中生代技术&iTechPlus上海

2017

年度技术峰会

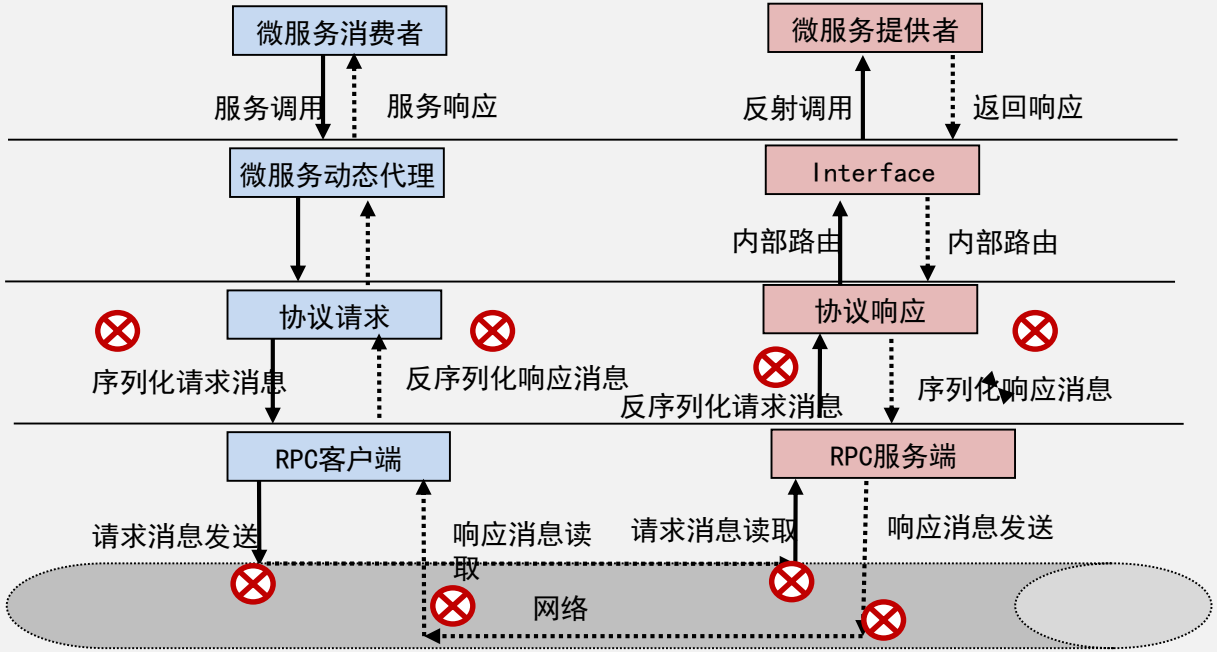
微服务故障隔离技术

李林锋

无处不在的故障-分布式调用故障

分布式调用之后，新引入的故障：

- ✓ 序列化与反序列化故障
- ✓ 分布式路由故障
- ✓ 网络通信故障

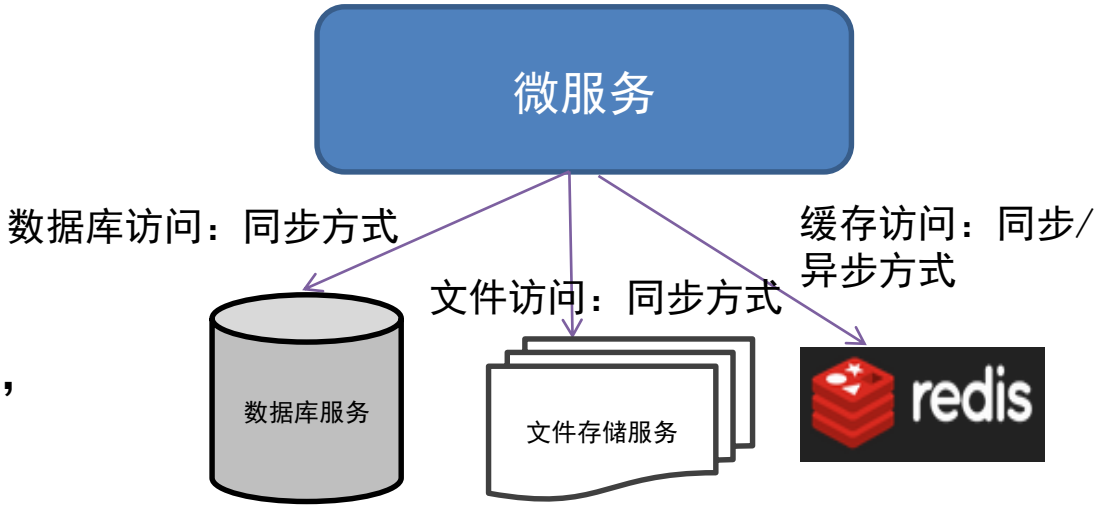


无处不在的故障-第三方服务依赖

