51CTO.com
技术成就梦想

UBM

2012 / 10 / 25 ► 2012 / 10 / 26
中国 · 北京
October 25-26,2012
Beijing · China

World of Tech
WOT

2012云计算架构师峰会

Cloud Computing Architects Summit China 2012

揭示企业级IT架构转型 分享最新技术的应用落地

主办单位：51CTO.com 技术成就梦想　联合主办：UBM

公有云平台下
企业级软件设计经验谈

徐子岩

2012年10月25日

关于我

- IGT科技开发（北京）研发中心
- Microsoft Windows Azure MVP
- 程序员
- 微博控：blogs.shaunxu.me
- 博客控：@老羊肖恩

内容

- 云计算和公有云计算平台
- 云平台对软件设计的（新）要求
  - 可扩展性和多实例
  - 缓存的运用
  - 消息队列和异步操作
  - 面向成本设计
- 总结

云计算平台和公有云平台

云计算平台

- 提供的服务级别
  - IaaS
  - PaaS
  - SaaS
- 服务对象
  - 通用云
  - 专业云
- 服务范围
  - 公有云
  - 私有云

公有云的特点

- 服务模式：平台提供商所有，企业按需使用
- 地理位置：程序和数据保存在提供商的数据中心
- 服务内容：通用服务，较少可定制性
- 销售模式：按需付费
- 服务级别：IaaS和PaaS为主

公有云的优点

- 零投入：基础设施和硬件
- 高可用性
- 快速部署
- 全球化
- 按需付费
- 可度量

公有云的缺点

- 数据安全
  - 数据存放在平台提供商的数据中心
  - 某些法律和规定不允许
- 访问速度
  - 有限的数据中心
- 可定制化
  - 只能提供有限的定制化要求

公有云服务包括

- 分布式计算
- 分布式存储
- 分布式数据库
  - 关系数据库
  - 非关系数据库
- 负载均衡
- 缓存
  - 分布式缓存
  - 分布式缓存服务
- 消息队列
- CDN

主要服务

附加服务

软件设计的（新）要求

软件设计的（新）要求

- 高可用性：各模块支持多实例和可扩展性
- 快速部署：模块各自独立互不干扰
- 全球化：多语言支持，跨时区支持
- 按需付费：面向成本设计
- 可度量：自动扩展
- 平台附加服务：灵活运用平台提供的服务
  - 缓存
  - 消息队列
  - 数据库和存储服务

多实例和可扩展性

可扩展性

- 一种对软件系统计算处理能力的设计指标
- 在系统扩展成长过程中，软件能够保证旺盛的生命力，通过很少的改动甚至只是硬件设备的添置，就能实现整个系统处理能力的线性增长，实现高吞吐量和低延迟高性能。

