SOA架构设计

技术部\基础架构部
宋伟
2016-06-06


浪漫经典全新演绎
午夜钟声叩响真爱
DISNEY
灰姑娘

目录

1，SOA简介

SOA(Service Oriented Architecture)面向服务架构，从语义上说，它与面向过程、面向对象、面向组件一样，是一种软件组建及开发的方式。与以往的软件开发、架构模式一样，SOA只是一种 体系、一种思想，而不是某种具体的软件产品。SOA要解决的主要问题是：快速构建与应用集成。SOA能够在实际应用中获得成功基于两个重要的因素：灵活性 和业务相关性。这使得它成为解决企业业务发展需求与企业IT支持能力之间矛盾的最佳方案

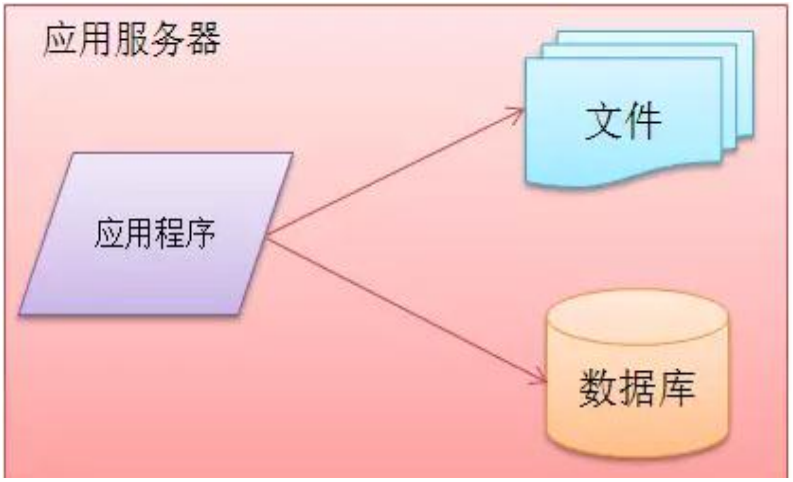
SOA是一种以业务为中心的 IT 架构方法，可以将您的业务作为彼此链接的、可重复的业务任务或服务来进行整合，SOA（面向服务架构）使得构建在各种这样的系统中的服务可以以一种统一和通用的方式进行交互

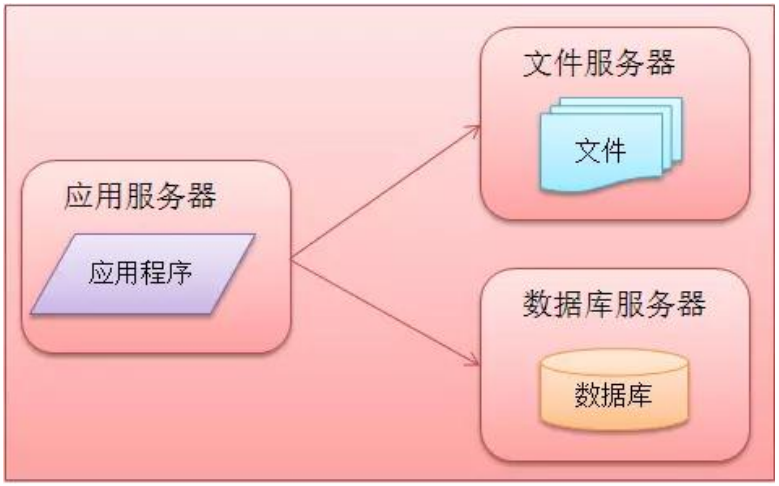
2，SOA之灰姑娘初见

网站架构结构

起初的网站鉴于用户量、访问量较少，只需要一台服务器足以，应用程序、数据库、文件等其所有资源放在一台服务器上就已经足够满足此时的需求，这时候网站的架构就几个简单组成部分如下图(左图)

随着网站业务需求的发展，越来越多的用户进行访问，此时一台服务器渐渐不能满足需求，数据的存储空间出现瓶颈。于是应用程序、数据库、文件三者面临分离，各自为首分配一台服务器，这三台服务器对硬件的要求各取所需，应用服务器处理大量的业务逻辑，需求更快更大的CPU；数据库服务器对数据库的处理需要快速搜索以及缓存，需求对内存更大，对硬盘读写能力更迅速；文件服务器需求放入大量的用户资源，对硬盘空间要求更大。此时的网站的架构组成部分展示如下图(右图)

