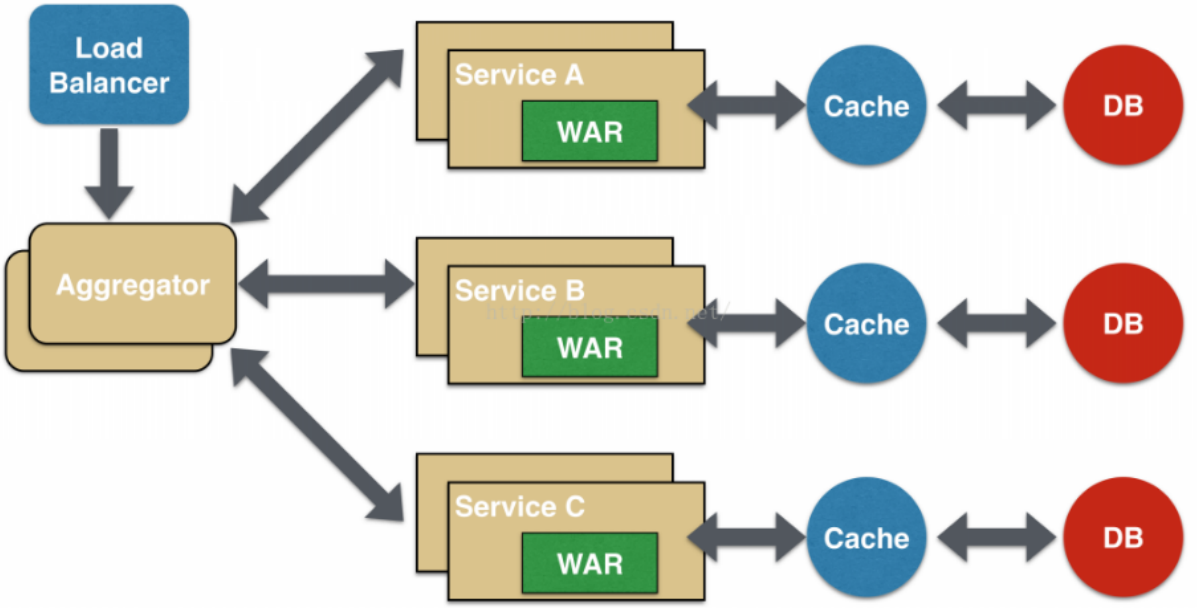
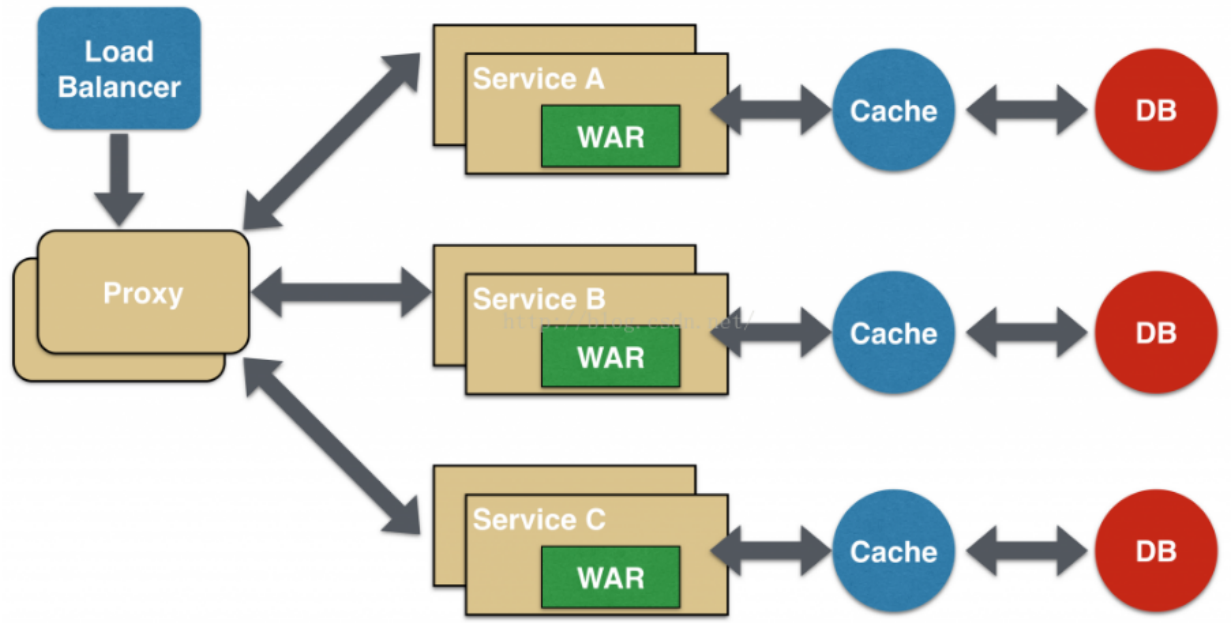
用Scale Cube方法设计应用架构，将应用服务按功能拆分成一组相互协作的服务。每个服务负责一组特定、相关的功能。每个服务可以有自己独立的数据库，从而保证与其他服务解耦。

