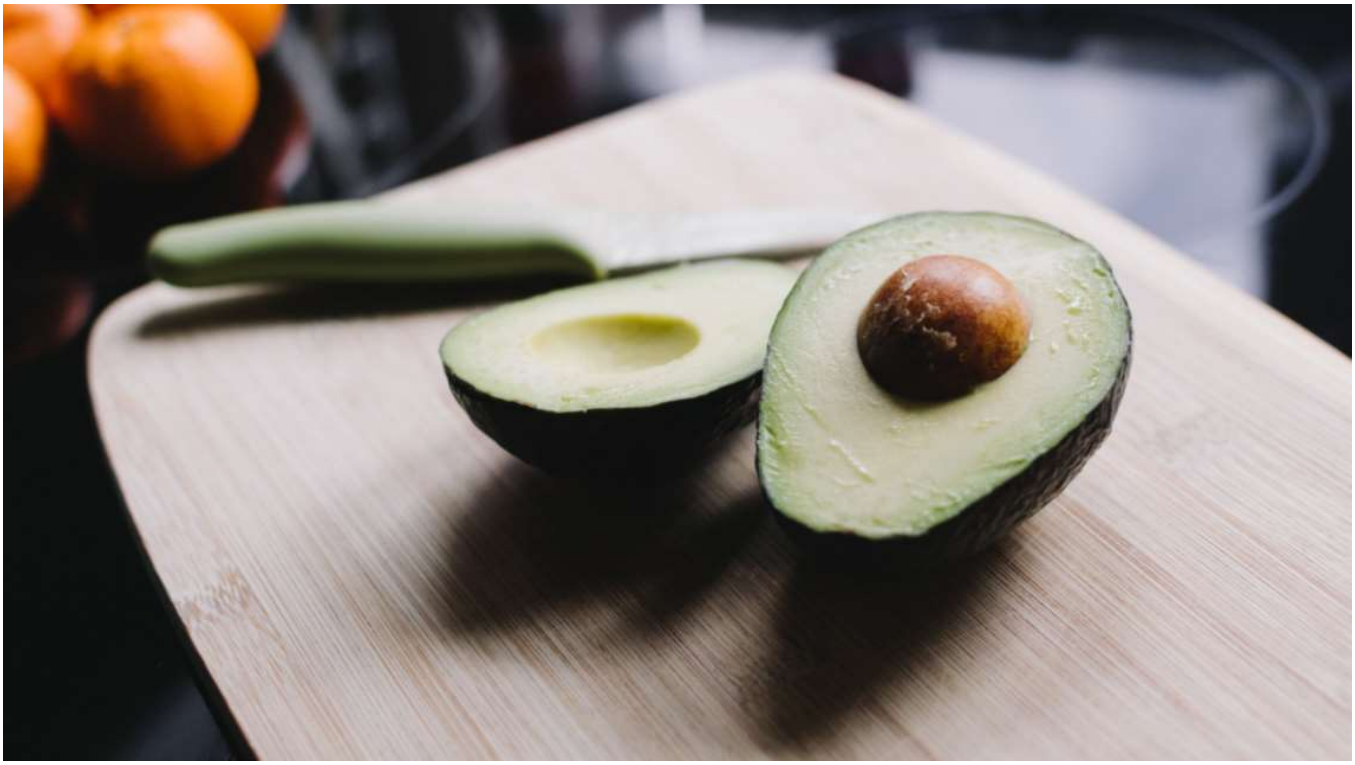
02 | 从单体应用走向服务化

2018-08-25 胡忠想



02 | 从单体应用走向服务化
朗读人：胡忠想　07′55″ | 3.64M

专栏上一期，我给你讲述了什么是微服务，以及微服务架构的由来。简单回顾一下，微服务就是将庞杂臃肿的单体应用拆分成细粒度的服务，独立部署，并交给各个中小团队来负责开发、测试、上线和运维整个生命周期。

那么到底什么时候应该拆分单体应用？拆分单体应用有哪些标准可依呢？

为了解答这两个问题，今天我将通过具体案例来阐述，希望你能够学会单体应用拆分成微服务的正确姿势。

什么时候进行服务化拆分？

从我所经历过的多个项目来看，项目第一阶段的主要目标是快速开发和验证想法，证明产品思路是否可行。这个阶段功能设计一般不会太复杂，开发采取快速迭代的方式，架构也不适合过度设计。所以将所有功能打包部署在一起，集中地进行开发、测试和运维，对于项目起步阶段，是最高效也是最节省成本的方式。当可行性验证通过，功能进一步迭代，就可以加入越来越多的新特性。

比如做一个社交 App，初期为了快速上线，验证可行性，可以只开发首页信息流、评论等基本功能。产品上线后，经过一段时间的运营，用户开始逐步增多，可行性验证通过，下一阶段就需要进一步增加更多的新特性来吸引更多的目标用户，比如再给这个社交 App 添加个人主页显示、消息通知等功能。