中生代技术 FRESHMAN TECHNOLOGY IT大咖说

微服务通常会依赖第三方服务，包括数据库服务、文件存储服务、缓存服务、消息队列服务等，这种第三方依赖同时也引入了潜在的故障：

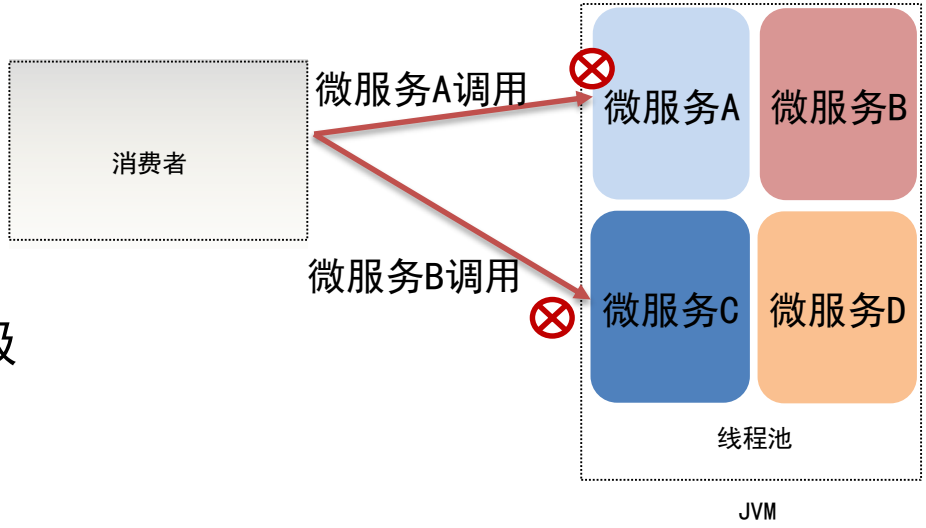
- ✓ 网络通信类故障
- ✓ “雪崩效用”导致的级联故障，例如服务端处理慢导致客户端线程被阻塞
- ✓ 第三方不可用导致微服务处理失败