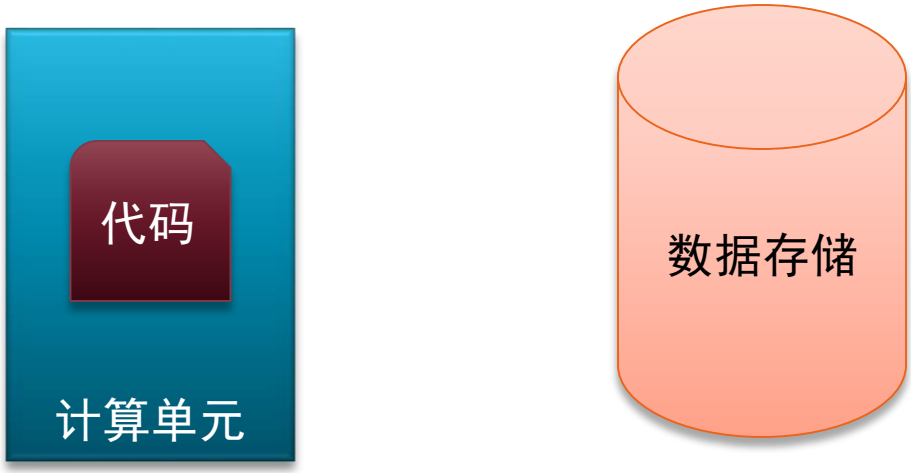
为什么要支持可扩展性

- 云计算平台提供支持
  - 负载均衡
  - 易于分配
  - 按需付费
- 对软件设计本身
  - 弹性和伸缩性
  - 高可用性
  - 成本控制

可扩展性分类

- 扩展方式
  - 纵向扩展
  - 横向扩展
- 扩展对象
  - 针对应用层的扩展
  - 针对数据层的扩展

纵向扩展



纵向扩展

- 较少或基本不需要软件设计的修改
- 有上限
- 贵
- 可能带来单点故障

横向扩展

横向扩展

