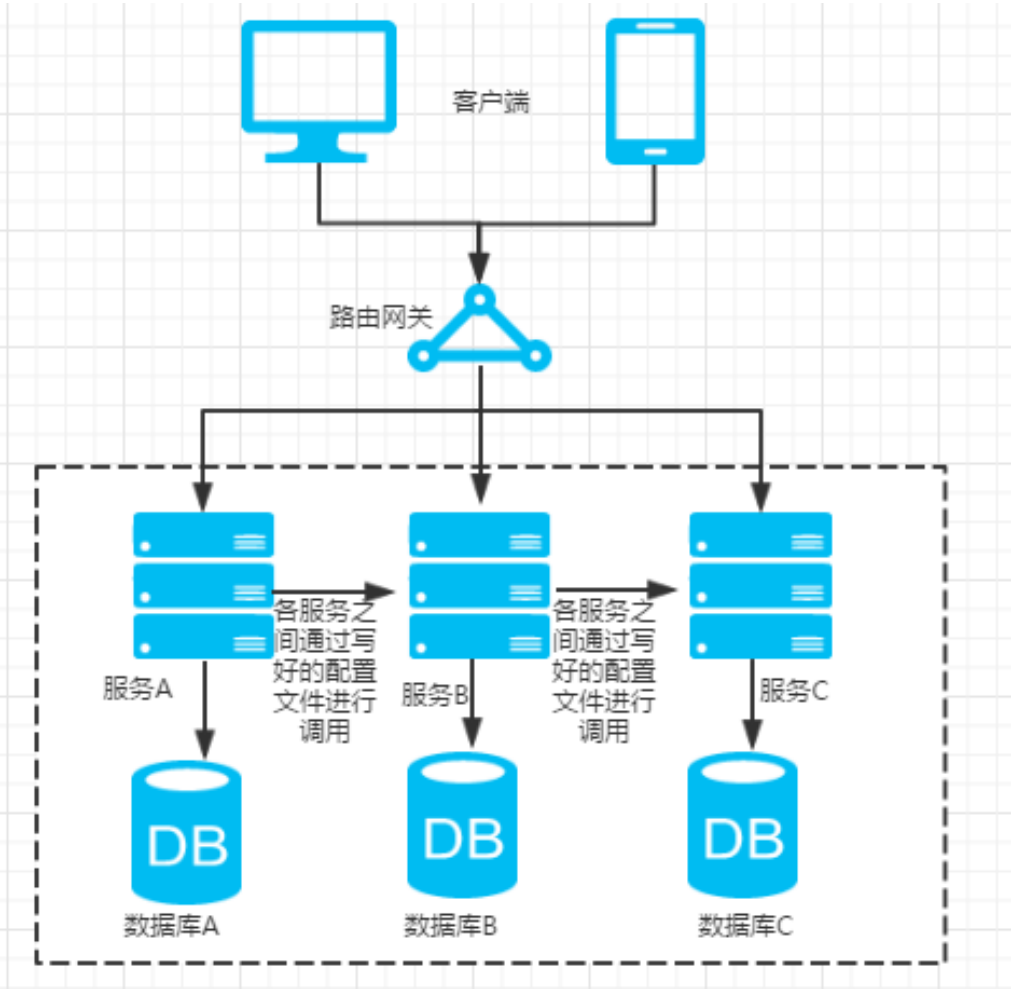
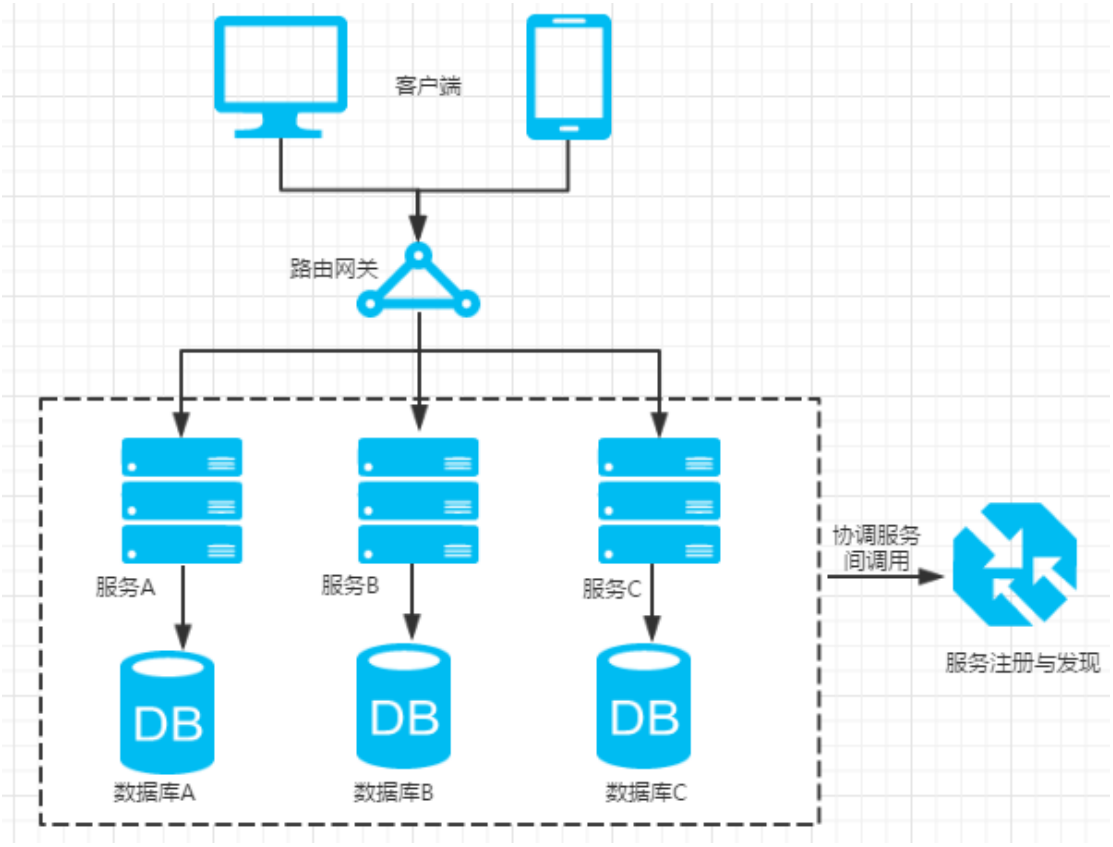
微服务中的服务注册与发现

