

微服务架构及服务Comb实践



[社区网站] <http://servicecomb.apache.org>

[Github] <https://github.com/apache?q=servicecomb>

什么是微服务架构？

一个既“简单”的又“强大”的后端架构模式。

简单：

- 每个服务相对较小并比较容易开发维护。
- 服务可以独立的部署。
- 更适合小团队（个人）开发管理
- 更容易实验和采纳新的技术。

适合同学们学习

Demo也可以作为生产系统的一部分

强大：

- 使大型的复杂应用程序可以持续的交付和持续的部署。
- 更容易测试
- 更容易对已有大型系统进行修改和扩展
- 容易进行的性能优化
- 更高的可用性
- 更容易的进行性能伸缩性

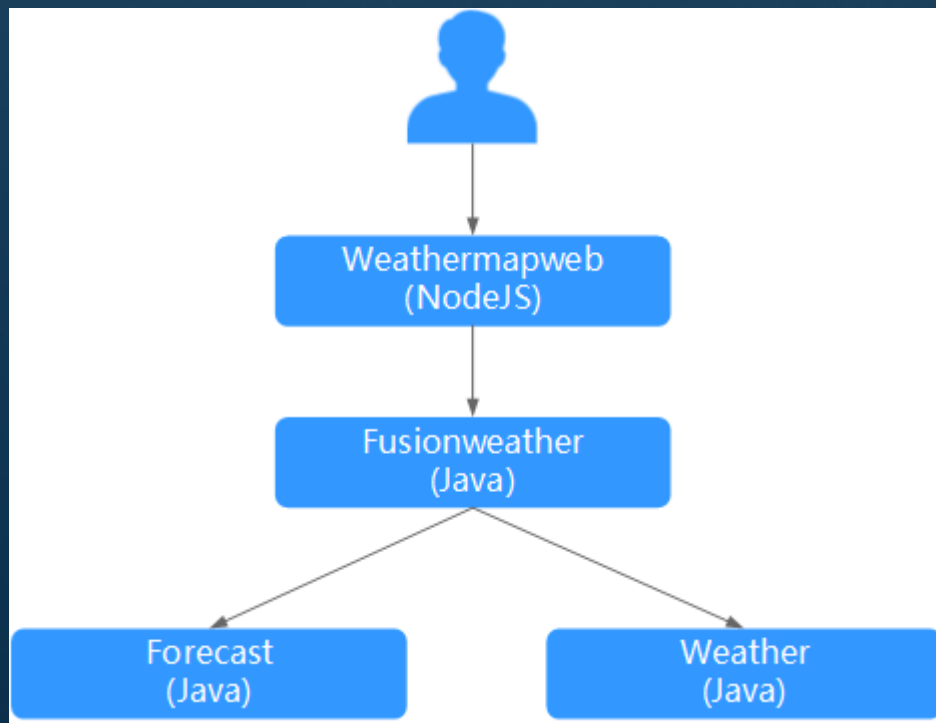


[社区网站] <http://servicecomb.apache.org>

[Github] <https://github.com/apache?q=servicecomb>

Demo简介

从一个简单的天气预报微服务应用谈起



- Weathermapweb : 界面服务 nodeJS
- Fusionweather : 汇聚服务, 为前台界面提供数据
- Forecast : 预报天气
- Weather : 当前天气
- <http://wyfrs4z7zxumdjp9.zizipo.cn-east-2.huaweicse.com/#!?city=shenzhen>



[社区网站] <http://servicecomb.apache.org>

[Github] <https://github.com/apache?q=servicecomb>

<input type="checkbox"/>	微服务名称	所属应用	版本数	实例数	框架
<input type="checkbox"/>	fusionweather	weathermap-...	1	1	ServiceComb 1.2.0.B01
<input type="checkbox"/>	weather	weathermap-...	2	2	ServiceComb 1.2.0.B01
<input type="checkbox"/>	forecast	weathermap-...	1	1	ServiceComb 1.2.0.B01
<input type="checkbox"/>	weathermapweb-zjam	weathermap-...	1	1	Meshier 1.6.2.1

weather#0.0.2 weatherma...
监控
阈值

0 0 0%
0 0
0 0

吞吐量 0/s
熔断状态 Closed

实例数	1	90th	0ms
中位数时延	0ms	99th	0ms
平均时延	0ms	99.5th	0ms

诊断结果 未诊断 (诊断)

