spring cloud|01 微服务简介

作者：qq692310342
原文链接：https://ld246.com/article/1565577776390
来源网站：链滴

单体架构

简介

在软件设计中，频繁被使用的就是我们的经典三层架构了。

1、表示层：用于直接与用户进行交互，通常是页面、UI等；

2、业务逻辑层（service）：用于处理业务，比如用户从表示层输入了消息就要经过业务逻辑层的逻处理之后再进行的相关操作；

3、数据访问层（dao）：用于与数据库进行交互；

而在单体应用中，我们就会把这三层都放在一个工程中，最终通过打成war包发布到服服务器的tomca的web-app中上线。可以说是十分方便的一个设计理念了！

优劣

单体应用的优势在于：

1、性价比非常高，通常只需要一台服务器就能够把项目跑起来；

2、开发的速度也比较快，运维较简单，项目架构比较简单明了，适合小型应用开发。

单体应用的劣势在于：

1、所有的业务相关操作都放在了一个服务器上，如果项目中某个业务出现了bug，不急时发现就会致整个项目的瘫痪，最终宕机，而这对于一个大型网站来说无疑是十分致命的问题；

2、业务越来越复杂的时候，单体应用的代码量就会越来越多，导致最后的代码可读性、可维护性越越差，最终只能进行重构；

3、用户越来越多的时候，单体应用的并发能力有限；

由此可见：在应用初期，单体应用从成本、开发时间和运维等方面都有优势，但是单体应用会随着业量和用户量的增加，会暴露出的缺点也是显而易见的，所以到了现在的这个完全互联网的时代，单体用已经不适应时代的发展了！

微服务

什么是微服务

微服务最初是由Martin Fowler在2014年的一篇文章中提出来的，简单来说，就是将单一的程序开发一个微服务，每个服务运行在自己的进程中，通常使用HTTP RESTful API的通信风格，独立部署的工！

微服务特点

1、微服务单元按业务来划分：

服务到底要多“微”，这是一个很难的界定的概念，可以从三个方面来定义：

1、根据代码量定义

2、根据开发时间的长短来定义

3、根据业务大小来划分

按业务划分的微服务是主流，各个微服务独立部署，独立运行在进程中。微服务单元是高度组件化的块，并且提供了稳定的模块边界，**服务与服务之间没有任何耦合。**

2、微服务通过HTTP来互相通信

微服务之间通过简单的HTTP来调用，更多的是使用RESTful API的风格来调用，实现了服务与语言和台无关，例如：使用JAVA写的微服务可以消费使用Python写的服务。

服务之间通信也可以通过轻量级的消息总线来实现，例如：RabblitMQ、Kafaka等，通过发布-订阅设计模式来实现服务之间通信。

服务与服务之间通信的数据格式一般使用的是json和xml，这两个也是与语言、平台无关的，一般来说son更加高效、轻量。另一种是使用Protobuf进行数据的序列化，这种方式比json更加轻量，但是可性十分差，需要反序列化才能读懂，所以在Protobuf在通信协议和数据存储中经常被使用到。

3、微服务的数据库独立

服务会有他的独立的数据库，数据库之间没有任何的联系，这样的好处在于，随着业务的不断扩张，据库相对独立，数据量不会太大，易于维护；

但是随之而来的问题就是如何解决分布式的事务问题了；（这个在后续介绍）

4、微服务的自动部署

一个大工程里会有许多的微服务，如果让人工去手动部署的话，难免会出纰漏，但是随着技术的发展docker的容器化技术的出现，微服务的自动部署的出现，让微服务的部署越来越简便。

5、服务集中化管理

随着服务的增多，服务的管理也就越来越麻烦了，所以需要使用集中化的管理方式，市场上的主流框就是 Spring Cloud提供的Eureka注册中心来注册服务和发现服务，另外，zookeeper和Consul都是秀的服务集中化管理框架。

6、分布式的架构

分布式的系统是集群部署的，通常是由许多台计算机共同完成了一个微服务的部署，而分布式的架构通过HTTP来通信的，所以我们的微服务可以搭建在相隔万里的不同的两台计算机上，对于空间没有何束缚。

微服务架构是分布式的架构，而分布式的架构比单体架构更为复杂，主要要确定服务的独立性和服务准去可靠性，以及分布式事务、全局锁、全局Id等问题，都是分布式系统需要考虑的。

7、熔断机制

为了防止一个服务出现bug，导致的系统资源的耗尽而引起的雪崩效应，系统应该对微服务具有一个断机制；

熔断机制的意思就是：在一个微服务出现bug的时候，请求失败次数达到一定的阈值之后，通过熔断让这个微服务断开服务主体，并且快速返回想要显示的错误信息，过一段时间后再重新连接测试，如反复的一个机制来保护整个系统的安全运作。

Spring Cloud中对于服务的熔断提供了Hystrix来实现。

优劣

微服务的优势：

1、服务进行拆分，每个服务只是负责小小的一块内容，这让复杂问题简单化，开发、维护单个服务为简单；

2、微服务的系统是分布式的系统，服务与服务之间没有任何耦合，随着业务的增加，我们可以根据务再拆分服务，这让微服务系统具备很强大横向扩展能力；

3、微服务之间完全通过HTTP协议来进行通信，单个微服务内部高度耦合，服务与服务之间完全独立；

4、重写单个微服务的业务代码变得较为简单；

5、微服务在CAP理论中采用的是AP架构，具有高可用（Availability）和分区容错（Partition toleran e）的特点，高可用体现在系统7*24小时不断的服务，它要求系统具有大量的服务器集群配置，分区错性也让系统更加的健壮。

微服务的劣势：

1、微服务的复杂度比单体服务更为复杂，更难拆分，这让我们的服务的架构设计上应该设计出一个棒的架构！

2、分布式的事务处理，如何处理分布式事务是一个业界所一直存在的问题，一般的处理方式是分为阶段的提交:

第一个阶段：服务通过发起一个分布式事务，交给事务协调器TC进行处理，事务调节器TC通过向所参与该事物的服务节点发送处理事务的准备操作，所有的参与节点执行准备操作，将Undo和Redo信写进日志中，并且向事务管理器返回准备操作是否成功；

第二阶段：事务协调器TC在一定的时间阈值收集所有节点的准备操作是否成功，如果都成功，则通知有的节点执行提交操作，如果有一个失败了，则执行回滚操作！

微服务的设计原则

1、如果在LAMP单体架构够用的情况下，就该使用LAMP，因为它开发速度快，性价比高，但是随着务的发展，用户的激增，可以考虑把数据库读写分离、加缓存、加复杂均衡服务、将应用集群化部署等，如果还不够解决效率，那就可以考虑使用分布式系统，例如微服务的系统架构。

2、微服务在设计的时候一定要考虑到三大难题，服务故障的传播性、服务的划分、分布式事务的处。总之，微服务的设计是渐进的，并且是随着业务发展而发展的！

END

2019年8月12日10:42:44