

# 极客大学架构师训练营

李智慧



## 《架构师训练营》

# 系统架构概述：互联网系统面临怎样的挑战？

# 高并发，大流量

需要面对高并发用户，大流量访问。

- Google 日均 PV 数35亿，日均 IP 访问数3亿
- 微信在线用户数10亿
- 天猫双十一活动一天交易额3000亿。

# 高可用

系统7×24小时不间断服务。大型互联网的宕机事件通常会成为新闻焦点

# 海量数据

需要存储、管理海量数据。

- Facebook 每周上传的照片数目接近10亿
- 百度收录的网页数目有数百亿
- Google 有近百万台服务器为全球用户提供服务

# 用户分布广泛，网络情况复杂

许多大型互联网都是为全球用户提供服务的，用户分布范围广，各地网络情况千差万别。在国内，有各个运营商网络互通难的问题。而中美光缆的数次故障，也让一些对国外用户依赖较大的网站不得不考虑在海外建立数据中心。

# 安全环境恶劣

由于互联网的开放性，使得互联网站更容易受到攻击，大型网站几乎每天都会遇到黑客攻击情况。2011年国内多个重要网站泄露用户密码，让普通用户也直面一次互联网安全问题。



# 需求快速变更，发布频繁

和传统软件的版本发布频率不同，互联网产品为快速适应市场，满足用户需求，其产品发布频率也是极高的。Office 的产品版本以年为单位发布，而一般大型网站的产品每周都有新版本发布上线，至于中小型网站的发布就更频繁了，有时候一天会发布几十次。

# 渐进式发展

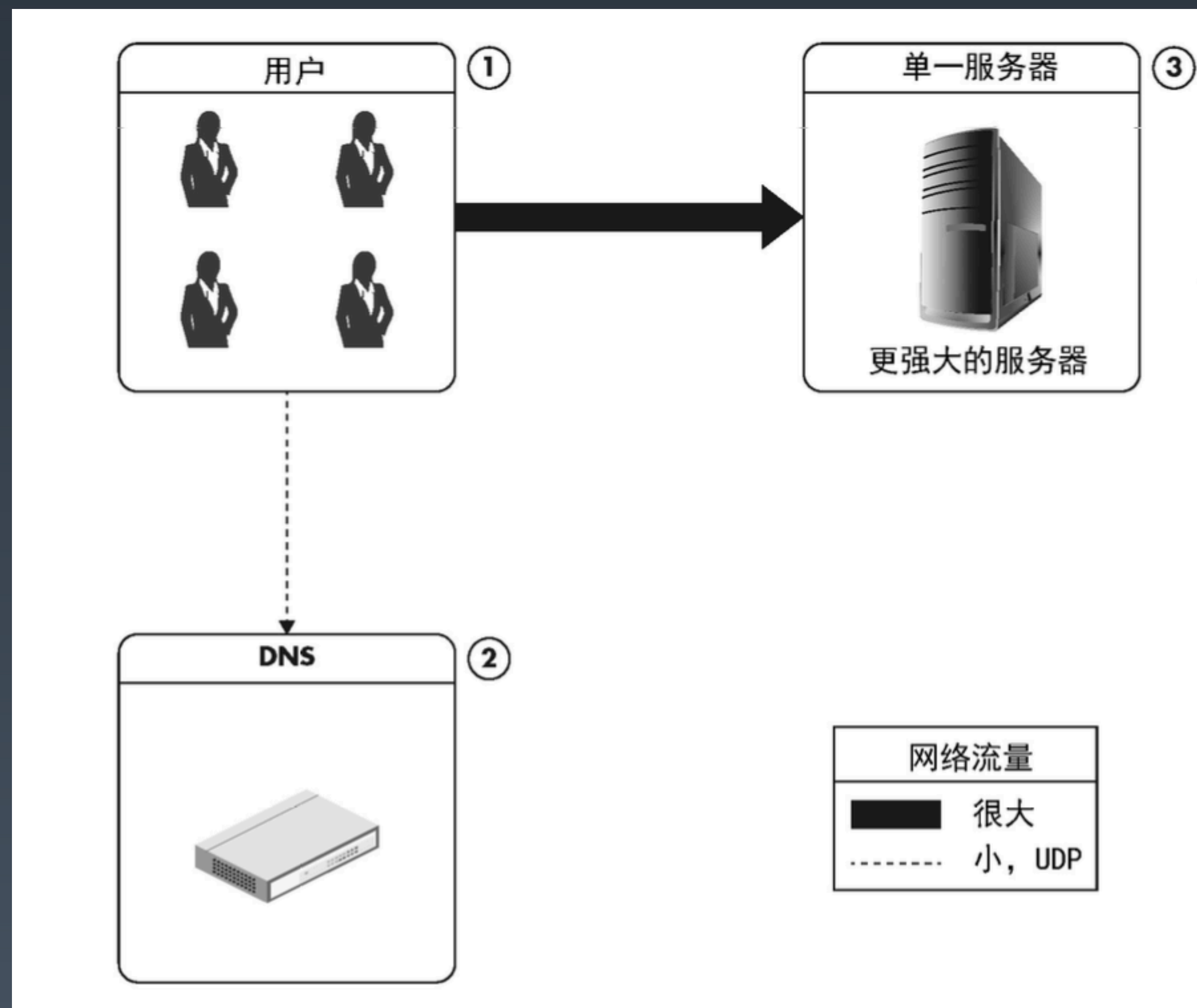
不同于传统软件产品或者企业应用系统，一开始就规划好所有的功能和非功能需求，几乎所有的大型互联网站都是从一个小网站开始，渐进的发展起来的。

- 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；
- 阿里巴巴则是在马云家的客厅里诞生的。

好的互联网产品都是慢慢运营出来的，不是一开始就开发好的。那些刚建立就投入巨资，有巨大背景的网站，后来发展都很惨淡。

# 应对高并发挑战的两个技术方向

# 垂直伸缩



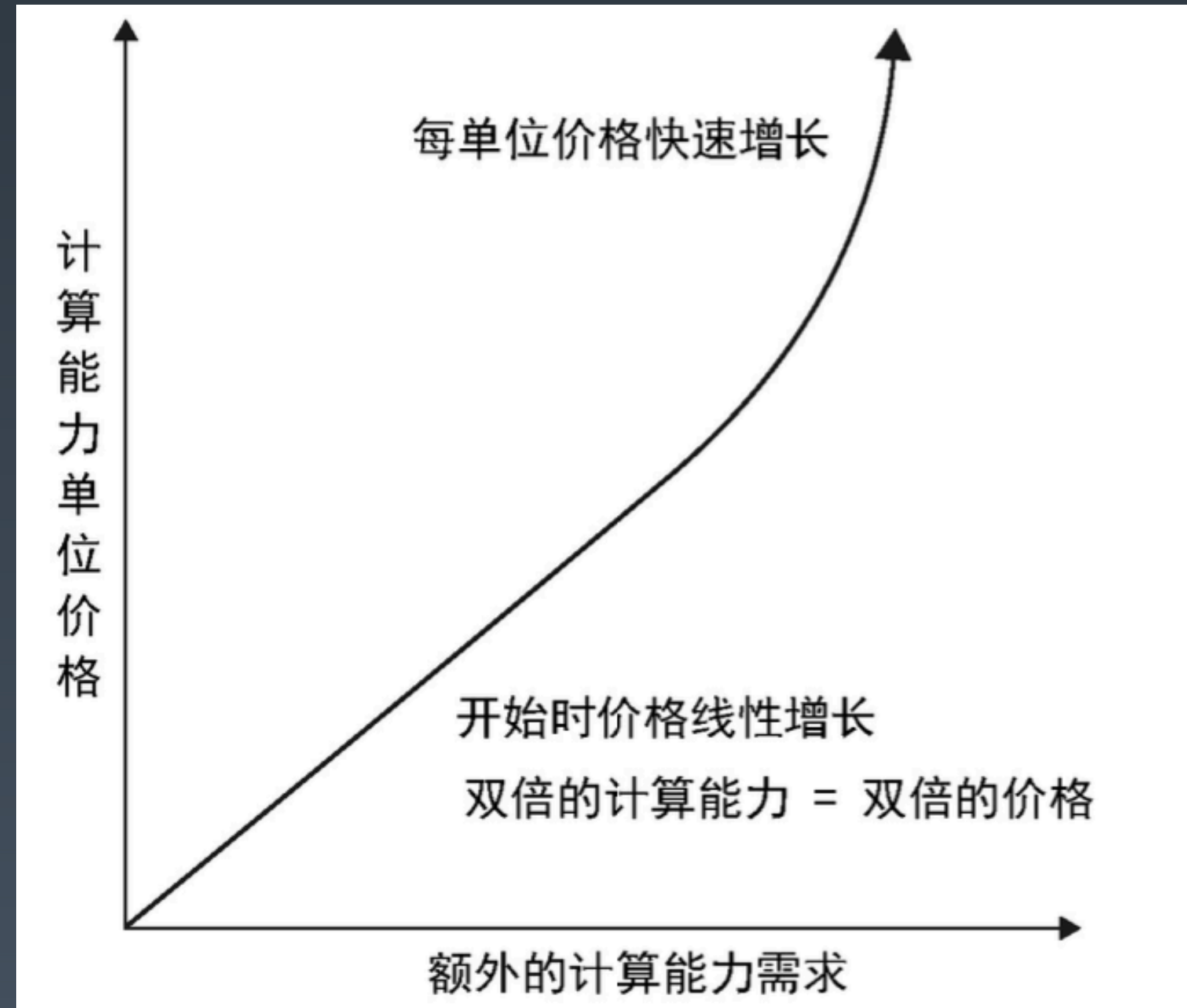
# 垂直伸缩

通过升级硬件和网络吞吐能力可以实现垂直伸缩。由于不需要改变应用架构，所以通常被认为是最简单的短期伸缩性方案。

- 通过使用 RAID（独立冗余磁盘阵列）增加 I/O 吞吐能力。
- 通过切换到 SSD（固态硬盘）改善 I/O 访问速度。
- 通过增加内存减少 I/O 操作。
- 通过升级网络接口或者增加网络接口提高网络吞吐能力。
- 更新服务器使用更多处理器或者更多超线程。

