大型网站软件系统的特点

与传统企业应用系统相比，大型互联网应用系统有以下特点

高并发，大流量：需要面对高并发用户，大流量访问

高可用：系统 7 x 24 小时不间断服务

海量数据：需要存储、管理海量数据，需要使用大量服务器。

用户分部广泛，网络情况复杂：许多大型互联网都是为全球用户服务的，用户分布范围广，各地网络情况千差万别。

安全环境恶劣

需求快读变更，发布频繁：

渐进式发展：几乎所有的大型互联网网站都是从一个小网站开始，渐进发展起来的。Facebook是伯克扎克同学在哈佛大学的宿舍里开发的。Google的第一台服务器部署在斯坦福大学的实验室里；阿里巴巴是在马云家客厅里诞生的。好的互联网产品都是慢慢运营出来的。

大型网站架构演化发展历程

大型网站的技术挑战主要来自庞大的用户，高并发的访问和海量的数据，任何简单的业务一旦需要处理以p计的数据和面对数以 亿计的用户，问题就会变得棘手，大型网站架构主要就是解决这类问题

**初始阶段网站**：应用程序、数据库、文件等所有的资源都在一台服务器上，典型案例：基于**LAMP**架构

**应用服务和数据服务分离**：单台不再适应业务的发展，越来越多的数据导致存储空间不足。将应用和数据分离后成三台Sever（应用服务器、文件服务器与数据库服务器）。分离后三台Server对硬件资源的需求各不相同：应用服务器需要更快更强大的CPU，而数据库服务器需要更快的硬盘和更大的内存，文件服务器则需要更大的硬盘；

**使用缓存改善网站性能**：二八定律80%的业务访问几种在20%的数据上。淘宝买家浏览商品集中在少部分 成交数多 评价良好的商品上；百度搜索关键字几种在少部分热门词汇上；只有经常登录的用户才会发微博 看微博，这部分用户只占用户数目的一部分。

把这一小部分数据缓存在内存中可以减少数据库的访问压力，提高网站的数据访问速度，改善数据库写入性能。

**网站使用的缓存分为两种**：缓存在应用服务器上的本地缓存和缓存再分布式缓存服务器上的远程缓存。本地缓存的访问速度更快一些，但是受应用服务器内存限制，其缓存数据量有限，而且会出现和应用程序争用内存的情况。远程分布式缓存可以使用集群的方式，部署大内存的服务器作为专门的缓存服务器，理论上做到不受内存容量限制。

**使用应用服务器集群改善网站并发处理**：

只要能通过一台服务器的方式改善负载压力，就可以以同样的方式持续增加服务器不断改善系统性能，从而实现系统的可伸缩性。通过负载均衡调度服务器，可将来自用户浏览器的访问请求分发到应用服务器集群中的任何一台服务器，使应用服务器的负载压力不再成为整个网站的瓶颈。

数据库读写分离：

网站使用缓存后，使绝大部分数据读操作访问都可以不通过数据库就能完成，但是仍有一部分操作（缓存不命中过期等）和全部的写操作需要访问数据库。在网站的用户达到一定规模后，数据库因为负载压力过高而成为网站瓶颈。

目前大部分主流数据库都提供主从热备功能，通过配置两台数据库主从系统，可以将一台数据库数据同步到另一台服务器上，实现读写分离，从而改善数据库负载压力。应用服务器写数据时访问主数据库，主数据库通过主从复制机制将数据更新同步到从数据库。读数据通过从数据库获取数据。为了便于应用程序访问读写分离后的数据库，通常在应用服务器端使用专门的数据访问模块，使数据库读写分离对应于透明。

使用反向代理和CDN加速网站响应：

由于中国复杂的网络环境，不同地区访问网站速度差别也极大。加速网站访问速度主要手段使用CDN和反向代理。

CDN和反向代理的基本原理都是**缓存**，区别在于CDN部署在网络提供商的机房，而反向代理则部署在网站的中心机房。使用CDN和反向代理的目的都是**尽早返回数据给用户**，一方面加快用户访问速度，另一方面也减轻后端服务器的负载压力。

