# 单块架构及其面临的挑战

系统中常见的三个层次：

**表示层:**聚焦数据显示和用户交互

**业务逻辑层:**聚焦业务逻辑处理

**数据访问层:**聚焦数据的存储于访问

**优势:**三层架构的出现，解决了系统间调用复制、制作不清的问题，也有效率低了层与层之间的依赖关系，成为软件架构的经典模式之一

虽然三层架构将系统在逻辑上分成了三层，但它并不是物理上的分层。对于这种功能集中、代码中心化、一个发布包、部署后运行在同一进程的应用程序，我们通常称之为单块架构应用

**不足:**随着业务的不断扩大，需求功能的持续增加，单块架构已经很难满足业务快速变化的需要，一方面，代码的可维护性、扩展性、灵活性在降低；；另一方面，系统的修改成本、构建以及维护成本在显著增加

**主要优势:**

易于开发

易于测试

易于部署

易于水平伸缩

**不足：**

代码量变大以后，后期维护困难。人员协调问题，效率问题显著

**互联网时代的产品特点:**

创新成本低、需求变化快、用户群体庞大

# 微服务架构

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，服务之间相互协调、互相配合，为用户提供最终价值。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成环境、类生成环境等。另外，尽量避免统一的、集中式的服务管理机制，对具体的应用服务而言，应根据业务上下文，选择合适的语言、工具对其进行构建

**静态语言:**

**优点**:在于其结果规范，存在编译期的语法检查、便于调试、类型安全性高，通常其继承关系简洁明了，IDE对其支持也更加友好;

**缺点:**需要编写更多的类型相关代码。因此要实现相同的功能代码量相对稍多，这类语言的典型代表有Java、C++等

**动态语言:**

**优点:**灵活性较高，运行时可以改变内存结构，无类型检查，无须写较多的类型相关的代码

**缺点:**不方便调试，无编译期检查。因此对个体的能力要求很高。尤其是项目复杂度高或者代码量较大的项目，如果没有足够搞的测试覆盖率，维护起来更是举步维艰。典型代码如JavaScript、Ruby或者Python等

**SOA:**对于复杂的企业IT系统，应按照不同的、可重用的粒度划分，将功能相关的一组功能提供者组织在一起为消费者提供服务

**SOA与微服务的区别**

|  |  |
| --- | --- |
| **SOA实现** | **微服务架构实现** |
| 企业级，自顶向下开展实施 | 团队级，自底向上开展实施 |
| 服务由多个子系统组成，粒度大 | 一个系统被拆分成多个服务，粒度细 |
| 企业服务总线，集中式的服务架构 | 无集中式总线，松散的服务架构 |
| 集成方式复杂(ESB/WS/SOAP) | 基础方式简单(HTTP/REEST/JSON) |
| 单块架构系统，相互依赖，部署复杂 | 服务能独立部署 |