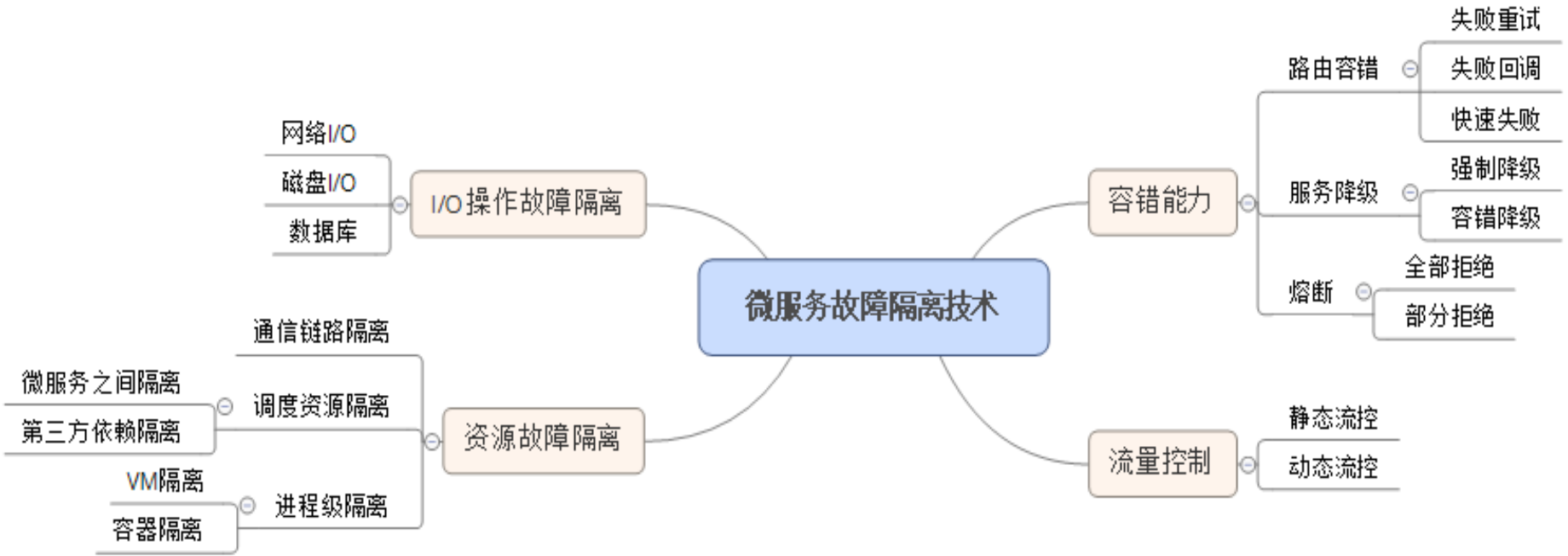
iTech+ 中生代技术 FRESHMAN TECHNOLOGY

无处不在的故障-多个微服务进程内合设导致故障扩散

- ✓ 处理较慢的微服务会阻塞其它微服务
- ✓ 某个微服务故障蔓延，可能导致整个进程不可用（OOM）
- ✓ 低优先级的微服务，抢占高优先级微服务的资源



服务故障隔离技术全景图



故障隔离技术：同步I/O到非阻塞I/O

中生代技术 FRESHMAN TECHNOLOGY IT大咖说

