这篇总结，迟迟没有写，唉，不应该：

Imooc上看的一篇关于dubble+zk实现的一个RPC微服务框架

感觉讲的还可以：

首先总结下rpc:

RPC就是一个远程过程调用的协议

实现原理：

生产者注册到服务器（Zookeper），消费者订阅服务器，有消息产生时则会做出相应的消费行为，没有消息则等待消息产生，生产者生产消息到服务器，

微服务就是采用的这个协议

典型的微服务dubble + zk

有几个角色：

生产者、zk、消费者、监听器（用于监听服务的执行次数，成功与否）

Dubble：

Dubble支持多种通讯协议，主要支持两大支持协议

RPC/Dubble协议：同种结构的项目，可以像在本地调用一样简单，适用于在公司内服务调用；

Rpc方式调用服务时要注意的几个点：

发布服务的实现类上添加dubble的@service注解，并设置其他参数，如调用方式、调用参数类型。

在类上添加@path(路径)注解，类似于@requertmapping()，其发布的方法也要添加这个注解，因为后期可能有多个服务，dubble要识别你要调用的是哪个服务

配置文件要加你要获取的服务的具体业务层，在发布方的哪个业务层下，给他加bean。

消费者要调用的时候需要把发布的服务jar包添加到本地仓库，才能像在本地调用一样。

发布的服务和调用方定义的配置版本要一致

当发布方修改代码后，要及时的更新jar包到仓库，否则调用方会异常，访问不到。

Rest方式就比较简单了

在发布方（生产者）的发布方法的实现类上加@parth注解、在其方法上也加@path注解声明路径、添加发送和响应的数据类型的注解、如果有参数也需要加关于参数的注解@queryparam（），然后调用方（消费方）直接http调用就行（看的视频里用的是okhttp3，感觉过程好繁琐，其实用httptempleat就可以直接调用，或者httpclean也可以，他说的是那种调用方式效率高，没有具体说明调用的优点。）

注意的几个地方：

还是生产者和调用者配置的版本要一致

http rest Api: 通用项目，一般用于给外包使用服务

他有两个重要的配置文件：

Spring-dubble.xml:

发布的服务所在的包的注解

配置两种调用的方式rpc、rest

消费注册服务配置

Dubble.properties:（从阿里官网上copy就行）

主要用来配置两大协议对应的端口信息

Zookeper：（他在这个RPC里主要担当了一个服务注册和订阅的功能）

看的这个项目主要分为四个模块

父模块：管理整个项目

子模块-api

子模块-model

子模块-server

引入依赖，加配置文件

在这里呢总结了几个问题：

1. 为什么用微服务这种rpc远程调用过程这种框架？

主要是用于在分布式、微服务，也就是说当业务越来越多时，可能子模块会有几十各甚至几百个，微服务也是由此而生，充当服务的管理者

1. Rpc调用的过程和我所知的http请求作用一样，区别在哪，优点？

前者性能更好，请求更高效，对于非常多的子模块来说是这样的

后者是直接调用的，没办法管理

1. Rpc有两种协议构建的请求方式，rpc/dubbo、rest这两个有什么区别？

前者是把请求的服务打包到调用方的模块里，实现像调用本地方法一样，直接把请求的接口交给spring管理，调用bean就可以实现，适用于在公司使用。

后者直接http根据发布方声明好的请求参数类型和响应类型，还有@path组成的请求路径就可以了。适用于给第三方提供服务。

最后再总结下我个人认为的消息中间件和微服务之间的关系，因为前两天学习dubbo+zk的微服务、activemq、ribbtmq总感觉他们很像，感觉真是一类的，其实并不是：

相同点，他们都有生产者、消费者

Xxxxxxxxxxxxxxxxx

又看了一遍，是我理解错了

Activemq（实现了jms规范）

博客地址：**https://blog.csdn.net/cs\_hnu\_scw/article/details/81040834**

有两种消息模型：

队列消息模型：生产者发出的每个消息只能提供给一个消费者

启动不分前后顺序，前提是要有生产的消息

主体订阅模式：生产者发出的消息被所有的消费者共享

必须先启动消费者发送消息，不然在消费者之前启动的生产消息是收不到的

消费实现：（生产和消费的实现是差不多的）（队列和主体订阅两种配置差不多）

下载安装activemq,并且启动，按照下面的写业务代码就可以了

//定义ActivMQ的连接地址

private static final String ACTIVEMQ\_URL = "tcp://127.0.0.1:61616";

//定义发送消息的队列名称

private static final String QUEUE\_NAME = "MyMessage";

//创建连接工厂

ActiveMQConnectionFactory activeMQConnectionFactory = new ActiveMQConnectionFactory(ACTIVEMQ\_URL);

//创建连接

Connection connection = activeMQConnectionFactory.createConnection();