相比传统SOA的服务实现方式，微服务更具灵活性、可实施性已经可扩展性，其强调的是一种独立厕所、独立部署、独立运行的软件架构模式



**微服务的本质：**

服务作为组件

围绕业务组织团队

关注产品而非项目

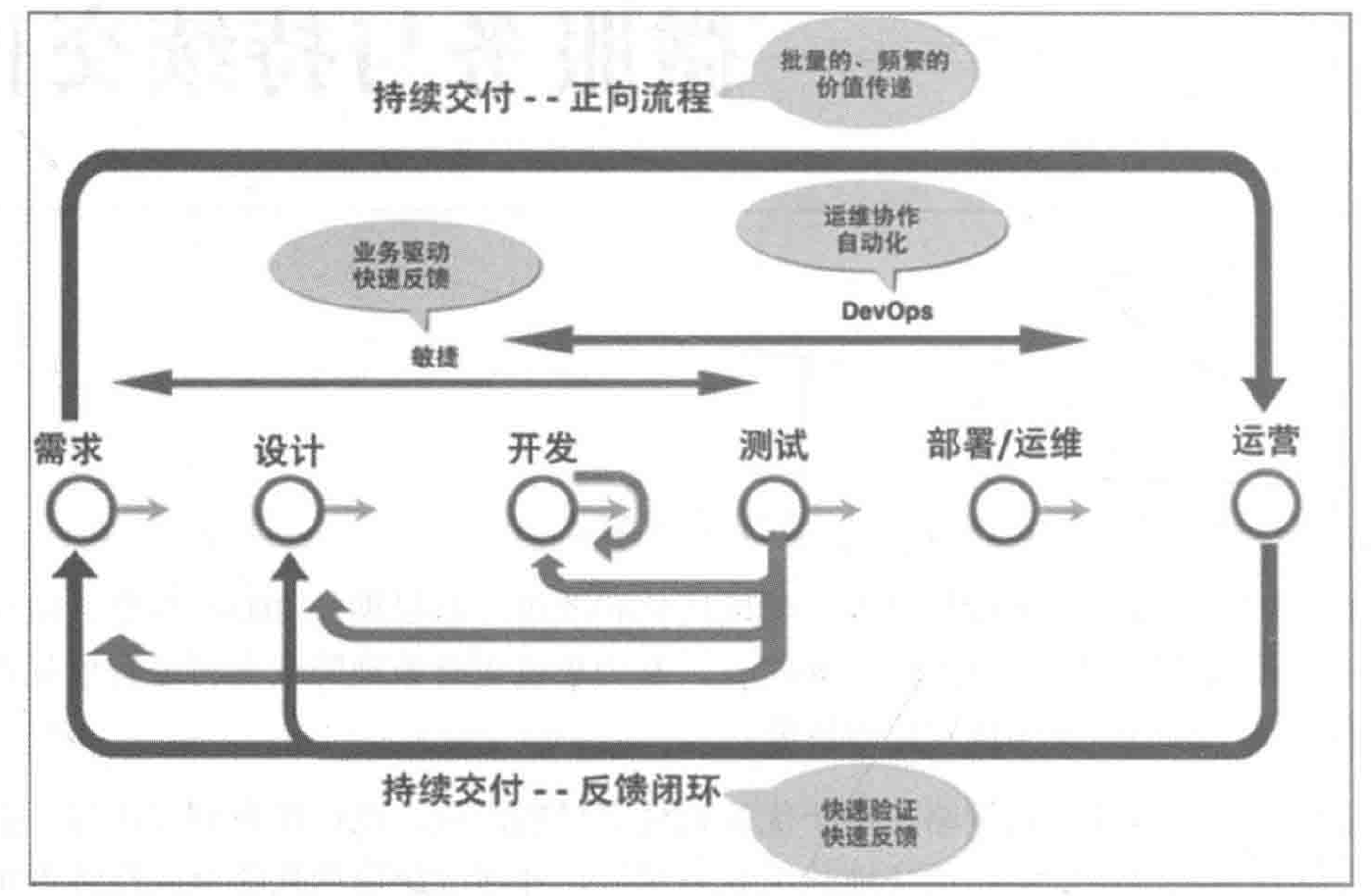
技术多样性

业务数据独立

基础设施自动化

演进式架构

**持续交付:**小、频、快



随着领域驱动设计、持续交付、按需虚拟化、基础设备自动化、小型自治团队、大型集群系统这些实践的流行，微服务也应运而生。它并不是被发明出来的，而是从现实世界中总结出来的一种趋势或模式

微服务就是一些协同工作的小而自治的服务

1. 很小，专注于做好一件事

内聚性：把因相同原因而变化的东西聚合到一起，而把因不同原因变化的东西分离开来

1. 自治性

一个微服务就是一个独立的实体。它可以独立地部署在PAAS上，也可以作为一个操作系统进程存在。我们要尽量避免把多个服务部署到同一台机器上，尽管现如今集群的概念已经非常模糊了

服务之间均通过网络调用进行通信，从而加强了服务之间的隔离性，避免紧耦合

部署微服务的好处

1. 技术异构性.

如果系统中的一部分需要做性能提升，可以使用性能更好的技术栈重新构建该部分

1. 弹性

如果系统中的一个组件不可用了，但并没有导致级联故障，那么系统的其他部分还可以正常运行

1. 扩展

庞大的单块服务只能作为一个整体进行扩展

1. 简化部署

在有几百万行代码的单块应用程序中，即使只修改一行代码，也需要重新部署整个应用程序才能够发布该变更，部署的效率就会变得很低

1. 与组织结构相匹配

微服务架构可以很好地将架构与组织结构相匹配，避免出现过大的代码块，从而获得理想的团队大小及生产力。服务的所有权也可以在团队之间迁移，从而避免异地团队的出现

1. 可组合性

分布式系统和面向服务架构声称的主要好处是易于重用已有功能。而在微服务架构中，根据不同的目的，人们可以通过不同的方式使用同一个功能

1. 对可替代性的优化

如果你在一个大中型组织工作，和可能接触过一些庞大而丑陋的遗留系统。这些系统无人敢碰，却对公司业务的运营至关重要

SOA是一种设计方法，其中包含多个服务，而服务之间通过配合最终会提供一系列的功能。一个服务通常一独立的形式存在于操作系统进程中。服务之间通过网络调用，而非采用进程内调用的方式进行通信