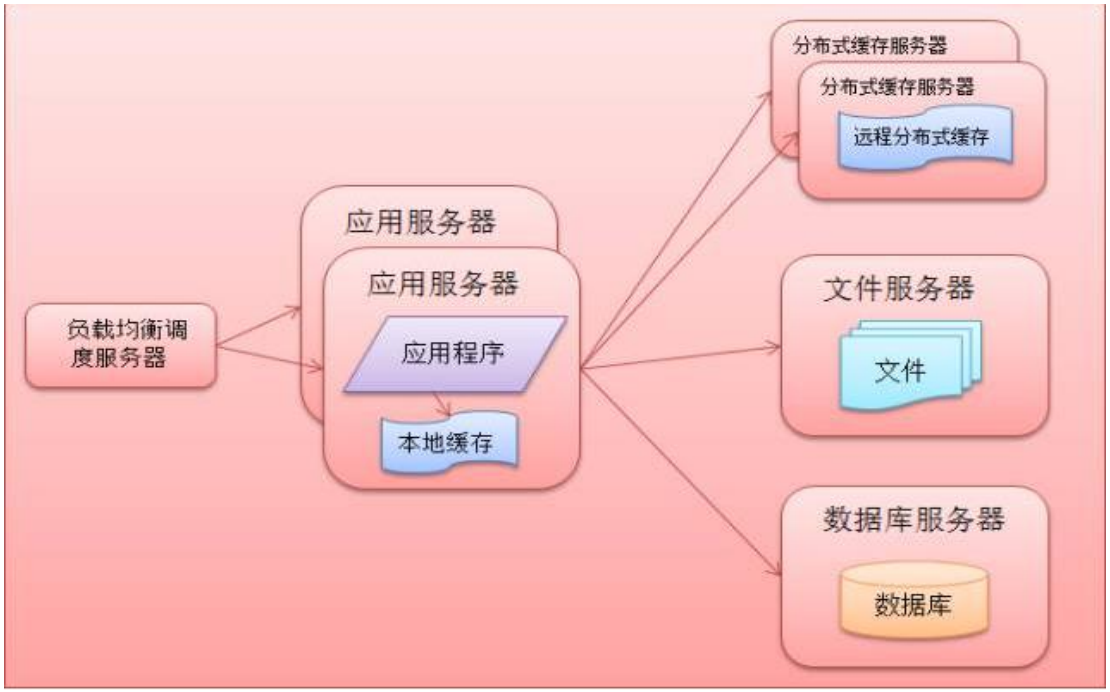




3，SOA之灰姑娘奇遇

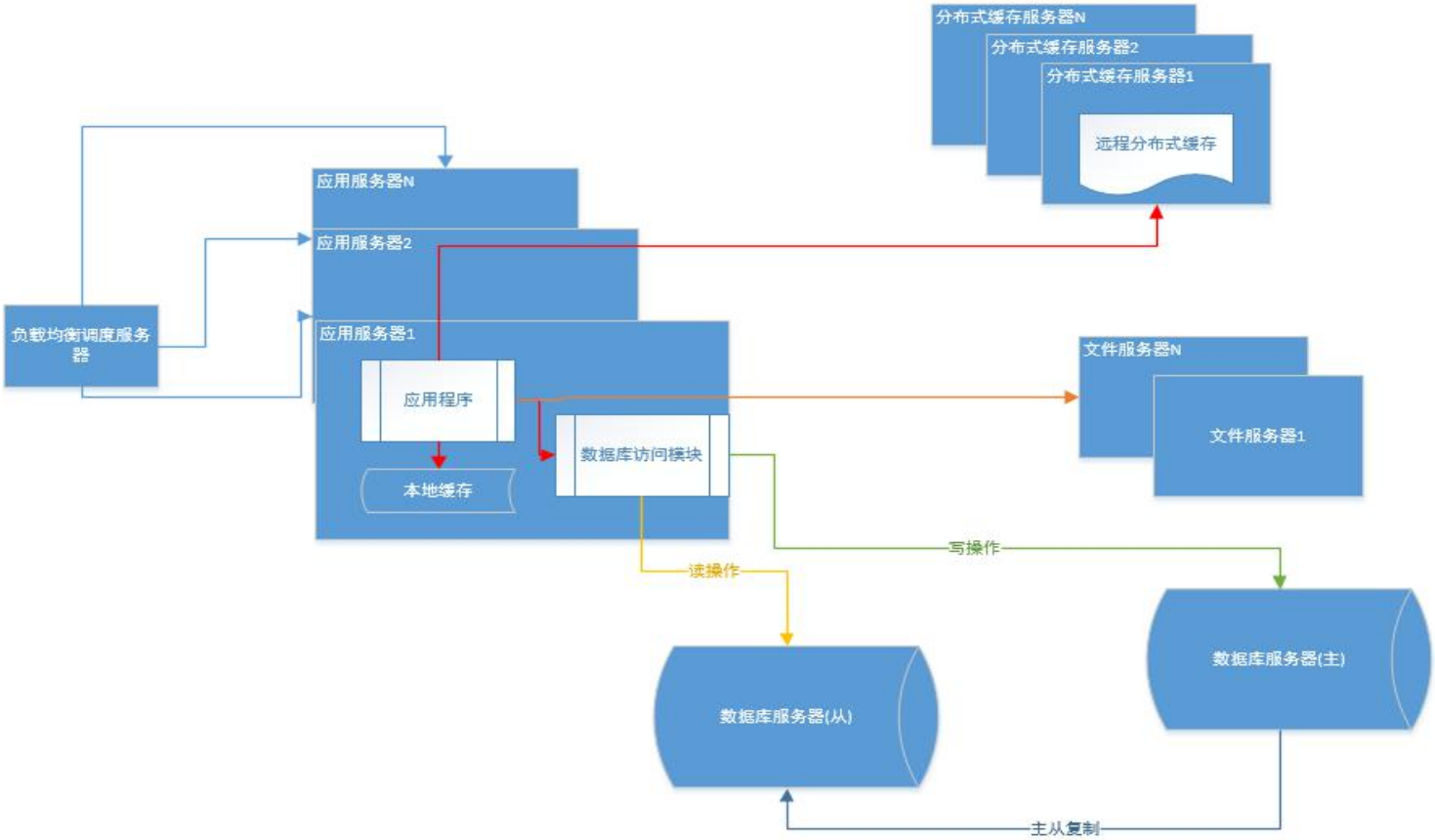
1： 缓存以及应用服务器集群

使用缓存后，数据库的压力得到缓解，但是在面临网站高峰期时，应用服务器处理单一的请求连接出现瓶颈，万事都有解决的办法，采用集群，集群多台应用程序服务器分布原有的应用程序服务器，从而实现了系统的可伸缩性