同步I/O主要弊端：

✓ I/O操作同步阻塞，

受制于网络和第三方处理速度

✓ I/O线程效率低，容易发生

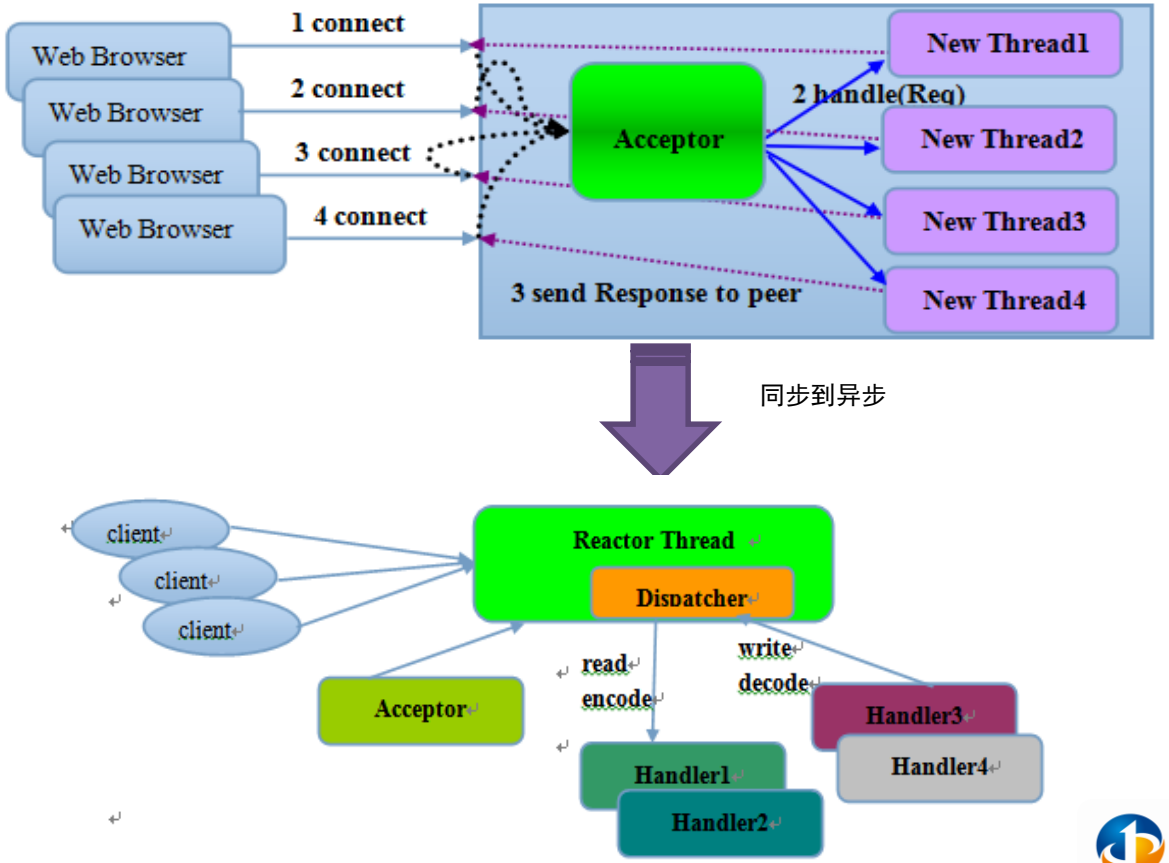
线程数量膨胀、通信队列积压

等问题

优化策略：

✓ TCP私有协议：建议直接基于Netty开发

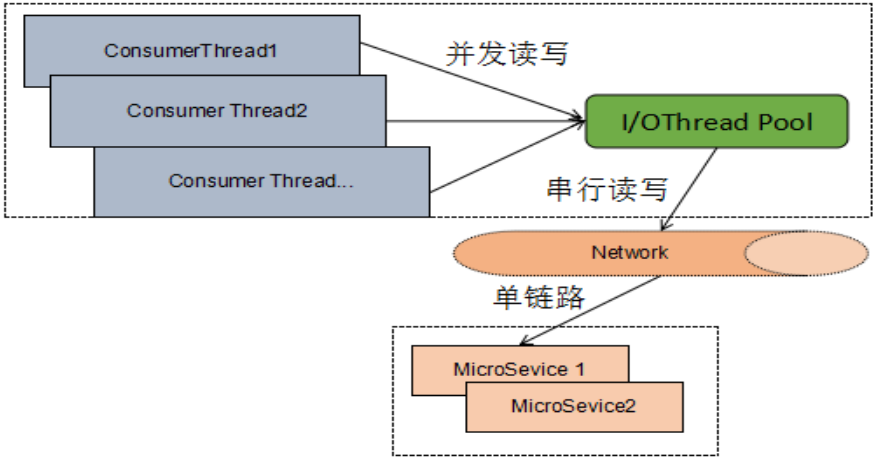
✓ HTTP/Restful/SOAP等：选择支持非阻塞I/O的Web框架。可以选择基于Netty构建的开源应用层协议栈框架



