讲师介绍

Hash　QQ：805921455

从事Java软件研发近十年。

前新浪支付核心成员、

咪咕视讯(中国移动)项目经理、

对分布式架构、高性能编程有深入的研究。

明天，你一定会感谢今天奋力拼搏的你

Dubbo入门

分布式系统开发技术

目录

课程安排

01

Dubbo概述

Dubbo框架介绍及使用

02

Dubbo核心功能

核心功能剖析

03

Dubbo协议

Dubbo协议分析、实现一个Dubbo客户端

04

总结

课堂知识总结

目录

Dubbo概述

Dubbo是什么

官网对于Dubbo的解释

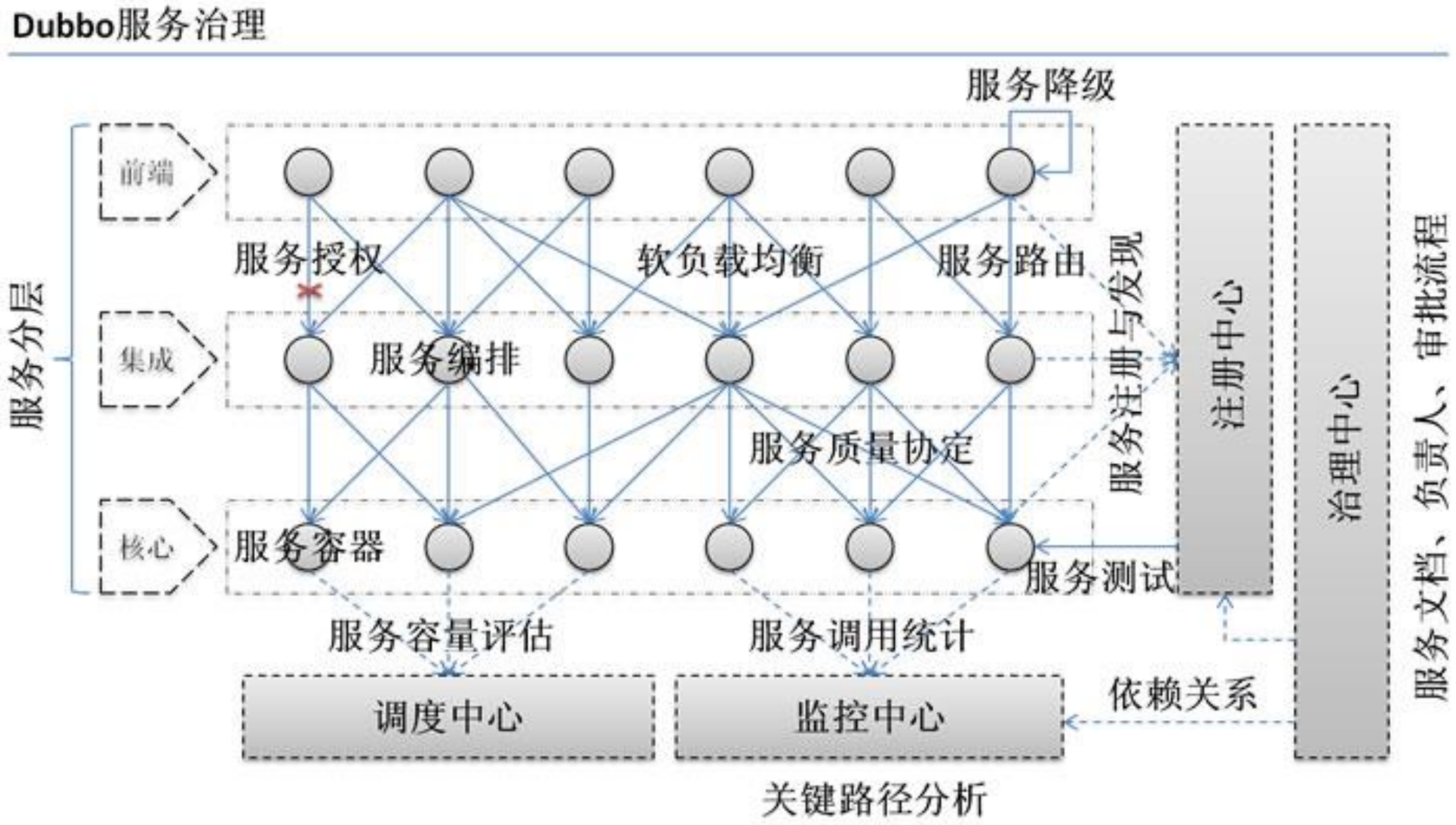
Apache Dubbo (incubating) |ˈdʌbəʊ| 是一款高性能、轻量级的开源Java RPC框架。