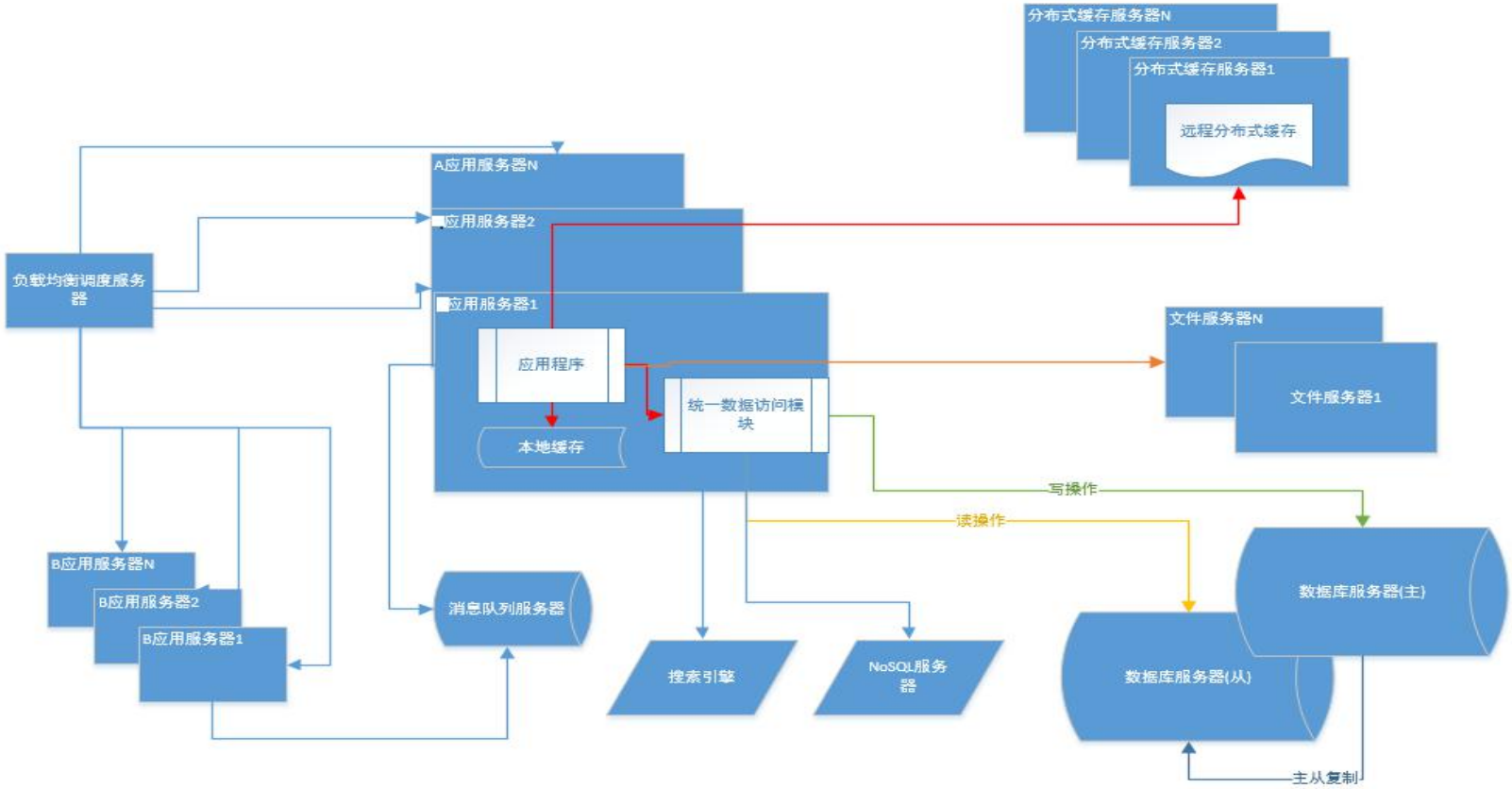
2：数据库读写分离

使用缓存，虽然使用户请求数据操作大部分不直接通过数据库，但是仍有一部分数据（缓存过期、缓存数据没有命中）读写操作需要访问数据库，面对这部分数据，可能出现数据访问负载压力，把数据库读写操作分离



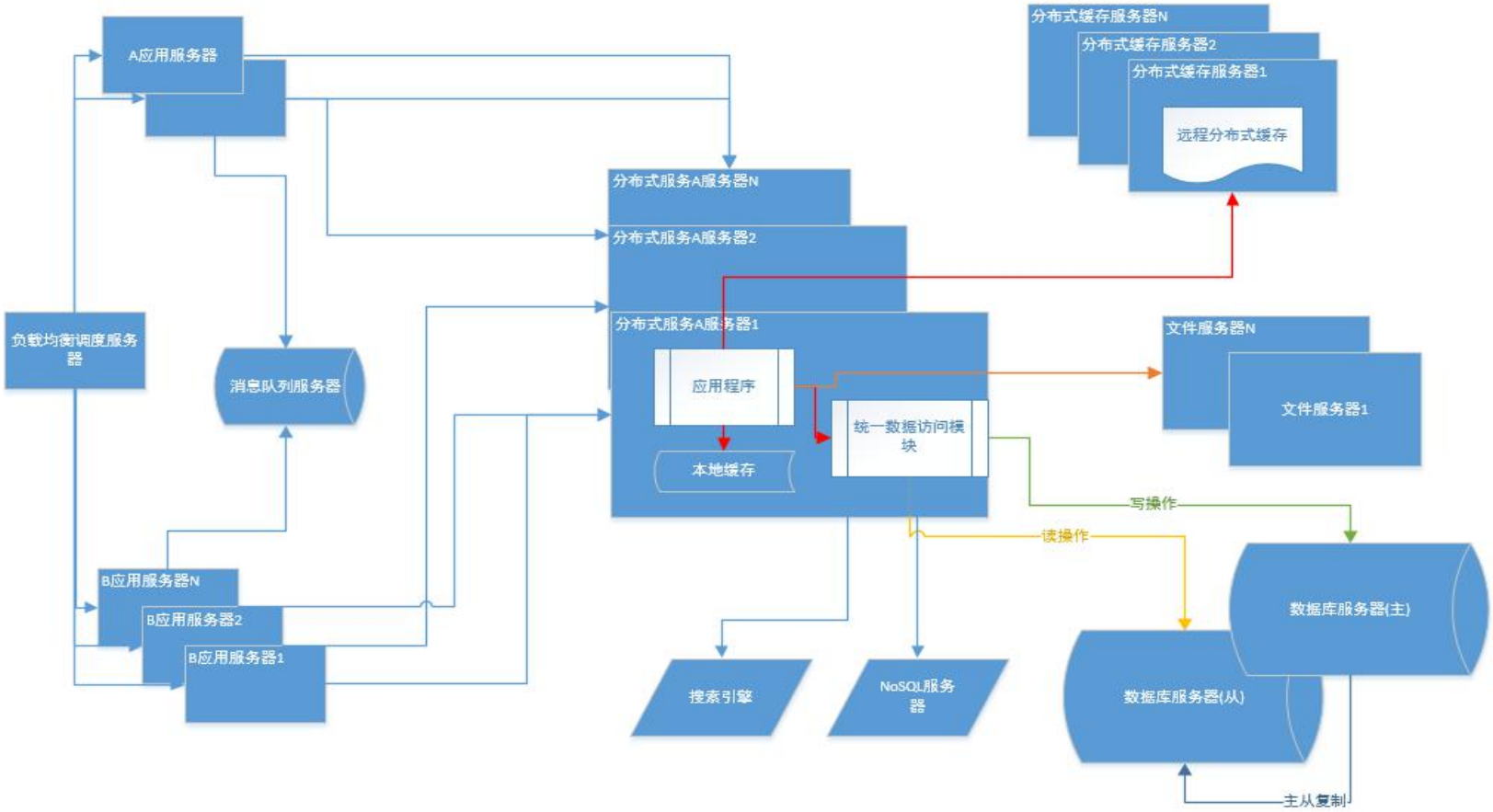
3：业务拆分

大型网站日益发展壮大，业务需求越来越复杂，使用分而治之手段分离整个网站的业务变成不同的产品线。具体到技术上，将一个网站拆分成许多不同的应用，每个应用独立部署，而应用与应用之间通过超链接关联，不过最多的还是通过访问同一个数据存储来构成一个关联的完整系统。



