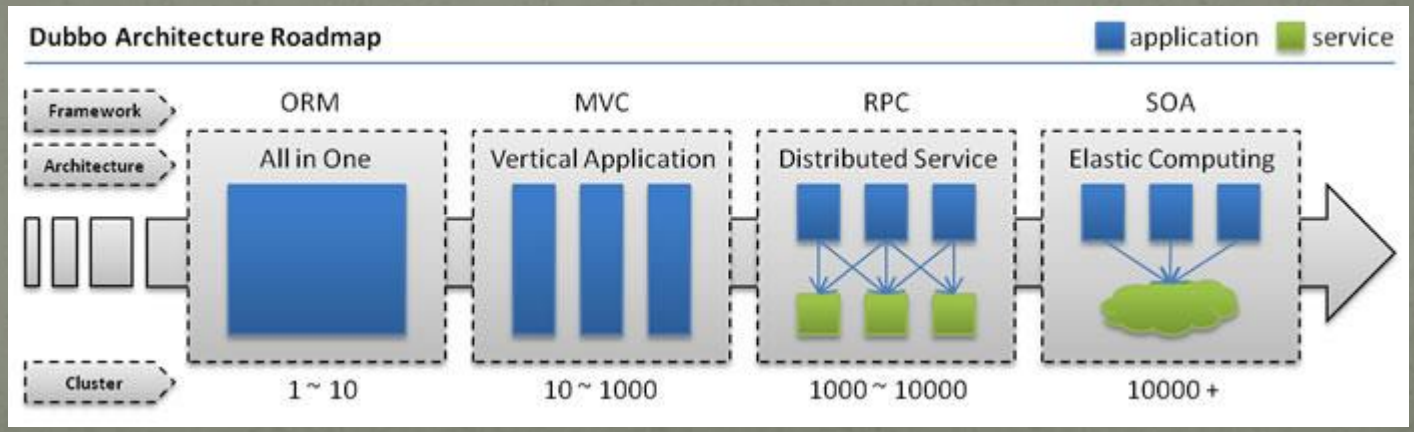
基于Dubbo服务化系统架构

目录

- 架构演进
- 现实需求
- Dubbo 简介与原理
- 为什么是 Dubbo
- 服务化最佳实践
- 都有谁在使用 Dubbo

架构演进

- 单一应用架构
- 垂直应用架构
- 分布式服务架构
- 流动计算架构



现实需求

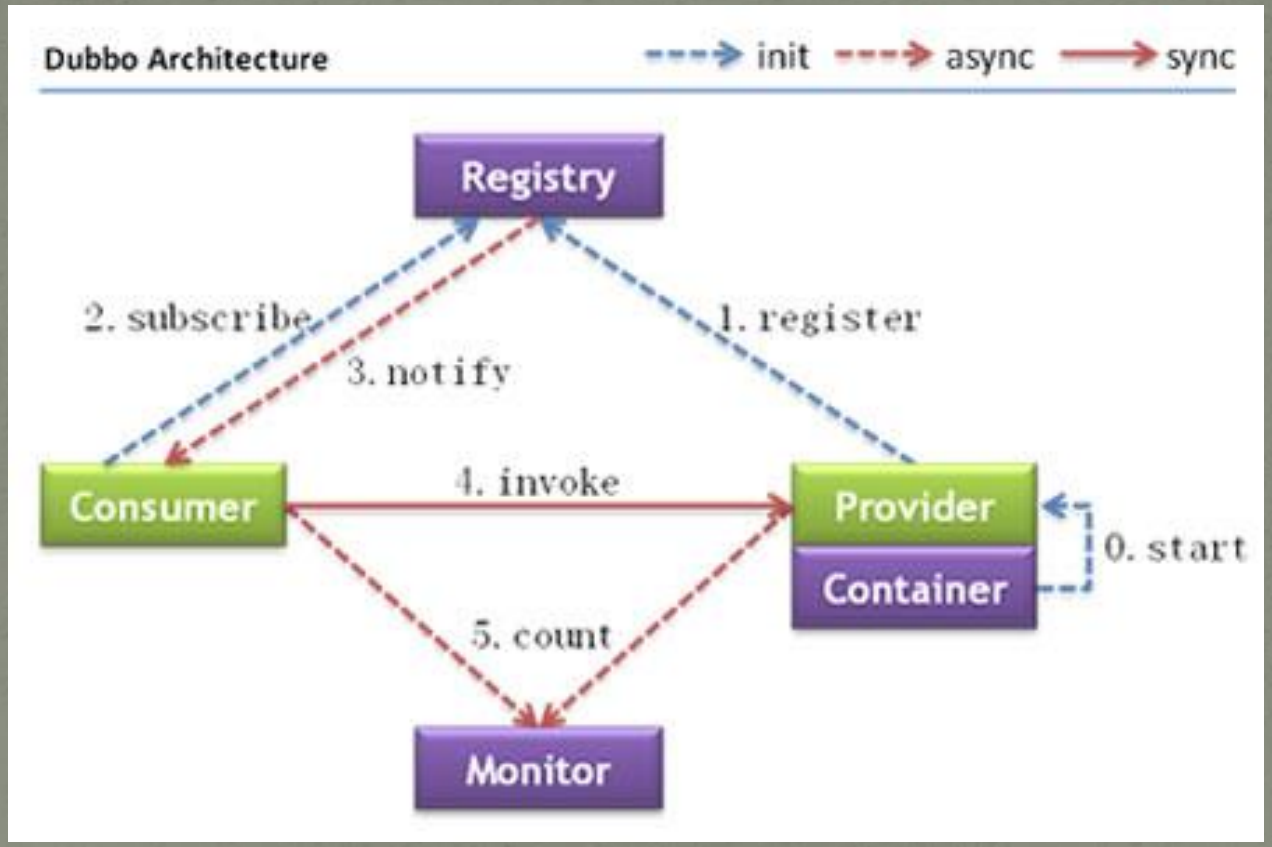
- 大规模服务化后
  - 管理繁重的服务URL配置及负载均衡需求（注册中心）
  - 梳理服务间的依赖关系
  - 服务器的规划参考（服务调用量、响应时间）

Dubbo 简介

- Dubbo 是什么？
  - Dubbo 是阿里巴巴公司开源的一个分布式服务框架，致力于提供高性能和透明化的 RPC 远程服务调用方案，以及 SOA 服务治理方案。

- Dubbo 有何特点？
  - 远程通讯：基于长连接的 NIO 框架抽象封装
  - 集群容错：提供多协议支持，以及软负载均衡，失败容错，地址路由，动态配置等集群支持。
  - 自动发现：基于注册中心目录服务，使服务消费方能动态的查找服务提供方，支持平滑减少或增加机器

Dubbo基本原理



为什么是Dubbo

- 透明化的远程方法调用，就像调用本地方法一样调用远程方法，只需简单配置，没有任何API侵入。
- 软负载均衡及容错机制，减少单点。
- 服务自动注册与发现，不再需要写死服务提供方地址，注册中心基于接口名查询服务提供者的IP地址，并且能够平滑添加或删除服务提供者。