服务过滤: 请选择服务
状态选择: All 有实例

```

graph TD
    A((weathermapweb-zjam#1.0)) --> B((fusionweather#0.0.5))
    B --> C((forecast#0.0.2))
    B --> D((weather#0.0.3))
    B --> E((weather#0.0.2))
  
```

负载均衡(0) 限流(0) 降级(0) 容错(0) 熔断(0)

错误注入(0) 黑白名单(0)

+ 新增

从Weather微服务应用谈起——简单性

数据库

表	内容	关键字段
regions	存储地区信息	id, regionName
weathers	存储当前时间的天气信息，包括，气温，风向....	id, temperature, pressure...



API

接口	内容
currentWeather	返回当前天气

定时任务

任务	内容
getWeatherTask	从OpenWeatherMap定期查询天气数据



[社区网站] <http://servicecomb.apache.org>

[Github] <https://github.com/apache?q=servicecomb>

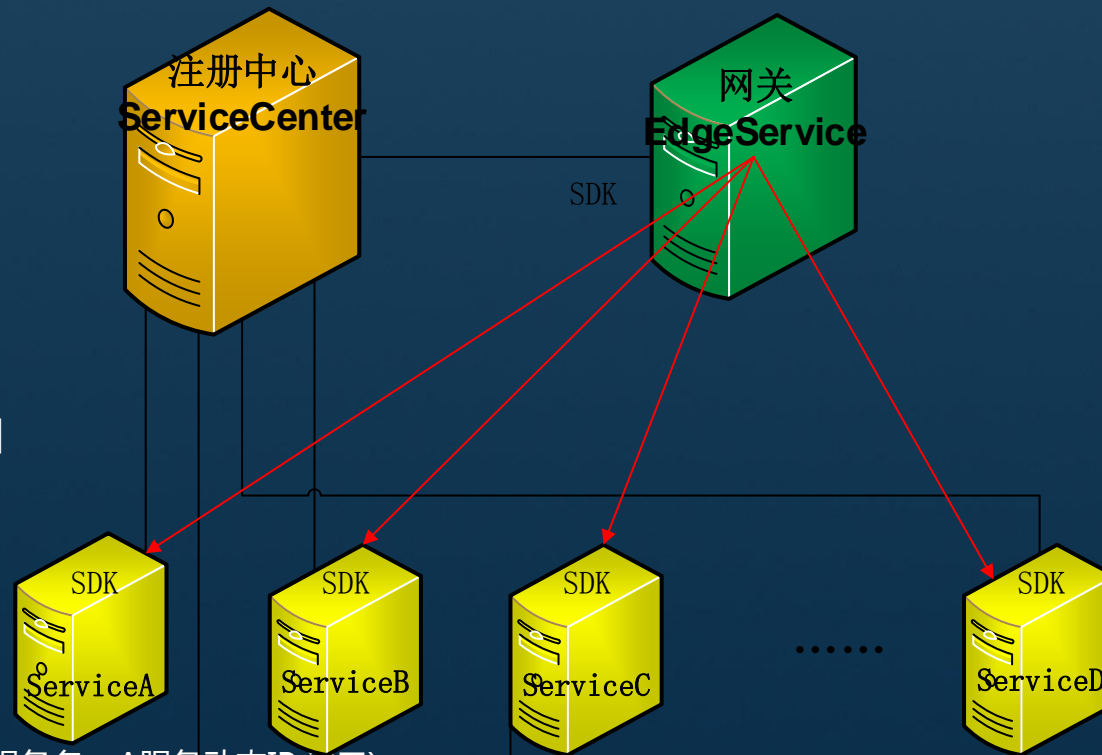
最简单的微服务 – 微服务的基本组件

目标：

- 服务与物理地址解耦
- 基本的健康检测

组件

- **1个服务（应用+数据库）**
- **服务注册中心**
 - a) <https://github.com/apache/servicecomb-center> 服务注册中心
 - b) 核心服务注册表 |---服务名---|---服务实例UID---|---物理地址(IP:端口)---
 - c) 表的维护（心跳存活性检测）
- **SDK**
 - a) <https://github.com/apache/servicecomb-java-chassis>
 - b) 和服务注册中心通信，共同维护表
- **网关**
 - a) 对外提供固定地址，提供路由转发功能
 - b) 网关固定IP:端口/服务名/URL -> 服务动态IP:端口/URL（网关固定IP:端口/A服务名 = A服务动态IP:端口）