一般情况下，这个时候就需要大规模地扩张开发人员，以支撑多个功能的开发。如果这个时候继续采用单体应用架构，多个功能模块混杂在一起开发、测试和部署的话，就会导致不同功能之间相互影响，一次打包部署需要所有的功能都测试 OK 才能上线。

不仅如此，多个功能模块混部在一起，对线上服务的稳定性也是个巨大的挑战。比如 A 开发的一个功能由于代码编写考虑不够全面，上线后产生了内存泄漏，运行一段时间后进程异常退出，那么部署在这个服务池中的所有功能都不可访问。一个经典的案例就是，曾经有一个视频 App，因为短时间内某个付费视频访问量巨大，超过了服务器的承载能力，造成了这个视频无法访问。不幸的是，这个网站付费视频和免费视频的服务部署在一起，也波及了免费视频，几乎全站崩溃。

根据我的实际项目经验，一旦单体应用同时进行开发的人员超过 10 人，就会遇到上面的问题，这个时候就该考虑进行服务化拆分了。

服务化拆分的两种姿势

那么服务化拆分具体该如何实施呢？一个最有效的手段就是将不同的功能模块服务化，独立部署和运维。以前面提到的社交 App 为例，你可以认为首页信息流是一个服务，评论是一个服务，消息通知是一个服务，个人主页也是一个服务。

这种服务化拆分方式是**纵向拆分**，是从业务维度进行拆分。标准是按照业务的关联程度来决定，关联比较密切的业务适合拆分为一个微服务，而功能相对比较独立的业务适合单独拆分为一个微服务。

还有一种服务化拆分方式是**横向拆分**，是从公共且独立功能维度拆分。标准是按照是否有公共的被多个其他服务调用，且依赖的资源独立不与其他业务耦合。

继续以前面提到的社交 App 举例，无论是首页信息流、评论、消息箱还是个人主页，都需要显示用户的昵称。假如用户的昵称功能有产品需求的变更，你需要上线几乎所有的服务，这个成本就有点高了。显而易见，如果我把用户的昵称功能单独部署成一个独立的服务，那么有什么变更我只需要上线这个服务即可，其他服务不受影响，开发和上线成本就大大降低了。

服务化拆分的前置条件

一般情况下，业务系统引入新技术就必然会带来架构的复杂度提升，在具体决策前，你先要认识到新架构会带来哪些新的问题，这些问题你和你的团队是否能够解决？如何解决？是自己投入人力建设，还是采用业界开源方案？

下面几个问题，是从单体应用迁移到微服务架构时必将面临也必须解决的。

- 服务如何定义。对于单体应用来说，不同功能模块之前相互交互时，通常是以类库的方式来提供各个模块的功能。对于微服务来说，每个服务都运行在各自的进程之中，应该以何种形式向外界传达自己的信息呢？答案就是接口，无论采用哪种通讯协议，是 HTTP 还是 RPC，服务之间的调用都通过接口描述来约定，约定内容包括接口名、接口参数以及接口返回值。

- 服务如何发布和订阅。单体应用由于部署在同一个 WAR 包里，接口之间的调用属于进程内的调用。而拆分为微服务独立部署后，服务提供者该如何对外暴露自己的地址，服务调用者该如何查询所需要调用的服务的地址呢？这个时候你就需要一个类似登记处的地方，能够记录每个服务提供者的地址以供服务调用者查询，在微服务架构里，这个地方就是注册中心。

- 服务如何监控。通常对于一个服务，我们最关心的是 QPS（调用量）、AvgTime（平均耗时）以及 P999（99.9% 的请求性能在多少毫秒以内）这些指标。这时候你就需要一种通用的监控方案，能够覆盖业务埋点、数据收集、数据处理，最后到数据展示的全链路功能。

- 服务如何治理。可以想象，拆分为微服务架构后，服务的数量变多了，依赖关系也变复杂了。比如一个服务的性能有问题时，依赖的服务都势必会受到影响。可以设定一个调用性能阈值，如果一段时间内一直超过这个值，那么依赖服务的调用可以直接返回，这就是熔断，也是服务治理最常用的手段之一。

- 故障如何定位。在单体应用拆分为微服务之后，一次用户调用可能依赖多个服务，每个服务又部署在不同的节点上，如果用户调用出现问题，你需要有一种解决方案能够将一次用户请求进行标记，并在多个依赖的服务系统中继续传递，以便串联所有路径，从而进行故障定位。