1 聚合器微服务设计模式



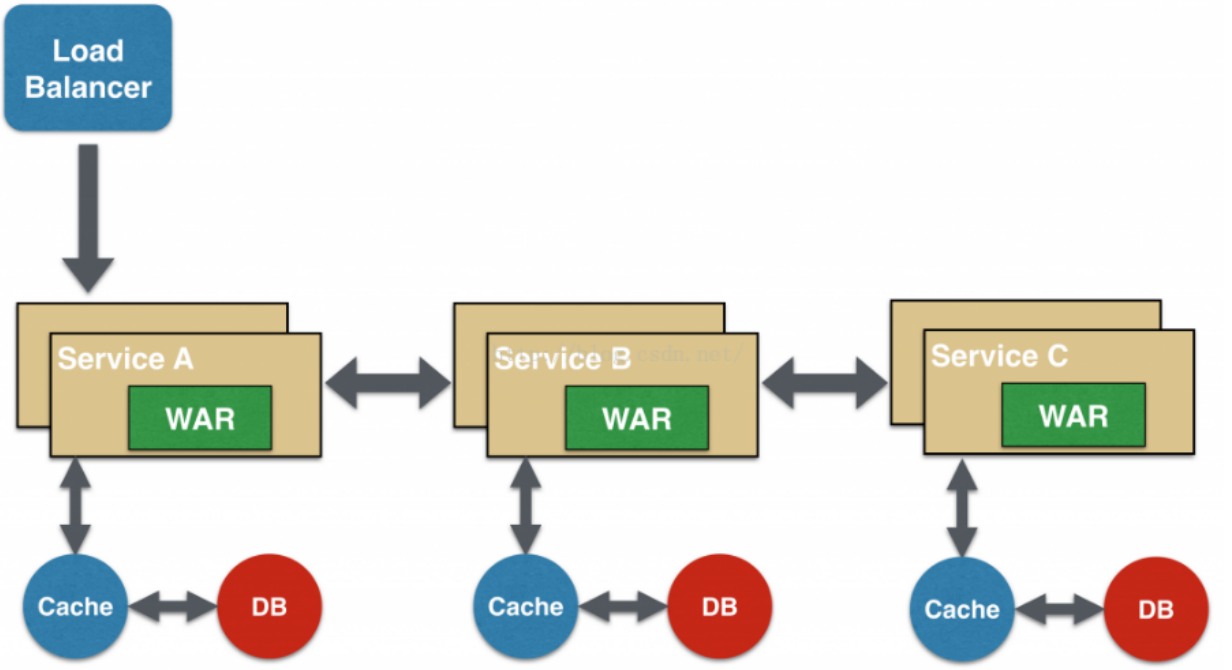
聚合器调用多个服务实现应用程序所需的功能。它可以是一个简单的Web页面，将检索到的数据进行处理展示。它也可以是一个更高层次的组合微服务，对检索到的数据增加业务逻辑后进一步发布成一个新的微服务，这符合DRY原则。另外，每个服务都有自己的缓存和数据库。如果聚合器是一个组合服务，那么它也有自己的缓存和数据库。聚合器可以沿X轴和Z轴独立扩展。

