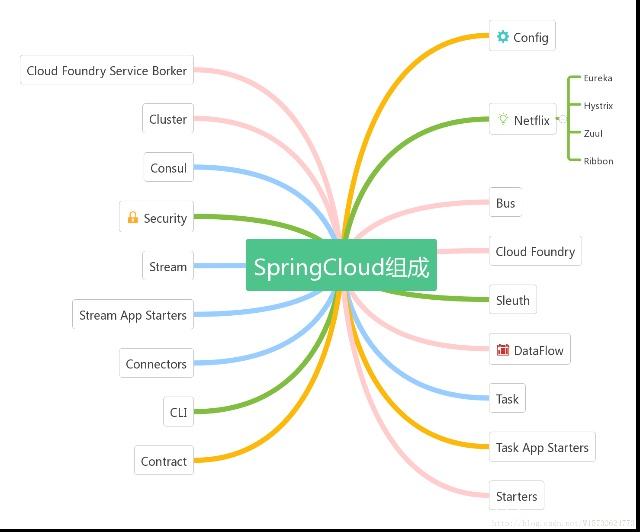
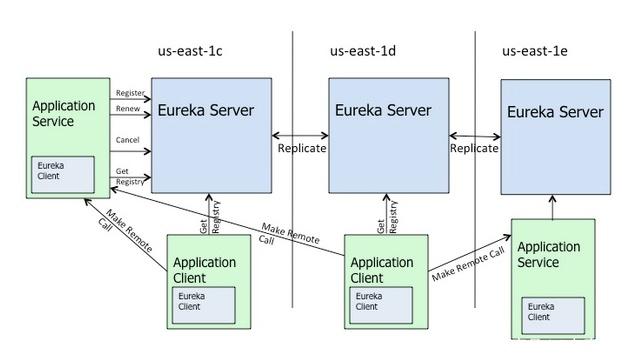
Java框架与面试

1. Java框架
2. Spring
3. SpringMVC
4. SpringBoot
5. SpringCloud



服务发现——Netflix Eureka客服端负载均衡——Netflix Ribbon断路器——Netflix Hystrix服务网关——Netflix Zuul分布式配置——Spring Cloud Config



作用：**实现服务治理**（服务注册与发现）

简介：Spring Cloud Eureka是Spring Cloud Netflix项目下的服务治理模块。

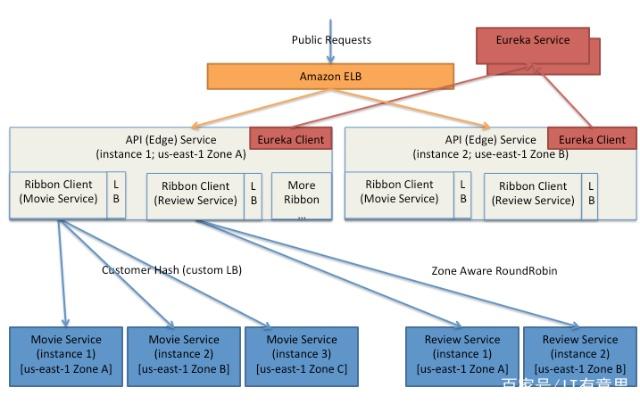
由两个组件组成：Eureka服务端和Eureka客户端。

**Eureka服务端用作服务注册中心。支持集群部署。**

Eureka客户端是一个java客户端，用来处理服务注册与发现。

在应用启动时，Eureka客户端向服务端注册自己的服务信息，同时将服务端的服务信息缓存到本地。客户端会和服务端周期性的进**行心跳交互**，以更新服务租约和服务信息。

Ribbon：

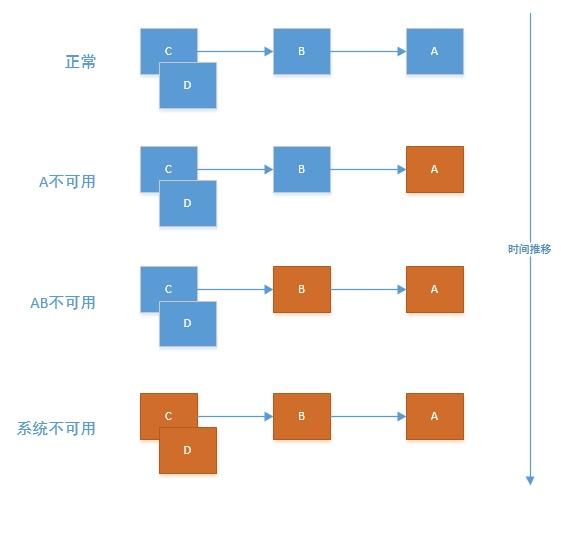


作用：Ribbon，主要提供客户侧的软件负载均衡算法。

简介：Spring Cloud Ribbon是一个基于HTTP和TCP的客户端负载均衡工具，它基于Netflix Ribbon实现。通过Spring Cloud的封装，可以让我们轻松地将面向服务的REST模版请求自动转换成客户端负载均衡的服务调用。

注意看上图，关键点就是将外界的rest调用，根据负载均衡策略转换为微服务调用。Ribbon有比较多的负载均衡策略，以后专门讲解。

**Hystrix：**



作用：断路器，保护系统，控制故障范围。

简介：为了保证其高可用，单个服务通常会集群部署。由于网络原因或者自身的原因，服务并不能保证100%可用，如果单个服务出现问题，调**用这个服务就会出现线程阻塞，此时若有大量的请求涌入，Servlet容器的线程资源会被消耗完毕，导致服务瘫痪。服务与服务之间的依赖性，故障会传播，会对整个微服务系统造成灾难性的严重后果，这就是服务故障的“雪崩”效应。**

1. 框架面试

（四）SpringCloud

,微服务架构是—种架构模式或者说是一种架构风格,它提倡将单一应用程序划分成-一组小的服务,每个服务运行在其独立的自己的进程中,服务之间互相协调、互相配合,为用户提供最终价值。服务之间采用轻量级的通信机制互相沟通(通常是基于HTTP的REST ful API)。毎个服务都围绕着貝体业务进行构建,并且能够被独立地部罟到生产环境、类生产环境等另外,应尽量避免统一的、集中式的服务管理机制,对具体的\_个服务而言,应根据业务上下文,选择合适的语言、工具对其进行构建,可以有一个非常轻量级的集中式管理来协调这些服务,可以使用不同的语言来编写服务,也可以使用不同的数据存储。

微服务 强调的是服务的大小,它关注的是某一个点,是具体解决某一个问题/提供落地对应服务的一个服务应用

狭意的看,可以看作 Eclipse!里面的个个微服务工程/或者 Module