# 垂直伸缩的缺点

- 达到某个程度后，增加计算能力需要的更多的花费。
- 垂直伸缩有物理极限。
- 操作系统的设计或者应用程序自身制约着垂直伸缩最多只能达到某个点。

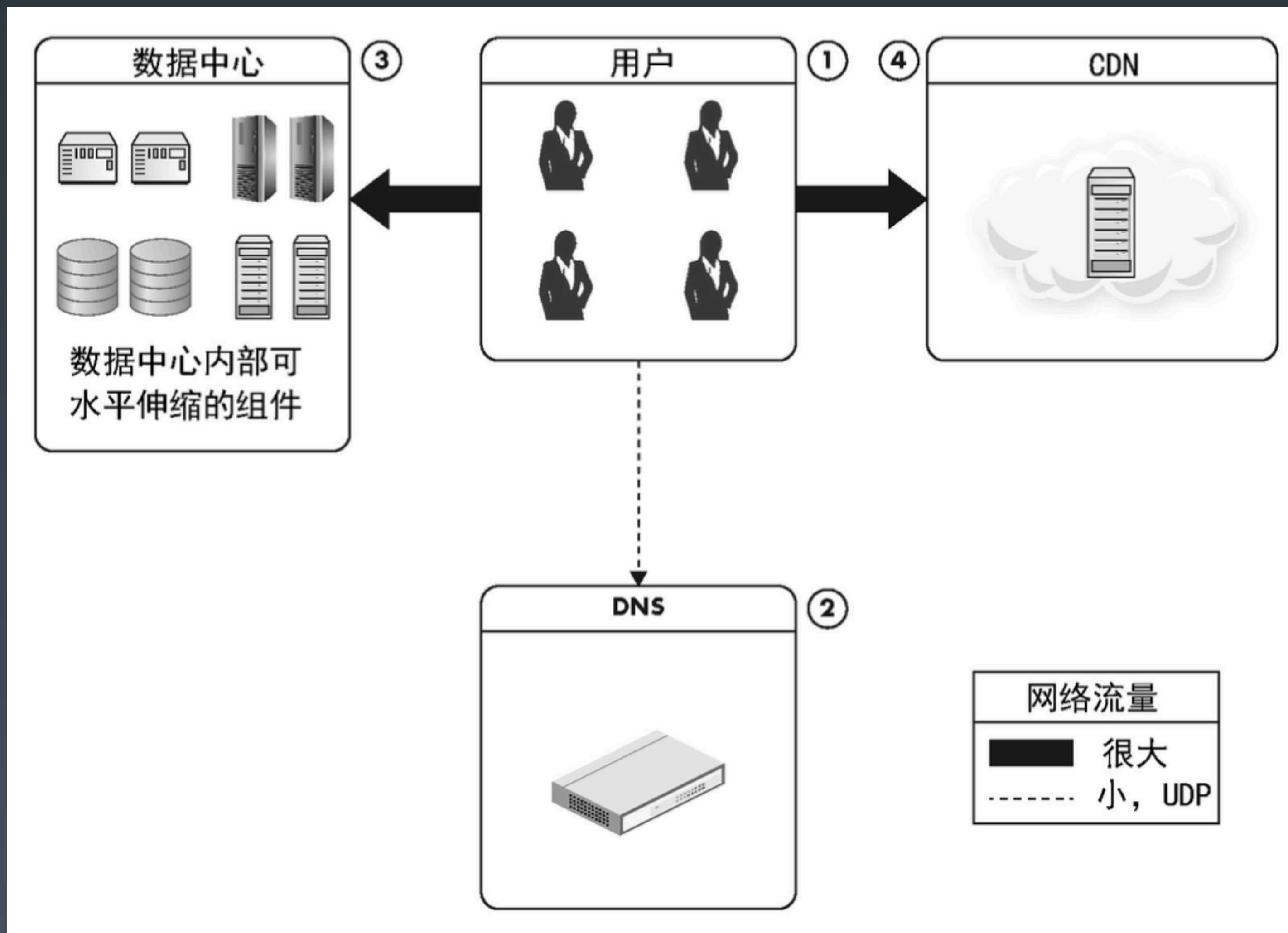


# 水平伸缩

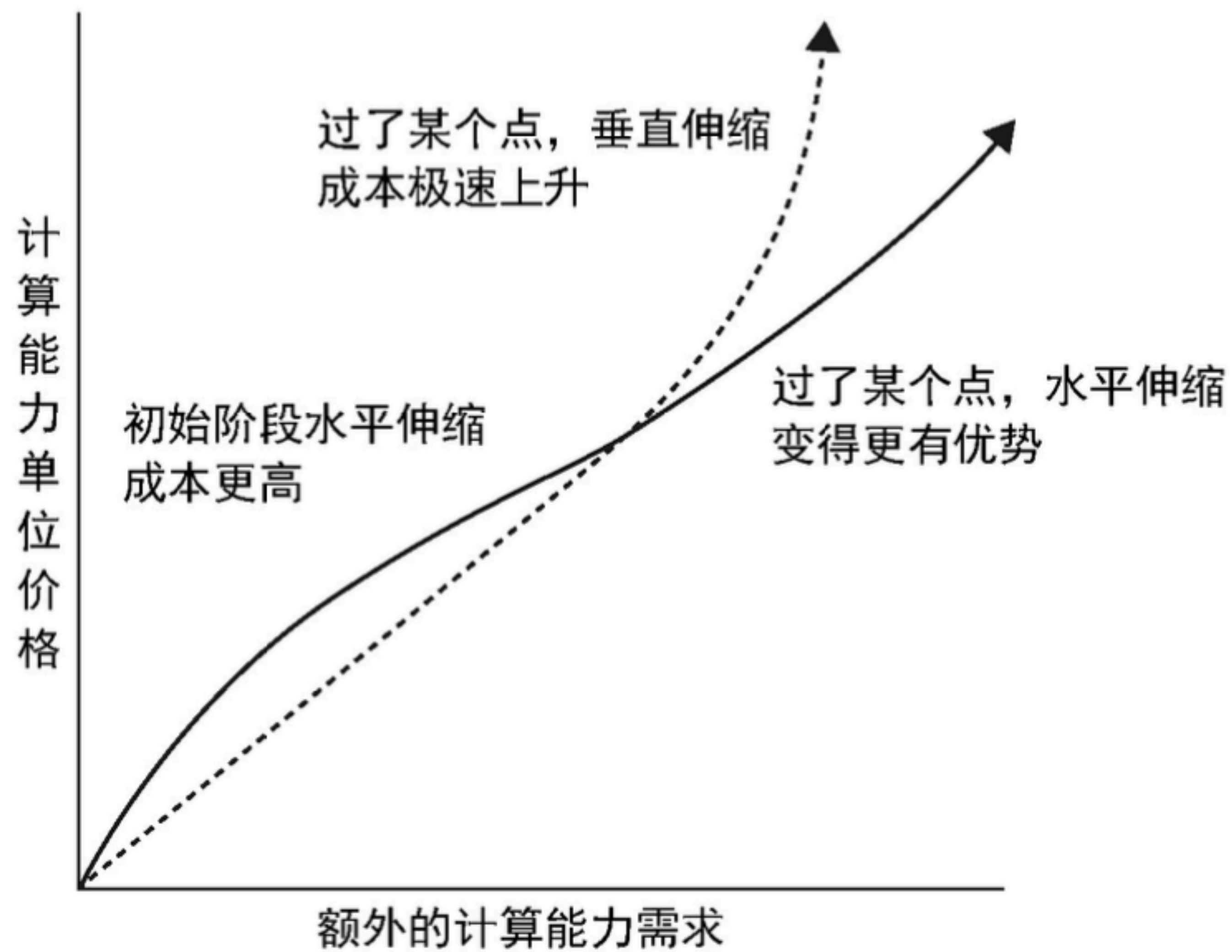
水平伸缩是指通过增加服务器提升计算能力的一类架构方法。

水平伸缩被认为是伸缩性的圣杯，水平伸缩可以克服垂直伸缩带来的单位计算成本随计算能力增加而迅速飙升的问题。

另外，水平伸缩总是可以增加更多服务器，这样，就不会像垂直伸缩那样遭遇到单台服务器的极限。

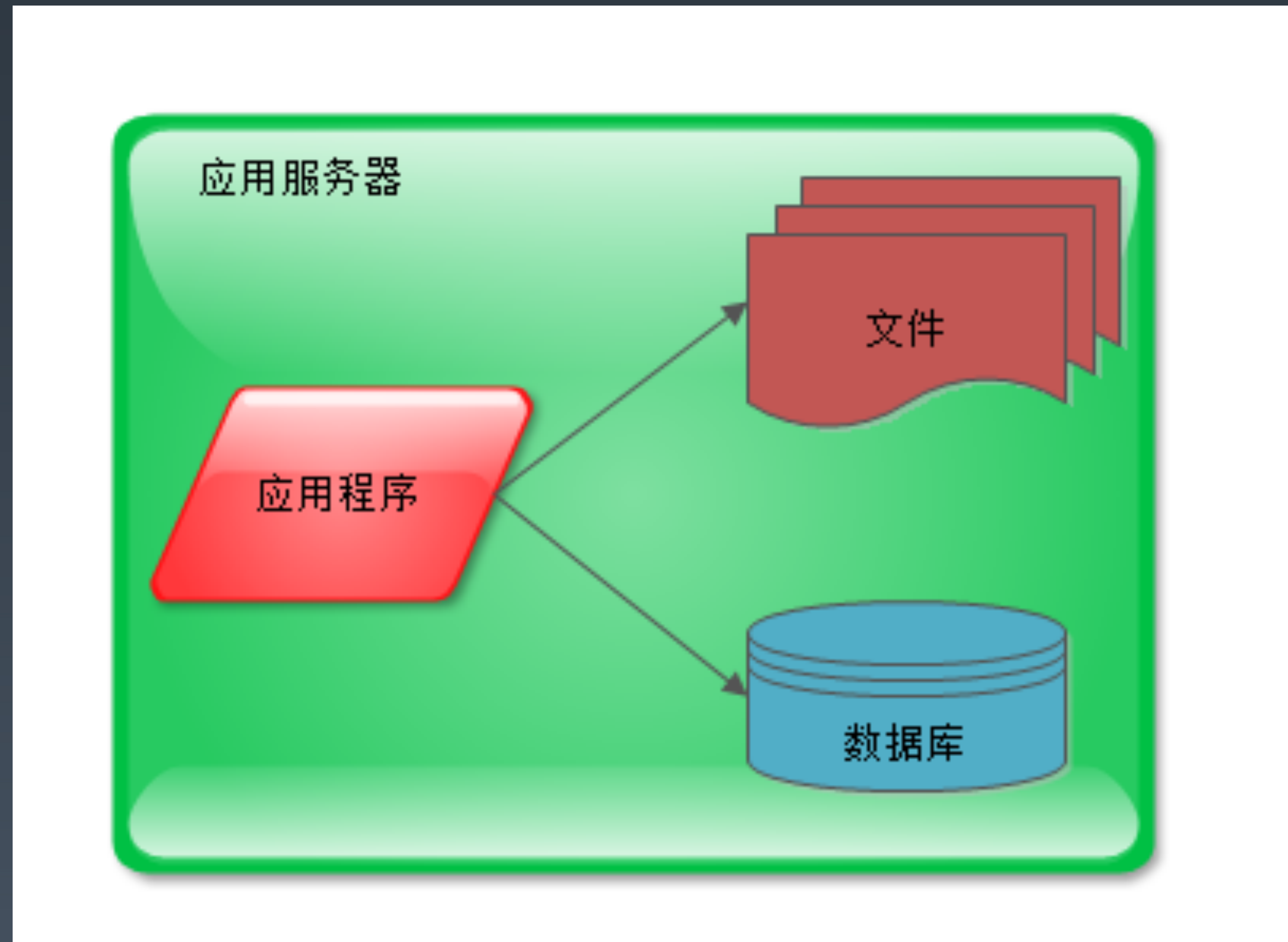




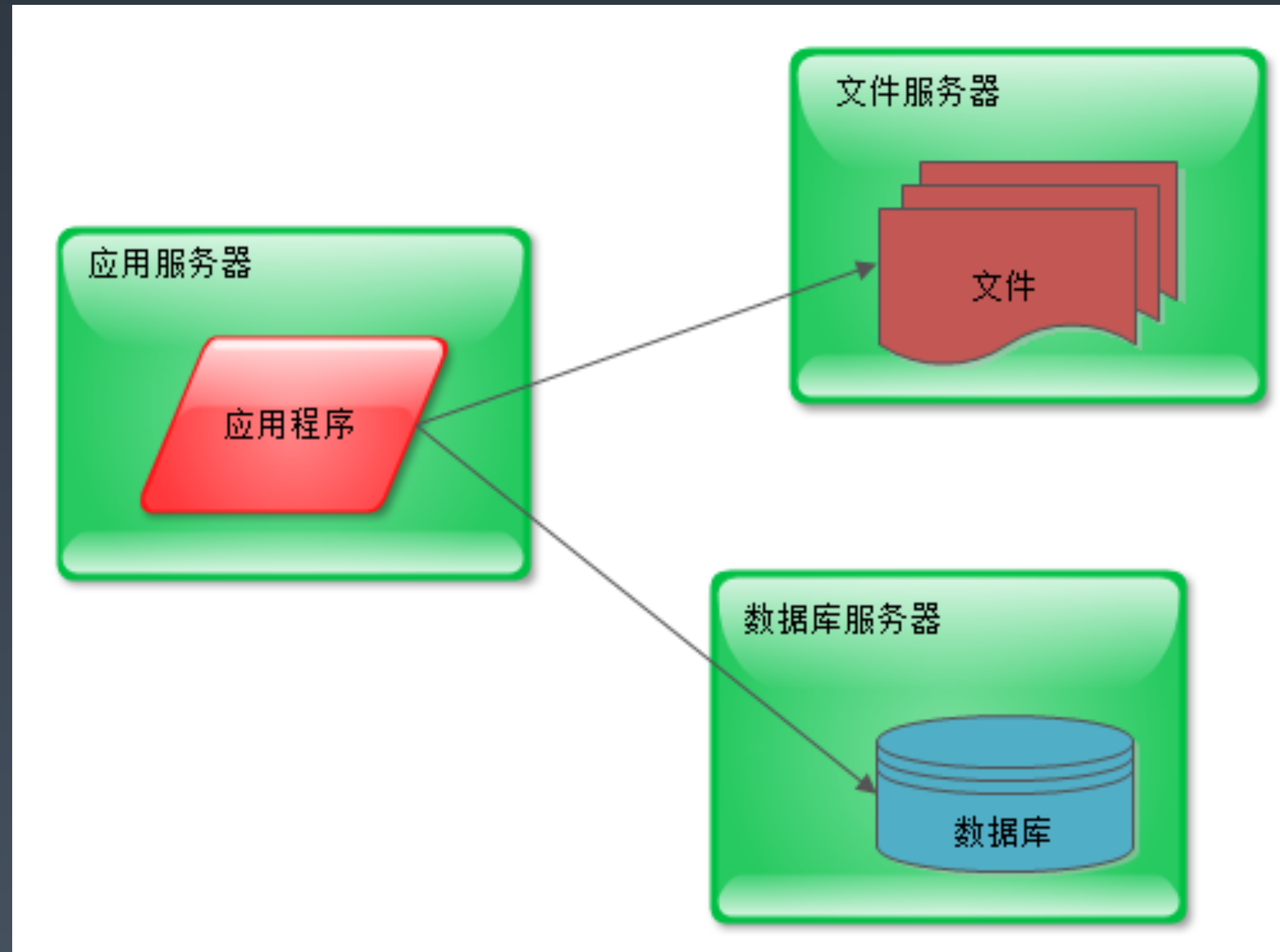


# 互联网架构演化

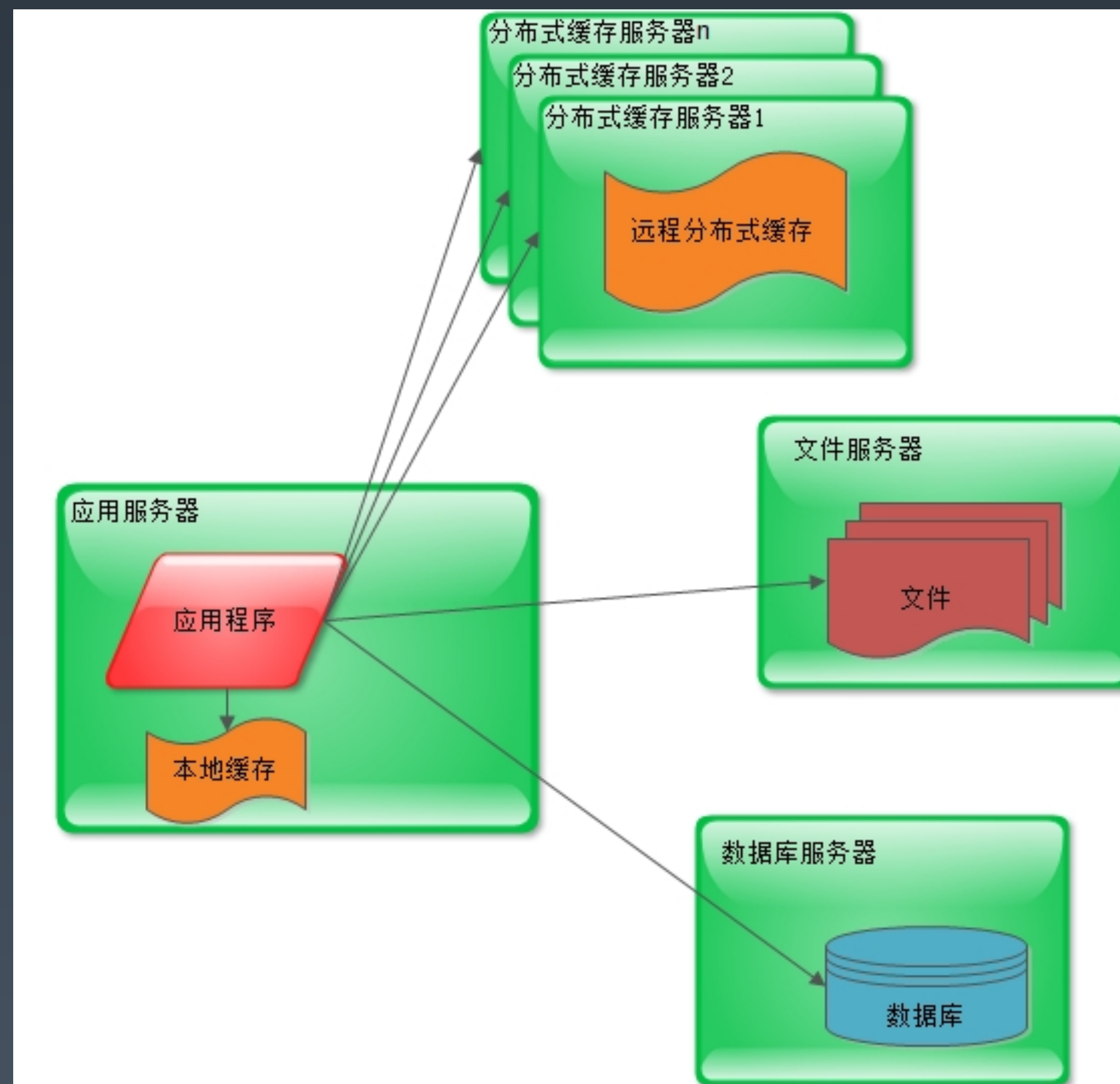
# 架构演化第零阶段：最简单的互联网应用架构



# 架构演化第一阶段：应用数据分离

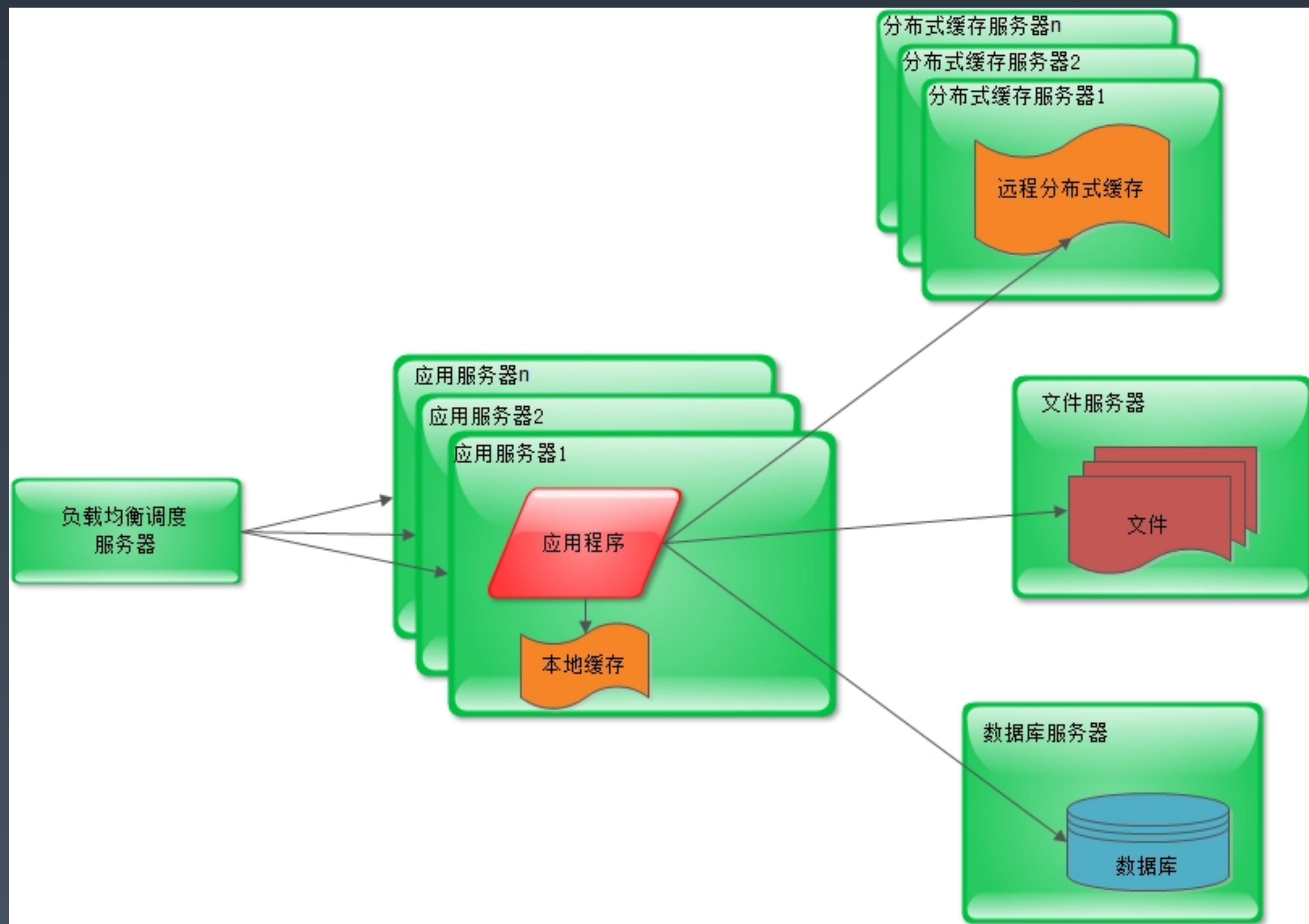