- 需要软件设计上的支持
- 无上限
- 便宜
- 解决单点故障

可扩展不等于高性能

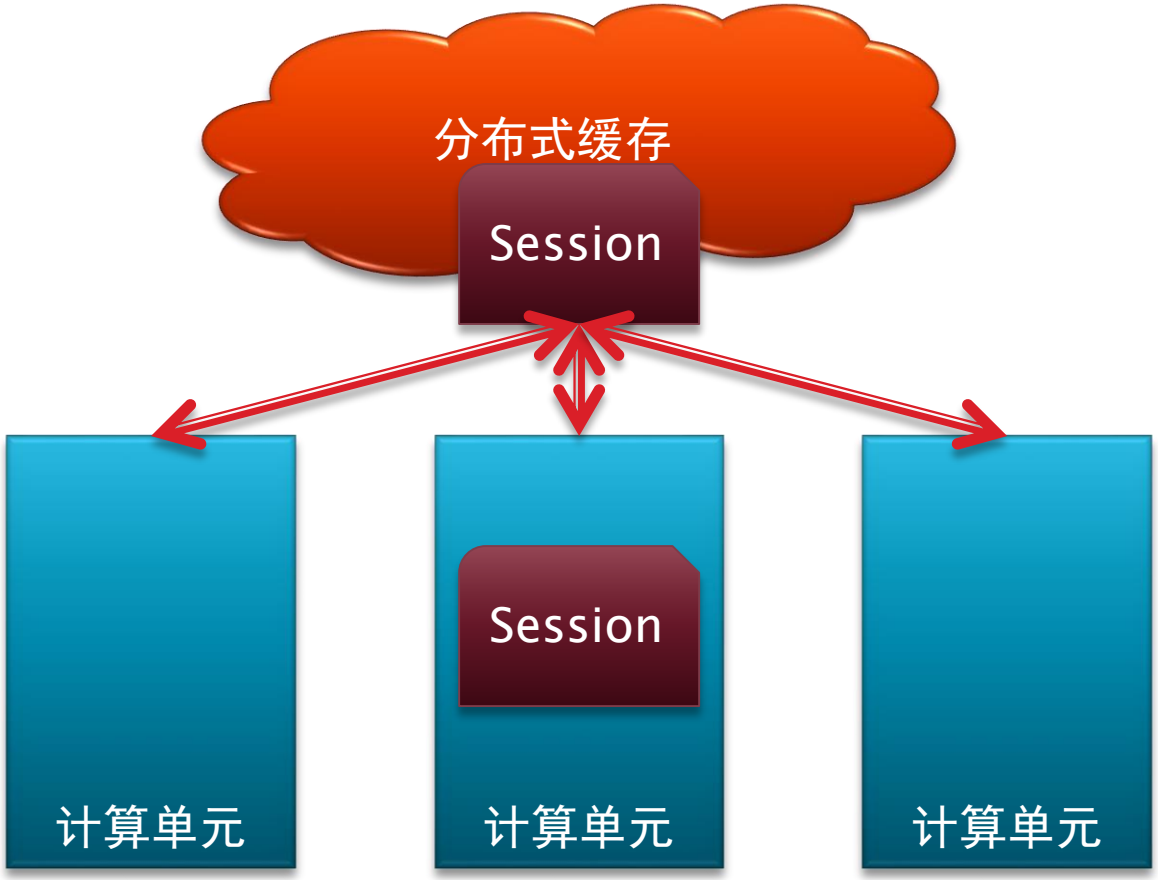
- 支持可扩展性的设计往往导致一定的性能损失
- 扩展后会带来极大的性能提升

应用层的横向扩展

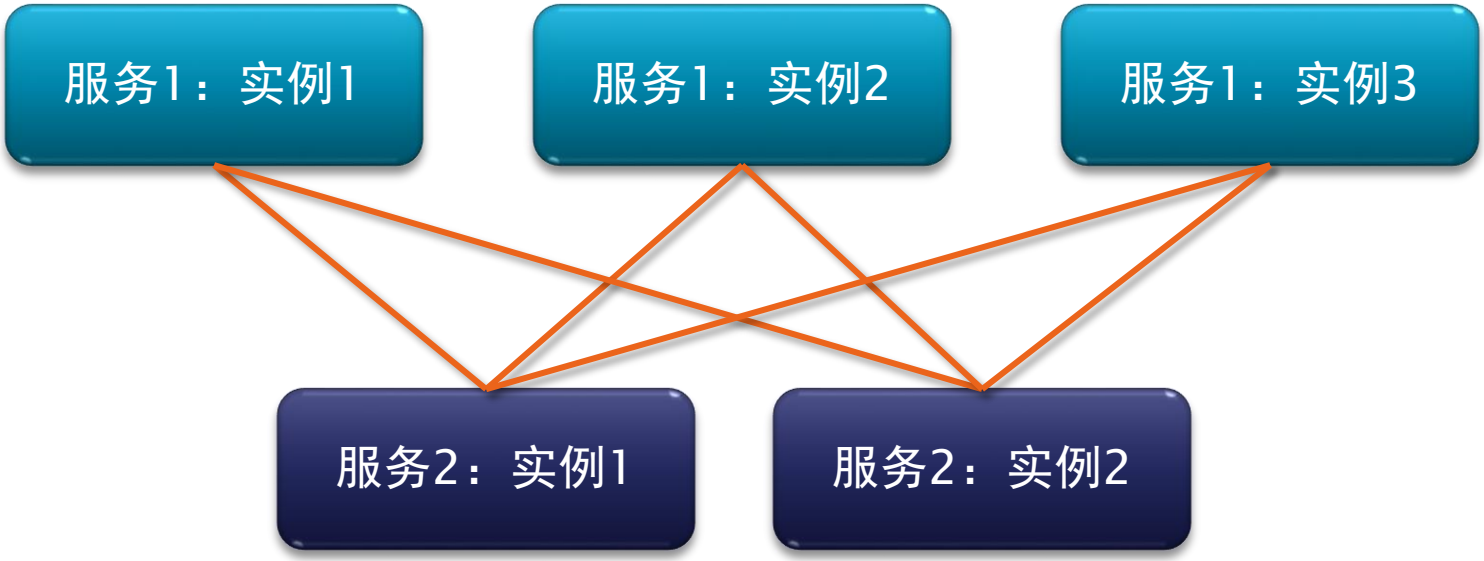
- 原则：不依赖于某个具体的资源
- 模式
  - 基于分布式资源
  - 基于中心资源
- 实践
  - 基于分布式缓存的Session
  - 基于分布式消息总线的服务端调用