4：分布式服务

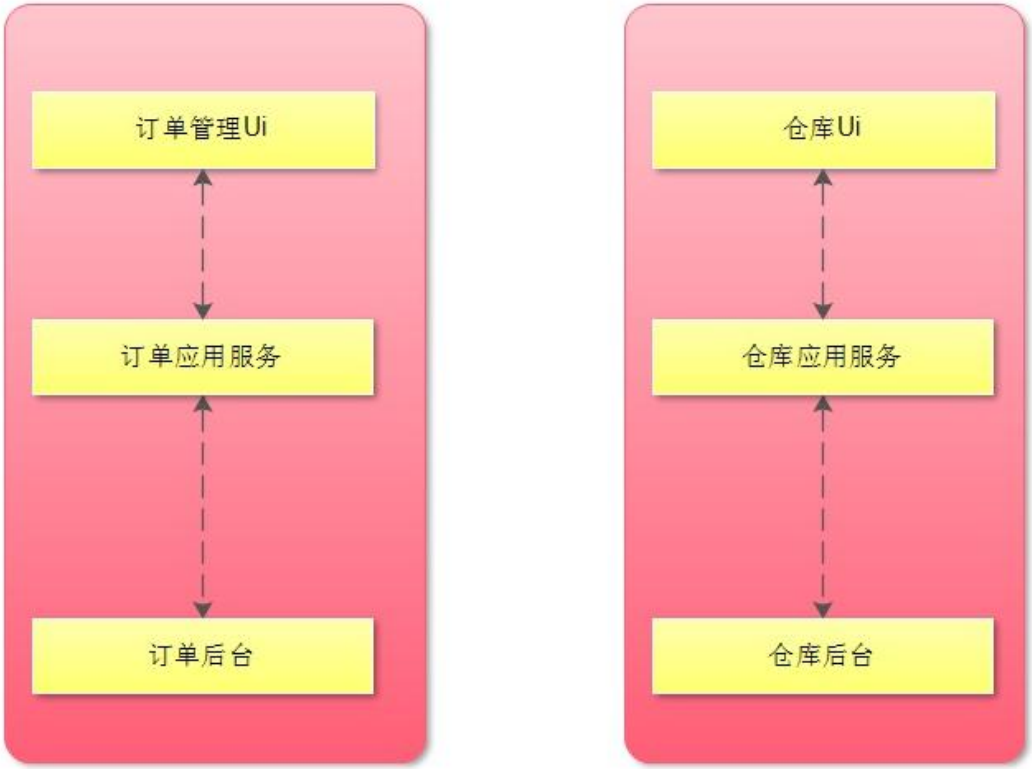
一个应用系统需要执行相同业务操作，那么可以将共同的业务提取出来，独立部署，由这些可复用的业务连接数据库，提供共用业务服务，而应用系统只需要管理用户界面，通过分布式调用共用业务服务完成具体业务操作



SOA架构服务层次

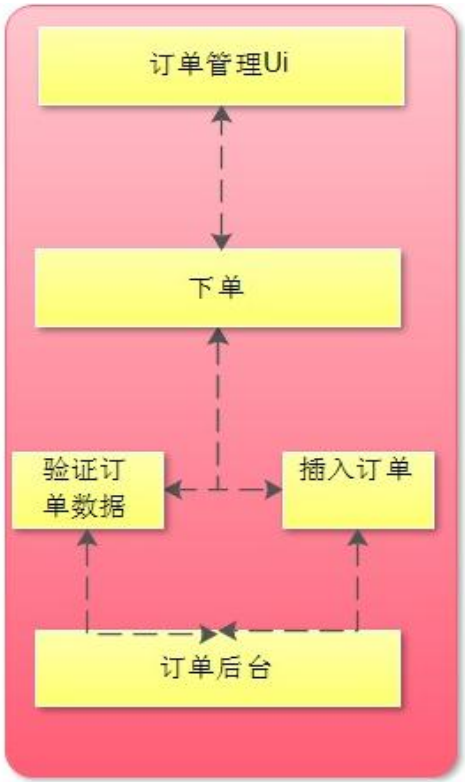
1：原子服务

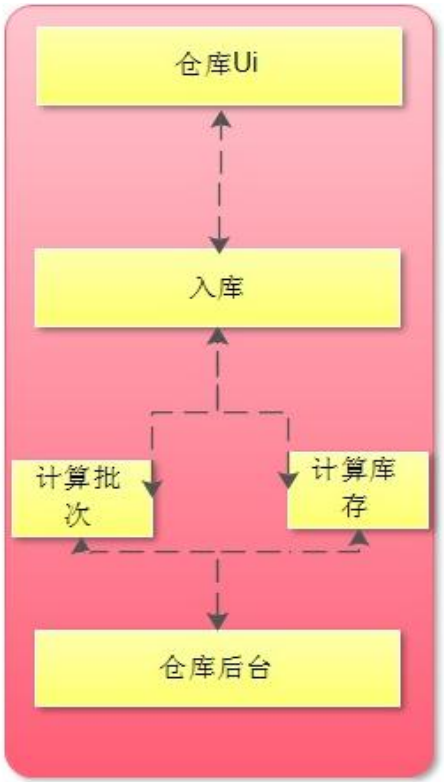
应用服务就是诸如：订单服务、仓库服务、销售服务、客户管理服务，这些服务直接对应不同的应用系统，直接服务这些应用系统的原子操作。订单服务直接原子性的插入订单，没有任何跨其他服务的分支逻辑。仓库服务只管自己的仓库逻辑。同样其他的应用服务只管好自己的职责，杜绝对其他服务的调用



2：组合服务

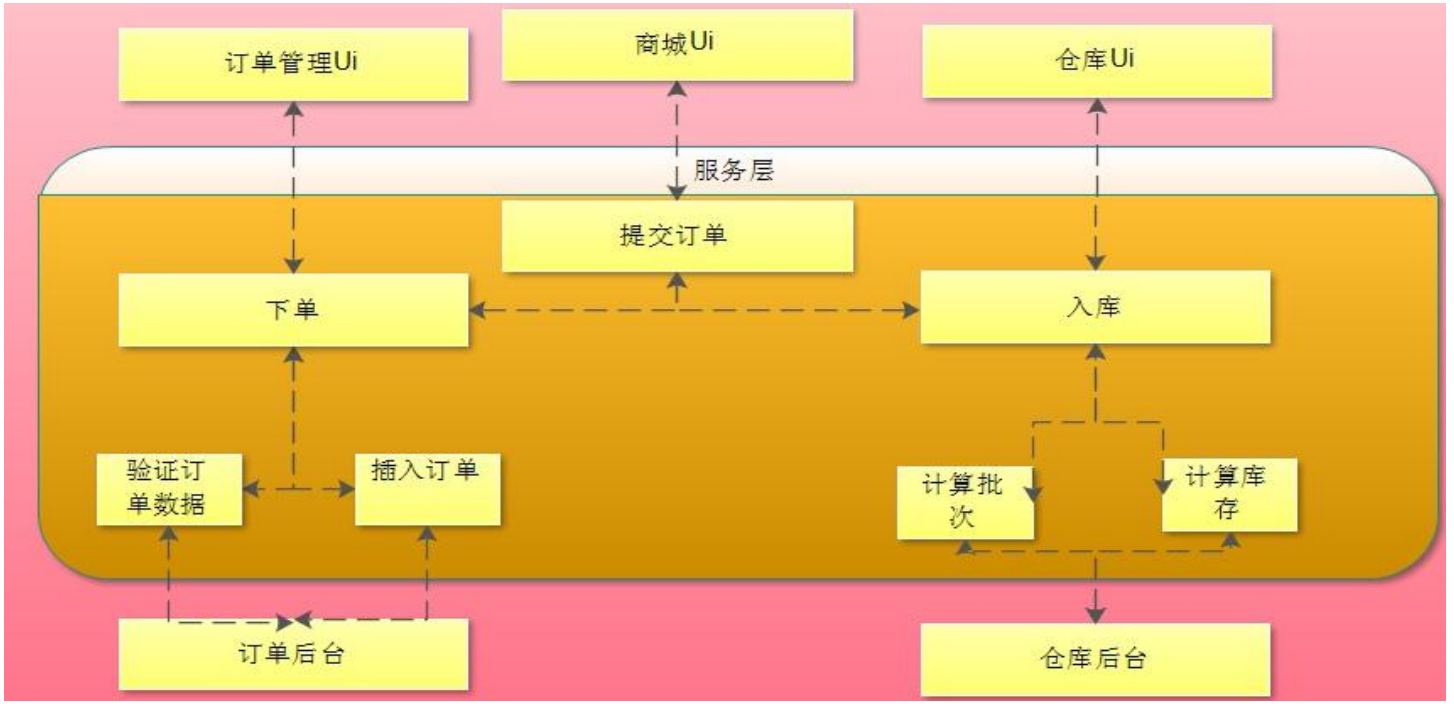
组合服务对下层的应用服务进行了组合，完成了一个基本的业务动作，应用服务中是最基本的基础性的原子性的操作。但是在复杂的业务需求下大部分业务功能都需要跨越多个应用线来完成一个最外层的企业动作。提交订单可能需要穿过很多应用线，订单管理、仓库、财务等等环节