它提供了三大核心能力：面向接口的远程方法调用，智能容错和负载均衡，以及服务自动注册和发现。

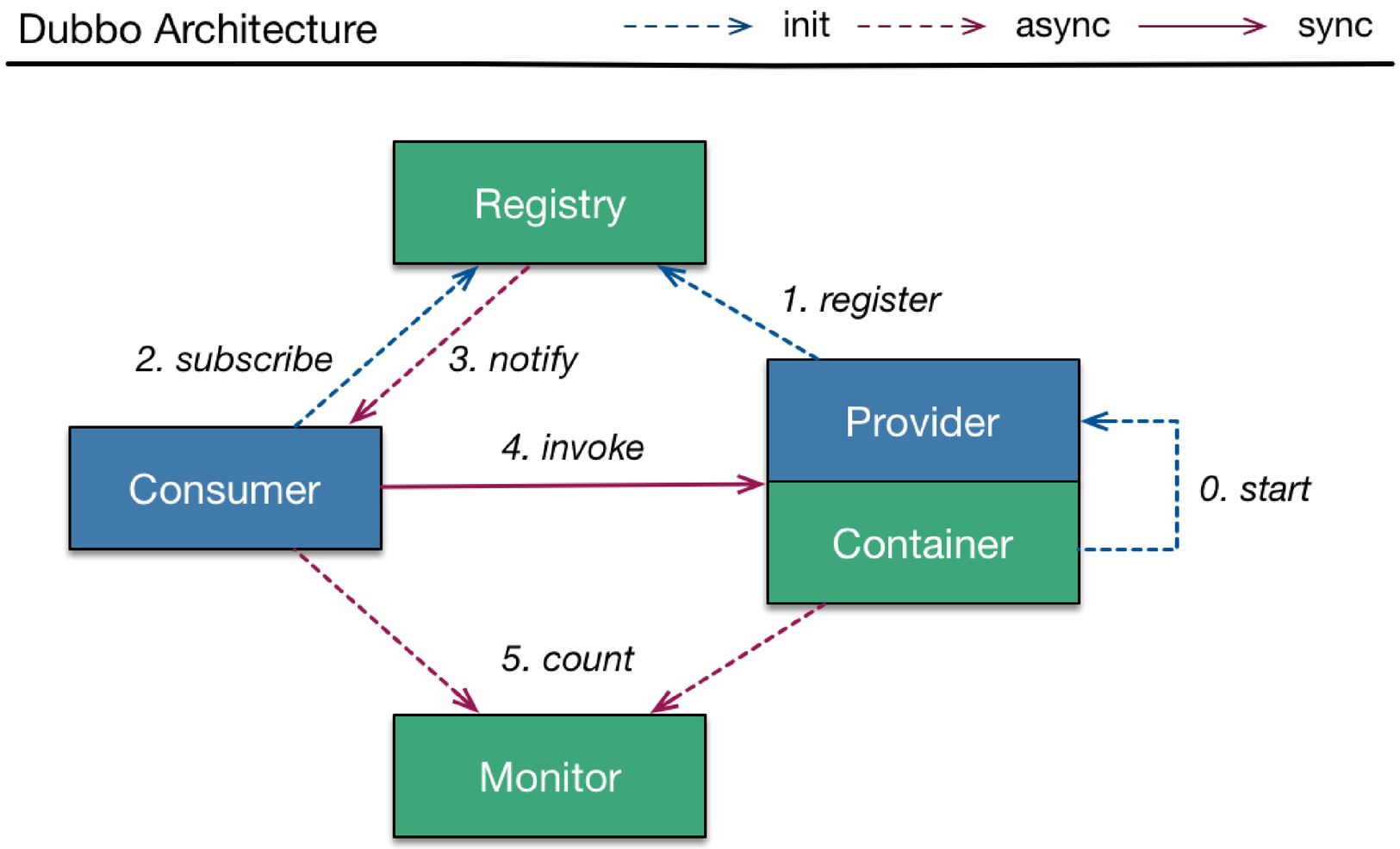
Dubbo能做什么?

功能	描述
服务开发（rpc应用开发）	RMI 或 Hessian 只能简单的暴露和引用远程服务，进行开发。通过配置服务的URL地址进行调用，F5 等硬件进行负载均衡。
服务软负载均衡	通过服务注册中心，动态地注册发现服务，使服务的位置透明，实现软负载均衡和容错机制，降低对硬件负载均衡器的依赖，减少部分成本。
服务依赖管理	服务间依赖关系错踪复杂时，人工难以描述，需要自动画出应用间的依赖关系图。
服务监控	统计服务每天的调用量、响应时间，作为容量规划的参考指标。将某台机器的权重一直加大，并在加大的过程中记录响应时间的变化，直到响应时间到达阈值，记录此时的访问量，再以此访问量乘以机器数反推总容量。
服务治理	可在线动态调整机器权重、服务分组隔离、禁启用服务。

Dubbo能做什么?