基于分布式缓存的Session

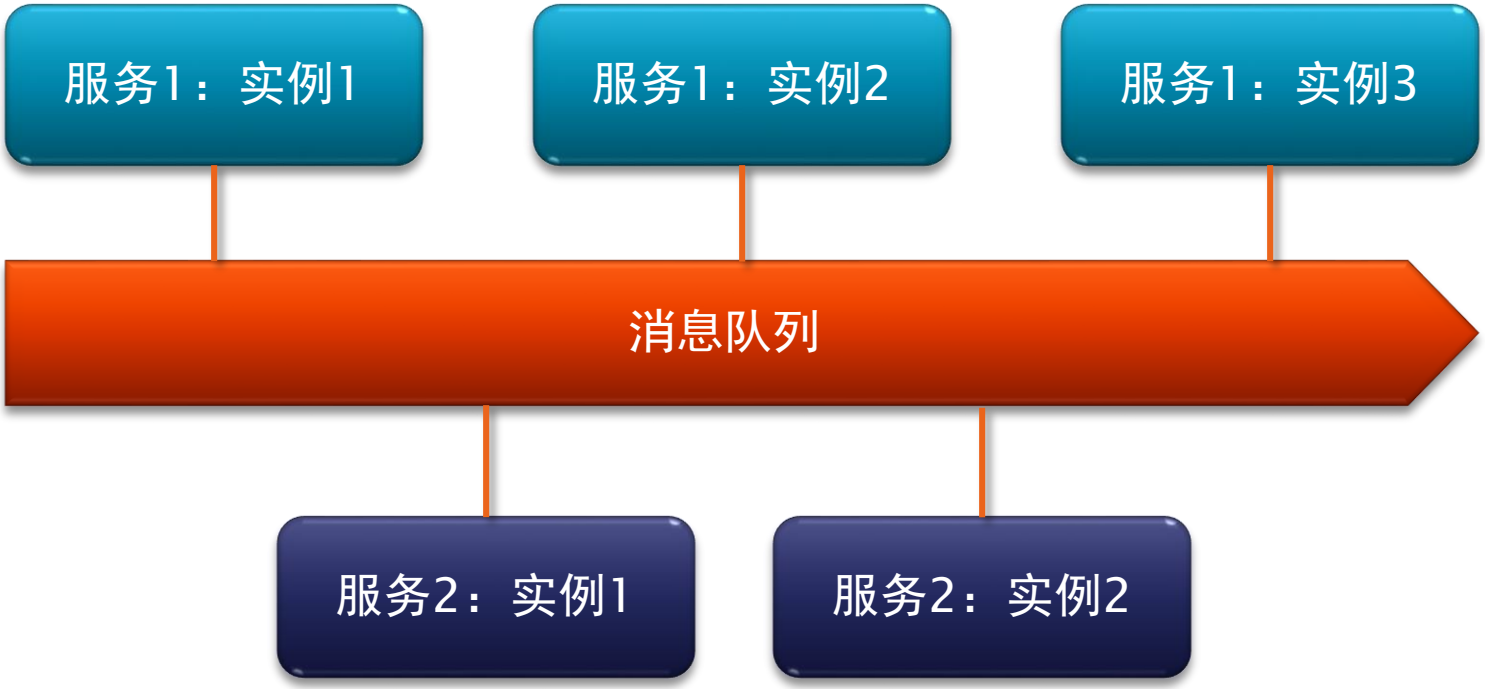
基于分布式缓存的Session



基于分布式消息总线的服务端调用



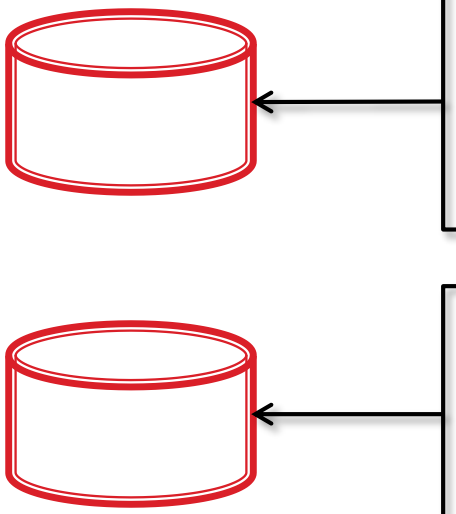
基于分布式消息总线的服务端调用



数据层的横向扩展

- 模式
  - 横向分片
  - 纵向分片
  - 混合分片

数据横向分片



ID	Name	Avatar
1	Jack	… …
2	Anna	… …
3	Daniel	… …
…	…	… …
101	Susan	… …
102	John	… …
103	Frank	… …