使用反向代理和CDN加速网站响应：CDN和反向代理的基本原理都是**缓存**，区别在于CDN部署在网络提供商的机房，而反向代理则部署在网站的中心机房。

首先访问的服务器是反向代理服务器，如果缓存着用户请求的资源，就将其直接返回给用户。

使用CDN和反向代理的目的都是**尽早返回数据给用户**，一方面加快用户访问速度，另一方面也减轻后端服务器的负载压力。

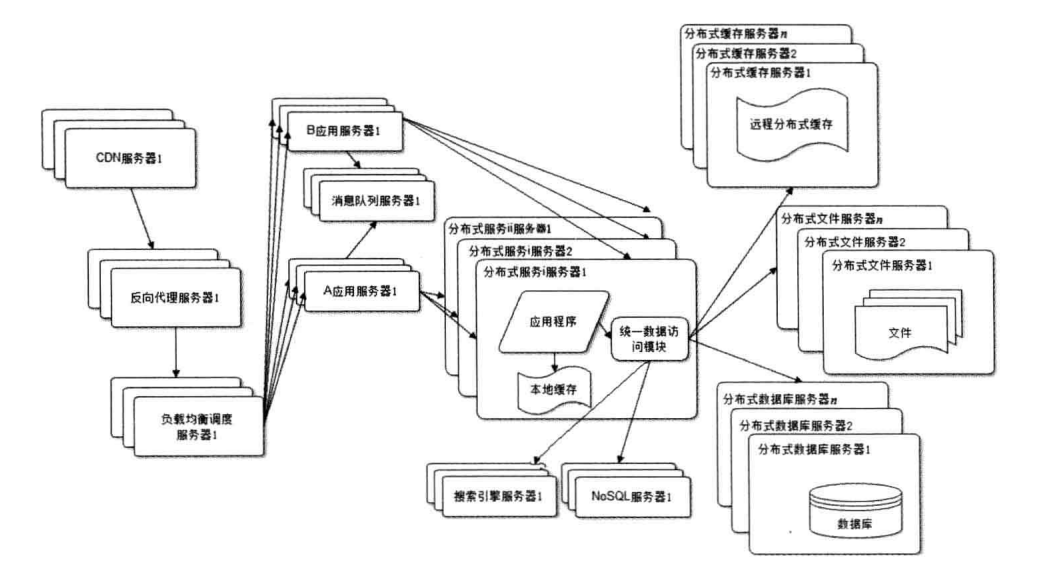
使用分布式文件系统和分布式数据库系统：随着网站业务的发展，两台DB服务器依然不能满足需求，文件系统也一样需要使用分布式系统。分布式数据库是网站数据库拆分的最后手段，只有单表数据规模非常庞大的时候才使用。网站更常用的是业务分库，将不同业务的数据库部署在不同的物理服务器上。

使用NoSQL和搜索引擎：NoSQL和搜索引擎都是源自互联网的技术手段，对可伸缩的分布式特性具有更好的支持。应用服务器则通过一个统一数据访问模块访问各种数据，减轻应用程序管理诸多数据源的麻烦。

业务拆分：通过分而治之的手段将整个网站业务分成不同的产品线，如淘宝将首页、商铺、订单、卖家、买家等拆分成不同的产品线，分归不同的业务团队负责。各个应用之间可以通过建立一个超链接建立关系，也可以通过消息队列进行数据分发，当然最多的还是通过访问同一个数据存储系统来构成一个关联的完整系统。

分布式服务：随着业务拆分越来越小，存储系统越来越庞大，应用系统的整体复杂度呈指数增加部署维护越来越困难。

既然每一个应用系统都需要执行许多相通的业务操作，比如用户管理、商品管理等，那么可以将这些共用的业务提取出来，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，通过分布式服务调用共用业务服务完成具体业务操作。



网站架构演化的价值观

网站的价值在于它能为用户提供什么价值，在于网站能做什么，而不在于它是怎么做的，所以在网站还很小的时候就去追求网站的架构是舍本逐末，得不偿失的。小型网站最需要做的就是为用户提供好的服务来创造价值，得到用户的任何，活下去野蛮生长。