Dubbo服务治理
服务降级
前端
服务授权
软负载均衡
服务路由
服务分层
集成
服务编排
服务注册与发现
注册中心
治理中心
服务质量协定
核心
服务容器
服务测试
服务文档、负责人、审批流程
服务容量评估
服务调用统计
依赖关系
调度中心
监控中心
关键路径分析

Dubbo架构



Provider 服务提供者:
我们的service，实际执行业务逻辑的服务层。

Consumer 服务消费者:
专门调用service的，不关注service具体实现的应用层。

Registry 注册中心:
存储Provider，consumer信息的中介。

Monitor：Dubbo负责收集服务调用信息的监控中心。

Dubbo架构调用流程

Dubbo调用关系

1. 服务容器负责启动，加载，运行服务提供者。
2. 服务提供者在启动时，向注册中心注册自己提供的服务。
3. 服务消费者在启动时，向注册中心订阅自己所需的服务。
4. 注册中心返回服务提供者地址列表给消费者，如果有变更，注册中心将基于长连接推送变更数据给消费者。
5. 服务消费者，从提供者地址列表中，基于软负载均衡算法，选一台提供者进行调用，如果调用失败，再选另一台调用。
6. 服务消费者和提供者，在内存中累计调用次数和调用时间，定时每分钟发送一次统计数据到监控中心。

Dubbo架构特点

Dubbo 架构具有以下几个特点，分别是连通性、健壮性、伸缩性、以及向未来架构的升级性。

特点	说明
连通性	注册中心、监控中心宕机不影响连通性，两个组件可选，服务消费端可直连。
健壮性	服务提供者无状态，任意一台宕机不影响使用，全宕机，无限重连。
伸缩性	注册中心集群，可动态增加部署实例，客户端自动发现新的注册中心。可动态增减服务者实例，注册中心将推送新的服务提供者信息给消费者。
升级性	升级到流动式计算，目前的架构毫无压力

Dubbo的依赖

必须依赖

JDK1.6+，理论上 Dubbo 可以只依赖 JDK，不依赖于任何三方库运行，只需配置使用 JDK 相关实现策略

缺省依赖

通过 `mvn dependency:tree > dep.log` 命令分析

```
[INFO] +- com.alibaba:dubbo:jar:2.5.9-SNAPSHOT:compile
[INFO] |  +- org.springframework:spring-context:jar:4.3.10.RELEASE:compile
[INFO] |  +- org.javassist:javassist:jar:3.21.0-GA:compile
[INFO] |  \- org.jboss.netty:netty:jar:3.2.5.Final:compile
```

可选依赖

其他三方相关依赖的jar包，有用到则需要依赖。

Dubbo的使用方式

服务提供端

1. 独立的服务（以普通的java程序形式）
2. 集成在应用中（在应用中增加远程服务能力）

消费客户端

1. 在应用中调用远程服务。
2. 也可是在服务提供者中调用远程服务。

Dubbo的三种配置方式

使用Dubbo注解

使用简单，有一定的侵入性，需要实现类需要依赖Dubbo注解

集成Spring XML

使用稍显麻烦，可做到无侵入性，方便以后改用其他RPC框架

使用原生API

编程开发麻烦，一般用于测试、开放API的场景

使用Dubbo的步骤

1. 引入dubobo相关依赖
2. 配置dubbo框架（提供了3中配置方式）
3. 开发服务
4. 配置服务
5. 启动、调用

```
<dependencies>
<dependency>
 <groupId>com.alibaba</groupId>
 <artifactId>dubbo</artifactId>
 <version>2.6.6</version>
</dependency>
 <!-- 这里我们使用netty -->
<dependency>
<groupId>io.netty</groupId>
<artifactId>netty-all</artifactId>
<version>4.1.32.Final</version>
</dependency>
 </dependencies>
```

Dubbo的使用

```
public interface DemoService {
String sayHello(String value);
}

DemoService demoService = (DemoService) context.getBean("demoService"); // 获取远程服务代理
String hello = demoService.sayHello("world"); // 执行远程方法
System.out.println(hello); // 显示调用结果
```

```
<!-- 生成远程服务代理，像本地bean一样使用demoService -->
<dubbo:reference id="demoService" interface="edu.dongnao.study.dubbo.DemoService" />
```

```
public class DemoServiceImpl implements DemoService {
    public String sayHello(String value) {
        return value + ">>>>>>>>result";
    }
}
```

```
<!-- 声明需要暴露的服务接口 -->
<dubbo:service interface="edu.dongnao.study.dubbo.DemoService" ref="demoService" />
```

SpringBoot中集成Dubbo

方式一，@EnableDubbo 注解

1. 引入对应的jar
2. 在springboot 的启动类上加 @EnableDubbo 注解开启dubbo（服务提供者、消费者的是一样的，扫描的包可能不一样）
3. 在application.yml中配置dubbo

具体引入jar和配置，见代码示例

SpringBoot中集成Dubbo

方式二，dubbo-spring-boot-starter方式

1. 引入dubbo-spring-boot-starter 及对应的dubbo jar
2. 在application.yml完成和方式一相同的配置。在application.yml中通过dubbo.scan.base-packages参数指定dubbo扫描的包（服务提供者、消费者设置方式一样）

具体引入jar和配置，见代码示例

最主要的区别，引入的jar不一样，从而扫描包的方式不一样。

谢谢观看