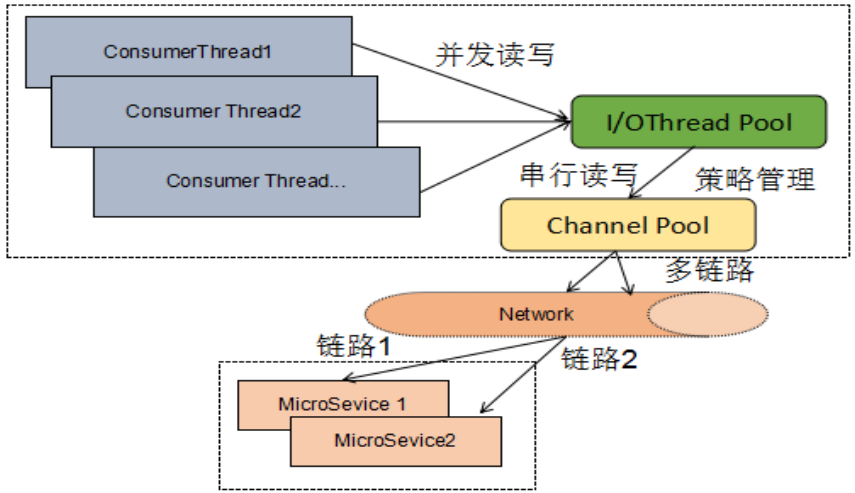
iTech+ 中生代技术 FRESHMAN TECHNOLOGY

故障隔离技术：RPC通信链路隔离

中生代技术 FRESHMAN TECHNOLOGY IT大咖说



单链路到多链路



隔离策略：

- ✓ 微服务节点之间支持配置多链路
- ✓ 微服务链路支持不同的隔离策略：

例如根据消息码流大小、根据微服务的优先级等策略，实现链路级的隔离

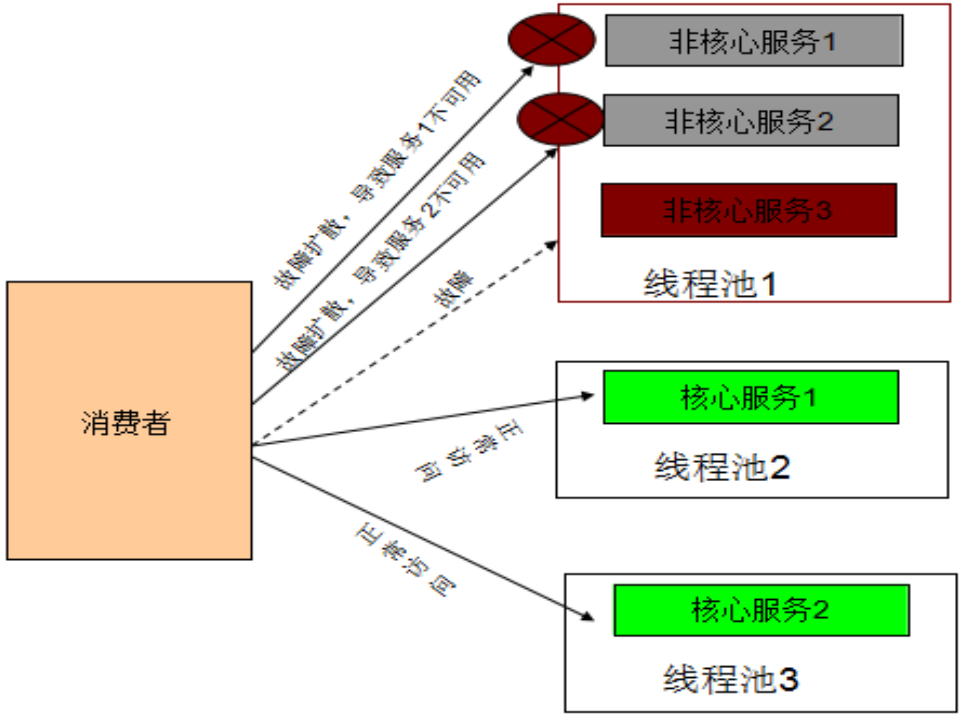
iTech+ 中生代技术 FRESHMAN TECHNOLOGY

故障隔离技术：微服务调度隔离

中生代技术 FRESHMAN TECHNOLOGY IT大咖说

关键技术点：

- ✓ 微服务发布时支持指定线程池/线程组
- ✓ 微服务线程池支持独享和共享两种模式
- ✓ 微服务和线程池监控，识别故障微服务，动态调整到故障隔离线程池中
- ✓ 支持按照微服务优先级调度微服务，即微服务线程支持微服务优先级调度



