在**[微服务](http://lib.csdn.net/base/microservice" \o "微服务知识库" \t "http://blog.csdn.net/forezp/article/details/_blank)[架构](http://lib.csdn.net/base/architecture" \o "大型网站架构知识库" \t "http://blog.csdn.net/forezp/article/details/_blank)**中，我们将业务拆分成一个个的服务，服务与服务之间可以相互调用（RPC）。为了保证其高可用，单个服务又必须集群部署。由于网络原因或者自身的原因，服务并不能保证服务的100%可用，如果单个服务出现问题，调用这个服务就会出现网络延迟，此时若有大量的网络涌入，会形成任务累计，导致服务瘫痪，甚至导致服务“雪崩”。

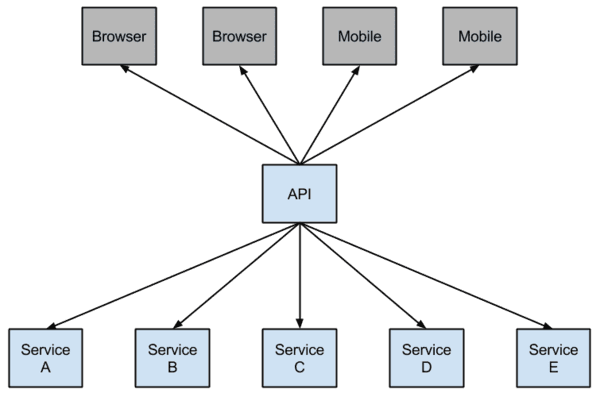
为了解决这个问题，就出现断路器模型。

### 一、断路器简介

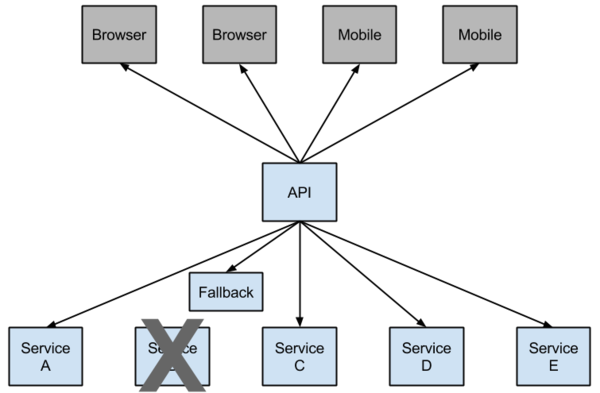
Netflix has created a library called Hystrix that implements the circuit breaker pattern. In a microservice architecture it is common to have multiple layers of service calls.

. —-摘自官网

Netflix已经创建了一个名为Hystrix的库来实现断路器模式。 在微服务架构中，多层服务调用是非常常见的。



较底层的服务如果出现故障，会导致连锁故障。当对特定的服务的调用达到一个阀值（hystric 是5秒20次） 断路器将会被打开。



断路打开后，可用避免连锁故障，fallback方法可以直接返回一个固定值。

### 二、准备工作

基于上一篇文章的工程，首先启动：   
基于上一节的工程，启动eureka-server 工程；启动service-hi工程，它的端口为8762;

### 三、在ribbon使用断路器

改造serice-ribbon 工程的代码：

在pox.xml文件中加入：

<dependency>

<groupId>org.springframework.cloud</groupId>

<artifactId>spring-cloud-starter-hystrix</artifactId>

</dependency>

在程序的入口类加@EnableHystrix：

@SpringBootApplication

@EnableDiscoveryClient

@EnableHystrix

public class ServiceRibbonApplication {

public static void main(String[] args) {

SpringApplication.run(ServiceRibbonApplication.class, args);

}

@Bean

@LoadBalanced

RestTemplate restTemplate() {

return new RestTemplate();

}

}

改造HelloService类，加上@HystrixCommand，并指定fallbackMethod方法。

@Service

public class HelloService {

@Autowired

RestTemplate restTemplate;

@HystrixCommand(fallbackMethod = "hiError")

public String hiService(String name) {

return restTemplate.getForObject("http://SERVICE-HI/hi?name="+name,String.class);

}

public String hiError(String name) {

return "hi,"+name+",sorry,error!";

}

}

启动：service-ribbon 工程，当我们访问[http://localhost:8764/hi?name=forezp](http://localhost:8764/hi?name=forezp" \t "http://blog.csdn.net/forezp/article/details/_blank),浏览器显示：

hi forezp,i am from port:8762

此时关闭 service-hi ,工程，当我们再访问[http://localhost:8764/hi?name=forezp](http://localhost:8764/hi?name=forezp" \t "http://blog.csdn.net/forezp/article/details/_blank)，浏览器会显示：

hi ,forezp,orry,error!