大型网站架构发展过程中几个容易出现的误区：

一味追求大公司的解决方案

为了技术而技术：网站技术是为业务而存在的，除此毫无意义。在技术选型和架构设计中，脱离网站业务发展的实际，一味追求时髦的新技术，可能将网站技术发展引入崎岖小道，架构之路越走越难。

企图用技术解决所有问题：

最典型的是2012年12306故障事件后，各路专业非专业人士众说纷纭地帮12306的技术架构出谋划策，解决其大规模并发访问的问题。

而其真正的问题不在于它的技术架构，而在于它的业务架构：12306根本就不应该在几亿中国人一票难求的情况下以窗口售票的模式在网上售票（零点开始出售若干天后的车票）。后来在售票方式上引入排队机制、整点售票调整为分时段售票，控制住了并发访问的量。

网站架构模式

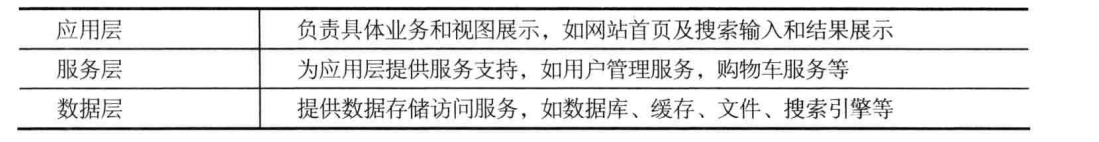
模式：每一个模式描述了一个在我们周围不断重复发生的问题及该问题解决方案的核心。这样，你就能一次又一次地使用该方案而不必做重复工作。

为了解决发型网站面临的高并发访问、海量数据处理、高可靠运行等一系列问题与挑战，大型互联网公司在实践中提出了许多解决方案，以实现网站高性能、高可用、易伸缩、可扩展、安全等各种技术架构目标。这些解决方案又被更多网站重复使用，从而逐渐形成架构模式。

分层：

分层是企业应用系统中最常见的一种架构模式，将系统在横向维度上切分为几个部分，每个部分负责一部分相对比较单一的职责，然后通过上层对下层的依赖和调用组成一个完整的系统。

网络的7层通信协议是一种分层结构；计算机硬件、操作系统、应用软件也可以看做是一种分层结构。网站架构也采用分层结构



通过分层，可以更好地将一个庞大的软件系统切分为不同的部分，便于分工合作开发和维护；各层之间具有一定的独立性，只要维持调用接口不变，各层可以根据具体问题独立演化发展而不需要其他层必须做出相应调整。

应用层可再细分为视图层和业务逻辑层；服务层也可以细分为数据接口层（适配各种输入输出数据格式）和逻辑处理层。

分割：

将不同的功能和服务分割开来，包装成高内聚低耦合的模块单元，一方面有助于软件的开发和维护；另一方面，便于不同模块的分布式部署，提高网站的并发能力和功能扩展能力。

应用层：可将不同业务进行分割，服务处也可根据需要将服务分割成合适的模块

分布式：

分层和分割的一个主要目的是为了切分后的模块便于分布式部署，将不同模块部署在不同的服务器上，通过远程调用协同工作。

常用的几种方案：

分布式应用和服务：将分层和分割后的应用和服务模块分布式部署，除了可以改善网站性能和并发性加快开发和发布速度，减小数据库连接消耗资源外还可以使不同应用复用共同的服务便于业务扩展

分布式静态资源：js css 图片等静态资源独立分布式部署，采用独立域名。

分布式数据 和存储：除了对传统关系数据库进行分布式部署外，为网站应用而生的各种Nosql产品几乎都是分布式的。

分布式计算：网站除了要处理在线业务，还有很大一部分后台业务要处理，包括搜索引擎索引构建、数据仓库的数据分析统计等。这些业务的计算规模非常庞大，目前普遍使用Hadoop及其MapReduce分布式计算框架进行此类批处理计算，其特点是移动计算而不是移动数据，将计算程序分发到数据所在的位置以加速计算和分布式计算。此外还有可以支持网站线上服务器配置实时更新的分布式配置；分布式环境下实现并发和协同的分布式锁；支持云存储的分布式文件系统等。