传统的项目中，某个服务访问另一个服务，可以通过在配置文件中记录其他服务静态地址的形式进行访问，通常这个配置文件也很少更新，模式如下图：



而在微服务中，每个功能可能都是一个独立的服务，如果通过传统的方式配置每个应用，配置文件会变得很复杂多变，所以我们需要服务的注册与发现。



注册与发现的原理

服务的注册与发现是通过服务注册表实现的，应用端（每个服务）通过配置文件像服务注册表提交自己的注册信息，当服务启动时，服务注册表会检索到该应用，并将该应用的网络地址添加到表中，同样当服务终止，服务注册表会删除服务的地址。服务注册表是通过心跳机制实现的。当其他应用访问已注册的服务时，负载均衡会通过服务注册表，实现服务的发现。

本文将介绍一个服务注册与发现：Zookeeper。

什么是 Zookeeper

Zookeeper 是 Apache 公司为分布式应用设计的协调服务，起初是 Hadoop 的子项目，现在是一个独立项目。ZooKeeper 的目标就是封装好复杂易出错的关键服务，将简单易用的接口和

性能高效、功能稳定的系统提供给用户。

Zookeeper 功能强大，本文主要介绍 Zookeeper 在微服务架构中的应用。

Zookeeper 的特点

Zookeeper 作为服务注册与发现的解决方案，它有如下优点：

1. 它提供的简单 API
2. 支持多语言的客户端
3. 通过 Watcher 机制实现 Push 模型，服务注册信息的变更能够及时通知服务消费方。

Zookeeper 管理服务的方式依赖于之前提到的服务注册表；Zookeeper 相比于其他注册与发现解决方案，多了角色，Zookeeper 有三个角色：Leader，Follower，Observer。

Zookeeper 中的服务在启动的时候会根据之前的启动状况推选一个 Leader 节点（由于必须推选出来一个结果，所以 Zookeeper 集群的服务器数量通常是奇数），推选出 Leader 节点后，其他节点成为 Follower 节点。Leader 节点是 Zookeeper 集群工作的核心，负责进行选举投票的发起和决议，更新系统状态；Follower 节点是 Zookeeper 集群状态的跟随者，用于接收客户端的请求并向客户端返回结果，在选举过程中参与投票。

Observer 可以接受客户端连接，将写请求转发给 Leader 节点，但 Observer 不参加投票过程，只同步 Leader 节点的状态，Observer 的目的是为了扩展系统，提高读取速度。