



携程技术中心

# 携程技术沙龙

## 开源配置中心Apollo的设计与实现

分享人：宋顺



## 宋顺

- 携程框架研发部技术专家。
- 2016年初加入携程，主要负责中间件产品的相关研发工作。
- 毕业于复旦大学软件工程系，曾就职于大众点评，担任后台系统技术负责人。

# 目录

CONTENTS

- 1 What is Apollo
- 2 Why Apollo
- 3 Apollo at a glance
- 4 Apollo in depth

- 小A是XX团队主力开发，有一天产品说要上线一个迪士尼门票内购功能
- 由于迪士尼门票很火爆，产品一拍脑袋说，每个用户限购5张！
- 于是小A在代码里是这么写的

```
private static final int MAX_QTY_PER_USER = 5; //产品需求限购5张
if (qty > MAX_QTY_PER_USER) {
    throw new IllegalStateException(
        String.format("每个用户最多购买%d张!", MAX_QTY_PER_USER));
}
```

- 第二天中午，由于内购实在太火爆，产品急匆匆的跑过来对小A说，赶紧改成每人1张！
- 小A只好放弃了午饭，改代码、回归测试、上线，整整花了1个小时才搞定。。。



- 小B是YY团队主力开发，有一天产品说要上线一个欢乐谷门票内购功能
- 由于欢乐谷门票很火爆，产品一拍脑袋说，每个用户限购5张！
- 小B吸取了小A的教训，二话不说把配置写在了Apollo配置中心

Key	Value	备注
max-qty-per-user	5	产品需求限购5张

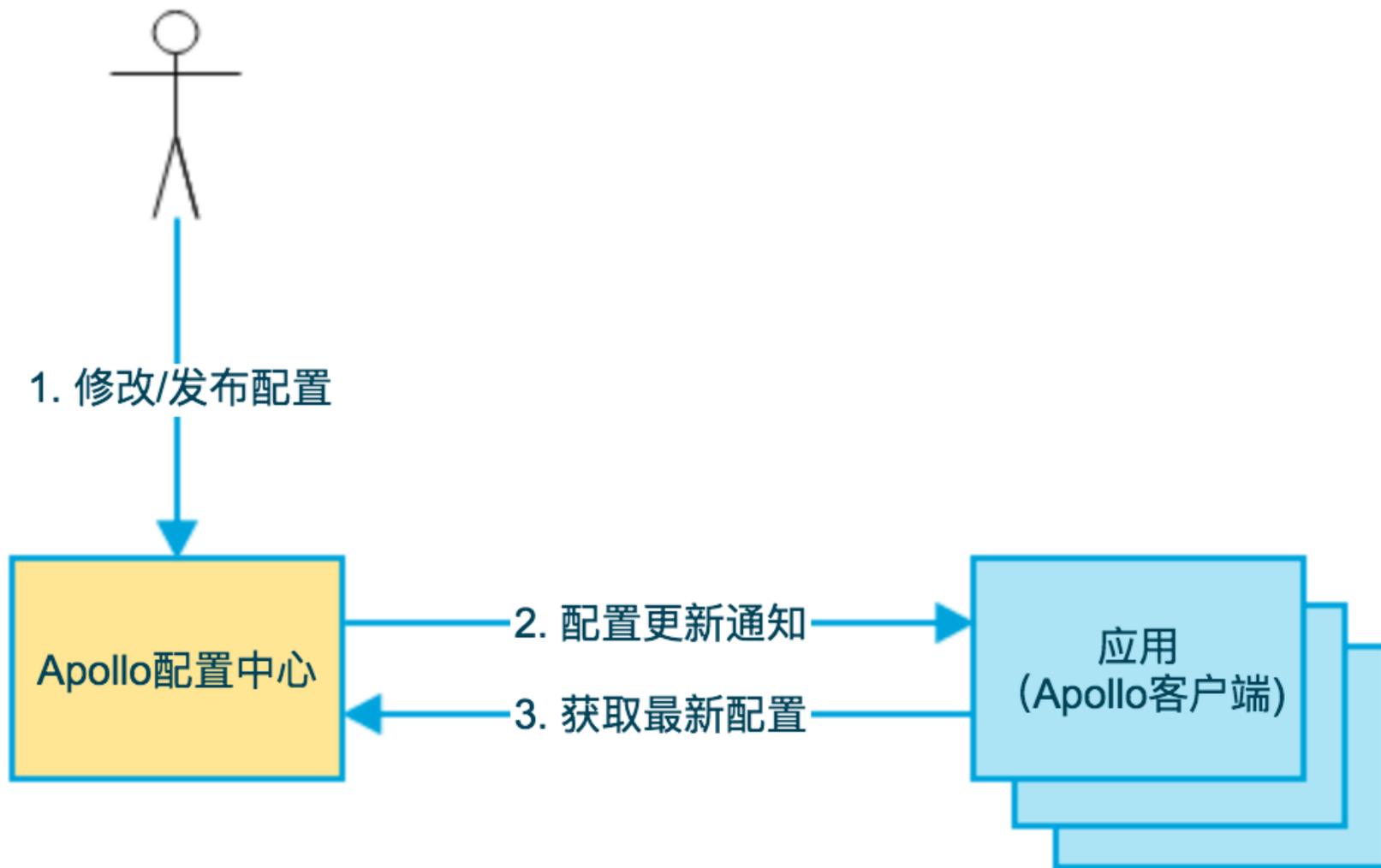
- 第二天中午，由于内购实在太火爆，产品急匆匆的跑过来对小B说，赶紧改成每人1张！
- 小B不紧不慢的说：10秒内搞定~



