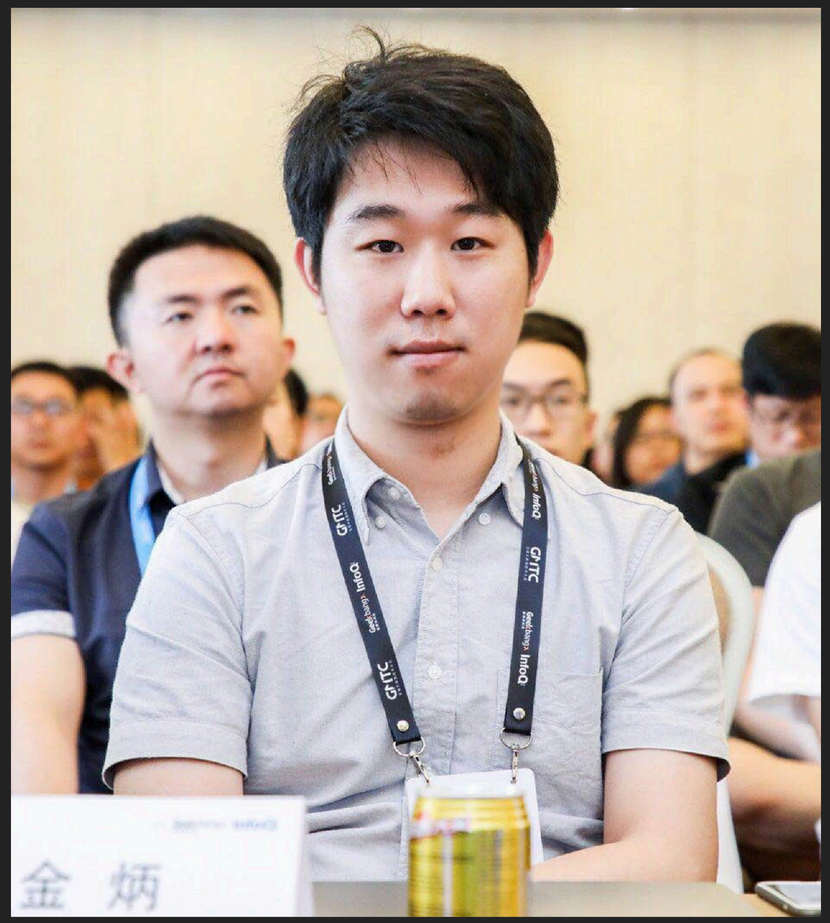
企业级Node建设之基建保障篇

网易-严选事业部

金炳

个人介绍



Name: 金炳

Web developer @网易严选 基础技术部

Github: stone-jin

知乎：金炳

公众号：Node全栈进阶

Doing

目前致力于Node应用框架开发维护及生态建设，
实践Node应用在Serverless、FaaS场景下的迁移和落地，
探索service mesh在Node应用中的价值。

目录

一、应用场景

网易严选
网易旗下自营电商品牌：
原创生活类

产品架构（简化版）


电商前台业务
APP/H5商城
线下店
小程序/PC
渠道平台
....
支撑
电商后台业务
商品中心
订单中心
库存中心
会员中心
促销中心
CRM系统
评价中心
采购中心
WMS
物流中心
风控中心
客服系统
财务管理
营销中心
数据中心
支付中心