- 一致性
- 完整性

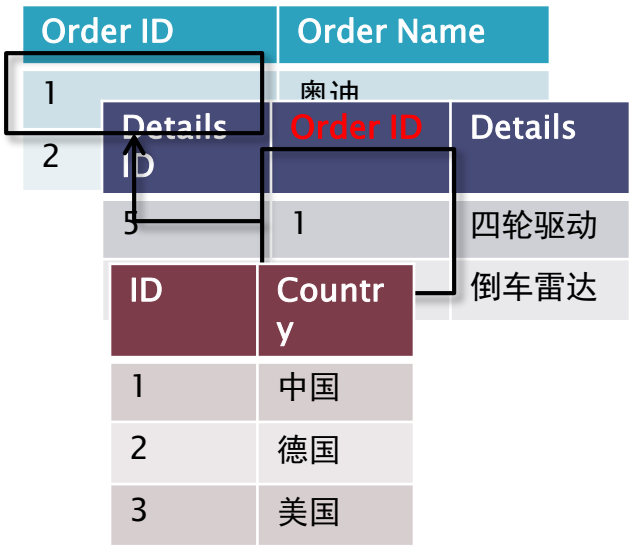
- 扇出查询效率低
- 再分片操作
- 合并操作

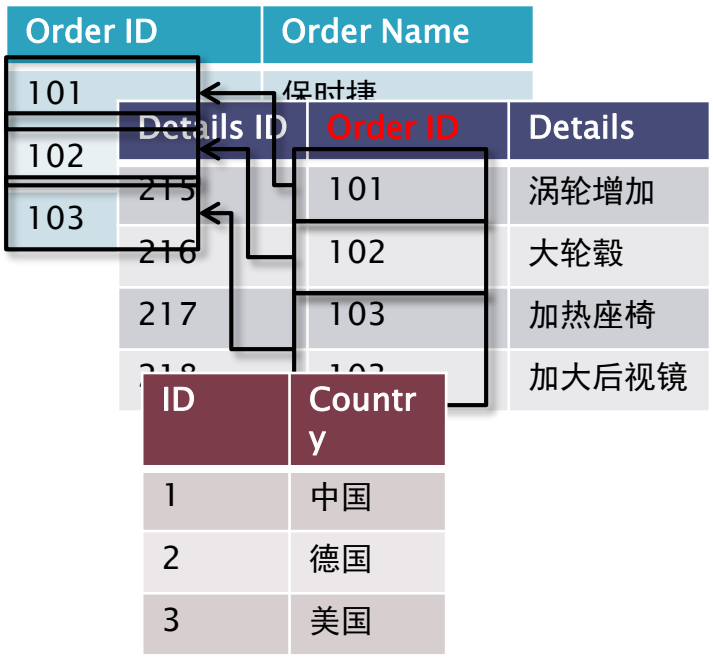
横向分片原则

- 基于数值
  - 整数
  - 时间
- 基于业务逻辑
  - 用户
  - 地域
- 基于算法
  - 取模
  - 哈希，一致性哈希

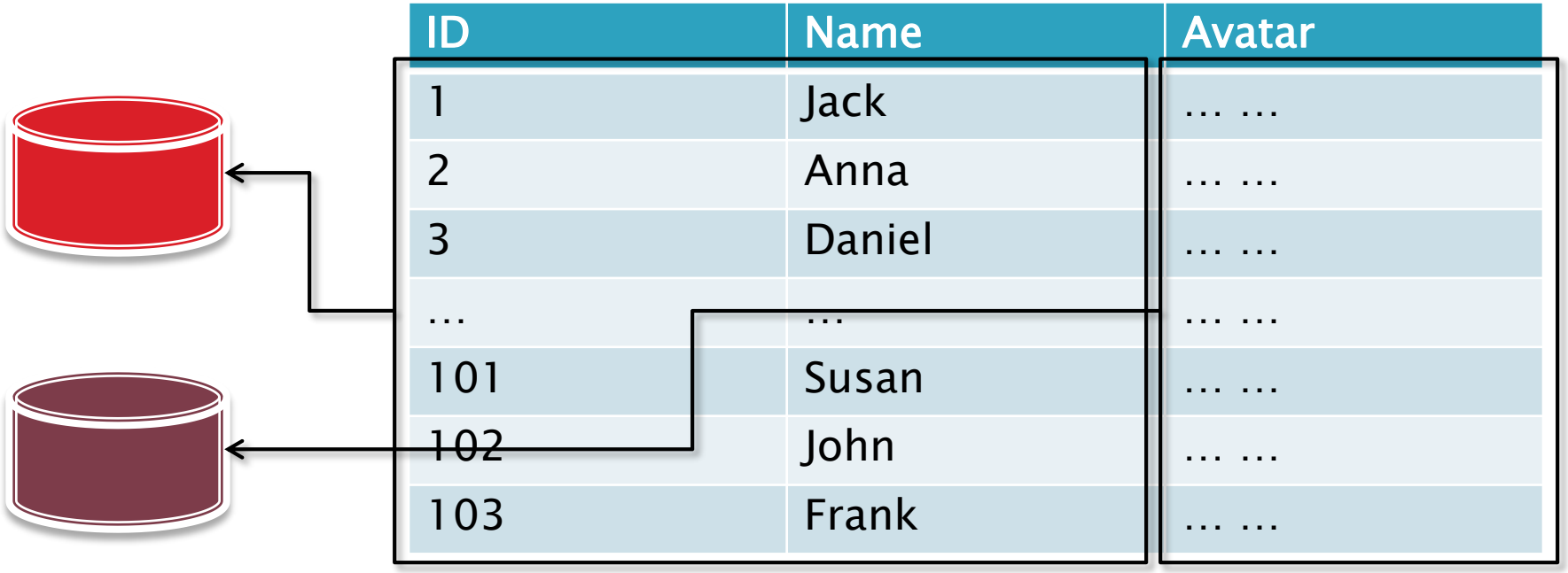
横向分片原则

Federation ID	Range
1	Min … 100
2	100 … 200
3	200 … MAX





数据纵向分片



ID	Name	Avatar
1	Jack	… …
2	Anna	… …
3	Daniel	… …
…	…	… …
101	Susan	… …
102	John	… …
103	Frank	… …

- 扇出查询效率高
- 面向成本分片

- 数据结构不一致
- 数据不完整
- 冗余数据

缓存

缓存的目的

- 提高性能
  - 内存访问替代数据库和网络访问
  - 直接获取结果无需重复计算
- 降低成本
  - 减少数据库和其他服务的使用频率
- 扩展性支持
  - 分布式缓存