2 代理微服务设计模式



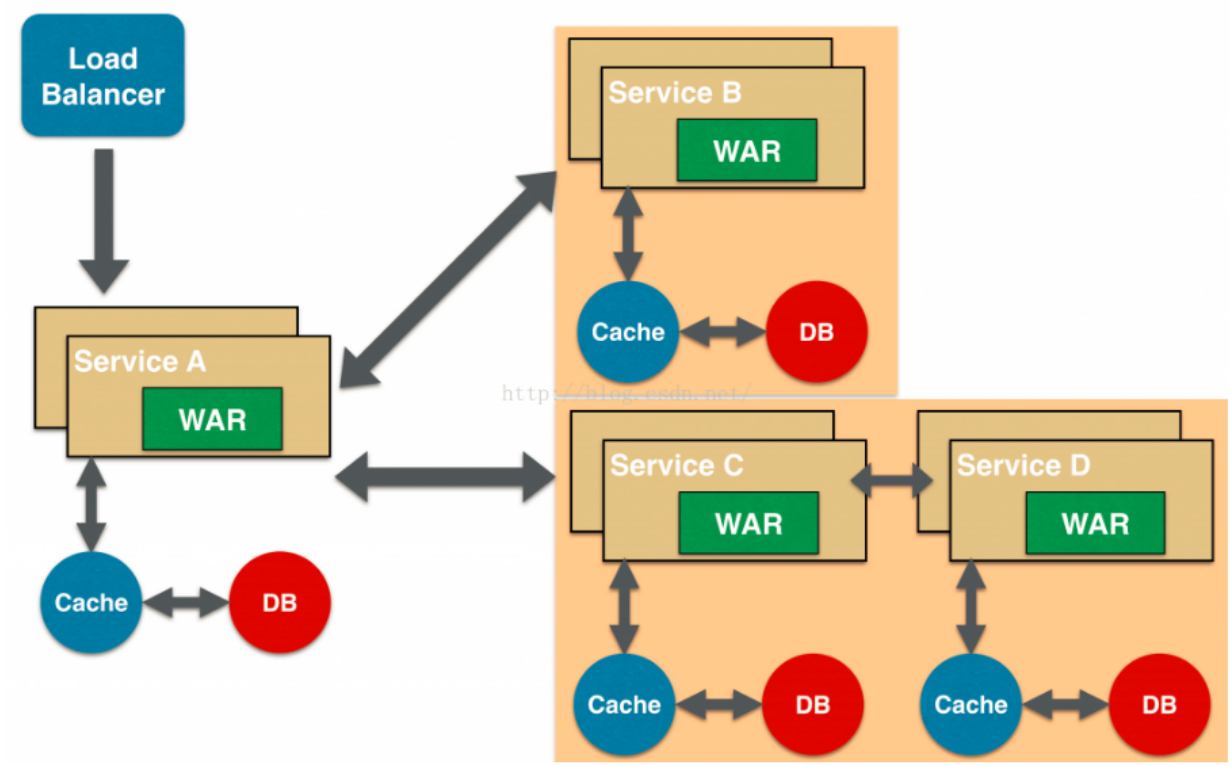
这是聚合器模式的一个变种，在这种情况下，客户端并不聚合数据，但会根据业务需求的差别调用不同的微服务。代理可以仅仅委派请求，也可以进行数据转换工作。

3 链式微服务设计模式



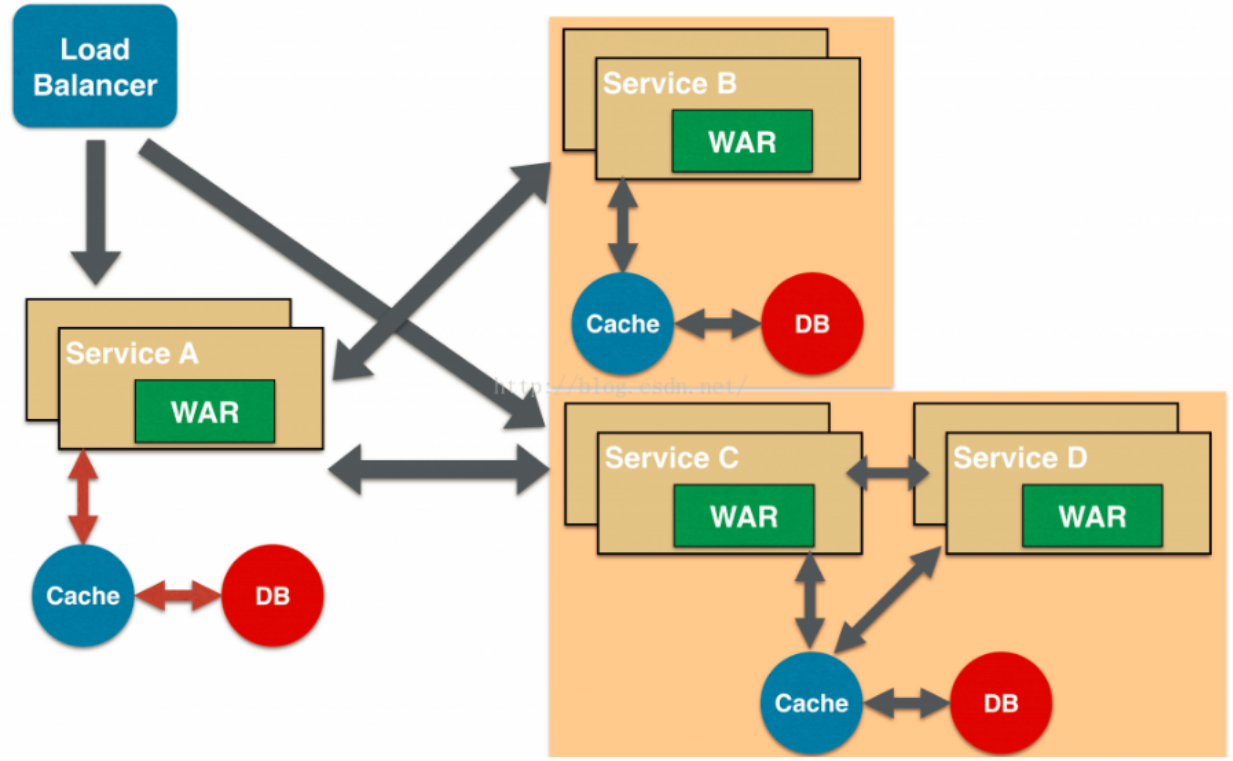
这种模式在接收到请求后会产生一个经过合并的响应，在这种情况下，服务A接收到请求后会与服务B进行通信，类似地，服务B会同服务C进行通信。所有服务都使用同步消息传递。在整个链式调用完成之前，客户端会一直阻塞。因此，服务调用链不宜过长，以免客户端长时间等待。