iTech+ 中生代技术 FRESHMAN TECHNOLOGY

故障隔离技术：第三方服务依赖隔离

中生代技术 FRESHMAN TECHNOLOGY | IT大咖说 本止于技术

关键技术点：

- ✓ 第三方依赖隔离可以采用线程池 + 响应式编程（例如RxJava）的方式实现
- ✓ 对第三方依赖进行分类，每种依赖对应一个独立的线程/线程池
- ✓ 微服务不直接调用第三方依赖的API，而是使用异步封装之后的API接口
- ✓ 异步调用第三方依赖API之后，获取Future对象。利用响应式编程框架，
- ✓ 可以订阅后续的事件，接收响应，针对响应进行编程

iTech+ 中生代技术 FRESHMAN TECHNOLOGY

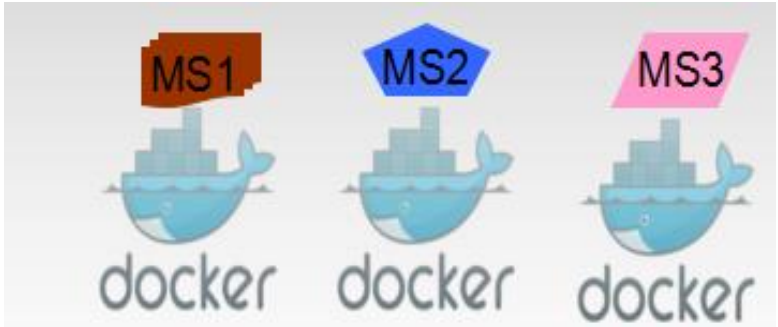
故障隔离技术：微服务进程隔离（Docker容器）

中生代技术 FRESHMAN TECHNOLOGY IT大咖说 本止于技术

关键技术点：

✓ 微服务独立开发、打包和部署

✓ 基于Docker部署微服务，可以实现细粒度的资源隔离，实现微服务的高密度部署。



优势：

✓ 高效：微服务的启动和销毁速度非常快，可以实现秒级弹性伸缩

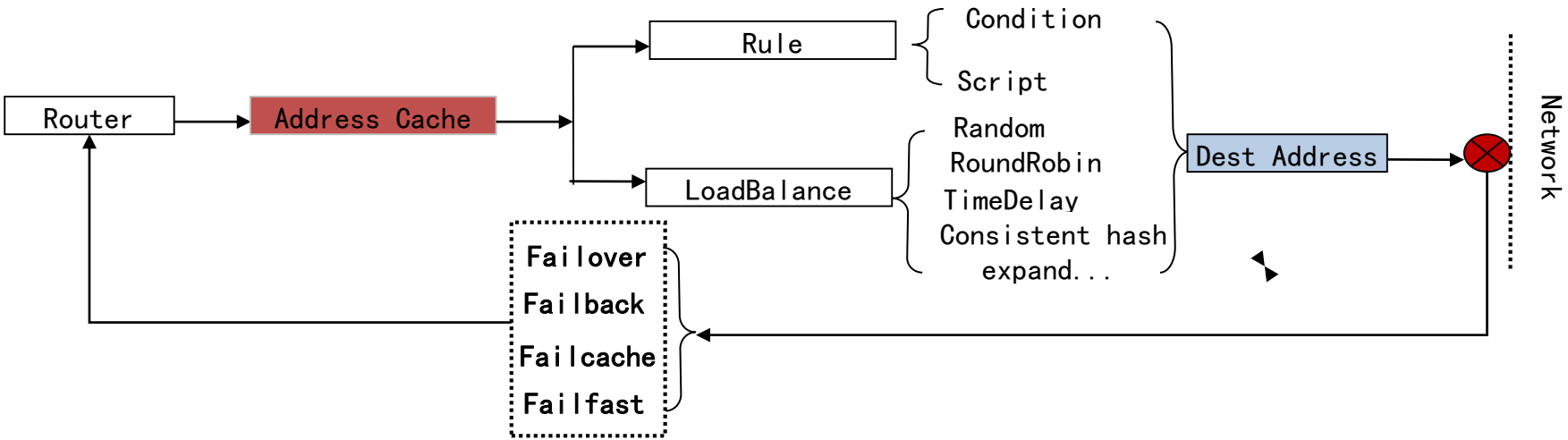
✓ 高性能：Docker容器的性能接近裸的物理机，综合性能损耗 < 5%

✓ 可移植性：“一次编写，到处运行”