为什么是Dubbo

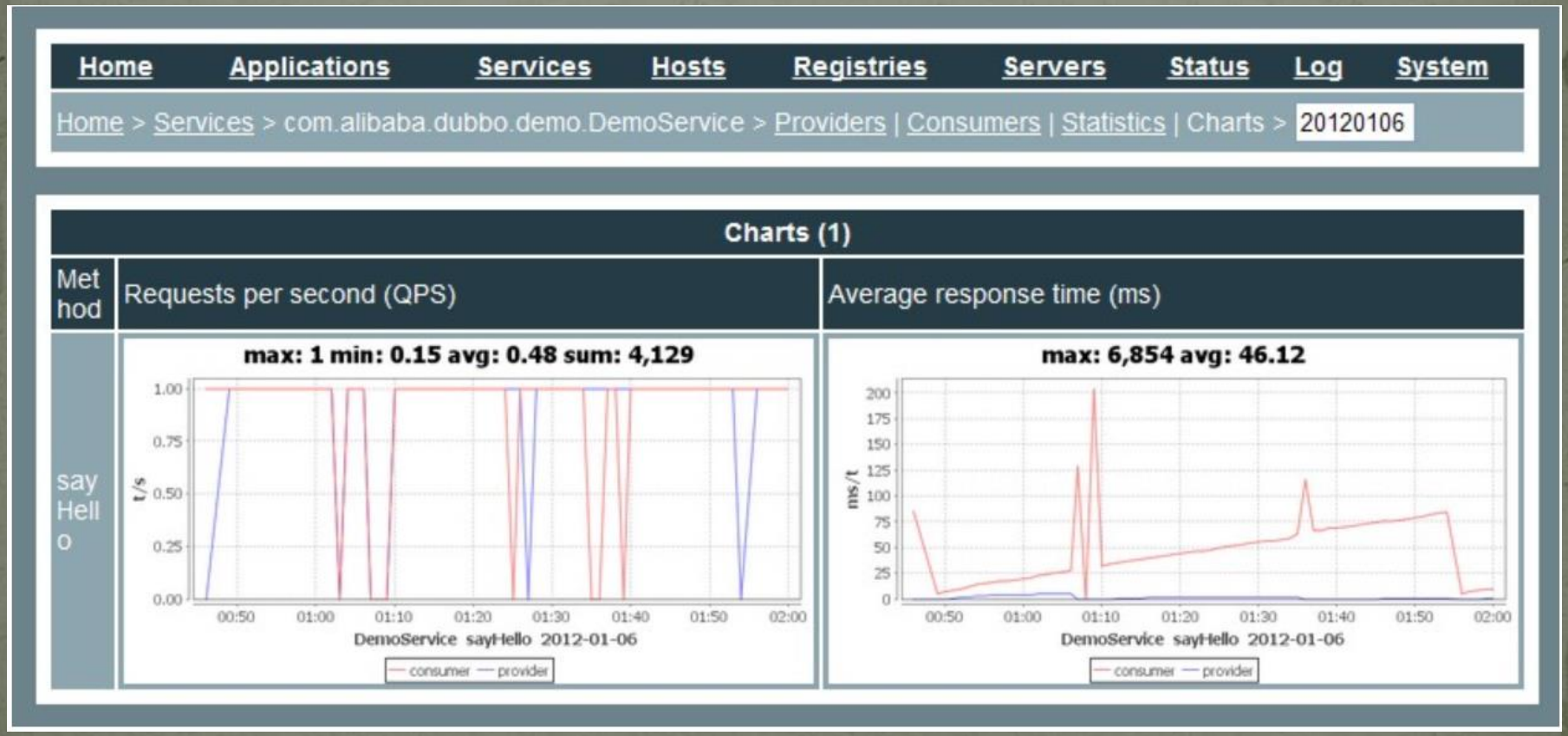
- 支持多种协议
  - Dubbo协议、Hessian协议、HTTP协议、RMI协议、WebService协议、Thrift协议、Memcached协议、Redis协议
- 支持多种序列化
  - Kryo、FST、Dubbo Serialization、Hessian、FastJson、Json、Java Serialization

为什么是Dubbo

- 卓越的性能
  - 运用socket长连接，减少握手
  - 运用NIO及线程池在单连接上并发拼包处理消息
  - 二进制流压缩数据，比常规HTTP等短连接协议更快
  - 在阿里巴巴内部，每天支撑2000多个服务，30多亿访问量，最大单机支撑每天近1亿访问量

为什么是Dubbo

- 完善的服务监控管理



为什么是Dubbo

服务提供者页面

为什么是Dubbo

服务消费者页面

为什么是Dubbo

添加路由规则页面

服务化最佳实践

接口服务

- 接口文档
- 加密解密
- 流量/安全控制
- ...
- 接口系统（api）

服务治理

- 授权和监控
- 注册和发现
- 分布式事务
- Zookeeper Dubbo

业务系统

- 案件办理子系统
- 接处警子系统
- 人口信息管理子系统
- 重点人员管理子系统
- 水警管理子系统
- 绩效考核子系统
- 接处警子系统
- 地址管理子系统
- 机构单位管理子系统
- ...