这就证明断路器起作用了。

### 四、Feign中使用断路器

如果你使用了feign，feign是自带断路器的，并且是已经打开了。如果使用feign不想用断路器的话，可以在配置文件中关闭它，配置如下：

feign.hystrix.enabled=false

基于service-feign我们在改造下,只需要在SchedualServiceHi接口的注解中加上fallback的指定类就行了：

@FeignClient(value="service-hi",fallback=SchedualServiceHiHystric.class)

public interface SchedualServiceHi {

@RequestMapping(value = "/hi",method = RequestMethod.GET)

String sayHiFromClientOne(@RequestParam(value = "name") String name);

}

SchedualServiceHiHystric类：

@Component

public class SchedualServiceHiHystric implements SchedualServiceHi {

@Override

public String sayHiFromClientOne(String name) {

return "sorry "+name;

}

}

启动四servcie-feign工程，打开[http://localhost:8765/hi?name=forezp](http://localhost:8765/hi?name=forezp" \t "http://blog.csdn.net/forezp/article/details/_blank),注意此时service-hi还没打开,网页显示：

sorry forezp

打开service-hi，网页显示；

>

hi forezp,i am from port:8762

这证明断路器起到作用了。

五、Circuit Breaker: Hystrix Dashboard (断路器：hystrix 仪表盘)

基于service-ribbon 改造下：

pom.xml加入：

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-actuator</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.cloud</groupId>

<artifactId>spring-cloud-starter-hystrix-dashboard</artifactId>

</dependency>

在主程序入口中加入@EnableHystrixDashboard注解，开启hystrixDashboard：

@SpringBootApplication

@EnableDiscoveryClient

@EnableHystrix

@EnableHystrixDashboard

public class ServiceRibbonApplication {

public static void main(String[] args) {

SpringApplication.run(ServiceRibbonApplication.class, args);

}

@Bean

@LoadBalanced

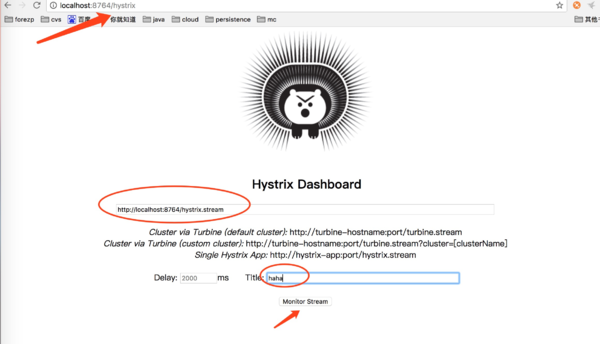
RestTemplate restTemplate() {

return new RestTemplate();

}

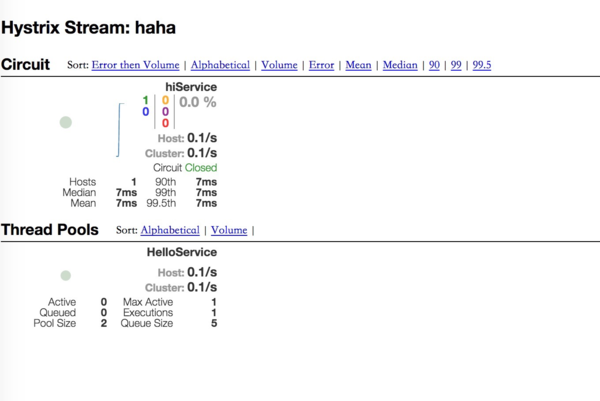
}

打开浏览器：访问[http://localhost:8764/hystrix](http://localhost:8764/hystrix" \t "http://blog.csdn.net/forezp/article/details/_blank),界面如下：



点击monitor stream，进入下一个界面，访问：[http://localhost:8764/hi?name=forezp](http://localhost:8764/hi?name=forezp" \t "http://blog.csdn.net/forezp/article/details/_blank)

此时会出现监控界面：



勘误：有人反映feign的熔断器不起作用，springcloud版本的问题，用这个：

<dependencyManagement>

<dependencies>

<dependency>

<groupId>org.springframework.cloud</groupId>

<artifactId>spring-cloud-dependencies</artifactId>

<version>Camden.SR6</version>

<type>pom</type>

<scope>import</scope>

</dependency>

</dependencies>

</dependencyManagement>

问题Feign 默认是关闭熔断的新版本

feign:

hystrix:

enabled: **true开启**