iTech+ 中生代技术 FRESHMAN TECHNOLOGY

故障隔离技术：分布式路由容错

中生代技术 FRESHMAN TECHNOLOGY IT大咖说



- ✓ **失败自动重试：**微服务调用失败自动重试
- ✓ **失败自动切换：**当发生服务调用异常时，重新选路，查找下一个可用的微服务提供者
- ✓ **快速失败：**对于一些非核心的服务，希望只调用一次，失败也不再重试
- ✓ **失败回调：**提供异常回调接口，执行微服务消费者自定义的失败处理逻辑

iTech+ 中生代技术 FRESHMAN TECHNOLOGY

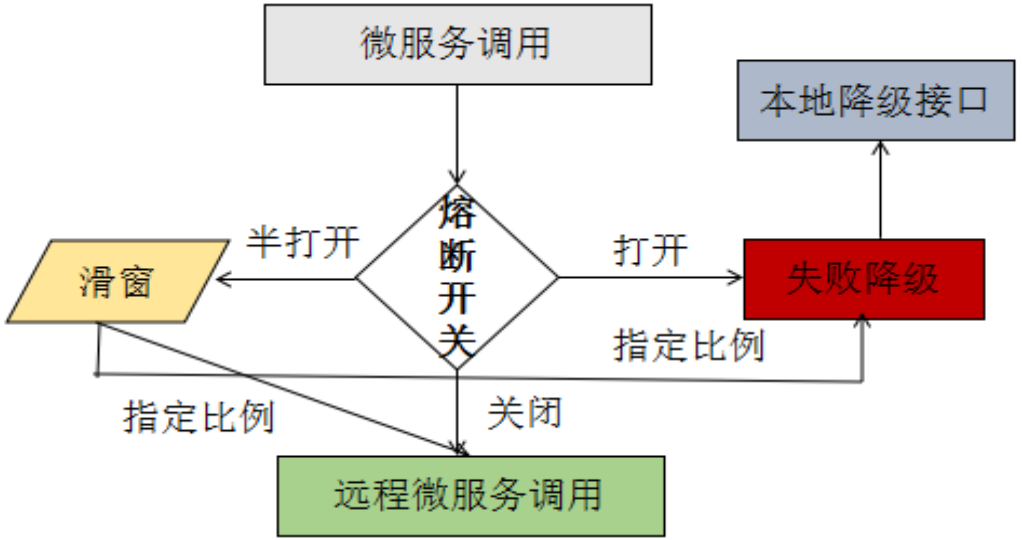
故障隔离技术：服务降级

- ✓ **强制降级：**不发起远程服务调用，执行本地降级策略，例如本地Mock方法
- ✓ **容错降级：**当非核心服务不可用时，可以对故障服务做业务逻辑放通，以保障核心服务的运行，降级策略包括异常转换、本地放通方法调用

故障隔离技术：熔断机制

中生代技术 FRESHMAN TECHNOLOGY IT大咖说

- ✓ **熔断判断：**微服务调用时，对熔断开关状态进行判断，当熔断器开关关闭时，请求被允许通过熔断器
- ✓ **熔断执行：**当熔断器开关打开时，微服务调用请求被禁止通过，执行失败回调接口
- ✓ **自动恢复：**熔断之后，周期T之后允许一条消息通过，如果成功，则取消熔断状态，否则继续处于熔断状态



iTech+ 中生代技术 FRESHMAN TECHNOLOGY

故障隔离技术：微服务健康度模型

✓ 熔断器开关的状态取决于微服务的运行质量，微服务的运行质量通常由多种因素决定，具有多个衡量因子。通过对微服务健康度建模，可以实现对微服务运行质量的360°实时评估

