# Dubbo相关知识点笔记

Dubbo虽然已经不更新了，但是背靠阿里的技术实力，中文资料的丰富，很适合大多数中小型分布式项目的开发，在这里总结摘要一些知识点。

## Dubbo的通信协议

### dubbo

默认的，单一长连接、NIO异步通信。适合数据量小，并发大的场景，消费者数量多于提供者的情况。

传输个大文件、视频、图片什么的还是不要用了。

绝大多数的一般项目用这个就够了。

### RMI

这是JDK标准的远程调用协议，阻塞式短连接&JDK标准的序列化方式。

输入输出的数据包大小混合，消费者和提供者数量差不多，是比较适用的，可以传文件。

个人在项目中没用过，不怎么关注。

### 3、hessian

Hessian采用HTTP通信，由Servlet对外提供服务，在Dubbo中使用Jetty做服务器。很适合传输页面、文件。

### 4、Http

这个没啥好介绍的，主要是适合同时给APP、网页的Ajax提供服务。我在项目中到没这么用过，一般这种情况都是直接用SpringMVC打成war包，通过Keepalived+Nginx+Tomcat进行高可用和负载均衡。

### 5、WebService

老牌的数据交换技术了，SOAP协议，CXF框架实现，2010年之后我在项目中就没用过了。

### 6、thrift

Facebook的RPC框架，Dubbo加了一些头信息，不支持null值。

### 7、缓存

Dubbo支持缓存数据库Memcached、Redis。

## 通信协议的对比

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 协议 | 连接个数 | 连接方式 | 传输协议 | 传输方式 | 序列化 |
| Dubbo | 单 | 长 | TCP | NIO异步 | Hessian二进制 |
| Rmi | 多 | 短 | TCP | 同步 | Java标准二进制 |
| Hessian | 多 | 短 | HTTP | 同步 | Hessian二进制 |
| Http | 多 | 短 | HTTP | 同步 | 表单序列化 |
| WebService | 多 | 短 | HTTP | 同步 | SOAP |

## 注册中心

推荐是Zookeeper，还有如下：

1. Multicast，没有中心，只要节点的广播地址是一样的，都可以互相发现，这个也就是小规模应用，或者开发的时候用。

步骤是：

1. 提供者启动时广播地址。
2. 消费者启动时广播订阅请求。
3. 提供者收到订阅请求是，单播或组播自己的地址给消费者。
4. 消费者收到提供者地址，进行RPC调用。
5. Redis，要求各个服务器时间必须保持一致，因为需要用心跳监测脏数据，对服务器还是有一定压力的。

步骤是：

1. 提供者启动时提交自己的地址，发送注册事件。
2. 消费者启动时订阅注册&解除注册事件，提交自己的地址。
3. 消费者收到注册和解除注册事件后，去获取提供者地址列表。
4. 监控中心启动时订阅注册&解除注册、订阅&解除订阅事件。
5. 监控中心收到注册&解除注册事件后，去获取提供者地址列表。
6. 监控中心收到订阅&解除订阅事件后，去获取消费者地址列表。
7. Simple注册、监控中心，这是Dubbo自己实现一个服务。
8. Zookeeper，步骤如下：
9. 提供者启动时写入自己的URL。
10. 消费者启动时订阅提供者URL列表，并写入自己的URL。
11. 监控中心订阅所有消费者提供者URL目录。

## 提供者如何实现失效踢出

以Zookeeper注册中心为例，有一个heartbeat\_monitor心跳监测线程，检查提供者是否存活，如果检测不到心跳，就把提供者从提供者列表中移除，反之加入。

## 在不影响旧版本的情况下升级新版本服务

使用Dubbo就比较方便，因为在配置提供者和服务者的时候可以指定版本号。

## 服务调用链过长

还是应该结合业务来梳理模块，一旦某个业务操作的调用链过长的时候，就要考虑把这一块整合一下。

## Dubbo核心配置

dubbo:application name应用名

dubbo:registry注册中心

dubbo:protocol使用协议和端口

dubbo:reference订阅的服务接口 check启动时检测服务是否可用 version 多版本

dubbo:service提供者 cluster容错机制 retries失败重试次数 loadbalance负载算法

## 是否可以直连提供者

可以，在dubbo:reference通过配置url直接指向某个提供者，多地址用逗号分隔。

## 服务注册与发现的流程图

3、调用

提供者有变动时通知

1、注册

2、订阅提供者URL列表缓存至本地

注册自己的地址

Zookeeper

消费者

提供者

## 集群容错

默认的failover，再加上retries，代表消费者调用提供者出现异常时，自动切换，默认试两次。

failfast，失败了就报异常，主要是用于非等幂性的写操作。当然这个也等价于failover，retres=0。

failsafe，失败就失败了，也不报异常，用于一些不重要的操作，比如写个不重要的日志什么的，错个1、2条的不影响。

failback，失败了重发，比如推送。

forking，调用所有提供者，谁最先返回结果就用谁的，这个太耗资源了。

broadcast，调用所有提供者，一个报错就报错，这个主要是用于通知提供者更新缓存啊，本地资源之类的操作。