针对上述问题，你必须有可行的解决方案之后，才能进一步进行服务化拆分。专栏后面的文章，我会给你逐一讲解相应的解决方案。

总结

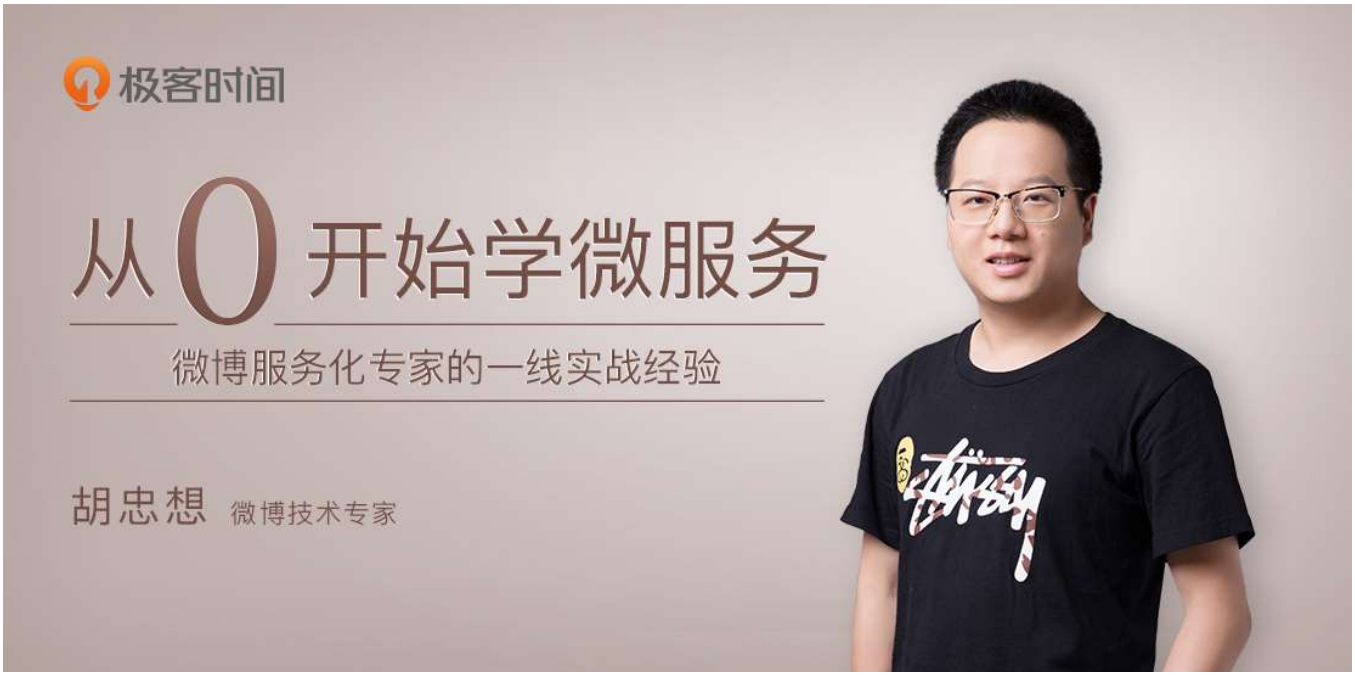
无论是纵向拆分还是横向拆分，都是将单体应用庞杂的功能进行拆分，抽离成单独的服务部署。

但并不是说功能拆分的越细越好，过度的拆分反而会让服务数量膨胀变得难以管理，因此找到符合自己业务现状和团队人员技术水平的拆分粒度才是可取的。我建议的标准是按照每个开发人员负责不超过 3 个大的服务为标准，毕竟每个人的精力是有限的，所以在拆分微服务时，可以按照开发人员的总人数来决定。

思考题

想想你现在的业务场景，如果是单体应用的话，是否需要进行服务化拆分？如果需要的话，你觉得纵向拆分还是横向拆分合适？具体可以拆分到什么粒度？

欢迎你在留言区写下自己的思考，与我一起讨论。



精选留言

常玉棋 3
我觉得目前单体能搞定的话就不要为了拆分而拆分，因为拆分后涉及到的问题有可能会让现有人员手忙脚乱，最好等做好了技术和人员准备再拆分。
2018-08-25

唐伯虎点蚊香 1
纵向拆分和横向拆分感觉还是不太理解
2018-08-25

科大大👏 1
当公司技术人员不足时就没必要考虑微服务了，好好做好单体架构，但是做的同时也要考虑未来有可能过渡到微服务，所以如果明确了以后会变成微服务的话单体架构有没有什么设计标准呢？
2018-08-25

why 0
十个人吗，但是四个人搞的微服务
2018-08-25

sunmeilin 0
我们现在也是单体应用，目前人员差不多只有我一个可以进行重构。我现在的思路是这样的，所有新开发的服务采用httpclient的形式调用用户的部分接口，这样的话慢慢的一块一块

的抽离，然后想在做完第一个服务之后采用springcloud+httpclient，大概10几个微服务，老师这样合理不？

2018-08-25

猿工匠 0

胡老师，早😁

2018-08-25

Stalary 0

想问一下，拆分后，我要调用同一个基础服务拿数据，那是不是调用方还要复制一份数据结构呢？还是直接操作json

2018-08-25

南山南的忧伤 0

阿忠伯，在设计架构时，是优先考虑当前所有的业务量，还是要满足以后潜在的业务量呢

2018-08-25