## 架构演化第二阶段：使用缓存改善系统性能

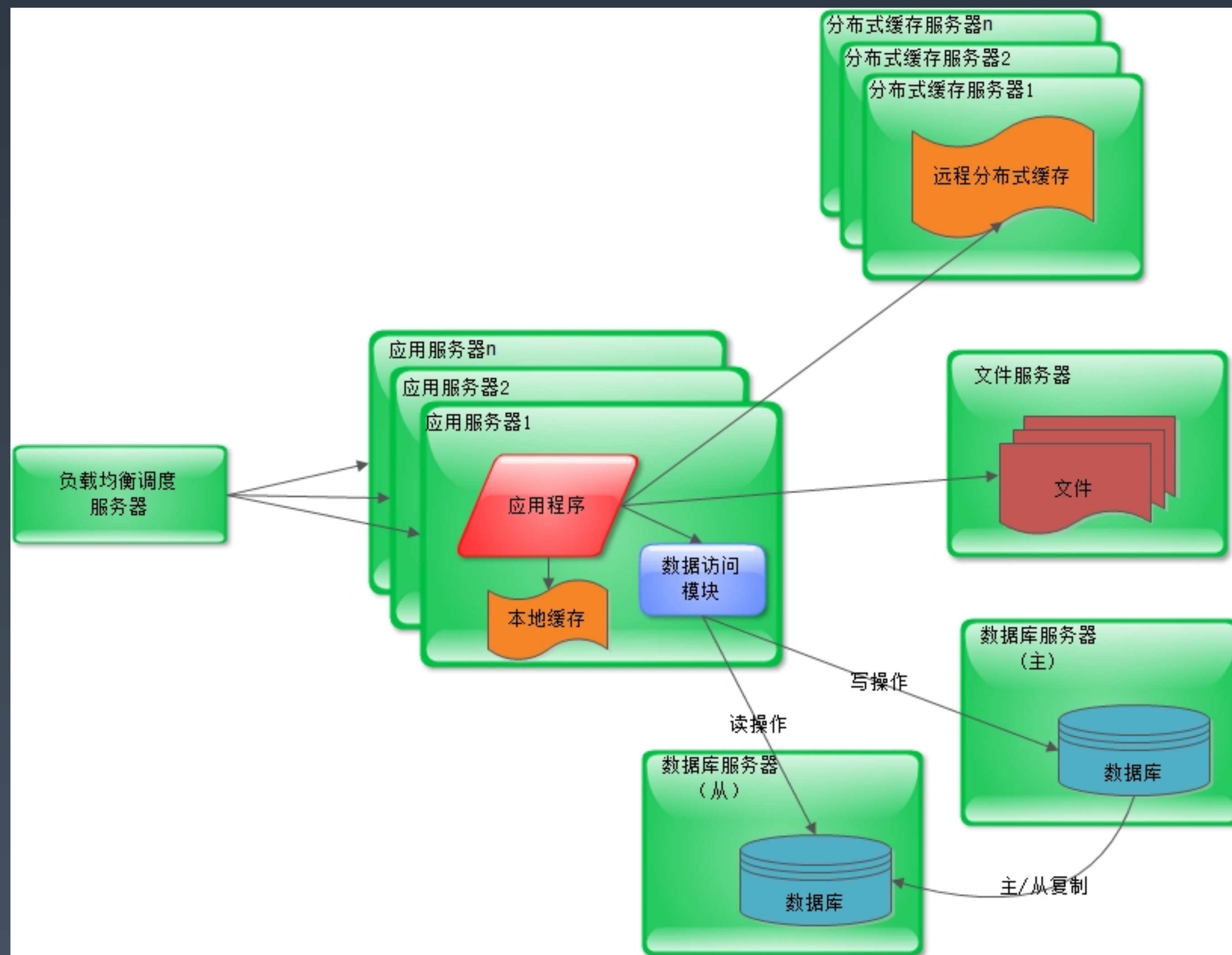




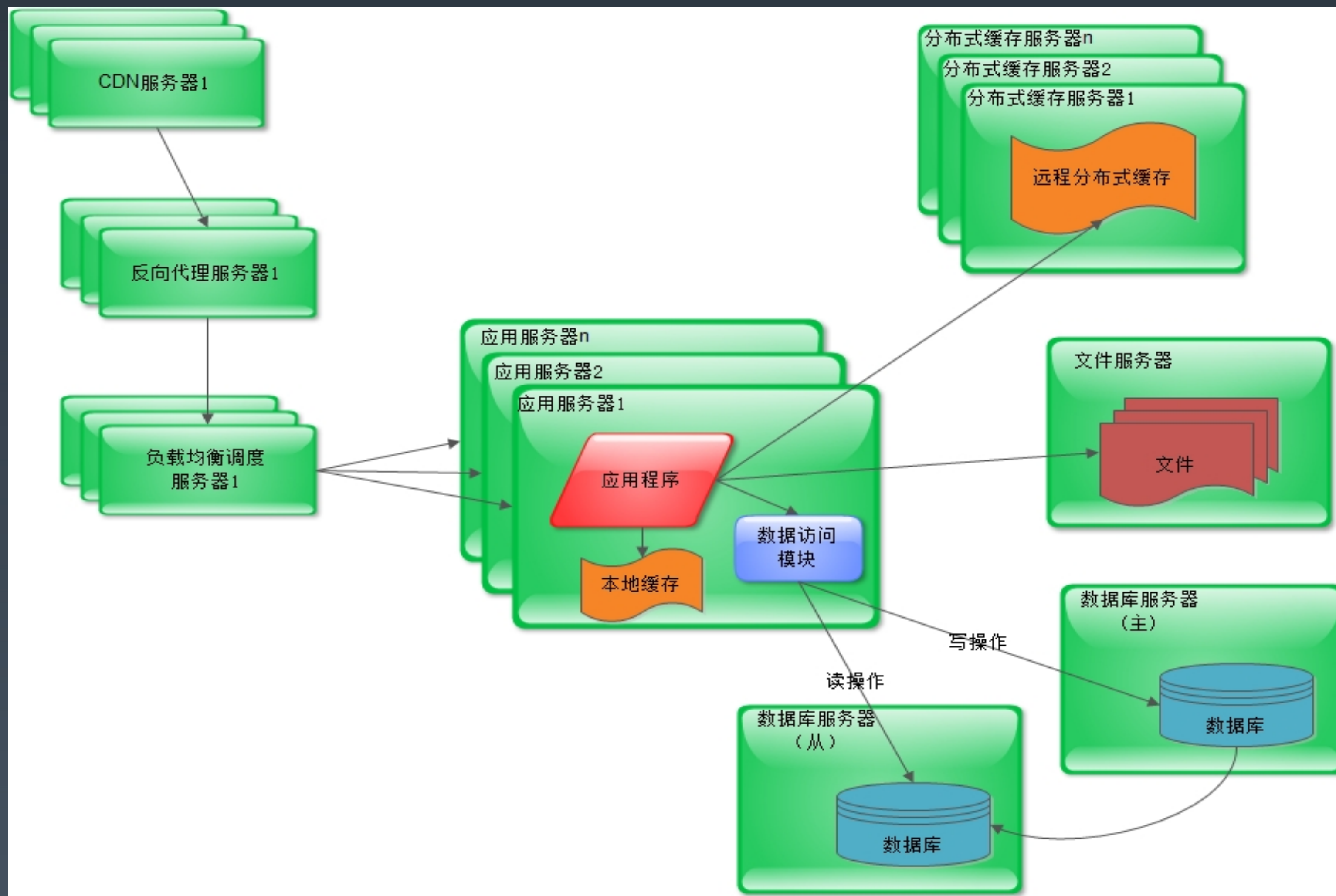
# 架构演化第三阶段：使用应用服务器集群改善系统的并发处理能力



# 架构演化第四阶段：数据库读写分离

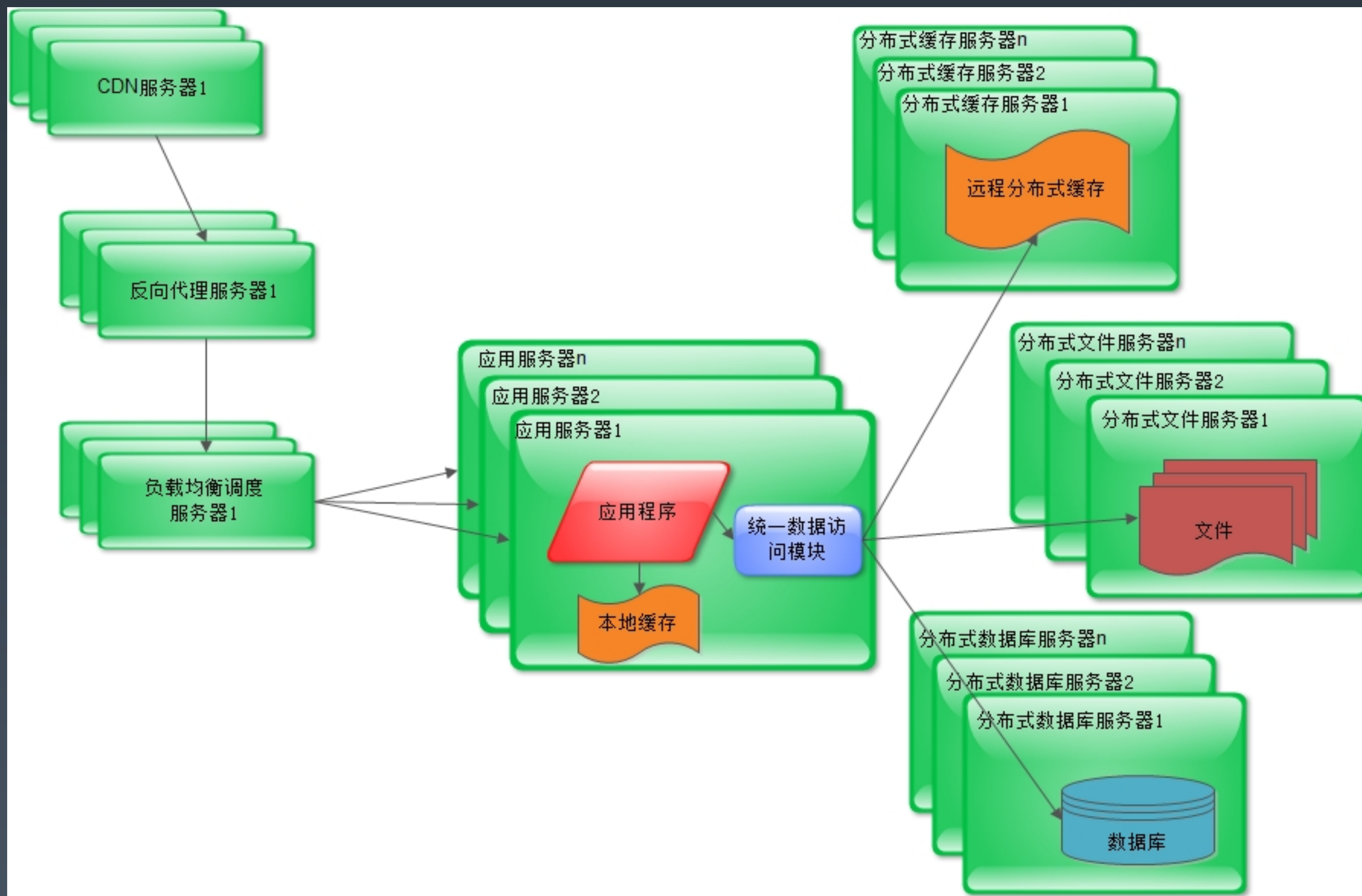


# 架构演化第五阶段：使用反向代理和 CDN 加速网站响应

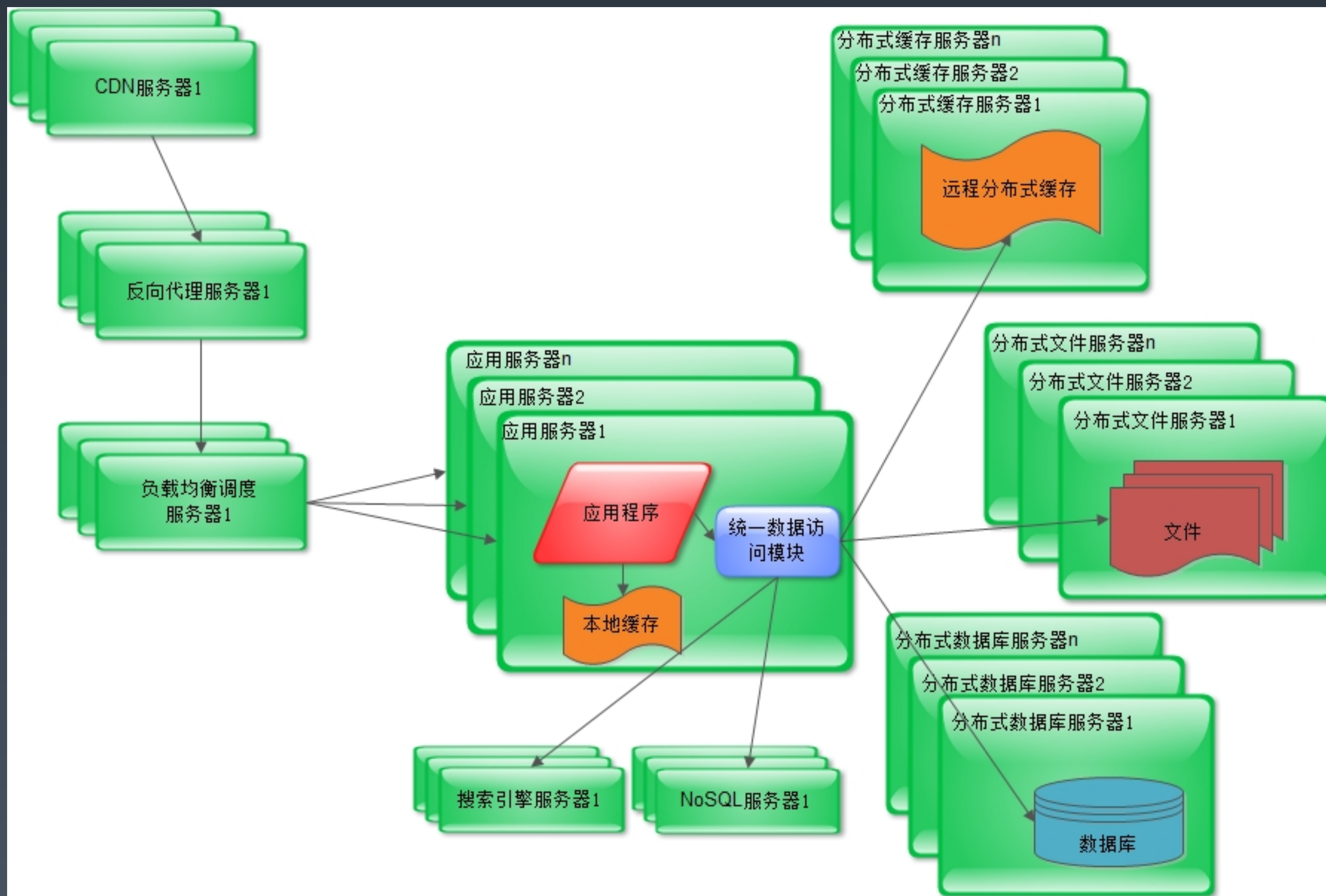




# 架构演化第六阶段：使用分布式文件系统和分布式数据库系统

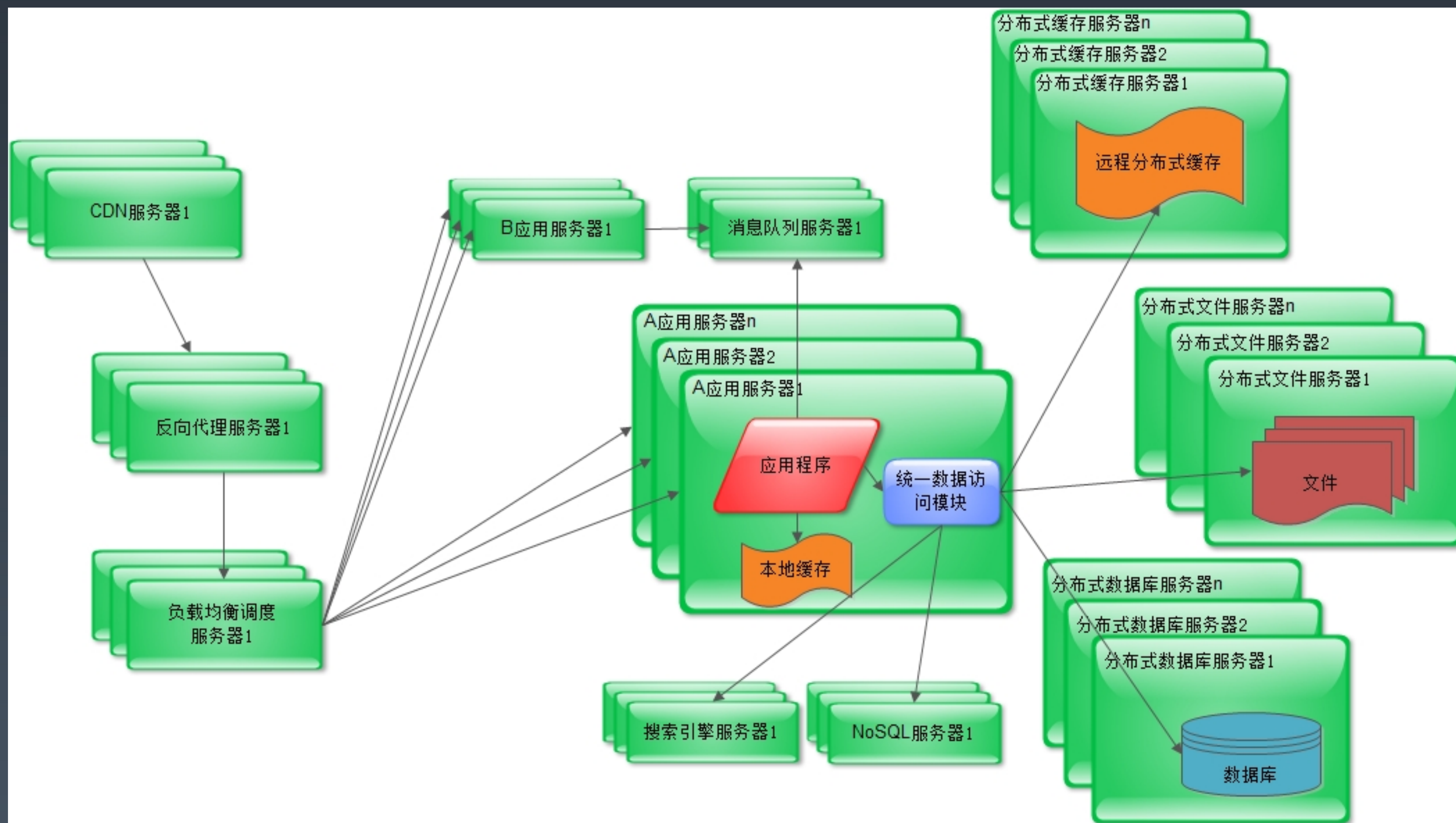


# 架构演化第七阶段：使用 NoSQL 和搜索引擎



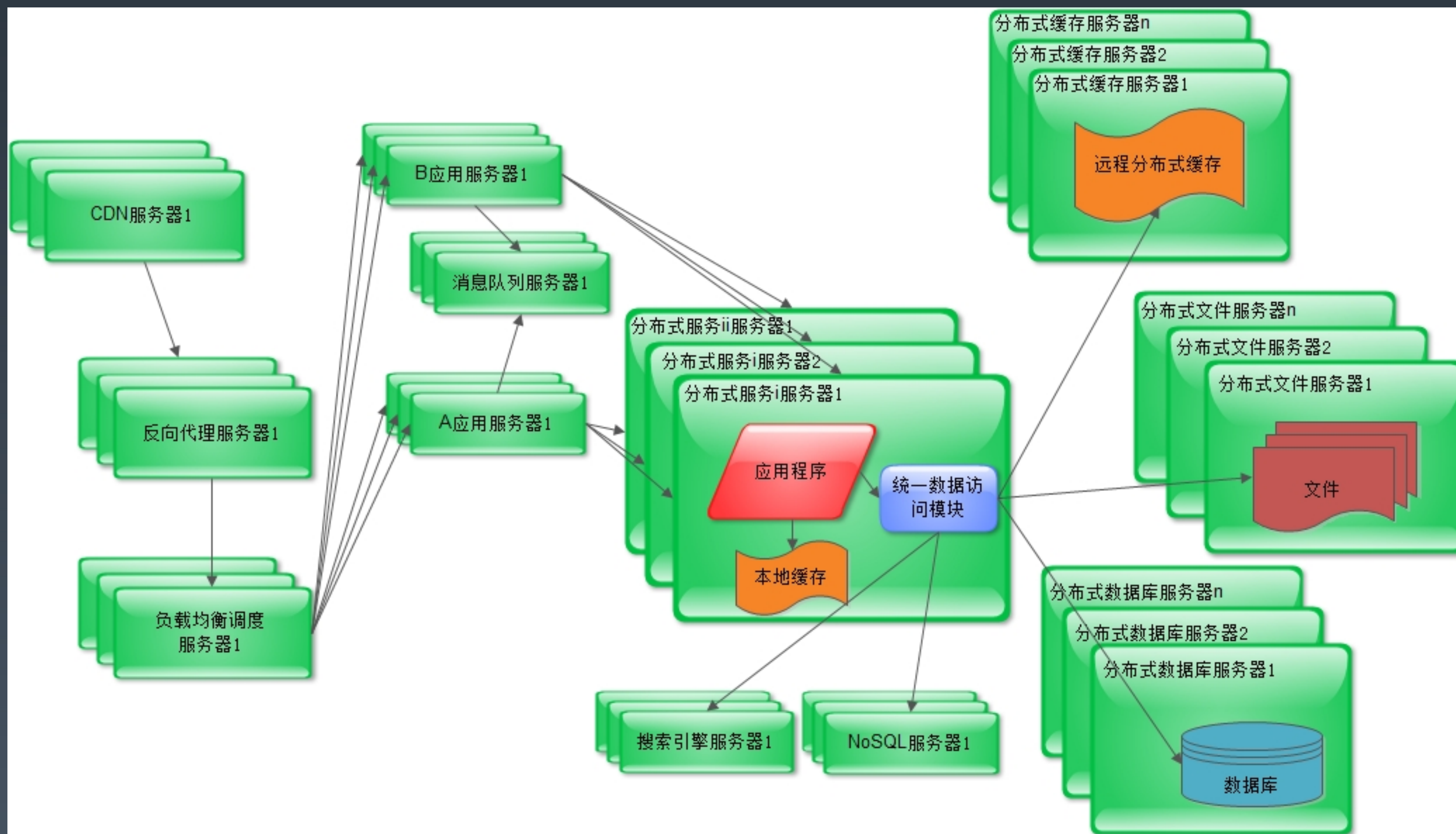


# 架构演化第八阶段：业务拆分





# 架构演化第九阶段：微服务及中台化



# 架构演化第十阶段 大数据与智能化

# 互联网架构模式

# 架构模式

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

模式的关键在于模式的可重复性，问题与场景的可重复性带来解决方案的可重复使用。