缓存的内容

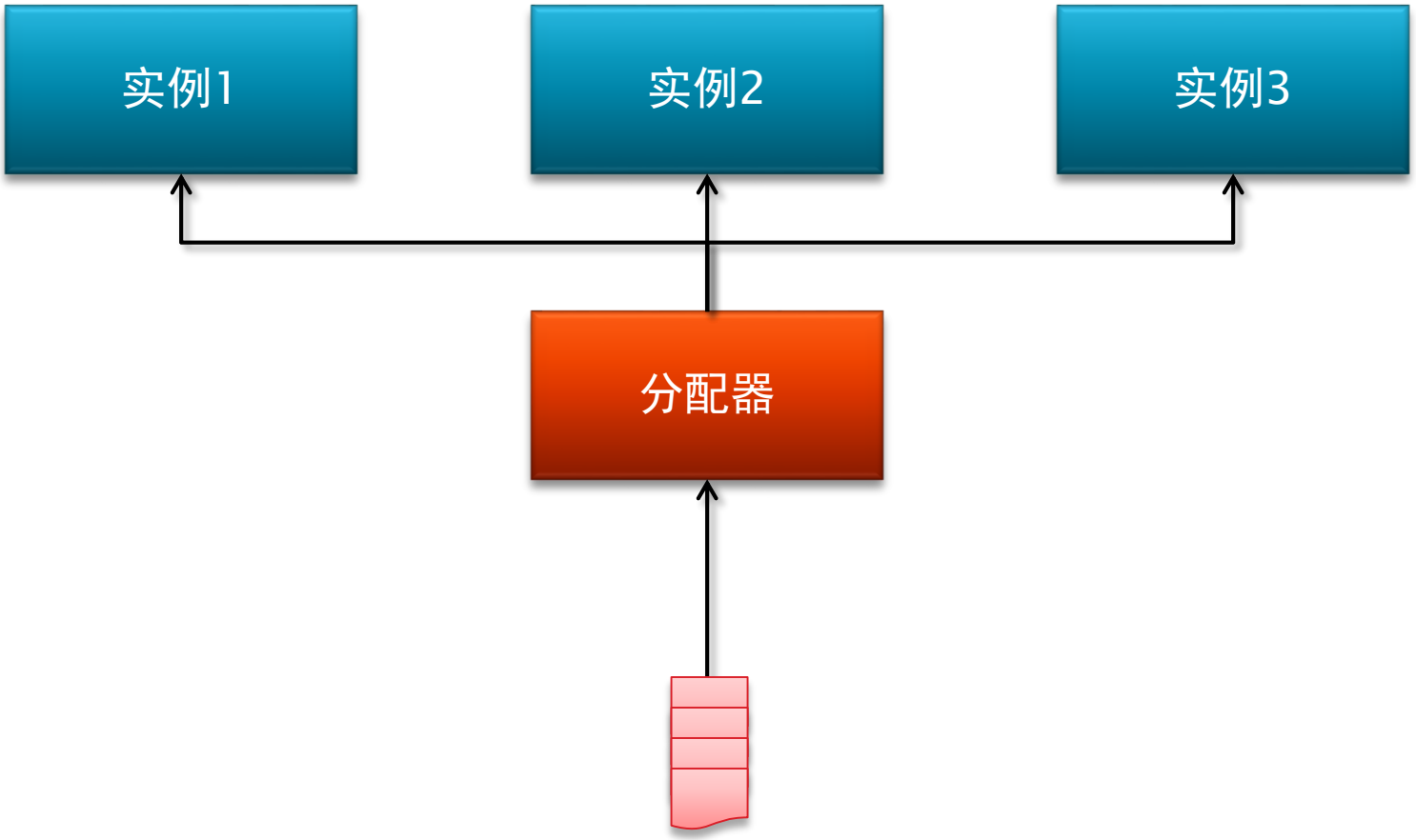
- 资源数据：一次写入不会改变
  - 国家、城市、语言
  - 部门、职务
  - 性别、称谓
- 引用数据：一处修改多处使用
  - 用户信息
  - 权限矩阵
- 活动数据：频繁更改和读取
  - 购物车
  - 会话