- 携程框架部门开源的统一应用配置中心
- 支持4个维度管理配置 ( Key-Value )
  - application (应用)
  - environment (环境)
  - cluster (集群)
  - namespace (命名空间)

- 配置是独立于程序的只读变量
  - DB Connection Str、Thread Pool Size、Buffer Size、Request Timeout、Feature Switch、Server Urls等
- 配置伴随应用的整个生命周期
  - 启动时读取配置，运行时根据配置调整行为
- 配置可以有多种加载方式
  - 程序内部hard code，配置文件，环境变量，启动参数，基于数据库等
- 配置需要治理
  - 权限控制、发布审核
  - 不同环境、集群配置管理

- 有治理能力的配置管理平台
  - 统一管理不同环境、不同集群的配置
  - 配置修改实时生效（热发布）
  - 版本发布管理
  - 灰度发布
  - 权限管理、发布审核、操作审计
  - 客户端配置信息监控
  - Java, .Net原生客户端, Spring支持



## Apollo 配置中心

[帮助](#) song\_s ▾

**环境列表** ?

FAT

**default** 集群

dev 集群

lpt 集群

UAT

PRO

default 集群

SHAOY 集群

SHAJQ 集群

**项目信息** ★

AppId: 100004458

应用名: apollo-demo

部门: 框架(FX)

负责人: song\_s

Email: song\_s@ctrip.com

[+ 添加集群](#)

[+ 添加Namespace](#)

[项目权限](#)

**私有**

[主版本](#) [灰度版本](#)

**application** [properties](#) 发布 回滚 发布历史 授权

[表格](#) [文本](#) [更改历史](#) [实例列表](#) 1

[过滤配置](#) [同步配置](#) [新增配置](#)

发布状态	Key ↑↓	Value	备注	最后修改人 ↑↓	最后修改时间 ↑↓	操作
已发布	timeout <span>灰</span>	3000		song_s	2017-01-18 19:03:59	<a href="#">编辑</a> <a href="#">删除</a>
已发布	kibana.url	http://1.1.1.2:5600		song_s	2016-11-25 20:57:27	<a href="#">编辑</a> <a href="#">删除</a>
已发布	elastic.cluster.name	es-cluster		song_s	2016-10-18 19:57:29	<a href="#">编辑</a> <a href="#">删除</a>
已发布	page.size	20		song_s	2016-12-27 14:58:56	<a href="#">编辑</a> <a href="#">删除</a>
已发布	zookeeper.address	10.1.12.2		song_s	2016-10-19 11:33:50	<a href="#">编辑</a> <a href="#">删除</a>

**关联**

**FX.apollo** [properties](#) 发布 回滚 发布历史 授权 创建灰度

[表格](#) [文本](#) [更改历史](#) [实例列表](#) 1

[同步配置](#)

覆盖的配置 filter by key ...