4 分支微服务设计模式



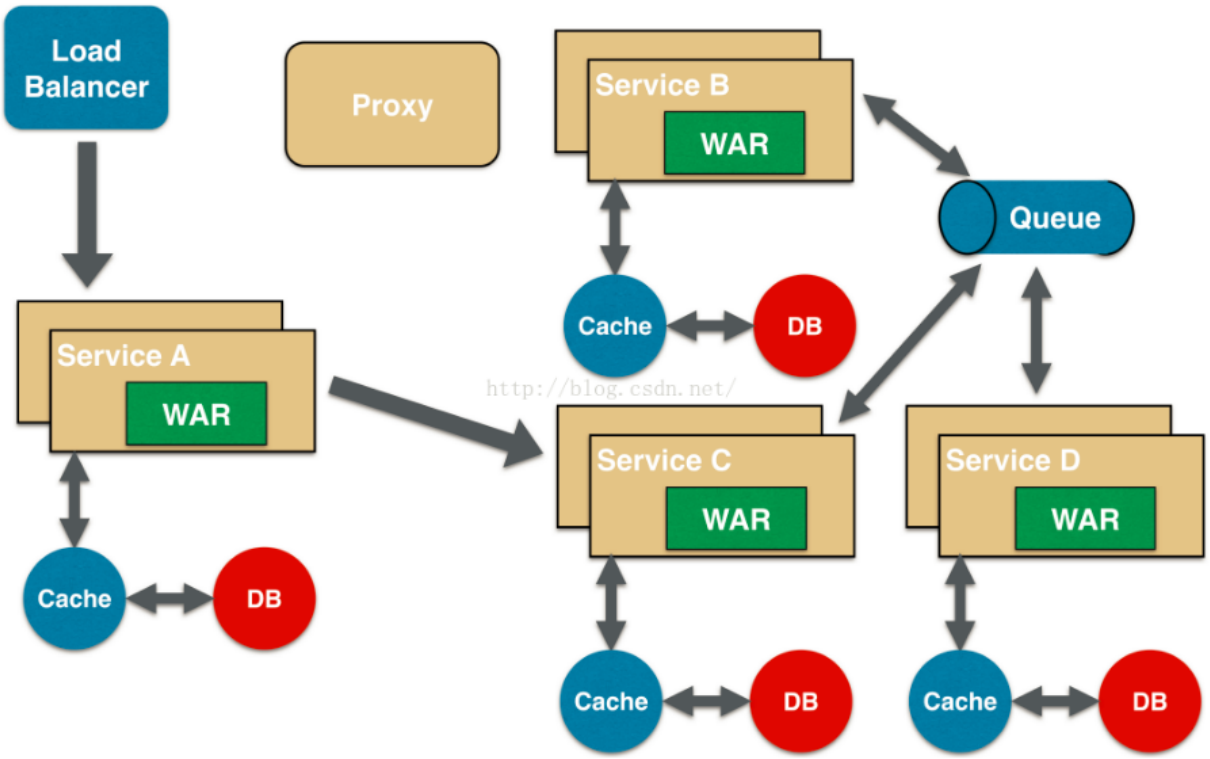
这种模式是聚合器模式的扩展，允许同时调用两个微服务链

5 数据共享微服务设计模式



自治是微服务的设计原则之一，就是说微服务是全栈式服务。但在重构现有的"单体应用（monolithic application）"时，SQL数据库反规范化可能会导致数据重复和不一致。因此，在单体应用到微服务架构的过渡阶段，可以使用这种设计模式

6 异步消息传递微服务设计模式



虽然REST设计模式非常流行，但它是同步的，会造成阻塞。因此部分基于微服务的架构可能会选择使用消息队列代替REST请求/响应