Node的应用场景

SSR

BFF

全栈

SSR（引用beidou）

我们以一个直观的表现来看一下同构和非同构的区别。

同构的价值

价值一：性能

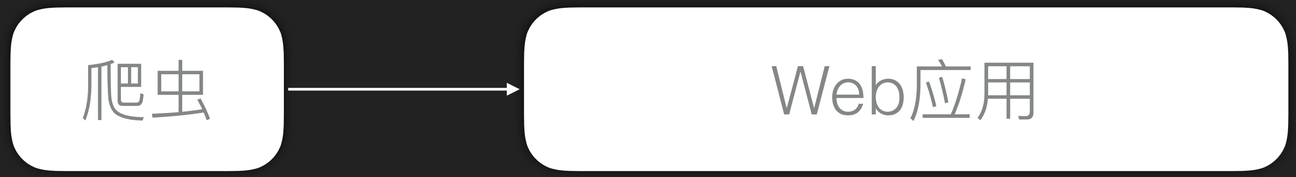
非同构模式

同构模式

通过Node直出，降低首屏渲染时间

价值二：SEO

SEO

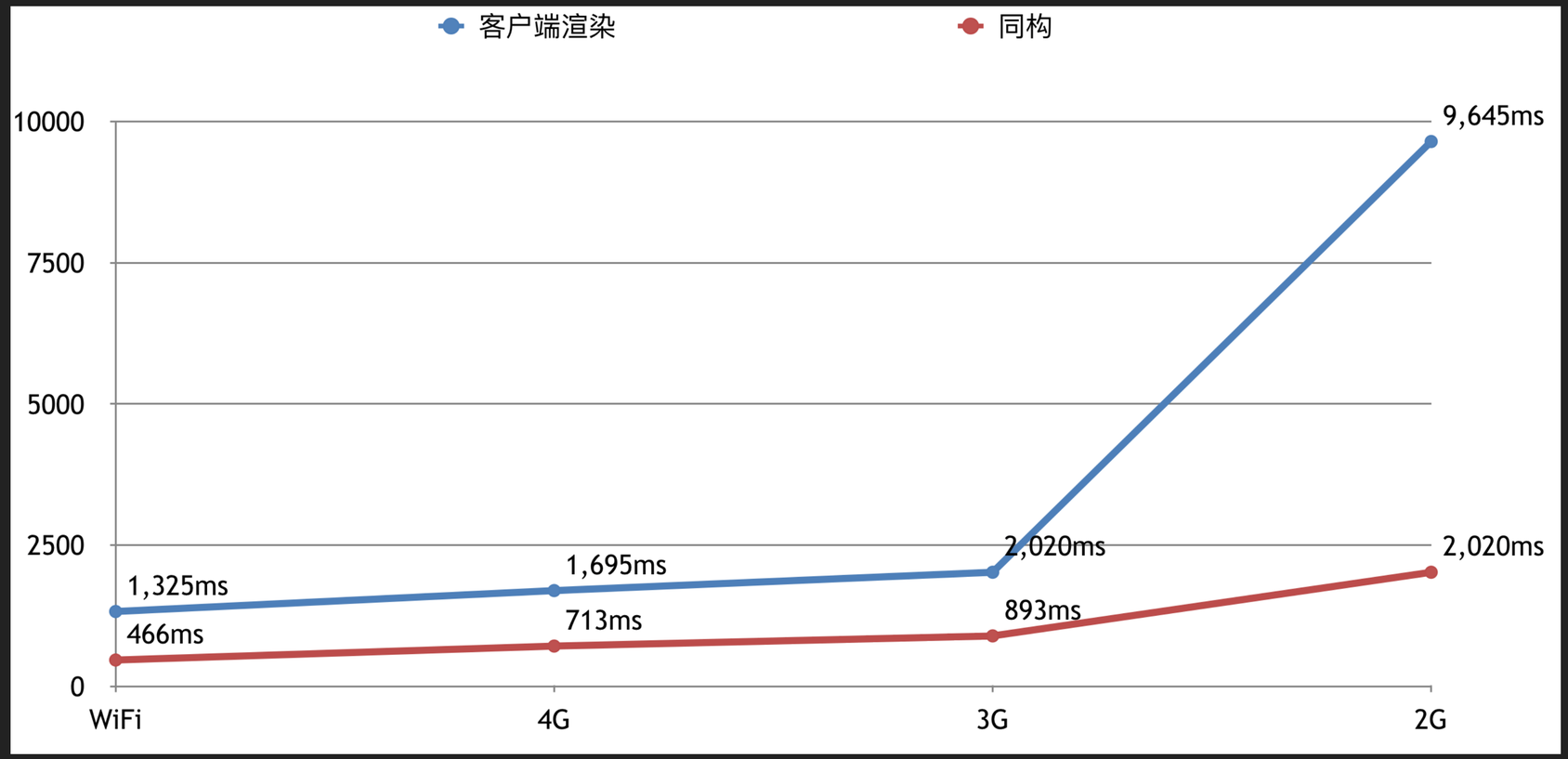


服务端渲染对搜索引擎的爬取有着天然的优势，
虽然电商体系对SEO需求并不强，但是也有很多业务依赖搜索引擎的流量导入

价值三：兼容性

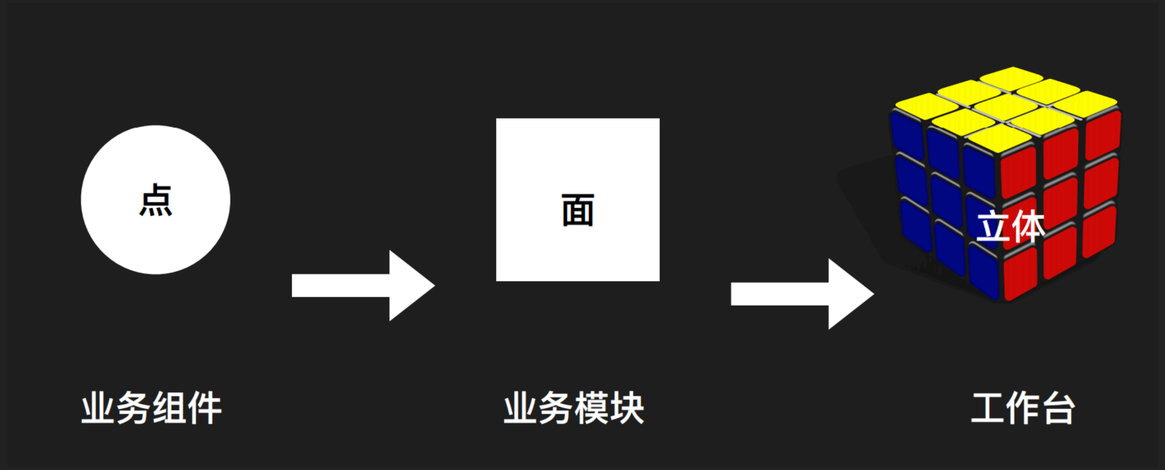
部分展示页，可能规避客户端兼容性问题，比如白屏。

适用场景



通常来说，网络状况越差，同构的优势越明显。

BFF



活动地址：https://fed.duiba.cn/

PPT：查看地址(53页)

视频：https://www.bilibili.com/video/av57520987/

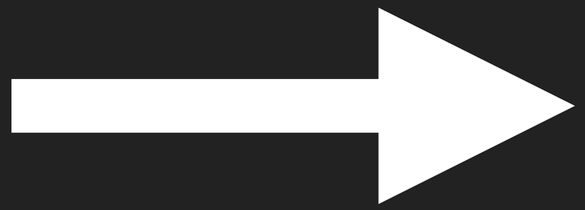
全栈

前端

后端

运维

....



有价值的东西

看了这些Node场景，不知道有没有这样一个问题?

是不是后端语言，都能做这个事情?

答案：是的！

那为什么要搞Node?

Node作为前端最友好的后端语言，你值得拥有！

搞Node或者其他后端时要注意什么?

1、首先多学习一些后端语言，没有错！这样你能找到服务端开发领域的最优解。

2、其次要多花一些精力在通用技能上：数据库、消息队列、高可用性、微服务治理。

3、最后不要花太多精力学每个语言的Web框架怎么用，这没有意义。

所以让我们一起学习通用技能：基建

涉密！！！

第二章：以对应方向的开源方案讲解

具体方案，可以做线下沟通

二、基建

2.0 Docker

背景：

雪花服务器	手动修改的nginx配置、代理、第三方SDK，你也不知道服务器环境现在装了什么了？
时间侵蚀	磁盘写满了，机房带宽有限了、硬件老化了等等
集群漂移	修改了Node版本了，修改了日志配置
Node.js特殊性	没有可直接运行的二进制包、依赖环境运行时、node_modules黑洞