缓存的选取

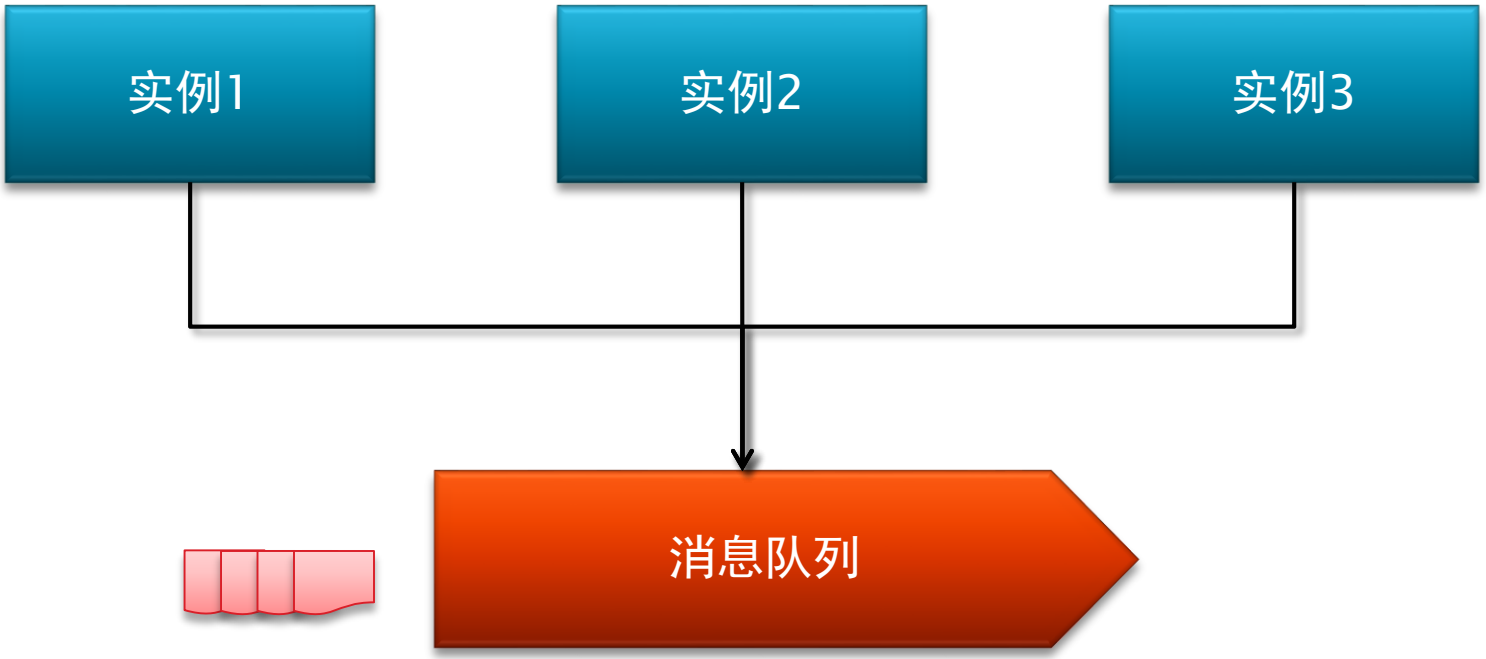
	本地缓存	分布式缓存	分布式缓存服务
优点	• 速度快	• 分布式 • 支持扩展	• 分布式 • 无需管理缓存节点
缺点	• 不易于扩展	• 速度慢	• 速度慢 • 可控性差
注意点	• 缓存同步 • 过期处理	• 缓存节点的配置 • 缓存不命中	• 混合本地缓存
适用场景	• 资源数据 • 缓存的缓存	• 引用数据 • 活动数据	

消息队列和异步操作

分配器模式的扩展性实现



消息队列模式的扩展性实现



基于消息队列的设计

- 低耦合：服务之间只通过数据（消息体）耦合
- 单一性：服务之关系自己能处理的消息和需要其它服务处理的消息
- 自制性：服务自动运行，尽力而为，无需调度
- 统一性：统一服务的每个实例完全一致，实例的增减无需额外配置

消息什么时候删除

- 隐式删除
  - 获取消息后立即删除
  - 一旦服务因故障没有完成操作，则此消息将无法追回，对应的操作也无法再吃执行。
- 显示删除
  - 服务执行完毕后在删除消息
  - 一旦服务因故障没有完成操作，则此消息可能会被二次（多次）执行，成为有毒消息。

有毒消息

- 什么是有毒消息
  - 没人能执行的消息
  - 没人能执行成功的消息
- 为什么会有有毒消息
  - 程序错误
  - 异常数据
- 有毒消息的危害
  - 增加队列长度
  - 服务操作频繁失败
  - 降低服务的执行能力

有毒消息的处理

- 消息弹出计数器
  - 大部分消息队列服务提供此功能
  - 设定阀值
  - 有毒队列
- 消息操作的幂等性
  - 幂等操作：重复执行
  - 非幂等操作：操作日志
- 有毒消息回注

面向成本设计

云平台的计费特点

- 计费项目
  - 计算服务
  - 存储服务
  - 数据访问服务
- 计费内容
  - CPU核心运算时间
  - 存储量
  - 流量
  - 访问次数
- 计算方法

考量点

- 计算资源
  - 资源大小
  - 使用时间
- 存储资源
  - 关系数据库
  - 分布式存储

应用程序的支持

- 计算资源
  - 可扩展性
  - 服务的独立性和自制性
- 存储资源
  - 纵向分片
  - 混合分片
- 自动扩展

总结

- 公有云平台的特点
- 应用程序设计的（新）挑战
- 目的
  - 速度快
  - 高可用
  - 更智能
  - 更省钱

联系我

- jfarrio@gmail.com
  @老羊肖恩
  blogs.shaunxu.me