互联网架构模式就是试图去描述那些为解决互联网系统高性能、高可用、易扩展、可伸缩、安全等目标，被很多互联网应用重复使用的一些解决方案，这些解决方案是互联网软件系统的重要组成部分。

# 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成几个部分，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依赖和调用组成一个完整的系统。



# 分割

如果说分层是将软件在横向方面进行切分，那么分割就是在纵向方面对软件进行切分。

系统越大，功能越复杂，服务和数据处理的种类也越多，将这些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，提高网站的并发处理能力和功能扩展能力。

# 分布式

对于大型网站，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，即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。分布式意味着解决同样的问题可以使用更多的计算机，计算机越多，CPU、内存、存储资源也就越多，能够处理的并发访问和数据量就越大。

- 分布式应用和服务
- 分布式静态资源
- 分布式数据和存储
- 分布式计算

# 集群

使用分布式虽然已经将分层和分割后的模块独立部署，但是对于用户访问集中的模块，比如网站的首页，还需要将独立部署的服务器集群化，即多台服务器部署相同应用构成一个集群，通过负载均衡设备共同对外提供服务。

# 缓存

缓存就是将数据存放在距离计算最近的位置以加快处理速度。缓存是改善软件性能的第一手段，现代 CPU 越来越快的一个重要因素就是使用了更多的缓存，在复杂的软件设计中，缓存几乎无处不在。大型网站架构设计在很多方面都使用了缓存设计。

- CDN
- 反向代理
- 本地缓存
- 远程缓存

# 异步

计算机软件发展的一个重要目标和驱动力是降低软件耦合性。事物之间越少直接关系，那么就越少被彼此影响，越可以独立发展。大型网站架构中，系统解耦合的手段除了前面提到的分层、分割、分布等手段，还有一个重要手段是异步，就是将一个业务操作分成多个阶段，每个阶段之间通过共享数据而不是直接调用的方法进行协作。

- 提高系统可用性
- 加快网站响应速度
- 消除并发访问高峰

# 冗余

互联网应用需要7×24小时连续运行，但是服务器总有可能可能会出现故障，特别是服务器规模比较大的时候，服务器宕机是必然事件。要想保证在服务器宕机的情况下网站依然可以继续服务，数据不会丢失，就需要一定程度的服务器冗余运行，数据冗余备份。