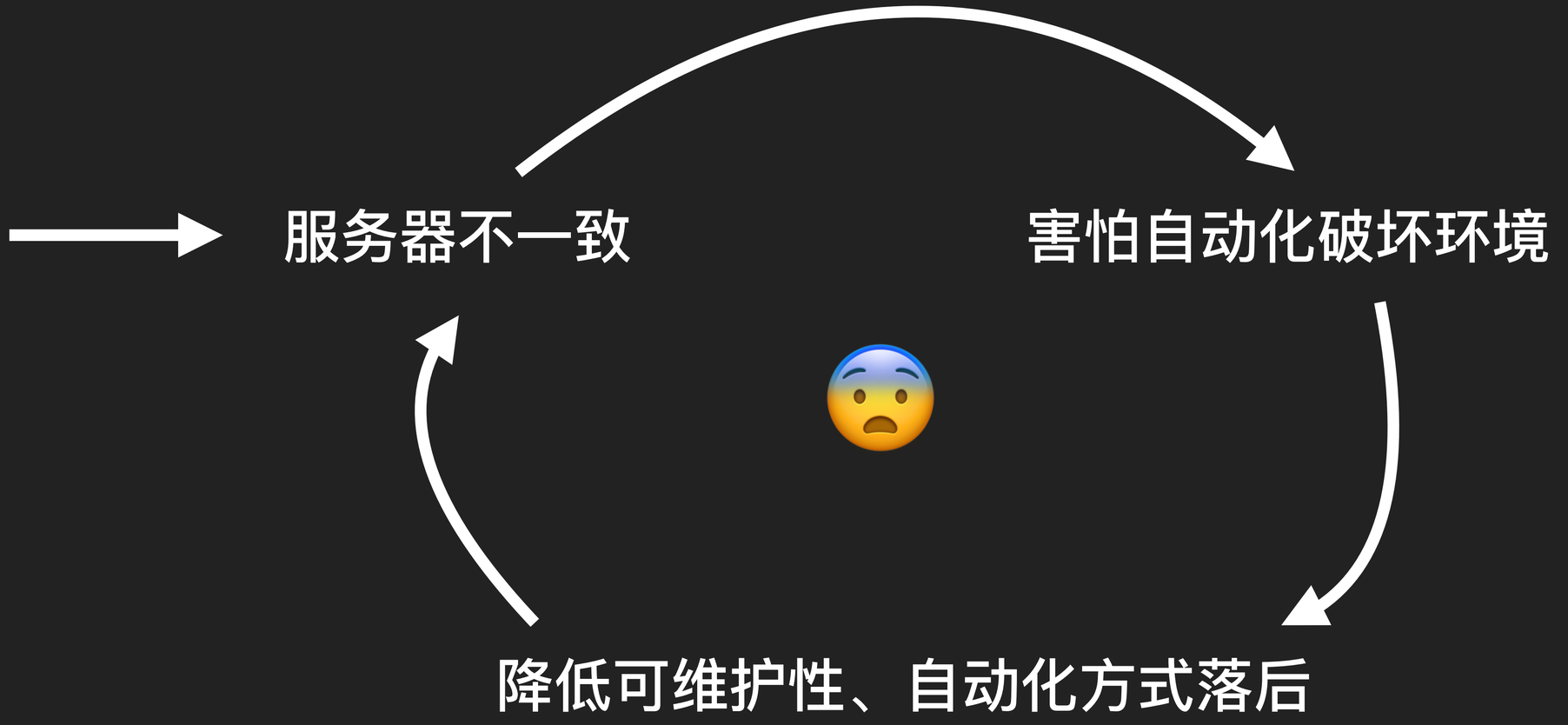
结果：

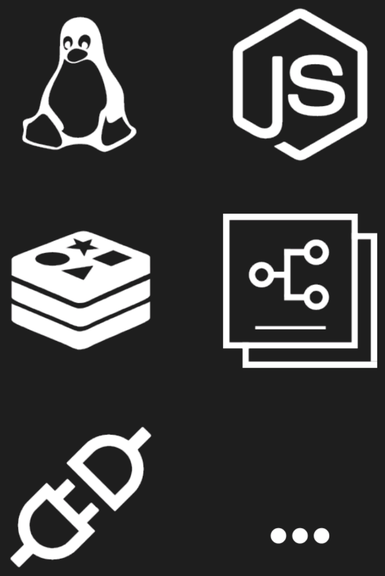
雪花服务器

时间侵蚀

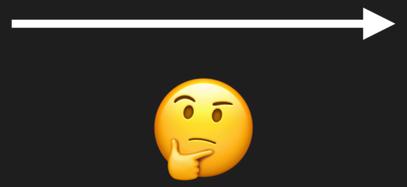
集群漂移

Node.js特殊性





设施难以维护



```
FROM node:10
WORKDIR /usr/src/app
COPY package*.json ./
RUN export http_proxy=...
RUN npm install
...
```