发布状态	Key ↑↓	Value	备注	最后修改人 ↑↓	最后修改时间 ↑↓	操作
已发布	servers	3.3.3.3,4.4.4.4		song_s	2017-01-17 15:11:06	<a href="#">编辑</a> <a href="#">删除</a>

公共的配置 (AppId:100003173, Cluster:default) filter by key ...

Key ↑↓	Value	备注	最后修改人 ↑↓	最后修改时间 ↑↓	操作
batch	2000	样例项目会使用到, 勿删。	song_s	2017-01-17 15:11:31	<a href="#">删除</a>
servers	1.1.1.1,2.2.2.2	样例项目会使用到, 勿删。	song_s	2016-10-12 14:03:34	

## • 添加/修改配置项

### Apollo 配置中心

[帮助](#)[song\\_s](#)[环境列表](#)

FAT

UAT

PRO

default

集群

SHAQY

集群

SHAQJ

集群

[项目信息](#)

AppId: 100004458

应用名: apollo-demo

部门: 框架(FX)

负责人: song\_s

负责人Email: song\_s@ctrip.com

application **properties**

Key ↑↓	Value	备注	最后修改人 ↑↓	最后修改时间 ↑↓	操作
request.timeout	200	请求超时时间 (毫秒)	song_s	2016-10-18 19:56:26	<input checked="" type="button" value="编辑"/> <input type="button" value="删除"/>
kibana.url	http://1.1.1.2:5601		song_s	2016-10-18 19:57:29	<input type="button" value="编辑"/> <input type="button" value="删除"/>
elastic.document.type	biz		song_s	2016-10-18 19:57:29	<input type="button" value="编辑"/> <input type="button" value="删除"/>
elastic.cluster.name	es-cluster		song_s	2016-10-18 19:57:29	<input type="button" value="编辑"/> <input type="button" value="删除"/>
elastic.cluster	2.2.2.2:9300,3.3.3.3,4.4.4.4:9300		song_s	2016-10-18 19:57:29	<input type="button" value="编辑"/> <input type="button" value="删除"/>
page.size	10		song_s	2016-10-18 19:57:29	<input type="button" value="编辑"/> <input type="button" value="删除"/>
zookeeper.address	10.1.12.1		song_s	2016-10-18 19:57:29	<input type="button" value="编辑"/> <input type="button" value="删除"/>

- 添加/修改配置项

The screenshot displays the Apollo Configuration Center interface. A modal dialog titled '修改配置项' (Modify Configuration Item) is open, showing the following details:

- Key:** request.timeout
- \* Value:** 100
- Comment:** 请求超时时间 (毫秒)

At the bottom of the dialog, there are two buttons: '关闭' (Close) and '提交' (Submit). The '提交' button is highlighted with a red border.

In the background, the main interface shows a sidebar with environment lists (FAT, UAT, PRO, default, SHAOY, SHAJQ) and project information (AppId: 100004458, 应用名: apollo-demo, 部门: 框架(FX), 负责人: song\_s). The main content area displays a table of configuration items:

Key	Value	App	Last Modified Time	Operation
elastic.cluster	2.2.2.2:9300,3.3.3.3,4.4.4.4:9300	song_s	2016-10-18 19:57:29	[Edit] [Delete]
page.size	10	song_s	2016-10-18 19:57:29	[Edit] [Delete]
zookeeper.address	10.1.12.1	song_s	2016-10-18 19:57:29	[Edit] [Delete]

## • 发布配置

### Apollo 配置中心

环境列表



FAT

UAT

PRO

default

集群

SHAOY

集群

SHAJQ

集群

项目信息



AppId: 100004458

应用名: apollo-demo

部门: 框架(FX)