3：服务编排

业务服务是最外层的服务，向下编排了组合服务。业务服务位于最上层，当需要有跨越多个应用线来完成的业务，这个业务就放入业务服务中。比如提交订单，先检查库存、扣减库存（冻结库存），然后下单，再往后通知财务，再往后通知物流等等都是一个复杂的企业服务线。这种最外层的业务逻辑如果你不进行SOA分层然后将其放入最外层的业务服务中，你把它放入任何一个应用线都会使系统调用混乱不堪。所以问题就是需要进行纵向的划分层次



分布式SOA下的数据一致性

1：分布式事务(DTC)

以往包括目前很多项目还是倾向于使用DTC来处理分布式事务，这个方案多数适用于一般的企业应用，业务、访问量、数据量要求都不是很高的情况下。用DTC很方便，事务的自动传播、事务的自动感知、事务的自动回滚和提交

2：异常补偿流程

提供正向或反向的操作来让数据在业务上是一致的

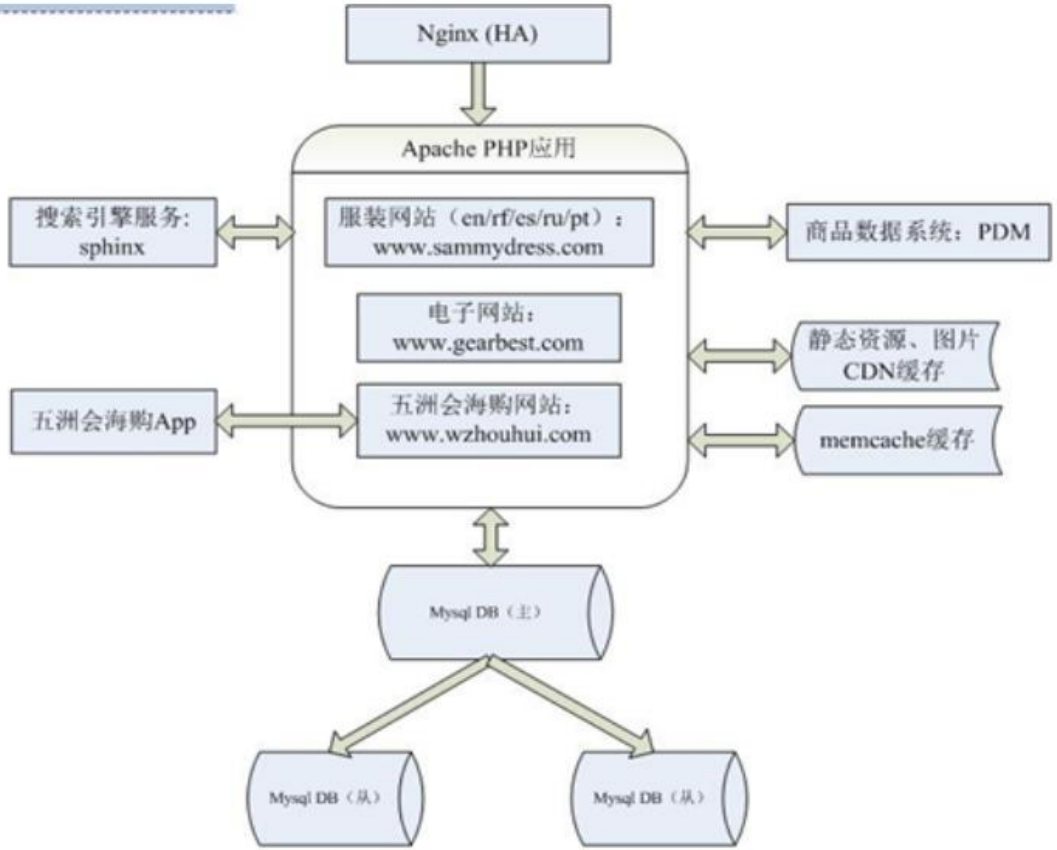
3：基于EDA松散事务

基于异步事件流来实现柔性的分布式事务,EDA简称”事件驱动架构“。多个系统之间通过传播”事件“来驱动整个业务的运转。系统之间没有紧耦合的同步调用的操作，都是通过发出异步的“事件”来通知下一个业务环节