代码容易维护

使用Dockerfile定义运行环境，提高可维护性、扩展性。

2.1 CI/CD

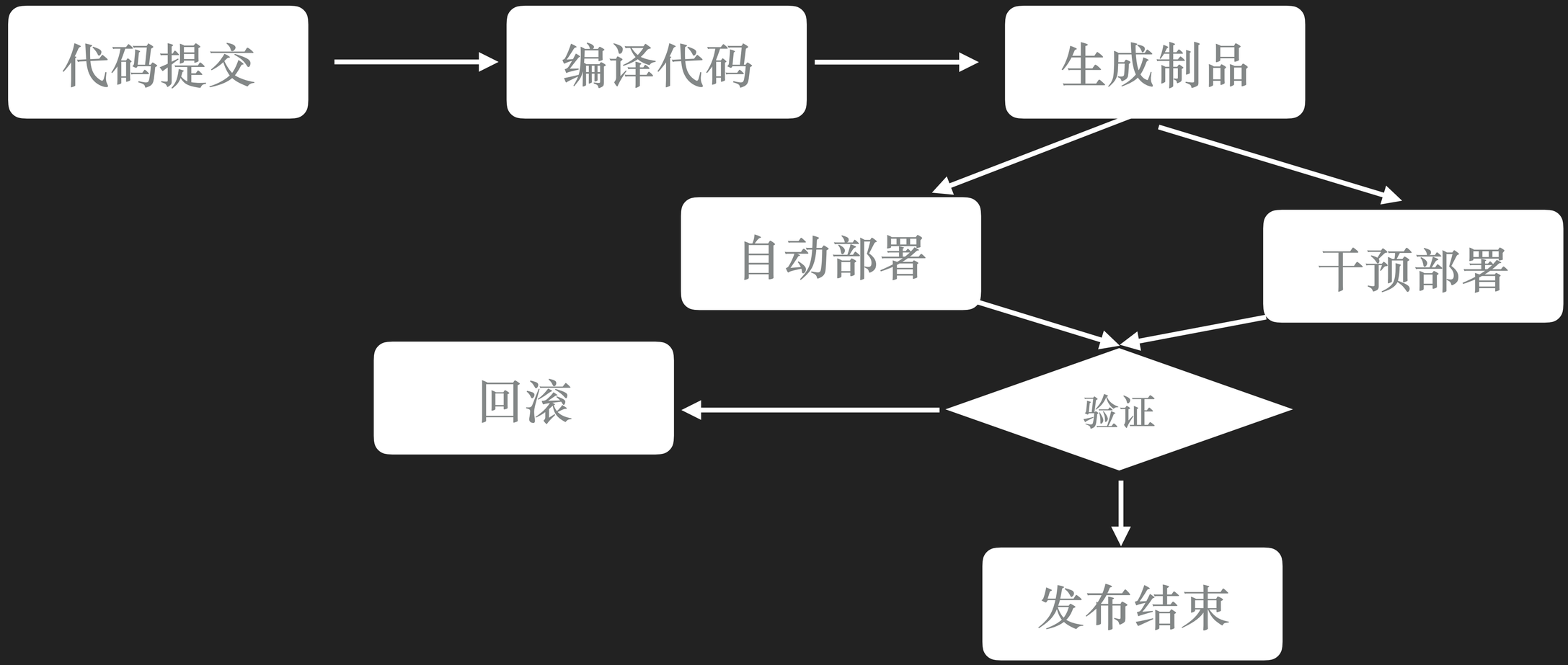
2.1 CI/CD

CI(持续集成）是一种软件开发实践，每次代码提交都通过自动化的构建。

CD(持续部署）是通过自动化的方式去部署这样的过程。

以gitlab-ci为例：

我们在项目中添加一个.gitlab-ci.yaml文件



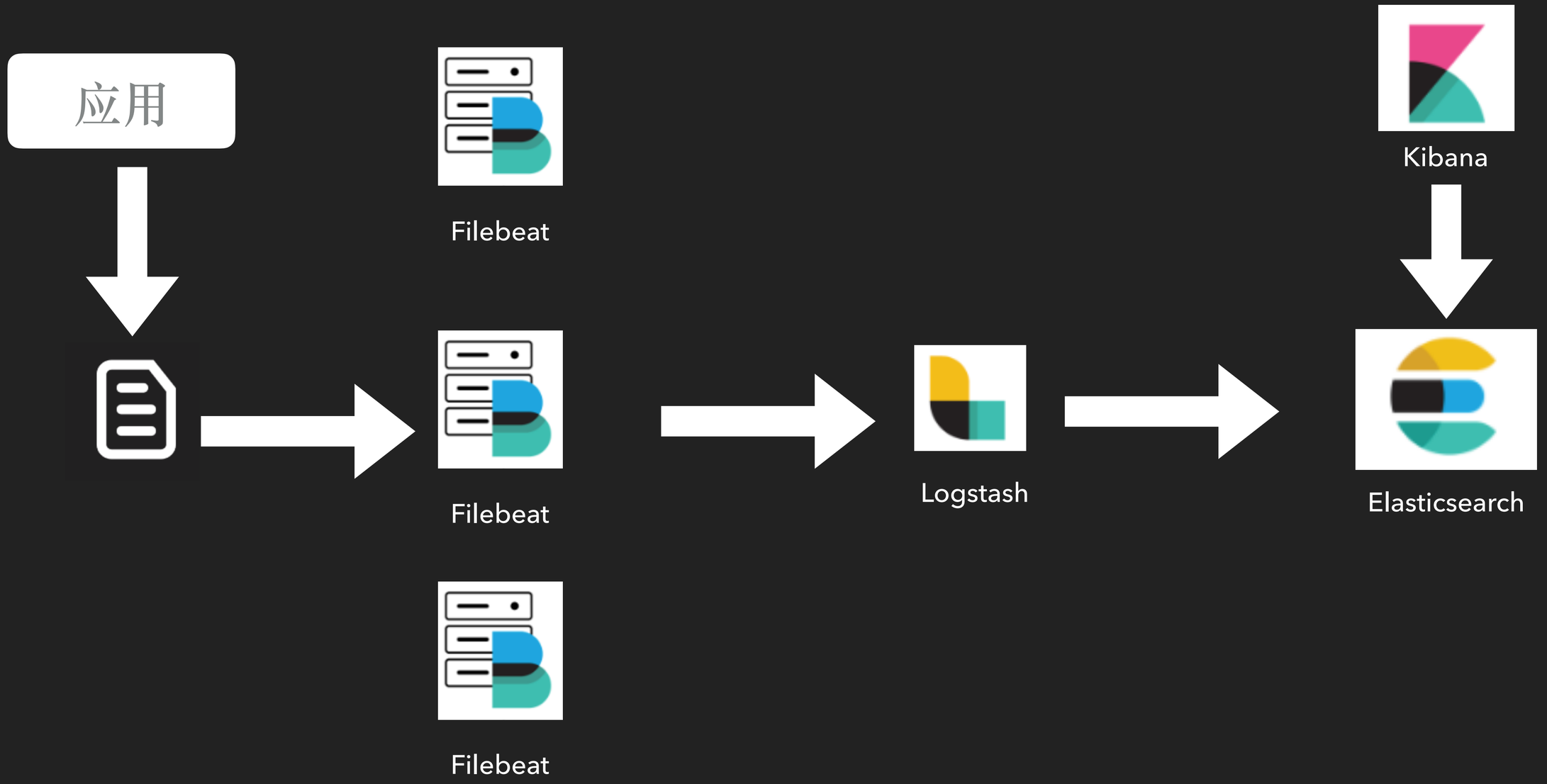
学会这种CI/CD类型的技能，让我们发布和回滚自动化。

2.2 日志平台

2.2 日志平台

线上跑着的，肯定有很多服务以及服务集群

以ELK方案为例：


应用
Filebeat
Filebeat
Filebeat
Logstash
Kibana
Elasticsearch

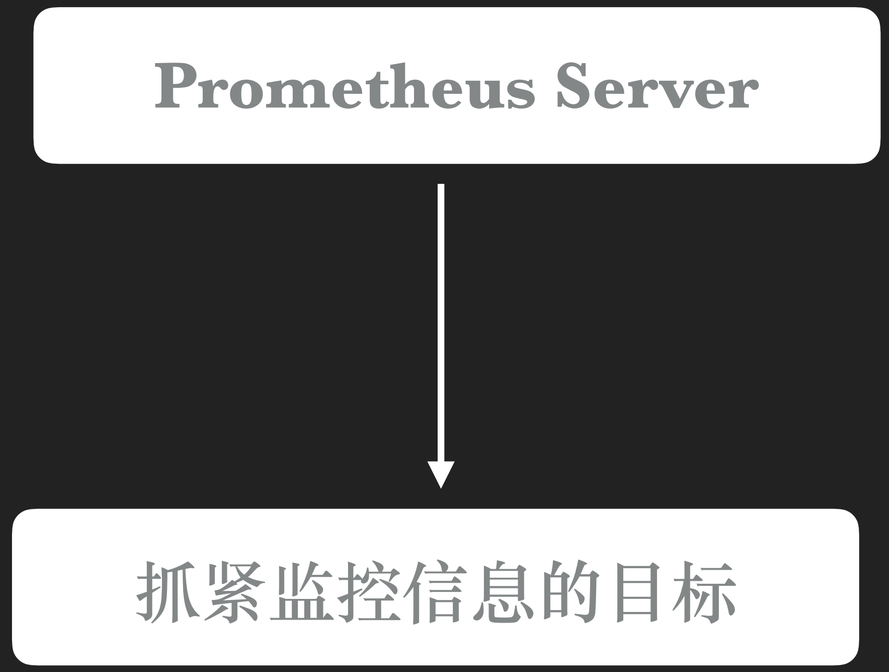
通过日志方案，将负责的项目的日志，聚合到一个平台进行查询。

2.3 监控与报警

2.3 监控与报警

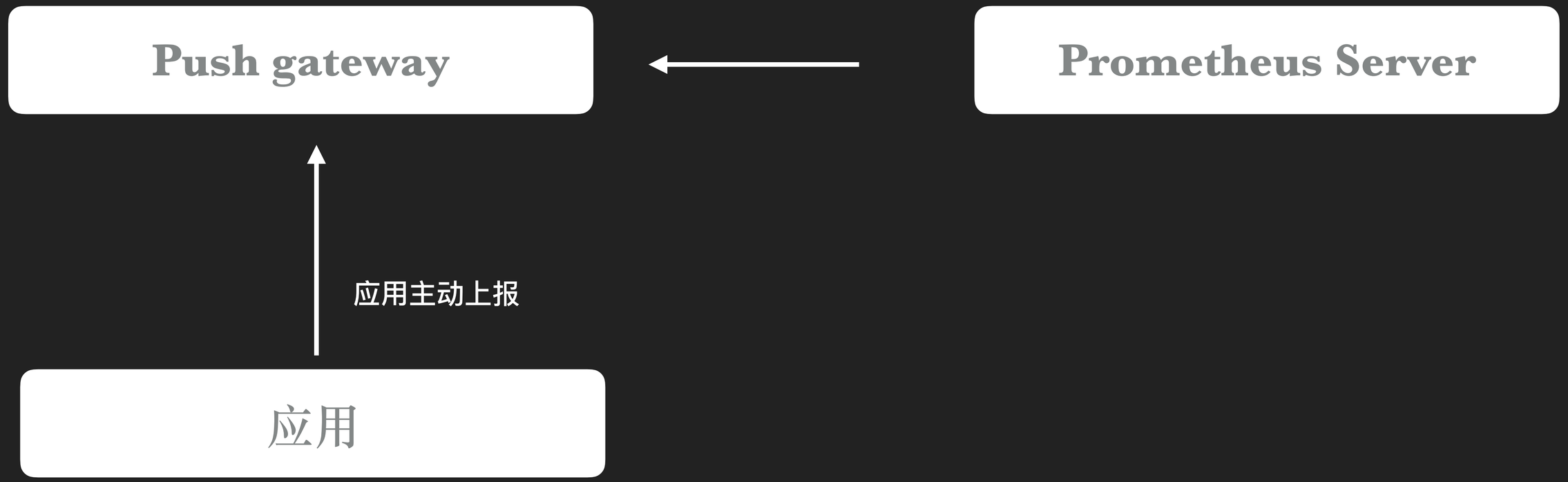
以Prometheus方案为例：

步骤一：



提供一个/metrics的接口，让服务端通过pull的形式来拉取

步骤二：　（不能拉取的部分，我们也可以推）

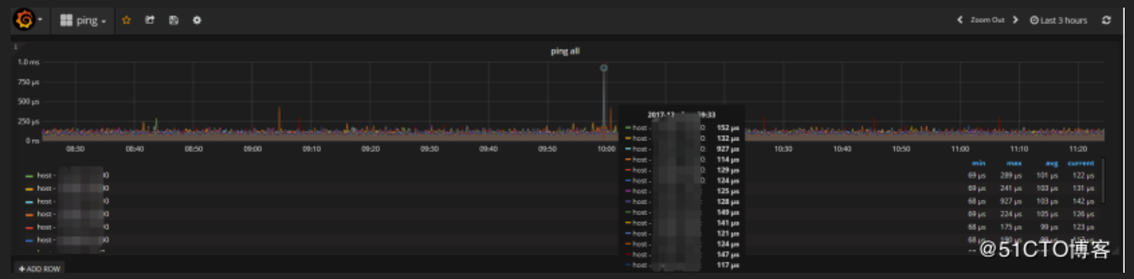


步骤三： 数据展示

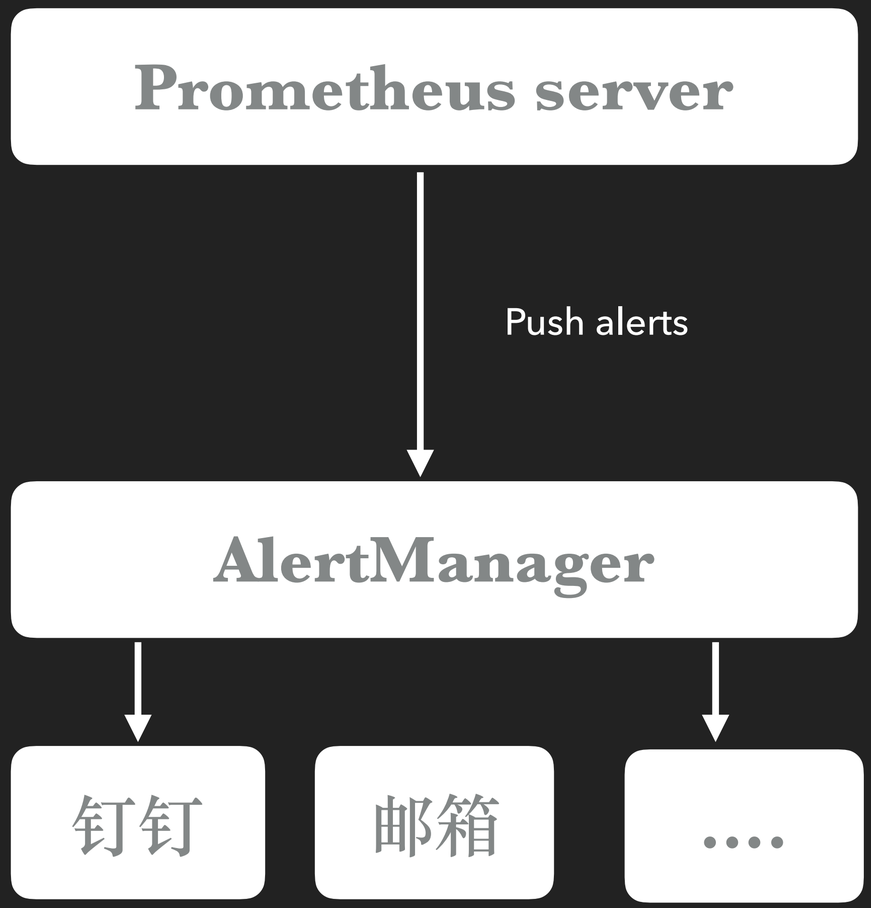
Grafana

PromQL查询

Prometheus server



步骤三： 数据展示



报警类型: FIRING

startsAt: 2019-03-11T09:42:13.131908754Z
job: kube-state-metrics
deployment: prometheus-webhook-dingtalk
instance: 10.46.0.0:8080
namespace: default
pod: p-kube-state-metrics-57d788d69-6jtw8
severity: critical
alertname: KubeDeploymentReplicasMismatch
service: p-kube-state-metrics
Describe: Deployment default/prometheus-webhook-dingtalk has not matched the expected number of replicas for longer than an hour.

报警类型: RESOLVED

startsAt: 2019-03-11T09:42:13.131908754Z
pod: p-kube-state-metrics-57d788d69-6jtw8
namespace: default
deployment: prometheus-webhook-dingtalk
job: kube-state-metrics
severity: critical
alertname: KubeDeploymentReplicasMismatch
service: p-kube-state-metrics
instance: 10.46.0.0:8080
Describe: Deployment default/prometheus-webhook-dingtalk has not matched the expected number of replicas for longer than an hour.