公共服务

- 对象存储（OSS）
- 消息中心（message）
- 配置中心（config）
- 用户中心（uic）
- 系统日志（logging）
- 蓝信网关（sms）
- 邮件服务（mail）
- 调度服务（scheduler）
- 工作流服务（workflow）
- ......

基础服务

- 静态资源
- SSM基础框架（common）
- 代码生成器

权限服务

- 集中权限
- 单点登录
- 系统组织
- 用户角色
- 权限管理

数据存储层

- Oracle集群
- Redis集群
- OSS(对象存储)
- ......

服务化最佳实践

- 分包
  - 建议将服务接口，服务模型，服务异常等均放在API包中，因为服务模型及异常也是API的 一部分，同时，这样做也符合分包原则：重用发布等价原则(REP)，共同重用原则(CRP)。
  - 如果需要，也可以考虑在API包中放置一份 spring 的引用配置，这样使用方，只需在spring 加载过程中引用此配置即可，配置建议放在模块的包目录下，以免冲突， 如：com/alibaba/china/xxx/dubbo-reference.xml 。

服务化最佳实践

- 粒度
  - 服务接口尽可能大粒度，每个服务方法应代表一个功能，而不是某功能的一个步骤，否则将 面临分布式事务问题，Dubbo暂未提供分布式事务支持。
  - 服务接口建议以业务场景为单位划分，并对相近业务做抽象，防止接口数量爆炸。
  - 不建议使用过于抽象的通用接口，如：Map query(Map)，这样的接口没有明确语义，会给后 期维护带来不便。

服务化最佳实践

- 版本
  - 每个接口都应定义版本号，为后续不兼容升级提供可能，
  - 建议使用两位版本号，因为第三位版本号通常表示兼容升级，只有不兼容时才需要变更服务 版本。
  - 当不兼容时，先升级一半提供者为新版本，再将消费者全部升为新版本，然后将剩下的一半 提供者升为新版本。

服务化最佳实践

- 兼容性
  - 服务接口增加方法，或服务模型增加字段，可向后兼容，删除方法或删除字段，将不兼容，枚举类型新增字段也不兼容，需通过变更版本号升级。
  - 各协议的兼容性不同

服务化最佳实践

- 枚举值
  - 如果是完备集，可以用Enum，比如：ENABLE, DISABLE。
  - 如果是业务种类，以后明显会有类型增加，不建议用Enum，可以用 String代替。
  - 如果是在返回值中用了Enum，并新增了Enum值，建议先升级服务消费方，这样服务提供 方不会返回新值。
  - 如果是在传入参数中用了Enum，并新增了Enum值，建议先升级服务提供方，这样服务消 费方不会传入新值。

服务化最佳实践

- 序列化
  - 服务参数及返回值建议使用POJO对象，即通过setter,getter方法表示属性的对象。
  - 服务参数及返回值不建议使用接口，因为数据模型抽象的意义不大，并且序列化需要接口实 现类的元信息，并不能起到隐藏实现的意图。
  - 服务参数及返回值都必需是byValue的，而不能byReference的，消费方和提供方的参数 或返回值引用并不是同一个，只是值相同，Dubbo不支持引用远程对象。

服务化最佳实践

- 异常
  - 建议使用异常汇报错误，而不是返回错误码，异常信息能携带更多信息，以及语义更友好。
  - 如果担心性能问题，在必要时，可以通过override掉异常类的fillInStackTrace()方法为空方法，使其不拷贝栈信息。
  - 查询方法不建议抛出checked异常，否则调用方在查询时将过多的try...catch，并且不能进行有效处理。

服务化最佳实践

- 服务提供方不应将DAO或SQL等异常抛给消费方，应在服务实现中对消费方不关心的异常 进行包装，否则可能出现消费方无法反序列化相应异常。
- 调用
  - 不要只是因为是Dubbo调用，而把调用try...catch起来。try...catch 应该加上合适的 回滚边界上。
  - 对于输入参数的校验逻辑在Provider端要有。如有性能上的考虑，服务实现者可以考虑在 API包上加上服务Stub类来完成检验。

谁在使用Dubbo

- 阿里巴巴B2B
- 京东（基于Dubbo的Hydra）
- 当当（基于Dubbo的Dubbox）
- ……

谢谢观看