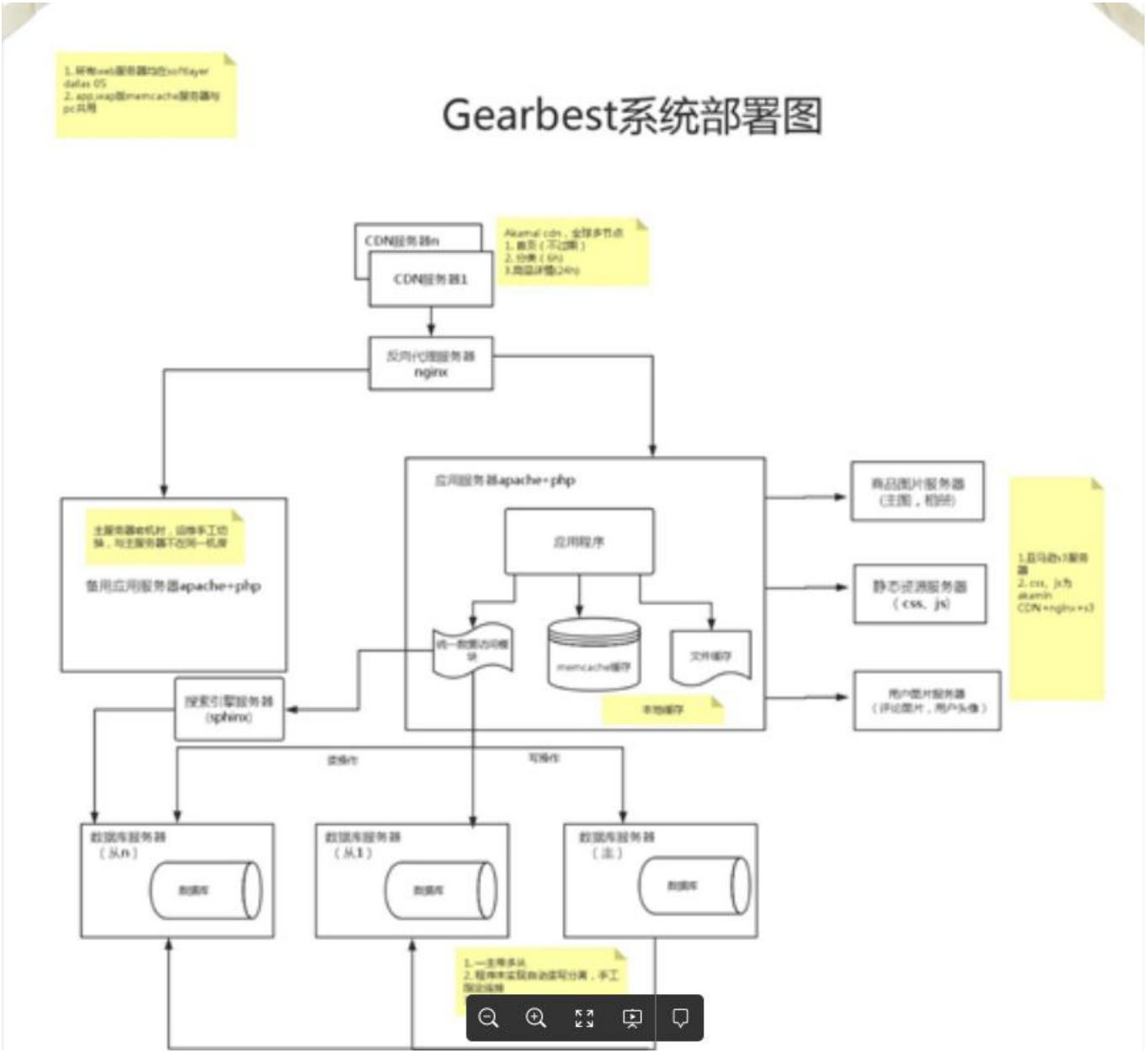
4，SOA之王妃长成

5，环球SOA灰姑娘之路

1：环球现有的电商架构



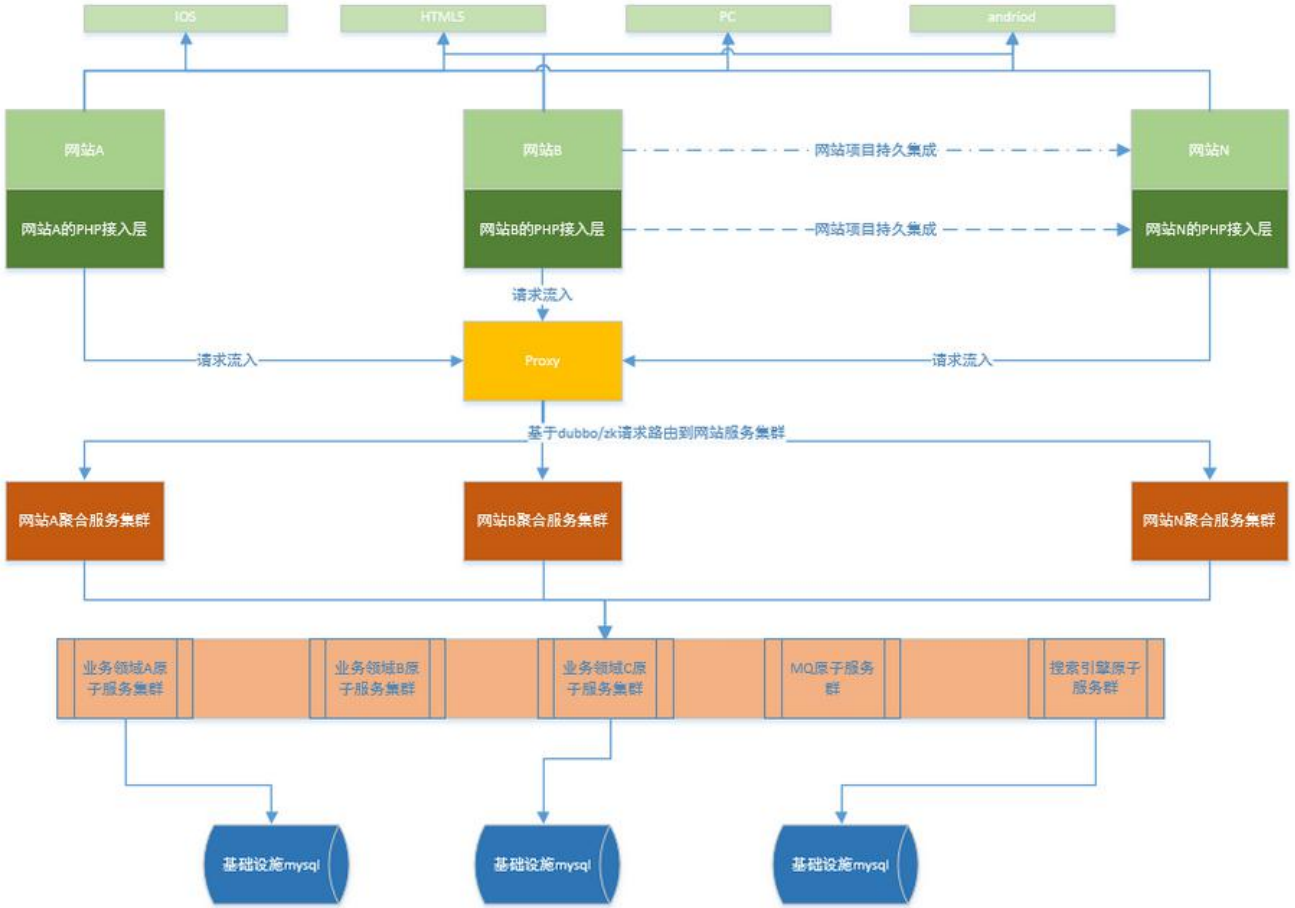
Gearbest系统部署图



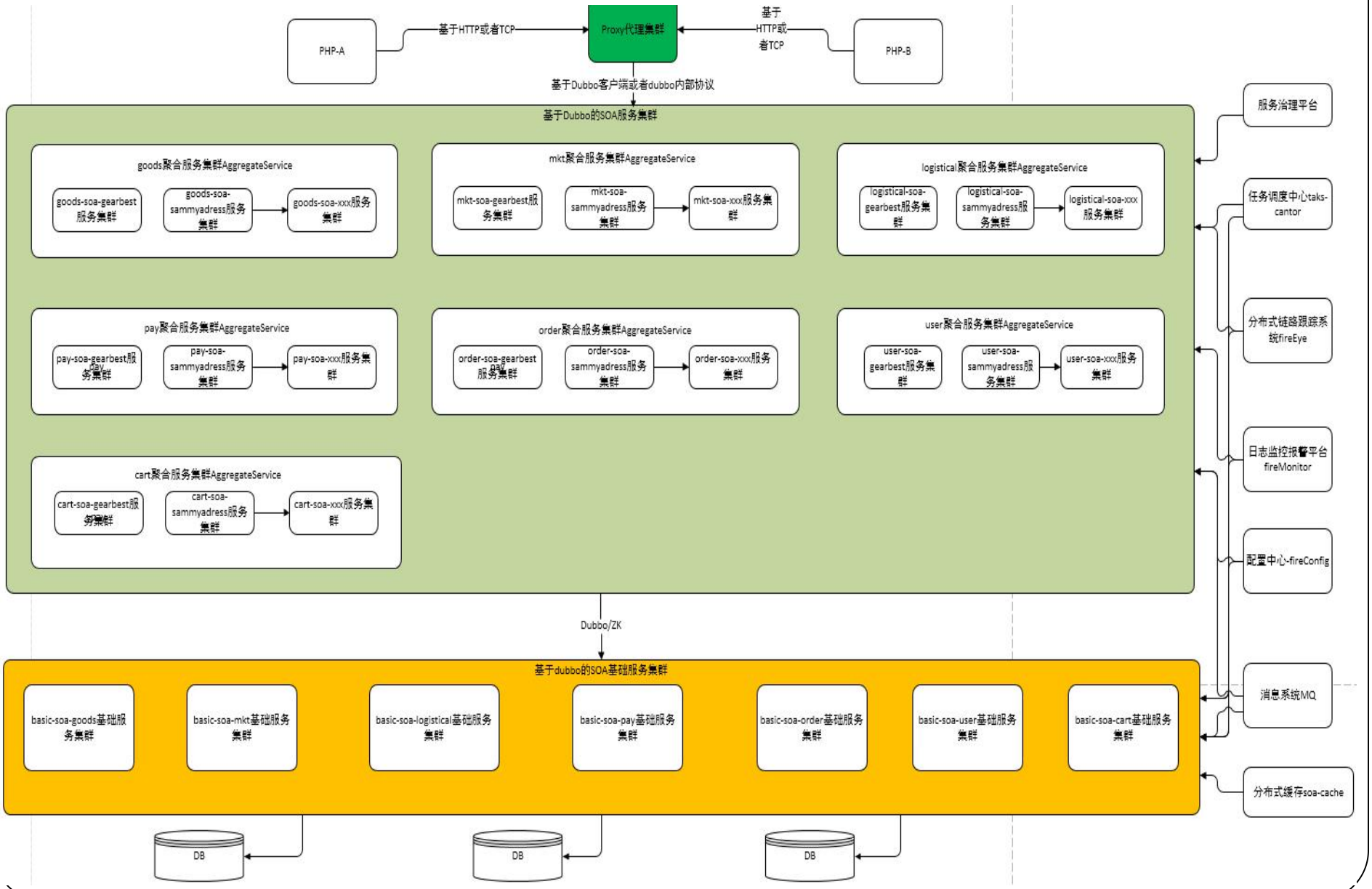

Please allow me to make a sad face
请允许我做一个悲伤的表情

What are we doing ?

2：业务拆分，分布式服务



3：环球SOA架构生态体系



4：当前架构实现目标

1： 增强可伸缩性，后续可以针对性能瓶颈灵活扩展
2： 降低服务耦合度，最大化隔离
3： 结合业务层面，抽离业务聚合点以及原子服务点，更细粒度控制服务
4： 提高多人协同开发效率、降低新人接入以及人员变动带来的影响
5: 利于中间组件持续接入，可以点对点细粒度集成
6： 提高后期服务监控、日志排查效率，更具针对性定位分析问题

Is it the end ?