[社区网站] <http://servicecomb.apache.org>

[Github] <https://github.com/apache?q=servicecomb>

3个组件的更多功能

- **服务注册中心 ServiceCenter**

- a) <https://github.com/apache/servicecomb-service-center>
- b) 当前实例的报表 (UI), 详细的API 接口列表
- c) 某服务实例数低于阈值指报警 社区召集 !

- **SDK**

- a) <https://github.com/apache/servicecomb-java-chassis> java SDK
- b) <https://github.com/apache/servicecomb-mesher> 多语言解决方案
- c) 重试
- d) 负载均衡
- e) 限流
- f) 降级 (隔离、熔断、容错)
- g) 分布式跟踪

- **网关**

- a) 路由
- b) 黑白名单
- c) 和认证鉴权集成



[社区网站] <http://servicecomb.apache.org>

[Github] <https://github.com/apache?q=servicecomb>

服务间的通信

背景：为什么服务间通信重要？

- 每个微服务有自己的数据库（层）
- 服务架构应用的质量很大程度取决于服务的拆分的高内聚，低耦合，不了解服务间通信，无法做出高质量的拆分。

不同的业务，涉及的不同的服务间通信有不同的要求

1. 关注事务性要求（ACID）
2. 关注对性能的要求

举例：商品维护和下订单是2个微服务，下订单系统对性能有较高要求，所以可以考虑把商品的价格从商品维护微服务同步到订单系统，该同步动作对性能没有要求，对事务性有要求。

常见的服务间通信模式：同（异）步通信，API组合，基于MQ的异步通信

事务性：**saga**, 事务性消息发布（一次仅一次），分布式事务等

很易用的服务间saga事务方案Servicecomb-Pack: <https://github.com/apache/servicecomb-service-pack>



[社区网站] <http://servicecomb.apache.org>

[Github] <https://github.com/apache?q=servicecomb>

微服务的其他组件

- **配置中心**

- a) 集中配置
- b) 动态配置
- c) 配置历史保留
- d) 配置回滚 社区召集！
- e) 灰度发布配置 社区召集！

- **认证鉴权**

- a) 认证
- b) 鉴权

- **Toolkit**

- a) 契约
- b) 脚手架

- **分布式日志**

<https://github.com/apache?utf8=%E2%9C%93&q=servicecomb&type=&language=>



[社区网站] <http://servicecomb.apache.org>

[Github] <https://github.com/apache?q=servicecomb>

微服务的“强大” - 以电商“抢购”场景举例

背景：因为竞争的加剧，电商系统越来越复杂，无论电商系统的规模大小，微服务几乎是事实标准。



[盘点12306:一个总在“崩溃中”的神奇网站_科技_腾讯网](#)

2012年9月29日 - 盘点12306:一个总在“崩溃中”的神奇网站 长假来临,即将踏上... 都长舒一口气,终于可以暂时告别那个“神话般存在”的12306网站了。但下一个...

12306网站 15年 峰值 1032 单/s
11.7w PV

[一加7 Pro首销一分钟销售额破亿 刘作虎:紧急从海外调货](#)



2019年5月21日 - 5月21日(今天)上午10点,一加7 Pro正式迎来全网首销。价格方面,6GB+...今天中午,一加CEO刘作虎在微博上宣布,一加7 Pro全网销售额1分钟破亿,同时...

