

## 02 为什么要使用微服务架构？

更新时间：2019-06-19 17:59:04



智慧，不是死的默念，而是生的沉思。

——斯宾诺莎

如今微服务架构越来越流行，被广泛应用在各行各业中，作为一线的开发人员其实也能深切感受到微服务热点，各大技术社区都会出现有关微服务架构的探讨。但究竟什么是微服务架构，使用微服务架构会给我们带来什么样的改变，本节专栏为大家揭开谜底。

说起微服务架构，我们先从它的发展由来、历史背景开始了解。

### 微服务诞生的历史背景

微服务的诞生并非偶然，它是在互联网高速发展、技术变化日新月异以及传统架构无法适应快速变化等多重因素共同推动下诞生的产物。

大约从2009年开始，Netflix 完全重新定义了它的应用程序开发和操作模型，拉开了微服务探索的第一步。直到2014年3月，Martin Fowler 写的一篇文章 《Microservices》 以更加通俗易懂的形式为大家定义了什么是微服务架构。

互联网时代的产品通常有两类特点：需求变化快和用户群体庞大。在这种情况下，如何从系统架构的角度出发，构建灵活、易扩展的系统，快速应对需求的变化；同时，随着用户的增加，如何保证系统的可伸缩性、高可用性，成为系统架构面临的新挑战。

如果还按照以前传统开发模式，开发一个大而全的系统已经很难满足市场对技术的需求。这时候分而治之的思想被提了出来，于是我们从单体架构发展到分布式架构，又从分布式架构发展到 SOA 架构，服务不断地被拆分和分解，粒度也越来越小，直到微服务架构的诞生。

当服务越来越小的时候，必然会面临其它很多问题，比如服务之间的调用更加频繁，服务的偶尔不可用变成常态。面对这些挑战，我们必须需要重新来构建一套技术来支持微服务架构。这套技术需要具备高可用、请求重试、熔断、容错、调用过程可监控等特性，当解决了这些问题后，微服务应运而生。

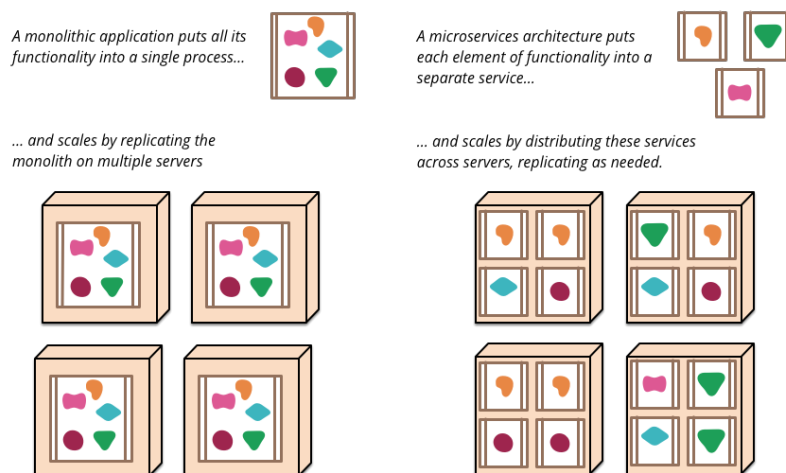
## 什么是微服务

简单来说，微服务架构是一种将一个单一应用程序开发为一组小型服务的方法。每个服务运行在其独立的进程中，服务与服务间采用轻量级的通信机制互相沟通（通常是基于 HTTP 的 RESTful API）。

每个服务都围绕着具体业务进行构建，并且能够被独立地部署到生产环境、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，对具体的一个服务而言，应根据业务上下文，选择合适的语言、工具对其进行构建。

微服务架构是一种架构模式，它提倡将单一应用程序划分成一组小的服务，服务之间互相协调、互相配合，为用户提供最终价值，一个大型复杂软件应用由一个或多个微服务组成。系统中的各个微服务可被独立部署，各个微服务之间是松耦合的。每个微服务仅专注于完成一件任务并很好地完成该任务。在所有情况下，每个任务代表着一个小的业务能力。