5：现有SOA存在的问题

1、RPC互相调用难以控制，很容易变成网状调用链并且形成回路

2、RPC调用链长度不可控

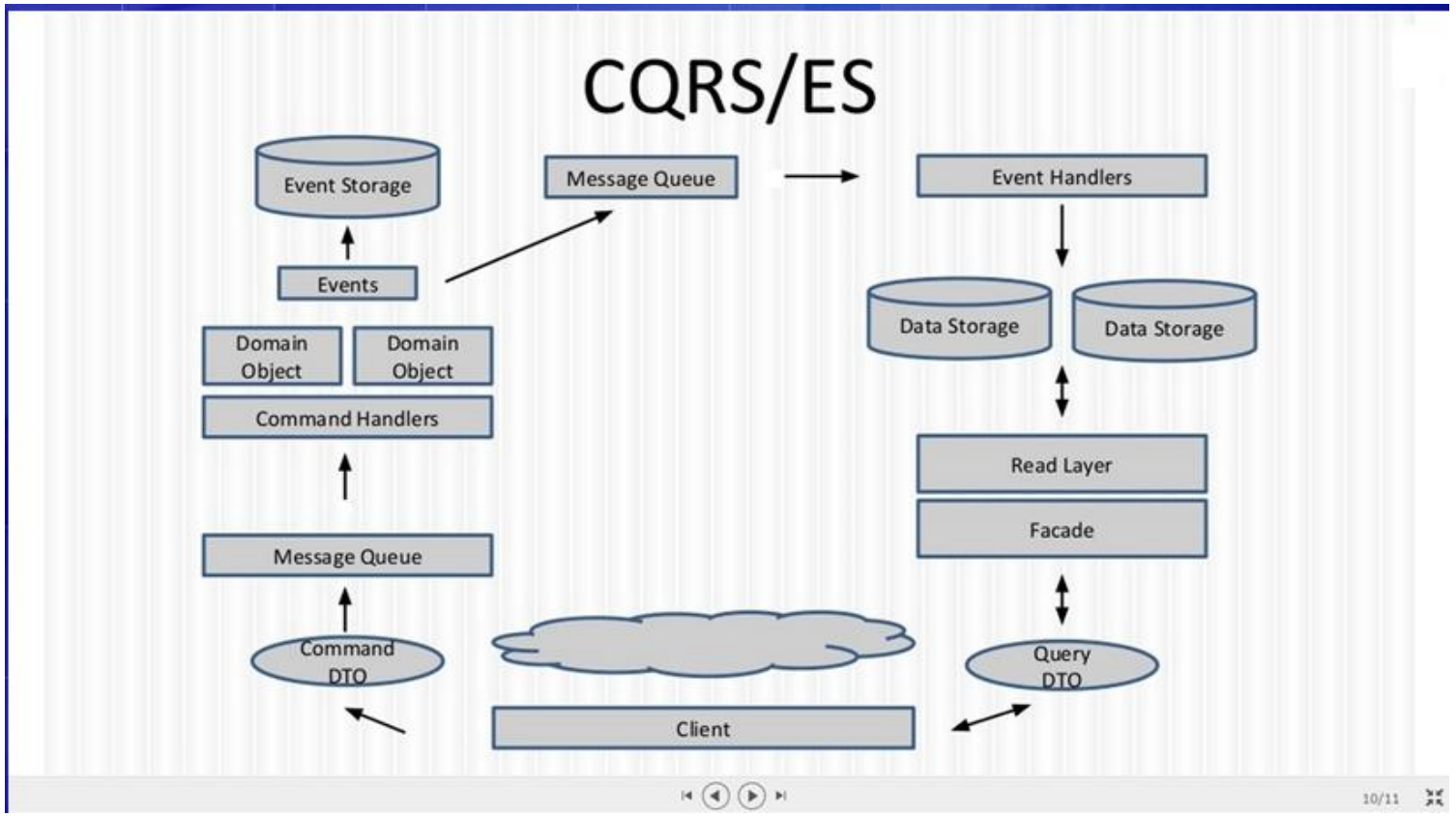
3、没有分布式事务，调用链中某点失败产生的各种错乱数据修复难度较高

4、极端情况下某个节点或者多个节点服务不可用，将会引起雪崩，整体服务或相关服务将不可用

5、系统业务容量估算不足也将很容易引起雪崩

6、复杂的网络调用，跨越的节点越多，服务的质量越难得到保障（响应时间、吞吐量、失败率等）

6： CQRS+DDD+EDA架构



7：CQRS + DDD+ EDA架构优势

1、CQ两端架构分离，相互不受约束，各自独立设计扩展

2、C端通常结合DDD，解决复杂业务逻辑；Q端轻量级查询，多种不同的查询视图通过订阅事件来更新

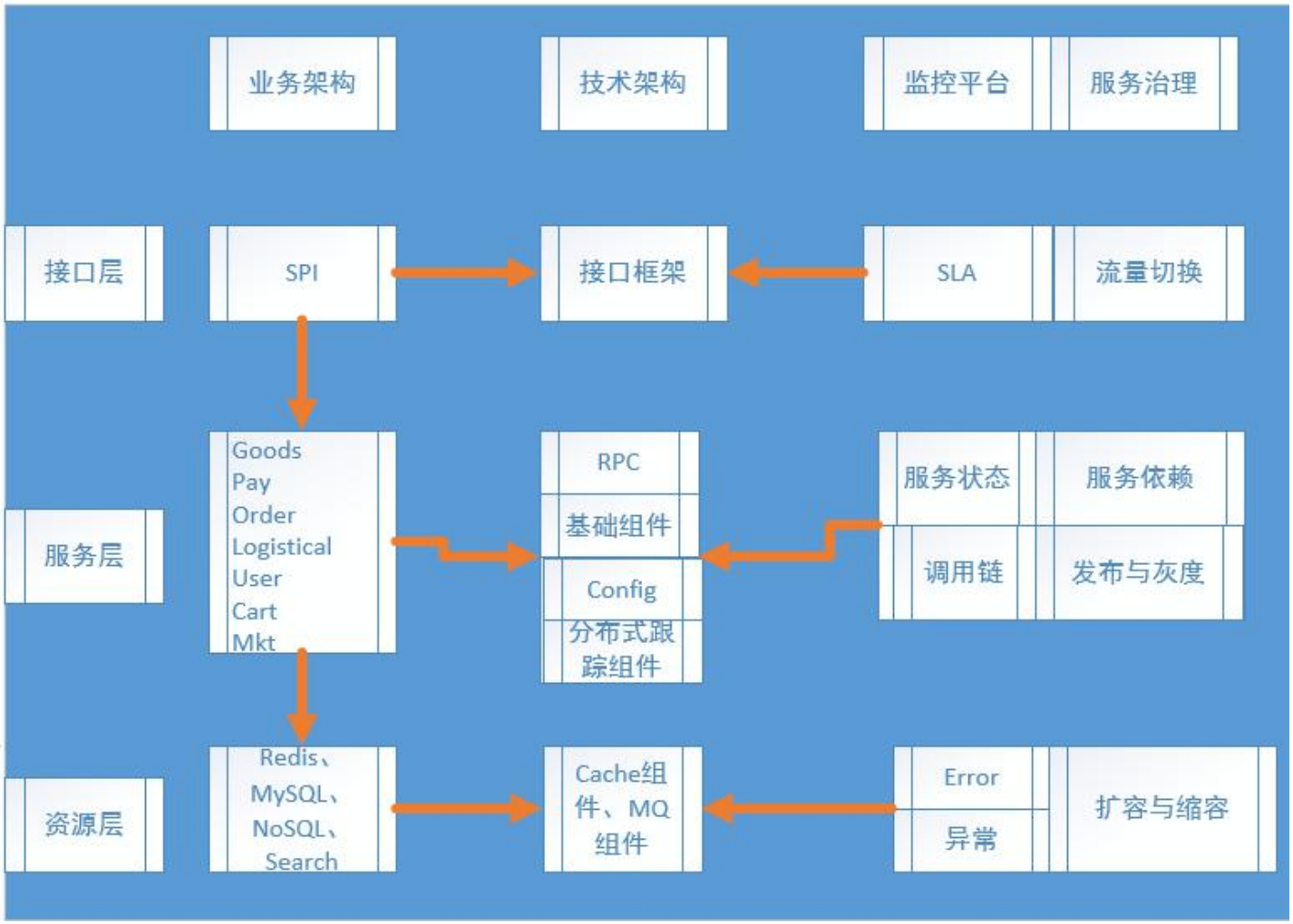
3、C端通过分布式消息队列水平扩展，天然支持削峰

4、EDA架构，整个系统各个部分松耦合，可扩展性好

5、架构层面做到无并发，实现Command的高吞吐

6、技术架构和业务代码完全分离，业务程序不需要关心技术问题

8: 环球电商架构蓝图


业务架构
技术架构
监控平台
服务治理
接口层
SPI
接口框架
SLA
流量切换
Goods
Pay
Order
Logistical
User
Cart
Mkt
RPC
基础组件
Config
分布式跟踪组件
服务状态
服务依赖
调用链
发布与灰度
服务层
资源层
Redis、MySQL、NoSQL、Search
Cache组件、MQ组件
Error
异常
扩容与缩容

6：总结

A SOA架构历经了长时间磨练才能成熟稳定：

- 没有完美的架构，存在即理由，任何架构都有一定的使用场景，需要根据实际业务，实际需求设计符合要求的架构
- 业务和架构的发展需要层层递进，先满足业务要求，然后不断优化重构，不可盲目推进，避免推倒重来的悲剧
- 一味的追随大公司解决方案，大公司的经验和成功固然重要，但是不能盲目的追从，要与实际的具体业务需求有所改动；
- 为了技术而技术，网站技术是为业务而存在的，但是一味的追求新技术，可能会导致结构技术之路越走越难；
- 企图用技术解决所有问题，技术虽是解决业务问题的，但也不是万能钥匙，有些业务的问题也是可以通过业务手段解决。

B Q&A：

- Why SOA？
- What are we doing？
- What is the final Goal？

谢谢！