# 自动化

在无人值守的情况下网站可以正常运行，一切都可以自动化是网站的理想状态。目前互联网的自动化架构设计主要集中在运维方面。

# 安全

互联网的开放特性使得其从诞生起就面对巨大的安全挑战，网站在安全架构方面也积累了许多模式：通过密码和手机校验码进行身份认证；登录、交易等操作需要对网络通讯进行加密，网站服务器上存储的敏感数据如用户信息等也进行加密处理；为了防止机器人程序滥用网络资源供给网站，网站使用验证码进行识别；对于常见的用于攻击网站的 XSS 攻击，SQL 注入，进行编码转换等相应处理；对于垃圾信息、敏感信息进行过滤；对转账交易等重要操作根据交易模式和交易信息进行风险控制。



# 互联网系统架构核心要素

## 如何衡量一个系统的架构设计

# 高性能

性能是互联网的一个重要指标，除非是没得选择，否则用户无法忍受一个响应缓慢的应用。一个打开缓慢应用会导致严重的用户流失，很多时候系统性能问题是系统架构升级优化的触发器。可以说性能是互联网系统架构设计的一个重要方面，任何架构设计方案都必须考虑可能会带来的性能问题。

也正是因为性能问题几乎无处不在，所以优化网站性能的手段也非常多，从用户端到数据库，从代码到机房部署，影响用户请求的所有环节都可以进行性能优化。