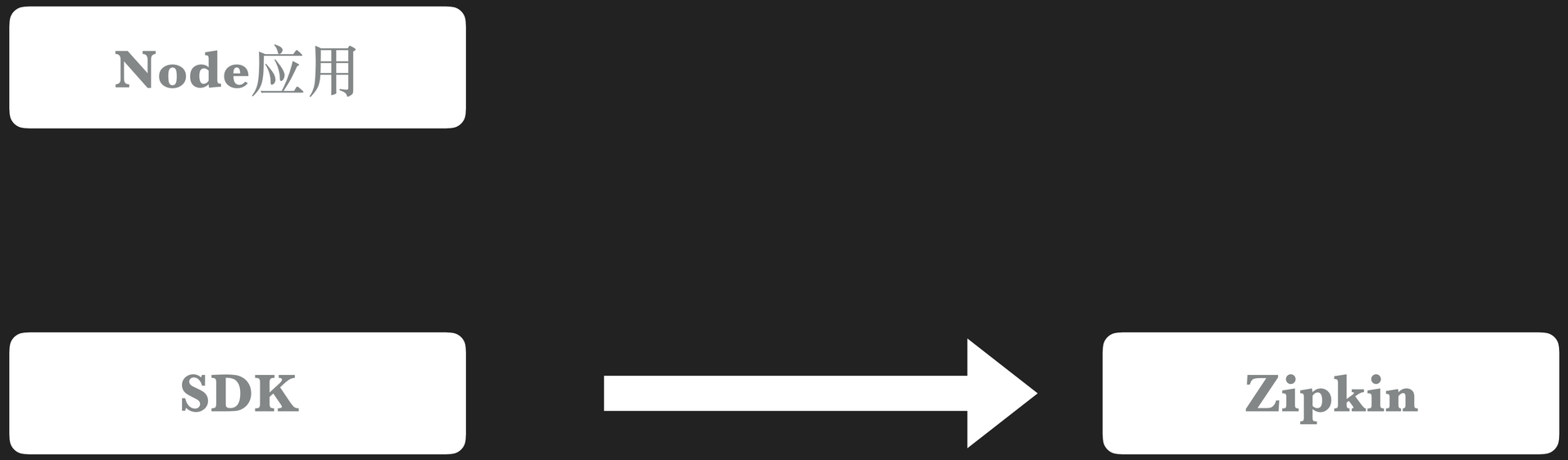
利用这类监控报警方案，让你的服务出错时，即时报警解决。

2.4 链路跟踪

2.4 链路跟踪

服务出错时，如何快速排错，我们Node接入一下链路跟踪

以Zipkin方案为例：



错误排查过程：

慢接口排查过程：

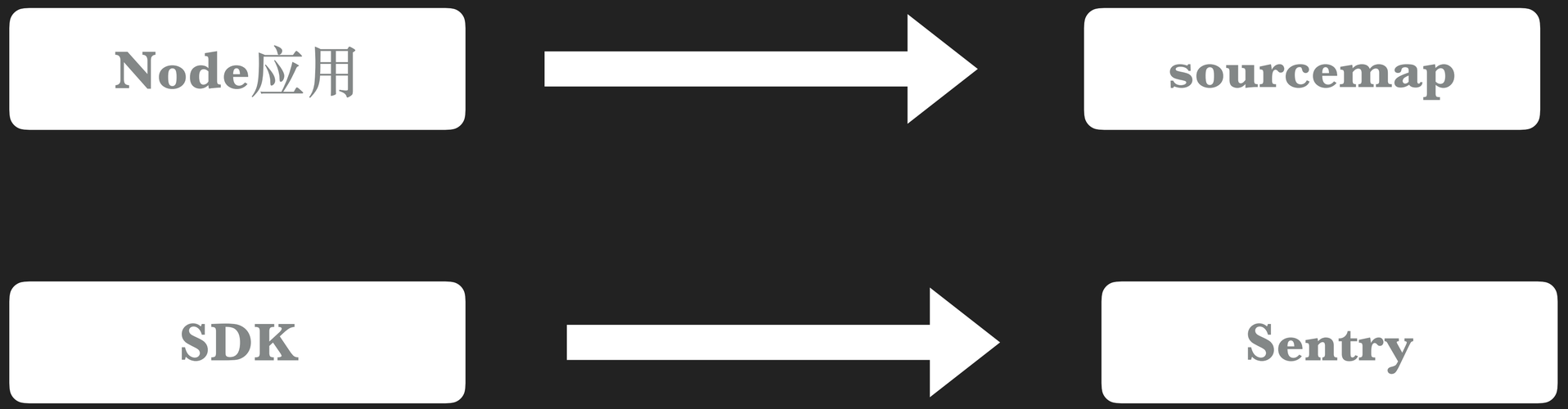
通过链路跟踪，我们知道了哪个服务出错了，哪些是慢接口。

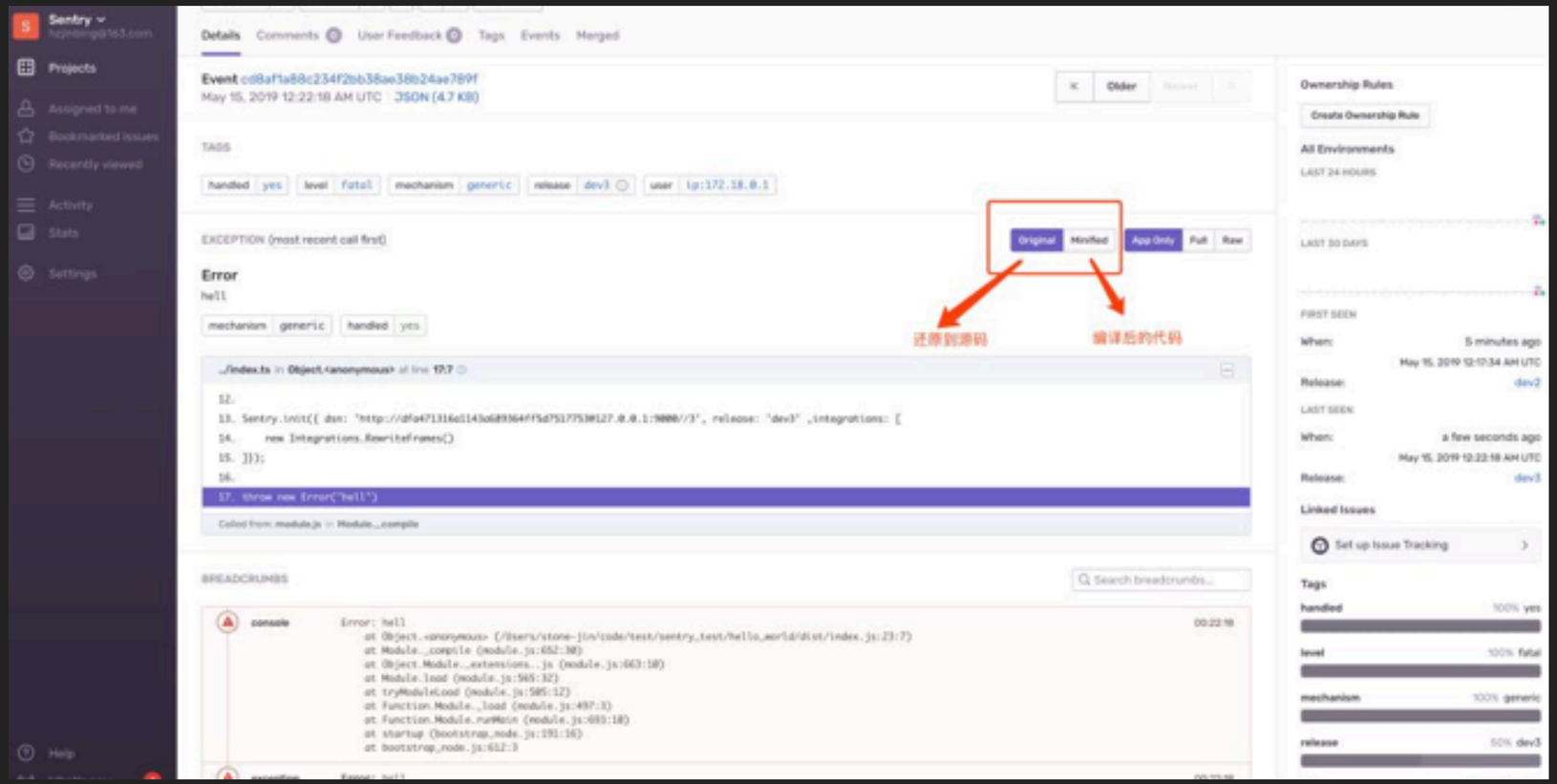
2.5 错误监控

2.5 错误监控

服务出错时，快速定位对应错误的堆栈。

以Sentry方案为例：



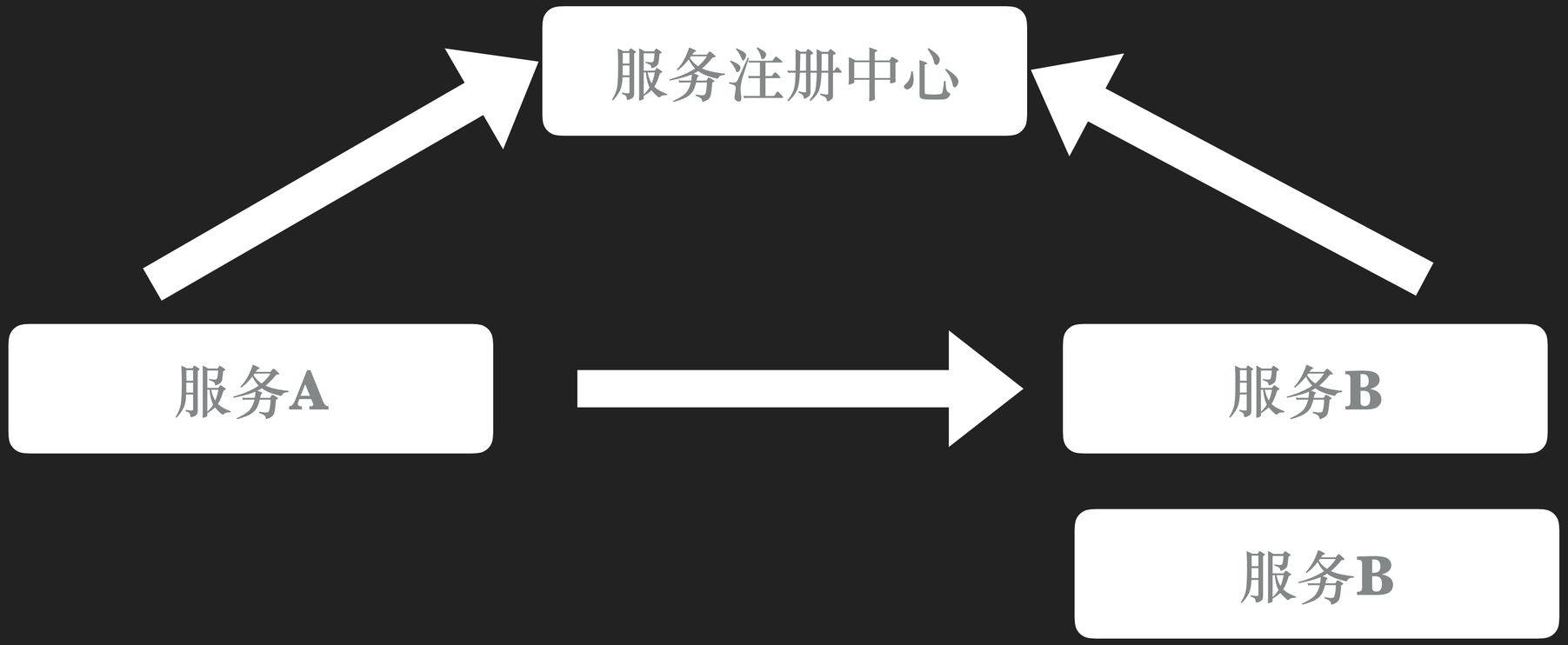


通过错误监控，快速查看错误的代码堆栈。

2.6 服务注册发现

2.6 服务注册发现

服务注册到服务注册中心，进行调用

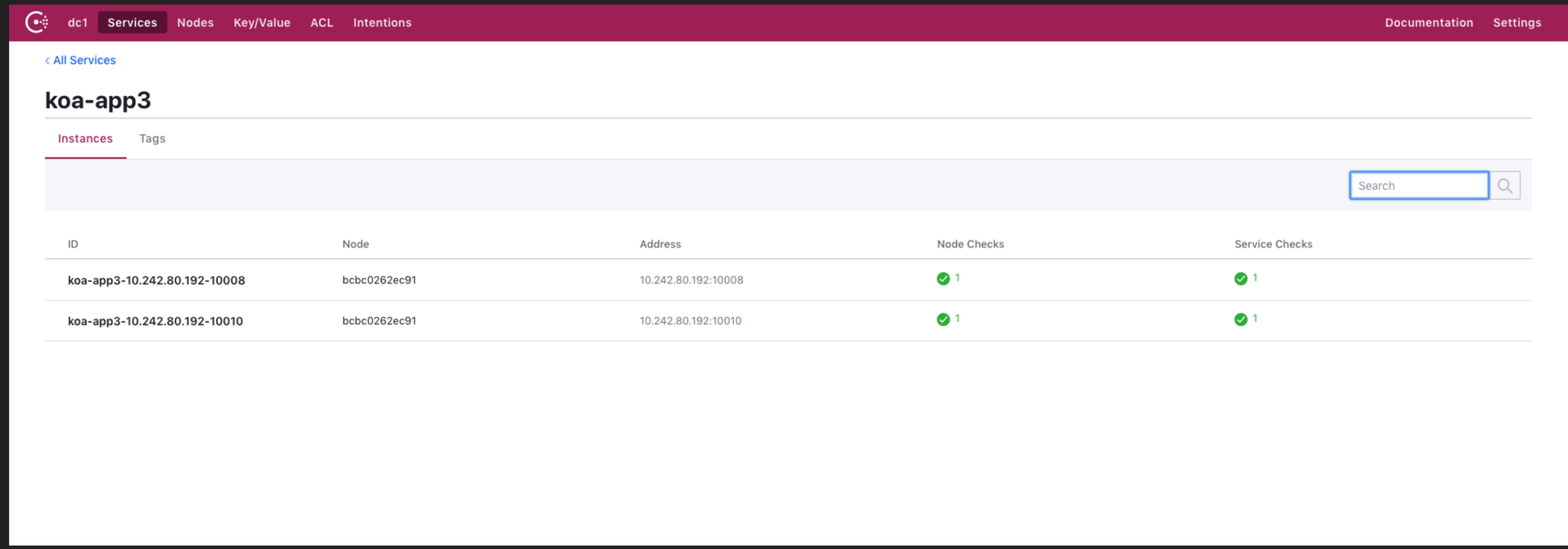


以consul方案为例：