快科技 - 百度快照

电商“抢购”“场景中企业中遇到的困难

1. 性能要求不断优化,和不能引入Bug之间的矛盾。
2. “变态”级别的高可用性要求。
3. 声誉和法律风险。
4. 抢购体验 → 商家的技术能力 → 产品质量。



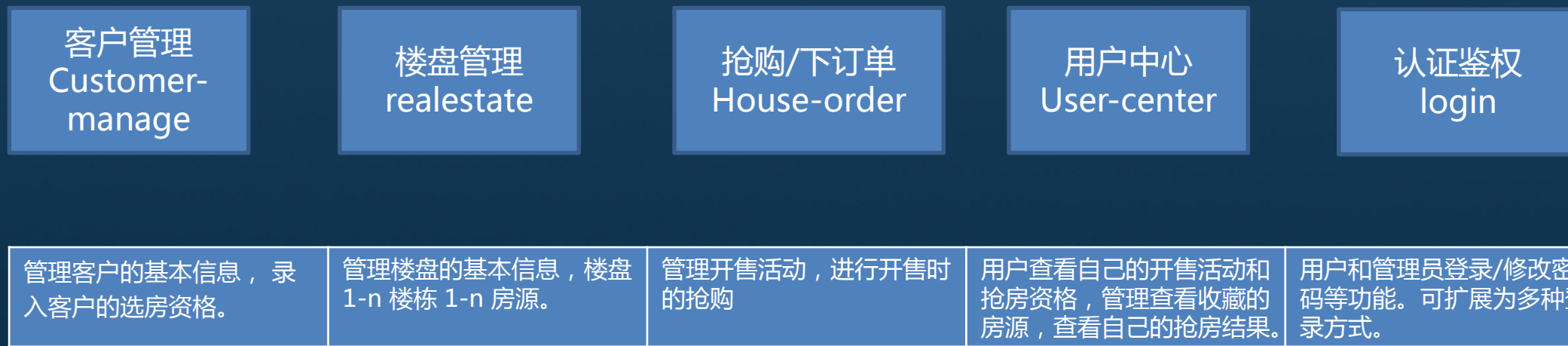
[社区网站] <http://servicecomb.apache.org>

[Github] <https://github.com/apache?q=servicecomb>

一个电商抢房系统举例简介

从一个简单的电商Demo开始。

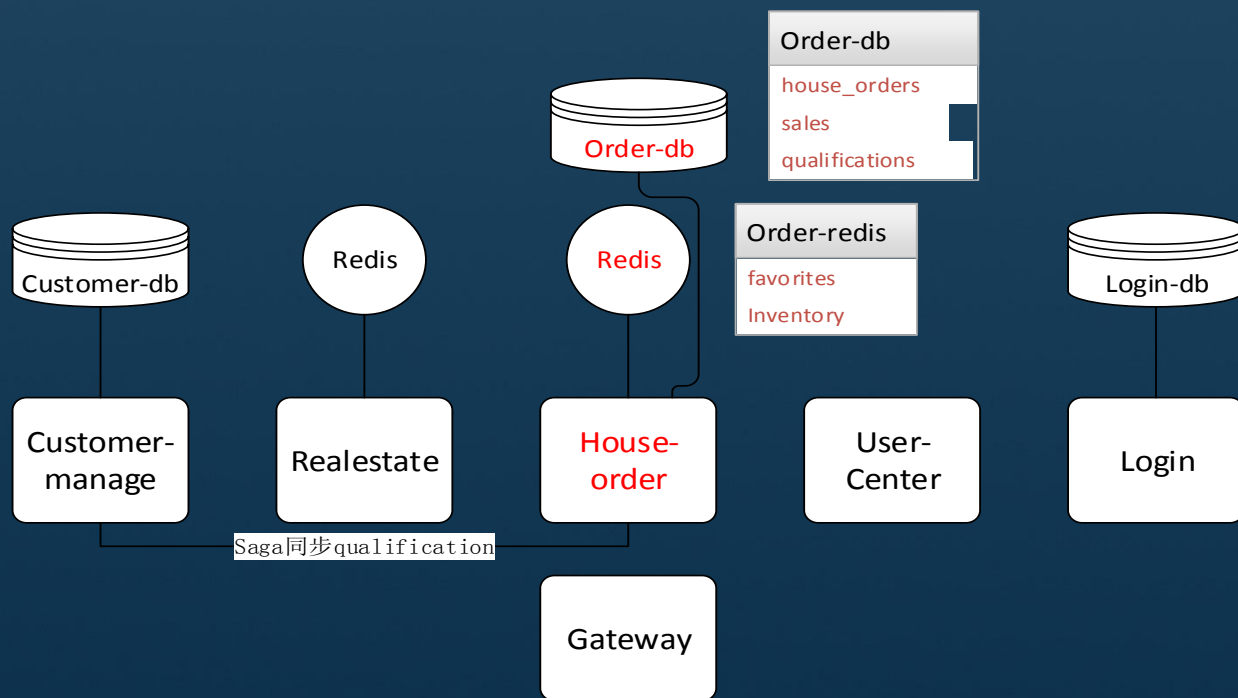
背景：开发商开盘时，让客户“抢购”其当前推售的所有房源，先抢到先得。



[社区网站] <http://servicecomb.apache.org>

[Github] <https://github.com/apache?q=servicecomb>

步骤1拆分：



1. 每个微服务对应自己数据层
2. 拆分经验：看业务，不看技术
3. 利用：saga, 事务性消息发布，最终一致性，同（异）步调用组合等微服务模式



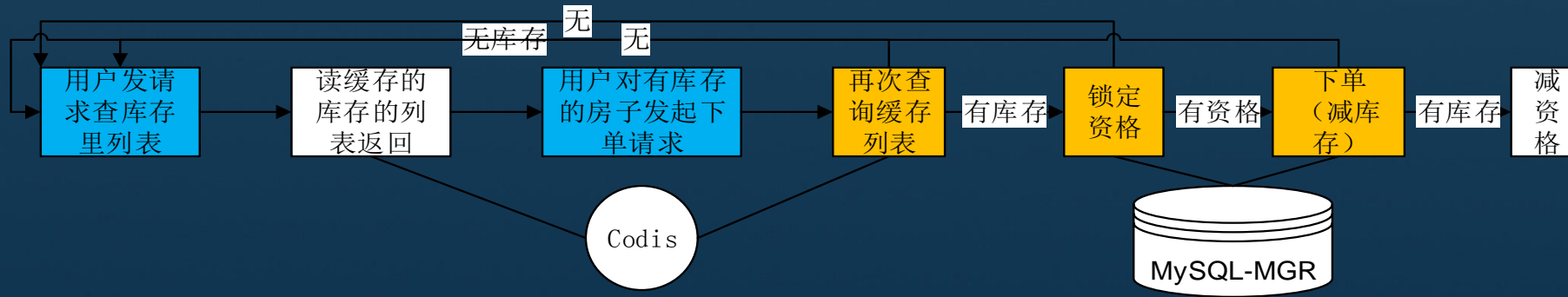
[社区网站] <http://servicecomb.apache.org>

[Github] <https://github.com/apache?q=servicecomb>

步骤2 优化

先关注数据层的优化

1. 锁资格 - (锁库存-下订单) CP MySQL-MGR 高带宽 NvmeSSD-持久化配置 索引-加锁 update-where
2. 库存缓存 AP Codis
3. 收藏数据 AP Codis



微服务层的优化

1. SeviceComb优异的异步通讯模型，REST over Vertx。
2. 透明RPC模式+CompletableFuture。把异步继续进行。
3. 低超时+重试策略配置。“高可用”的定义？A跟a 的区别 → 客户端无异常。
4. Zuul VS edge-service (更优)。
5. 生产环境需从最外层开始设置限流策略，并设置降级机制。



[社区网站] <http://servicecomb.apache.org>

[Github] <https://github.com/apache?q=servicecomb>

Thank You.

欢迎添加

ServiceComb小助手

加入微服务技术交流群，架构、设计、开发、解BUG、调优，总有您感兴趣的话题



欢迎关注

微服务蜂巢公众号

获取更多微服务技术干货、资讯文章



[社区网站] <http://servicecomb.apache.org>

[Github] <https://github.com/apache?q=servicecomb>