负责人: song\_s

application **properties** 有修改 1

发布

回滚

发布历史

授权

同步配置

表格

文本

更改历史

实例列表 4

过滤配置

新增配置

Key ↑↓	Value	备注	最后修改人 ↑↓	最后修改时间 ↑↓	操作
request.timeout	100	请求超时时间 (毫秒)	song_s	2016-10-18 19:59:51	
kibana.url	http://1.1.1.2:5601		song_s	2016-10-18 19:57:29	
elastic.document.type	biz		song_s	2016-10-18 19:57:29	
elastic.cluster.name	es-cluster		song_s	2016-10-18 19:57:29	
elastic.cluster	2.2.2.2:9300,3.3.3.3,4.4.4.4:9300		song_s	2016-10-18 19:57:29	
page.size	10		song_s	2016-10-18 19:57:29	
zookeeper.address	10.1.12.1		song_s	2016-10-18 19:57:29	

- 发布配置

Apollo 配置中心

应用ID/应用名  Go 帮助 song\_s

环境列表

- FAT
- UAT
- PRO
- default
- SHAOY
- SHAJQ

项目信息

AppId: 10  
应用名: ap  
部门: 框  
负责人: song\_s

发布

Changes	Key	Old Value	New Value	最后修改人	最后修改时间
	request.timeout	200	100	song_s	2016-10-18 19:59:51

\* Release Name:

Comment:

关闭 发布

- 客户端获取配置 ( Java API样例 )

```
Config config = ConfigService.getAppConfig();  
Integer defaultRequestTimeout = 200;  
Integer requestTimeout =  
    config.getIntProperty("request.timeout", defaultRequestTimeout);
```

- 客户端获取配置 ( Java API样例 )

```
Config config = ConfigService.getAppConfig();
config.addChangeListener(new ConfigChangeListener() {
    @Override
    public void onChange(ConfigChangeEvent changeEvent) {
        for (String key : changeEvent.changedKeys()) {
            ConfigChange change = changeEvent.getChange(key);
            System.out.println(String.format(
                "Found change - key: %s, oldValue: %s, newValue: %s, changeType: %s",
                change.getPropertyName(), change.getOldValue(),
                change.getNewValue(), change.getChangeType()));
        }
    }
});
```

- Spring集成样例

```
@Configuration
@EnableApolloConfig
public class AppConfig {}

@Component
public class SomeBean {
    @Value("${request.timeout:200}")
    private int timeout;

    @ApolloConfigChangeListener
    private void someChangeHandler(ConfigChangeEvent changeEvent) {
        if (changeEvent.isChanged("request.timeout")) {
            refreshTimeout();
        }
    }
}
```

- Core Concepts

- application (应用)

- 使用配置的应用

- 有唯一标识appId :

- Java: classpath:/META-INF/app.properties -> app.id

- .Net: app.config -> AppID

- environment (环境)