故障隔离技术：动态流控

中生代技术 FRESHMAN TECHNOLOGY IT大咖说 本止于技术

- 动态流控的最终目标是为了保命，并不是对流量或者访问速度做精确控制，通常需要对负载进行判断：
  - ✓ CPU使用率
  - ✓ 内存使用率
  - ✓ 内部消息队列积压率
- 动态流控是分级别的，不同级别拒掉的消息比例不同

iTech+ 中生代技术 FRESHMAN TECHNOLOGY

故障隔离技术：静态流控

中生代技术 FRESHMAN TECHNOLOGY IT大咖说

- ✓ 精确控制微服务的调用速率
- ✓ 平均主义不可行：由于微服务具备弹性伸缩、动态上线和下线等特性，因此集群中某个微服务实例的节点个数是动态变化的，采用传统的平均分配制无法做到精准的控制
- ✓ 动态配额申请制：根据微服务节点数和静态流控QPS阈值，拿出一定比例的配额做初始分配，剩余的配额放在配额资源池中。哪个微服务节点使用完了配额，就主动向服务注册中心申请配额

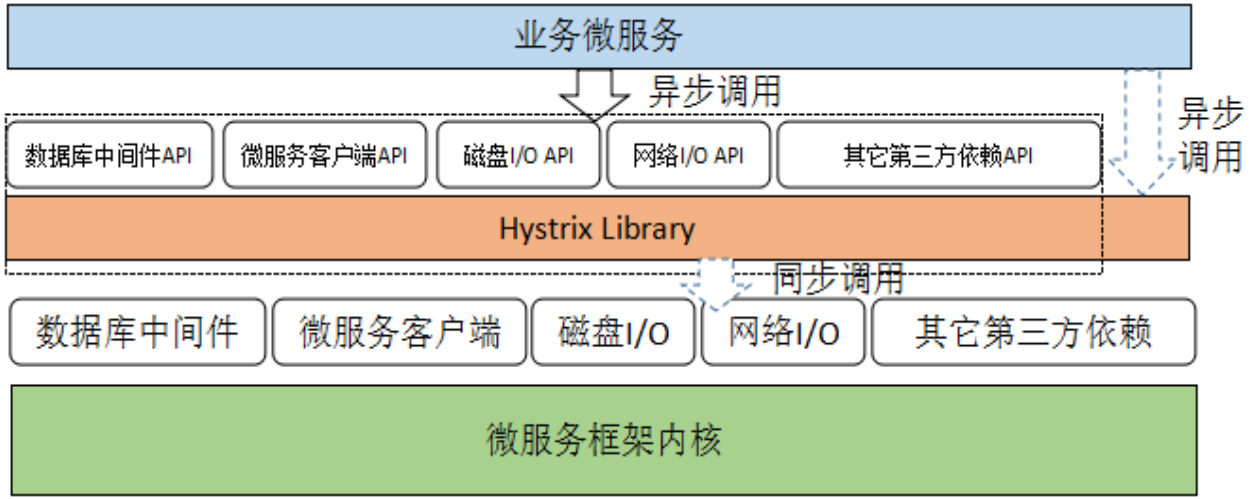
iTech+ 中生代技术 FRESHMAN TECHNOLOGY

故障隔离技术：集成业界成熟技术（Hystrix）

中生代技术 FRESHMAN TECHNOLOGY IT大咖说

可重用能力：

- ✓ 依赖隔离
- ✓ 熔断器
- ✓ 优雅降级
- ✓ Reactive编程
- ✓ 信号量隔离



集成策略：

- ✓ **第三方依赖隔离**：使用HystrixCommand做一层异步封装，实现业务的微服务调用线程和第三方依赖的线程隔离
- ✓ **依赖分类管理**：对第三方依赖进行分类、分组管理，根据依赖的特点设置熔断策略、优雅降级策略、超时策略等，以实现差异化的处理

iTech+ 中生代技术 FRESHMAN TECHNOLOGY

Thank you for watching