2019年1月23日星期三

**Dubbo**

今天看了小锋的dubbo视频，简单的了解了dubbo，小锋不是很推荐dubbo。

简单的dubbo使用

Zookeeper的安装

服务注册 将接口方法注册到zookeeper

消费端订阅服务

Dubbo admin 安装

Dubbo rpc服务接口单独一个形成一个工程，打成一个jar，可以供消费方和服务方共同使用。

Dubbo集群（app名称设置成一样就是一个集群） 通过admin设置服务实例权重，来达到负载均衡

**Springcloud**

Springcloud是在spring-boot的基础上构建起来的，服务之间通过http接口通信

注册中心 erueka

服务方将appname 和ip、port 注册到erueka

消费方通过RestTemplate来调用服务方接口

为了保证注册中心高可以用，erueka可以集群部署

为了保证服务的高可用，服务端会有多个实例注册到erueka提供服务。这样需要负载均衡来将请求合理的分发到不同的服务实例上面。

客户端负载均衡 ribbon

有没有服务端负载均衡解决方案？

客户端通过RestTemplate调用服务端http接口，还有ribbon负载均衡，这些都是非业务功能。但是却和业务功能代码写在一起，耦合性太高，混杂，代码不够清晰。因此引入feign

Feign将这些代码与业务代码隔离开来。使代码结构更清晰。

Hystrix 断路器。服务熔断，服务降级。主要功能是防止雪崩，导致整个应用响应时间变长，进而崩溃掉。

雪崩现象解释：一般前端（手机端、pc端、其他客户端）发起请求，请求通过网络发送到后台web服务器，包含应用的所有服务的入口，如果底层的某个服务响应时间很长，或者服务端种种原因不能及时返回结果（正常或者异常返回）。导致请求堆积在web服务器，长期不能释放，web服务器会对其很多请求线程，导致web服务器的资源（内存、cpu、带宽等）消耗，web服务器响应其他正常的请求会变慢，甚至不能访问。最后整个应用所有的服务都不能正常提供服务。

Hystrix机制，就是设置底层服务响应时间，如果超过这个时间还不能正常返回。就会放弃转向调用准备好的熔断方法，快速返回。这样web服务器就不会对其很多请求线程。不会影响其他的正常服务。熔断的服务处理是另一回事了。响应时间过长的服务需要告警，需要人工分析找到问题。

Hystrix服务监控，dashboard可以监控某一服务实例调用情况。有很对指标。不知道能不能监控到某一个方法？

集群监控turbine 监控服务的所有实例，就是具有同一个应用名的所有服务。

Feign和Hystrix可以整合，解耦。

Zuul路由网管服务。添加过滤器，实现网管功能，比如可以检查token等。

隐藏真正的服务。Url服务映射。

今天只是简单整体的了解了两种分布式解决方案。粗浅的认识，要精通，还需要不断的深入。

**2019年1月24日星期四**

今天开始看吴水成的 基于dubbo的分布式系统，他这个视频是很早就录制了，dubbo已经有了更新，不太一样了。我打算将整个视频看完，整体感觉一下，然后再动手操作实践一下。

将dubbo，以及整个开发集成环境，搭建部署一下。完整的学习dubbo，加上git上的源码，文档对着学习。最后再学习一下springcloud，和springboot，再对比一下。Dubbo和springcloud

Dubbo程序可以打包成可执行jar来部署服务。主类org.apache.dubbo.container.Main

Dubbo-admin安装

Dubbo-moniter 安装，并访问

视频里面用maven打包的方式比较特别，依赖的jar包拷贝到指定的lib文件夹中，jar包中的manifest文件会将依赖的jar引进来。还有部署维护dubbo程序的shell脚本挺新颖的。以前在工作中都是用servlet容器部署dubbo程序的。但是视频中解释说用servlet容器和spring容器启动dubbo程序都不好。但是现在dubbo官网上面也没有特别推荐说用哪一种。以前的文档已经找不到了。

Maven工程，不一定都是可以执行工程，配置文件单独分出来一个工程，主要是统一管理配置，供其他工程引用。还有dubbo接口定义的工程，全都是java interface，用户dubbo客户端和服务端引用。还有以下公用的工具类，dao层访问数据的代码，都是不能单独运行的。

Maven依赖 可以import 其他maven pom中配置的依赖，

<dependency>  
 <groupId>org.springframework</groupId>  
 <artifactId>spring-framework-bom</artifactId>  
 <version>${spring\_version}</version>  
 <type>pom</type>  
 <scope>import</scope>  
</dependency>

Maven的父工程只有一个pom文件，主要用于jar包版本管理，第一全局变量， 打包方式为pom

持续集成，敏捷开发的环境，jenkins、sonar、git、maven

视屏中使用的是svn，讲解了svn服务器的安装，并介绍了怎么使用。

Git是分布式的，照理说是没有服务端的。平时自己的代码都提交到github、gitlab上面。如果公司比较商业的代码应该不是放在别人的服务上面的。

Gitlab好像可以自己搭建一个，代码托管的git服务。

应该是这样的。

他这个视频不精简，太长了，今天学习的东西就这些了，明天从13讲继续。可以快进的地方就快进了。