- 配置对应的环境

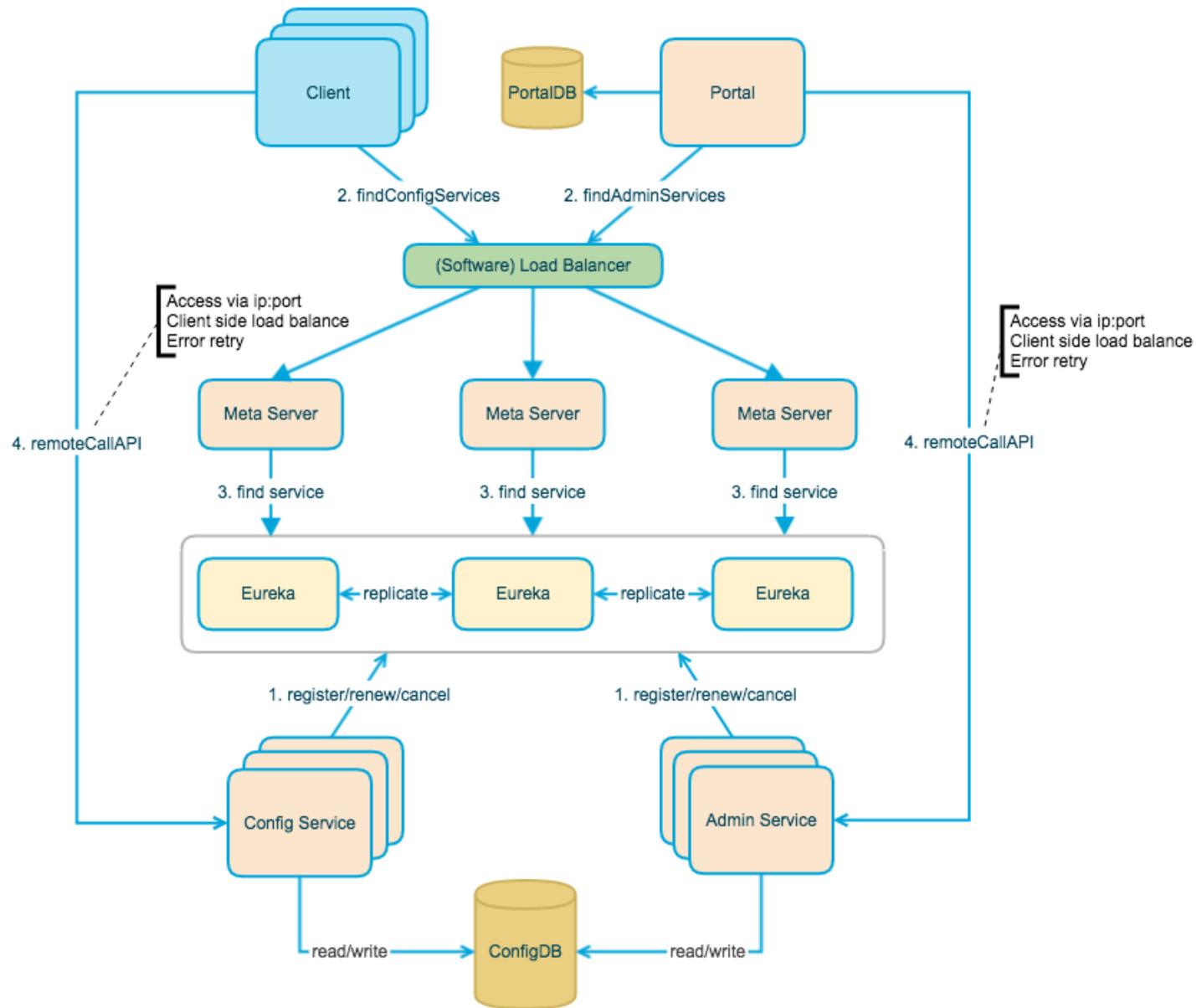
- DEV, FAT, UAT, PRO :

