SOA架构第一步是解耦，换句话说，就是业务拆分

解耦的几个层面

* 项目层面
* 代码层面
* 数据库层面

## 项目层面解耦

目前项目基本是**单体式应用**，即一个项目里包含了所有的业务，这种架构的问题是一旦项目挂了，所有业务都会挂掉

按照业务划分项目，如订单项目、购物车项目、支付项目、商品详情页项目，项目之间互相独立，任何一个项目挂掉，其他项目不收影响继续运行

## 代码层面解耦

目前服务的调用比较随意，谁需要服务谁写服务，导致代码大量冗余，无法重用已有代码

而SOA化能够最大程度的复用代码，提高开发效率

## 数据库层面解耦

目前所有表都在一个数据库中，**如果这个数据挂了，网站所有功能都会挂掉**

数据库解耦，会把数据库拆分成用户库、商品库、订单库等，挂掉其中一个业务库其他业务功能可以继续提供服务

数据库的水平拆分 垂直拆分，可以实现分布式数据库，解决了海量数据存储的问题

数据库拆分的几个原则：

* 大体包含用户库、商品库、订单库、结算库、基础设施库，初期可拆分成这5个库，等数据量大了可进一步拆分
* 去关联，去存储过程。库内减少关联， 库间不允许关联
* 每一个库对应一个小组，只有这个小组有权创建表、修改表。其他团队访问需要分级控制，L1为接口，L2为只读，L3为读写

## 目前架构图

如下图，可以看出项目直接访问数据库，项目之间不能复用业务逻辑



## 解耦后架构图（SOA架构）

如下图，新增了一个服务层，用于实现业务逻辑的复用；另外服务库层做了拆分

上层是项目，通过RPC调用中间的服务层，最底层是数据持久层



## 服务化系统逻辑图

下图是一个最基础的服务，包含了业务逻辑处理和数据库读写



下图是一个复杂点的服务，不仅调用了数据库，还依赖其他服务



## 总结

SOA化是一个长期的过程，不可能一蹴而就，心急吃不着热豆腐

先从数据库入手，去关联去存储过程，拆分数据库

找疼点，解决疼点的同时，顺手SOA

找有业务价值输出的功能点

不要删旧系统，用新系统慢慢替换旧系统