# 高可用

因为互联网分布式系统使用的服务器硬件通常是普通的商用服务器，这些服务器的设计目标本身并不保证高可用，也就是说，很有可能会出现服务器硬件故障，也就是俗称的服务器宕机。大型互联网系统通常都会有上万台服务器，每天都必定会有一些服务器宕机，因此系统高可用架构设计的前提是必然会出现服务器宕机，而高可用设计的目标就是当服务器宕机的时候，服务或者应用依然可用。

系统高可用的主要手段是冗余，应用部署在多台服务器上同时提供访问，数据存储在多台服务器上互相备份，任何一台服务器宕机都不会影响应用的整体可用，也不会导致数据丢失。

# 可伸缩

大型互联网应用通过集群的方式将多台服务器组成一个整体共同提供服务。所谓伸缩性是指通过不断向集群中加入服务器的手段来缓解不断上升的用户并发访问压力和不断增长的数据存储需求。

衡量架构伸缩性的主要标准就是是否可以用多台服务器构建集群，是否容易向集群中添加新的服务器。加入新的服务器后是否可以提供和原来的服务器无差别的服务。集群中可容纳的总的服务器数量是否有限制。

# 可扩展

不同于其他架构要素主要关注非功能性需求，扩展性架构直接关注系统的功能需求。互联网应用快速发展，功能不断扩展，如何设计系统的架构使其能够快速响应需求变化，是系统可扩展架构主要的目的。