- server.properties -> env

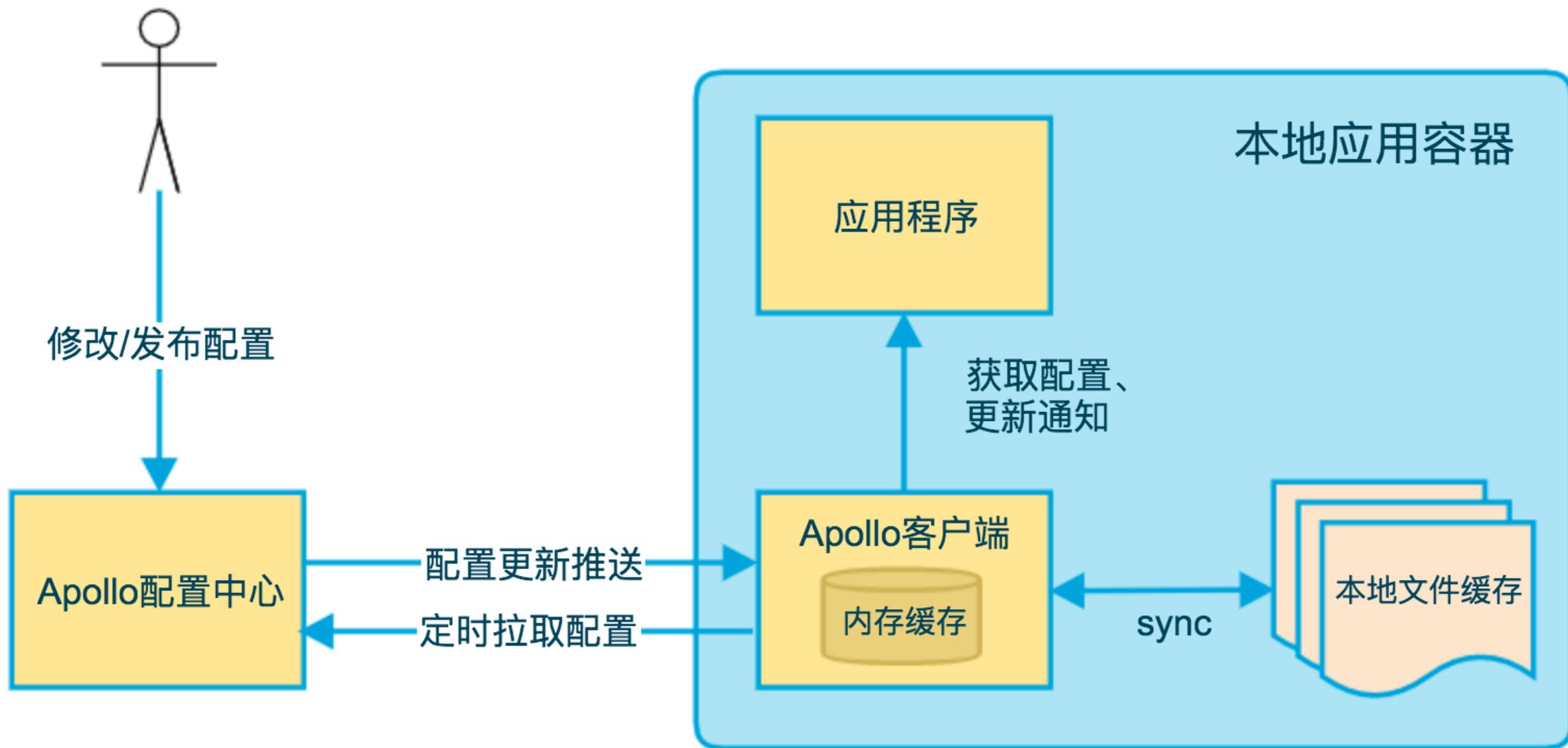
- C:\opt\settings\server.properties或/opt/settings/server.properties

- Core Concepts - cluster (集群)
  - 一个应用下不同实例的分组
  - 对不同的cluster, 可以有不一样的配置
    - 比如zk地址针对上海机房和成都机房可以有不一样的配置
  - 默认数据中心作为cluster
    - server.properties -> idc
    - C:\opt\settings\server.properties或/opt/settings/server.properties

- Core Concepts - namespace (命名空间)
  - 一个应用下不同配置的分组
  - 应用默认有自己的配置namespace – application
  - 也可以使用公共组件的配置namespace
    - 如RPC, DAL等
    - 可以通过继承方式对公共组件的配置做调整, 如DAL的初始数据库连接数