//打开连接

connection.start();

//创建会话

Session session = connection.createSession(false, Session.AUTO\_ACKNOWLEDGE);

//创建队列目标

Destination destination = session.createQueue(QUEUE\_NAME);

//创建消费者

javax.jms.MessageConsumer consumer = session.createConsumer(destination);

//创建消费的监听

consumer.setMessageListener(new MessageListener() {

消费者有个专门的监听方法，可以监听所指向的消息队列或者主题，只要有消息过来就消费他。

这种写法比较繁琐，而spring提供了和active整合的方案，简化了开发：

引入spring和active 相关依赖：

编写生产者的配置文件和消费者的配置文件，然后用spring为我们提供的JmsTemplate.send方法发送消息。

jmsTemplate.send(destination , new MessageCreator() {

@Override

public Message createMessage(Session session) throws JMSException {

TextMessage textMessage = session.createTextMessage(msg);

return textMessage;

}

});

System.out.println("现在发送的消息为： " + msg);

把写好的这个实现类加入到spring容器中

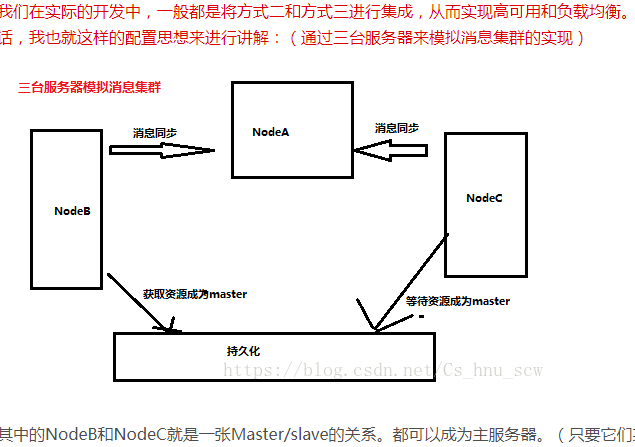
然后在写控制层或测试层调用发送消息的方法，用spring的大容器取出我们注册的bean然后调用其方法：ClassPathXmlApplicationContext classPathXmlApplicationContext = new ClassPathXmlApplicationContext("producer.xml");

ProduceService bean = classPathXmlApplicationContext.getBean(ProduceService.class);

最后关闭bean（作用就是为了关闭与activemq的连接，免得占用资源）

Activemq大都是做一个集群操作，防止单点故障，有三种集群，用的最多的是**主从集群()实现高可用 + Broker clusters：多个Broker之间同步消息。(实现负载均衡)**

**用三台服务器来讲就是i**

一个宕机，那么另一个立刻进行服务

从以上就可以看出来，其实active mq和微服务是完全不同的两个概念：

前者是生产者生产出消息到activemq,而消费者监听指定的队列或主题，如有消息会消费消息做其他的业务操作；而且他们是没有所谓的注册关系的，是直接生产消息到队列或主题，消费也是指定消费

主要用于异步操作比如说：订单业务，用户下订单后，要进行发短信通知、通知服务、等等一系列的操作，而这些操作并不需要立刻去完成，只是需要在后台异步执行，直到成功而已。为了就是减缓系统的瞬时压力，和执行的性能。这样也能防止当下订单完成后，订单系统宕机或者短信系统宕机而直接影响整个业务系统操作。

后者的话就不一样了，他的业务场景是生产者生产消息去调用远程服务，而远程服务要立刻消费，不存在存储消息这一说，而且是一对一的，两个的概念完全不一样好吧。

只有微服务有注册和订阅消息这么一说,

生产者注册服务到zk，消费者订阅zk的服务，当要调用服务时，直接去zk里访问服务，进而访问服务块

正常的业务场景是用户调用服务时，也就是消费者去调用生产者对的，两者之间从执行逻辑到业务场景都不一样。别把他们搞混了！！！！！！

在补充一点：通常启动activemq是以服务的方式启动的

下面我还得再总结下rabbit mq

唉，都是之前不认真：

由于时间关系，今天先写一点，明天完善：

博客地址：https://blog.csdn.net/hellozpc/article/details/81436980

先安装erlang，因为rabbitmq是erlang语言写的，下载安装rabbitmq，启动步骤：

先启用管理工具

1、双击这里写图片描述

2、进入C:\Program Files (x86)\RabbitMQ Server\rabbitmq\_server-3.4.1\sbin输入命令：

rabbitmq-plugins enable rabbitmq\_management

再启动：

停止：net stop RabbitMQ

启动：net start RabbitMQ

然后登陆浏览器登陆，端口默认为**15672**

**要输入用户名、密码 默认为： guest/ guest**

**有五种队列：（）**

**DOTO++++++++++++++++0904**