微服务架构是 SOA 架构的传承，但一个最本质的区别就在于微服务是真正分布式的、去中心化的。把所有的“思考”逻辑包括路由、消息解析等放在服务内部，去掉一个大一统的 ESB，服务间轻通信，是比 SOA 更彻底地拆分。微服务架构强调的重点是业务系统需要彻底的组件化和服务化，原有的单个业务系统会拆分为多个可以独立开发、设计、运行和运维的小应用，这些小应用之间通过服务完成交互和集成。



传统架构和微服务架构对比，图片来自文章《[Microservices](#)》。

## 微服务架构的优势

相比传统架构模式，微服务架构具有以下几方面的优势：

- 复杂度可控：在将应用分解的同时，规避了原本复杂度无止境的积累。每一个微服务专注于单一功能，并通过定义良好的接口清晰表述服务边界。由于体积小、复杂度低，每个微服务可由一个小规模开发团队完全掌控，易于保持高可维护性和开发效率；
- 独立部署：由于微服务具备独立的运行进程，所以每个微服务也可以独立部署。当某个微服务发生变更时，无需编译、部署整个应用。由微服务组成的应用相当于具备一系列可并行的发布流程，使得发布更加高效，同时降低对生产环境所造成的风险，最终缩短应用交付周期；
- 技术选型灵活：微服务架构下，技术选型是去中心化的。每个团队可以根据自身服务的需求和行业发展的现状，自由选择最适合的技术栈。由于每个微服务相对简单，故需要对技术栈进行升级时所面临的风险就较低，甚至完全重构一个微服务也是可行的；

- 容错：当某一组件发生故障时，在单一进程的传统架构下，故障很有可能在进程内扩散，形成应用全局性的不可用。在微服务架构下，故障会被隔离在单个服务中。若设计良好，其他服务可通过重试、平稳退化等机制实现应用层面的容错；
- 扩展：单块架构应用也可以实现横向扩展，就是将整个应用完整复制到不同的节点。当应用的不同组件在扩展需求上存在差异时，微服务架构便体现出其灵活性，因为每个服务可以根据实际需求独立进行扩展。

## 使用 Spring Cloud 开发微服务的优势

Spring Cloud 诞生时，正处于微服务概念在慢慢酝酿中。Spring Cloud 的研发融合了微服务架构的理念，实现了在 Java 领域内微服务架构落地的技术支撑，结合着 Spring Boot 相关特性，提供了基础组件一键式启动和部署能力，极大地简化了微服务架构的技术落地。

Spring Cloud 从设计之初就考虑了分布式架构演化所需的功能，如服务发现注册、配置中心、消息总线、负载均衡、断路器、数据监控等。这些功能都是以插拔的形式提供出来，方便我们在系统架构演进的过程中，可以合理选择需要的组件进行集成，从而在架构演进的过程中会更加平滑、顺利。

Spring Cloud 提供的功能，完美地解决了微服务实施过程中可能会遇到的困难。对比业内相关框架，在微服务领域 Spring Cloud 目前还没有真正的对手，从某种程度来讲 Spring Cloud 已经成为了微服务落地的技术标准。

微服务是为了解决项目快速变化、快速迭代的痛点，以及架构自然演进所诞生的一种架构解决方案。Spring Cloud 是微服务解决方案的技术落地。可以预见的是，未来微服务架构将会被越来越多的公司所采纳，同时 Spring Cloud 的应用范围也会越来越广泛，因此越早学习受益越大。

### 小结

微服务架构的诞生有着历史的必然性。随着互联网的不断迅速发展，原有的系统架构不断在接受着新的挑战，并发数超过几万、项目参与人数超过千人的项目也越来越多。如果在大规模系统协作下，如何保持系统的稳定性、可快速迭代性就变得非常紧迫。微服务架构便是在这样的历史背景下诞生，提供了一套大型项目开发的解决方案。

随着越来越多的公司采用微服务架构，可以预见的是，在未来微服务架构必将会成为 IT 行业最流行的系统架构。

参考：

<http://martinfowler.com/articles/microservices.html>

本文作者：纯洁的微笑、江南一点雨