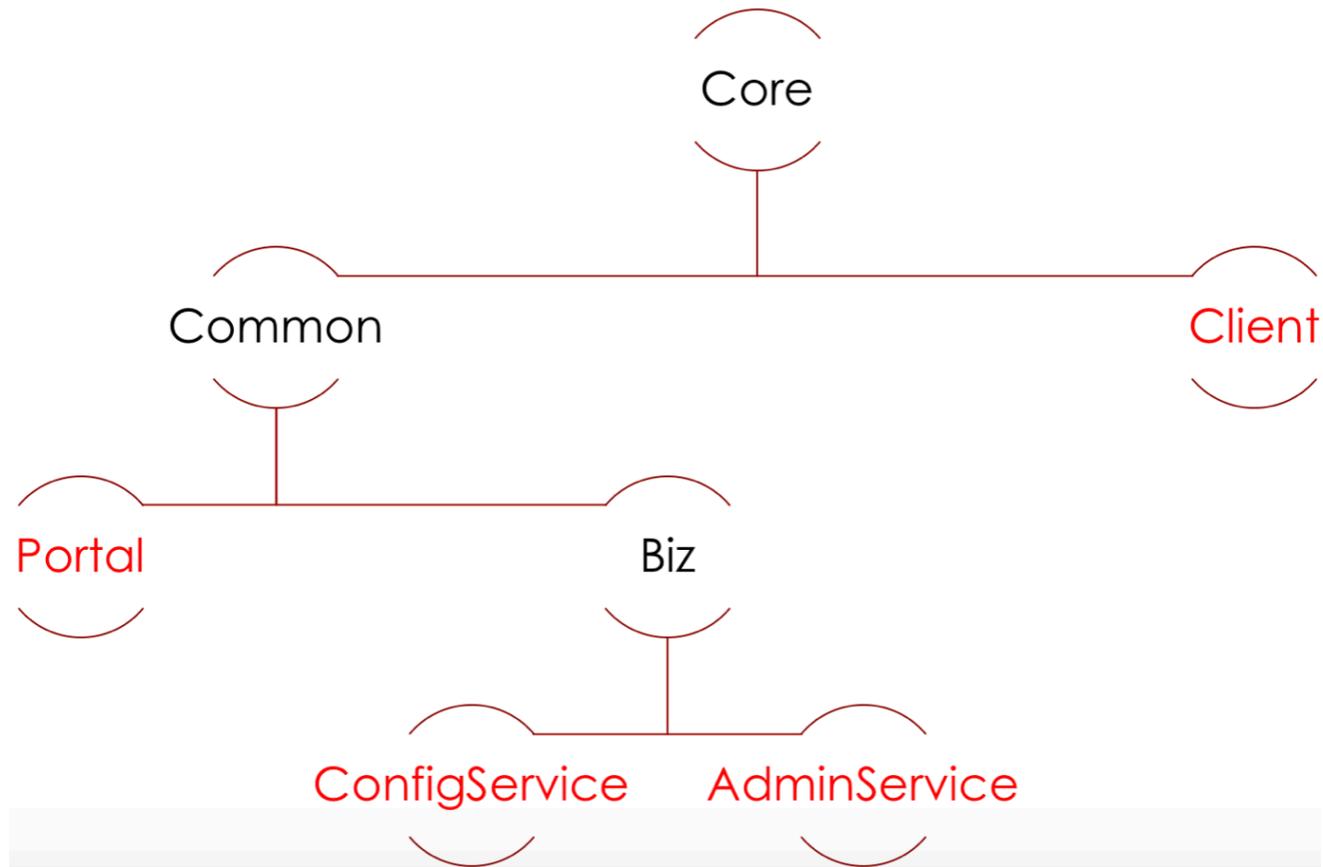
- 完整的 Service Registry 和 Service Discovery 实现
- 和 Spring Cloud 无缝集成
- Open Source



- 客户端Http Long Polling
  - 服务端保持30秒
  - 客户端断开自动重连
- 服务端async servlet
  - Spring DeferredResult

场景	影响	降级	原因
某台Config Service下线	无影响		Config Service无状态, 客户端重连其它Config Service
所有Config Service下线	客户端无法读取最新配置, Portal无影响	客户端重启时, 可以读取本地缓存配置文件。如果是新扩容的机器, 可以从其它机器上获取已缓存的配置文件, 具体信息可以参考 <a href="#">Java客户端使用指南 - 1.2.3 本地缓存路径</a>	
某台Admin Service下线	无影响		Admin Service无状态, Portal重连其它Admin Service
所有Admin Service下线	客户端无影响, Portal无法更新配置		
某台Portal下线	无影响		Portal域名通过SLB绑定多台服务器, 重试后指向可用的服务器
全部Portal下线	客户端无影响, Portal无法更新配置		
某个数据中心下线	无影响		多数据中心部署, 数据完全同步, Meta Server/Portal域名通过SLB自动切换到其它存活的数据中心
数据库宕机	客户端无影响, Portal无法更新配置	Config Service开启 <a href="#">配置缓存</a> 后, 对配置的读取不受数据库宕机影响	

- <https://github.com/ctripcorp/apollo>
- 服务端基于Spring Cloud和Spring Boot开发



# Summary

- 1 What is Apollo
- 2 Why Apollo
- 3 Apollo at a glance
- 4 Apollo in depth



携程技术中心

# THANK YOU!

---

## Q&A