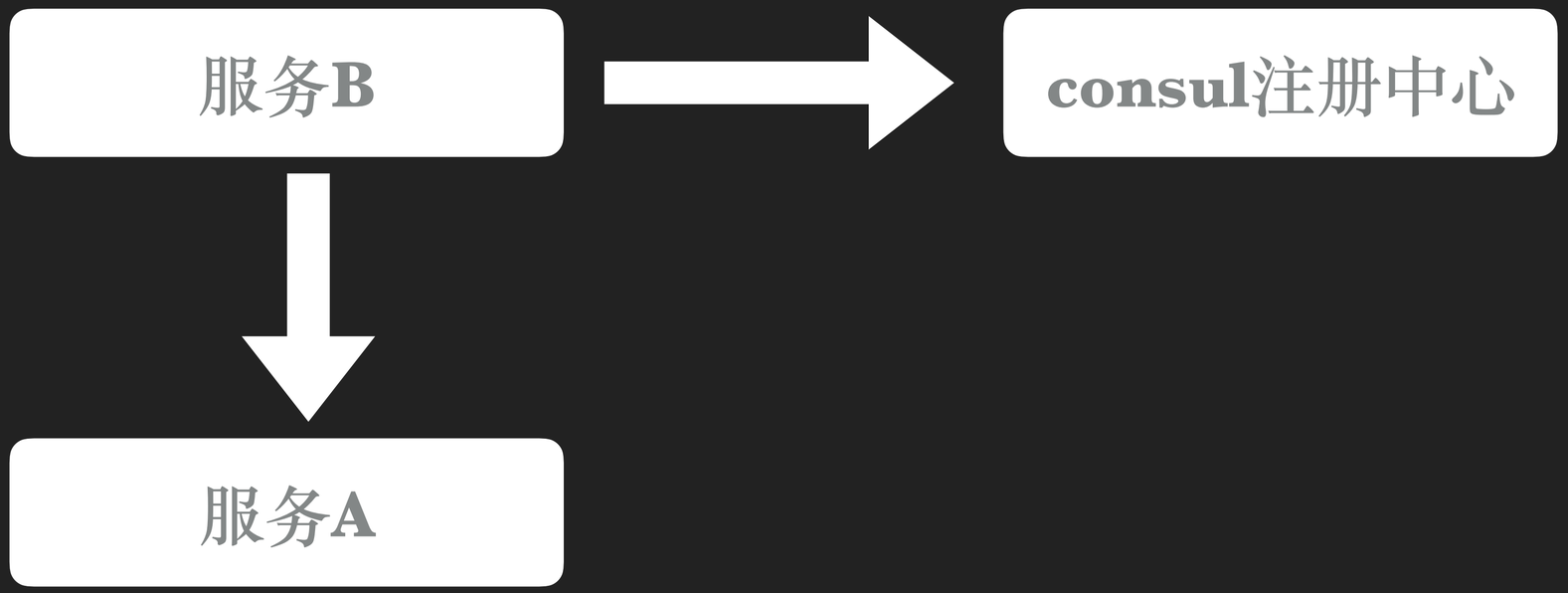
注册过程：

consul注册中心

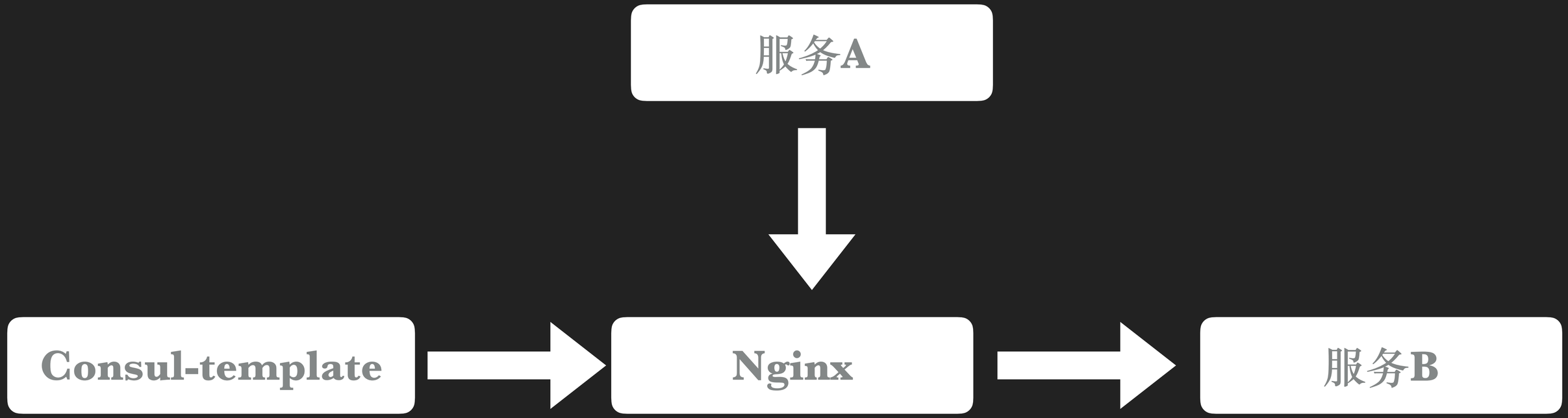
服务A



调用过程：



另外结合Nginx + consul-template：

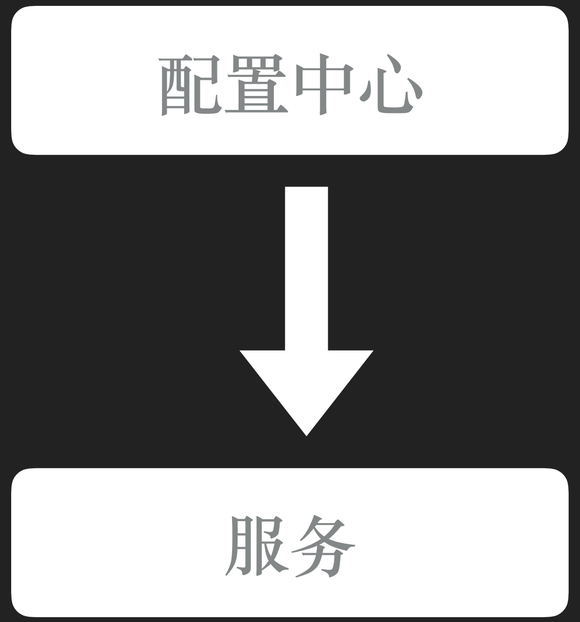


通过服务注册发现，提高我们服务的高可用性

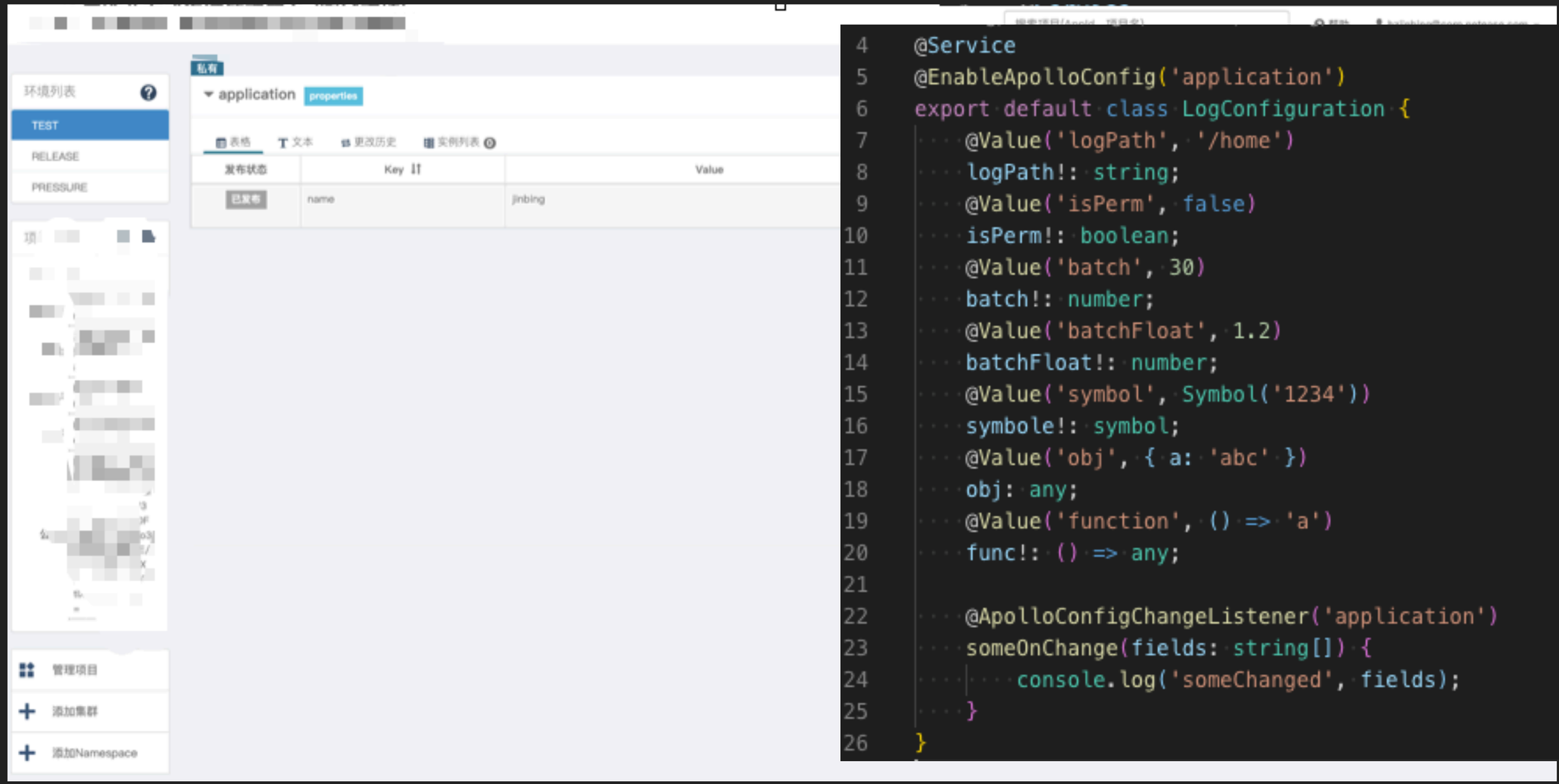
2.7 配置中心

2.7 配置中心

通过配置中心，动态配置服务的配置项



以apollo方案为例：




```
@Service
@EnableApolloConfig('application')
export default class LogConfiguration {
    @Value('logPath', '/home')
    logPath!: string;
    @Value('isPerm', false)
    isPerm!: boolean;
    @Value('batch', 30)
    batch!: number;
    @Value('batchFloat', 1.2)
    batchFloat!: number;
    @Value('symbol', Symbol('1234'))
    symbole!: symbol;
    @Value('obj', { a: 'abc' })
    obj: any;
    @Value('function', () => 'a')
    func!: () => any;

    @ApolloConfigChangeListener('application')
    someOnChange(fields: string[]) {
        console.log('someChanged', fields);
    }
}
```

通过配置中心，动态改变服务的配置项。

三、总结

那些伴随你程序员生涯的技能，值得你多花一些时间！