2.3.1 微服务架构

微服务架构是一种架构概念，旨在通过将功能分解到各个离散的服务中以实现对解决方案的解耦。它的主要作用是将功能分解到离散的各个服务当中，从而降低系统的耦合性，并提供更加灵活的服务支持。围绕业务领域组件来创建微型应用，这些应用可独立地进行开发、管理和迭代。在分散的组件中使用云架构和平台式部署、管理和服务功能，使产品交付变得更加简单。

微服务架构与传统的单体应用架构相比开发效率更高，减少了开发中的相互等待和冲突；维护更简单，代码耦合度极大的降低使得微型应用更加简单；更加灵活，构建时间更短；稳定性增强，不会因为一个小问题导致整个系统崩溃；扩展性增强，可以根据需求，对特定的功能组件进行扩展以提高系统的并发性。但是因为微服务架构是分布式架构，因此存在分布式的管理和调用消耗。

根据微服务的官方定义，微服务是一系列独立的服务共同组成的系统。每个独立的服务单独部署，并且相互隔离运行在自己的进程中。系统中每个服务为独立的业务开发，在运行中采用分布式管理。微服务系统强调强服务个体和弱通信，其高度容错性使得其在自动化运维和敏捷开发中得到了广泛应用。

2.3.2 微服务解决方案

微服务架构的具体实践中需要解决四个问题，首先是客户端如何访问服务，对此一般采用在服务和UI之间设置名为API Gateway的代理机构，由该机构提供服务的统一入口，聚合后台服务，节省访问流量并提供过滤、流量控制等API管理功能；其次是服务间的通信方式，服务之间的调用分为同步调用和异步调用，同步调用一致性强，但性能较差，如REST API的方式基于HTTP实现，在调用的过程中会有大量的多余通信消耗。异步调用往往通过消息队列实现，此方式在分布式系统中应用广泛，它既能降低系统耦合，又能实现调用的缓冲，有效的防止了系统因流量过大而崩溃的情况，不过异步调用也会导致一致性的减弱。第三个问题是服务的实现，即服务如何管理，服务之间如何管理。一般有两类做法，一种是基于客户端的服务注册与发现如图2.5所示，另一种是基于服务端的服务注册与发现如图2.6所示。当服务上线时，服务提供者将自己的服务信息注册到 ZK（或类似框架），并通过心跳维持长链接，实时更新链接信息。服务调用者通过 ZK 寻址，根据可定制算法，找到一个服务，还可以将服务信息缓存在本地以提高性能。当服务下线时，ZK 会发通知给服务客户端。最后一个问题是当服务崩溃时如何处理，对于此问题一般有重试、限流、熔断、负载均衡等方式处理。其中设计合适的负载均衡算法不仅可以应对服务崩溃，可以提高系统的并发性能。