衡量系统架构扩展性好坏的主要标准就是在系统增加新的业务产品时，是否可以实现对现有产品透明无影响，不需要任何改动或者很少改动既有业务功能就可以上线新产品。不同产品之间是否很少耦合，一个产品改动对其他产品无影响，其他产品和功能不需要受牵连进行改动。

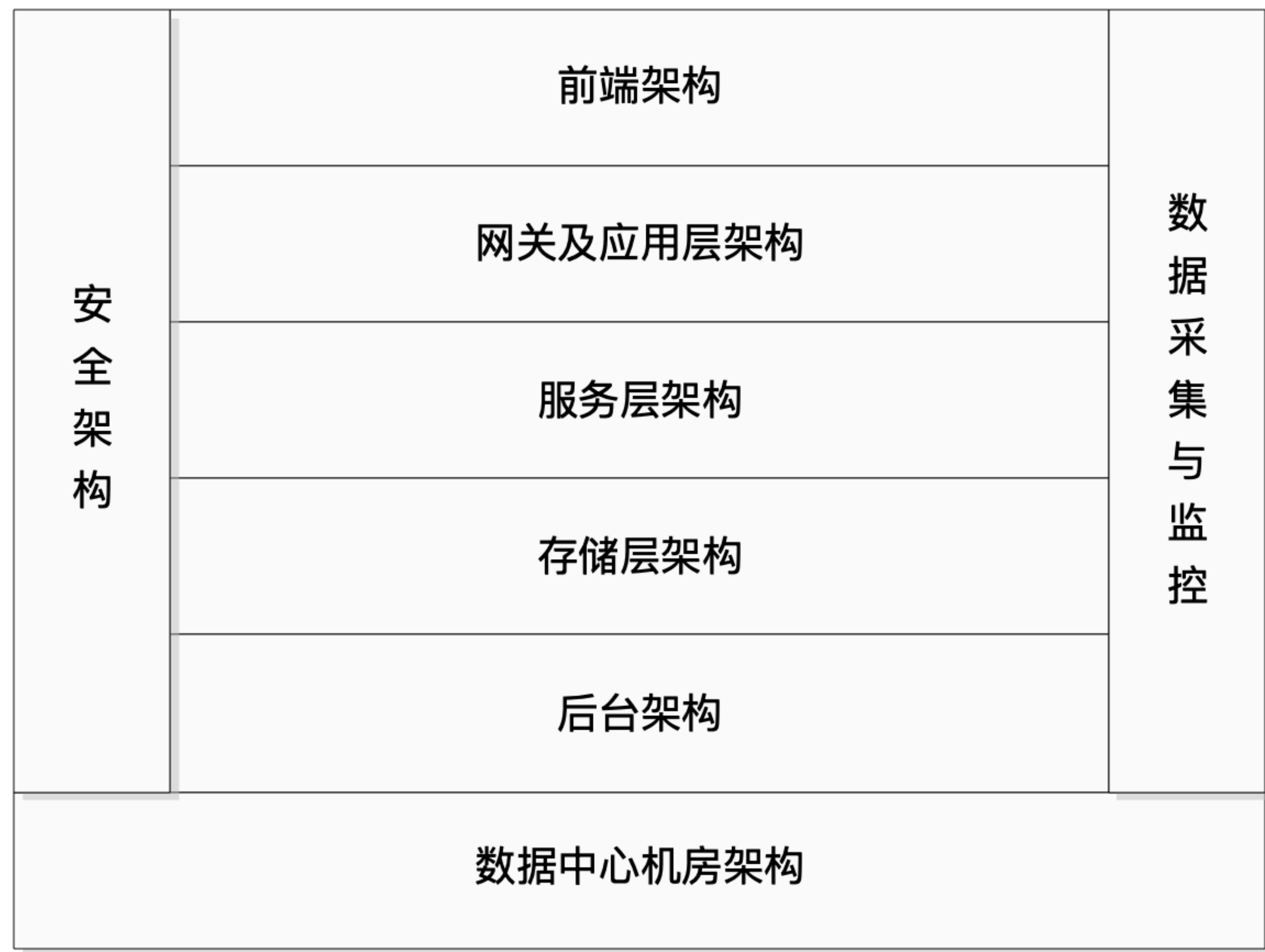
可扩展架构的主要手段是事件驱动架构和分布式服务。

# 安全

互联网是开放的，任何人在任何地方都可以访问系统。系统的安全架构就是保护系统不受恶意访问和攻击，保护网站的重要数据不被窃取。

衡量系统安全架构的标准就是针对现存和潜在的各种攻击与窃密手段，是否有可靠的应对策略。

# 互联网架构技术一览





# 前端架构

- App 及 Web 开发技术
- 浏览器及 HTTP 优化技术
- CDN
- 动静分离
- 图片服务
- 反向代理
- DNS

# 网关及应用层架构

- 网关架构
- 负载均衡
- 动态页面静态化
- 业务拆分

# 服务层架构

- 微服务框架
- 分布式消息队列
- 分布式缓存
- 分布式一致性（锁）服务

# 存储层架构

- 分布式文件
- 分布式关系数据库
- NoSQL 数据库

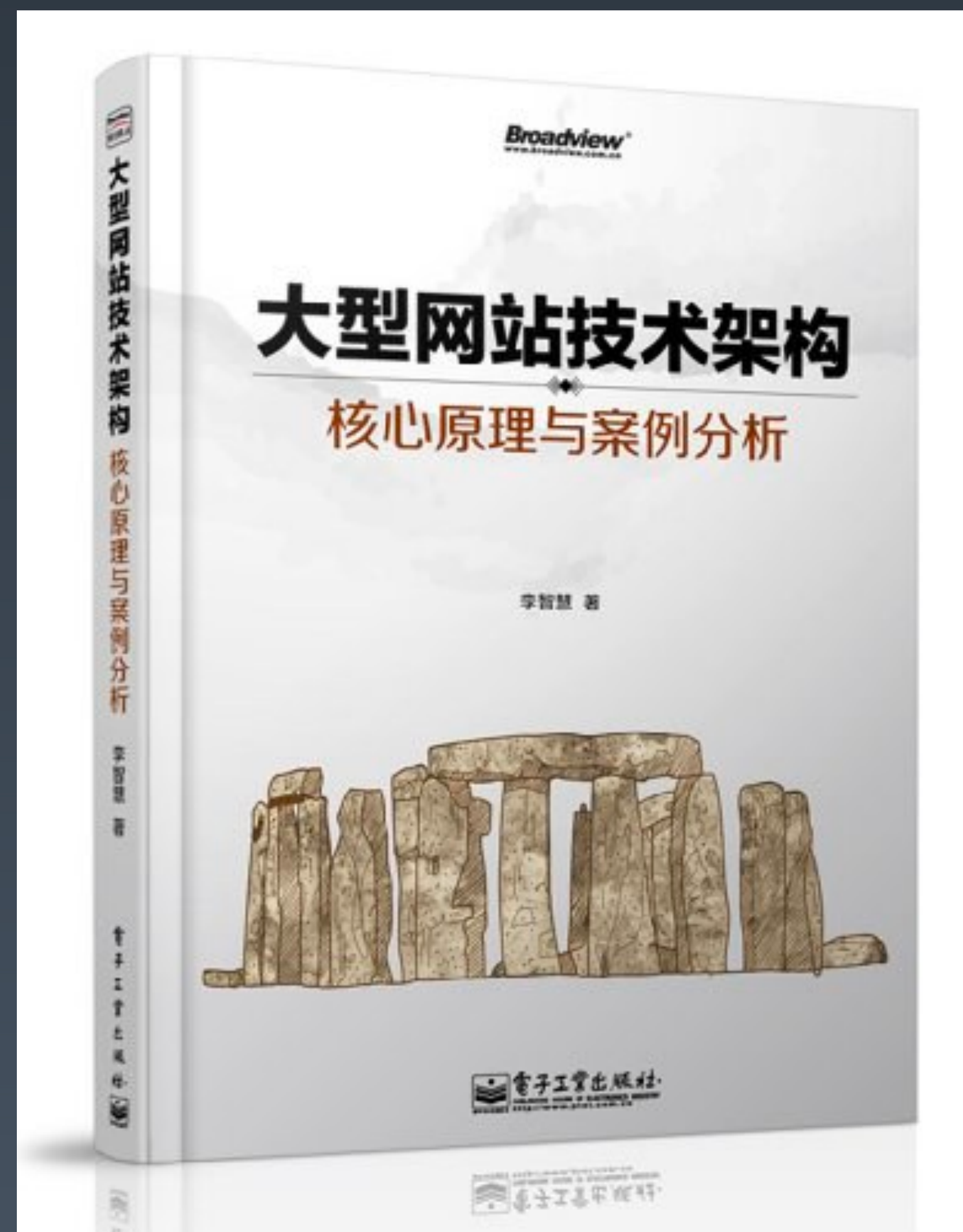
# 后台架构

- 大数据平台
- 搜索引擎
- 推荐引擎
- 数据仓库

# 运维与安全

- 数据采集与展示
- 数据监控与报警
- 攻击与防护
- 数据加密与解密

# 推荐书籍



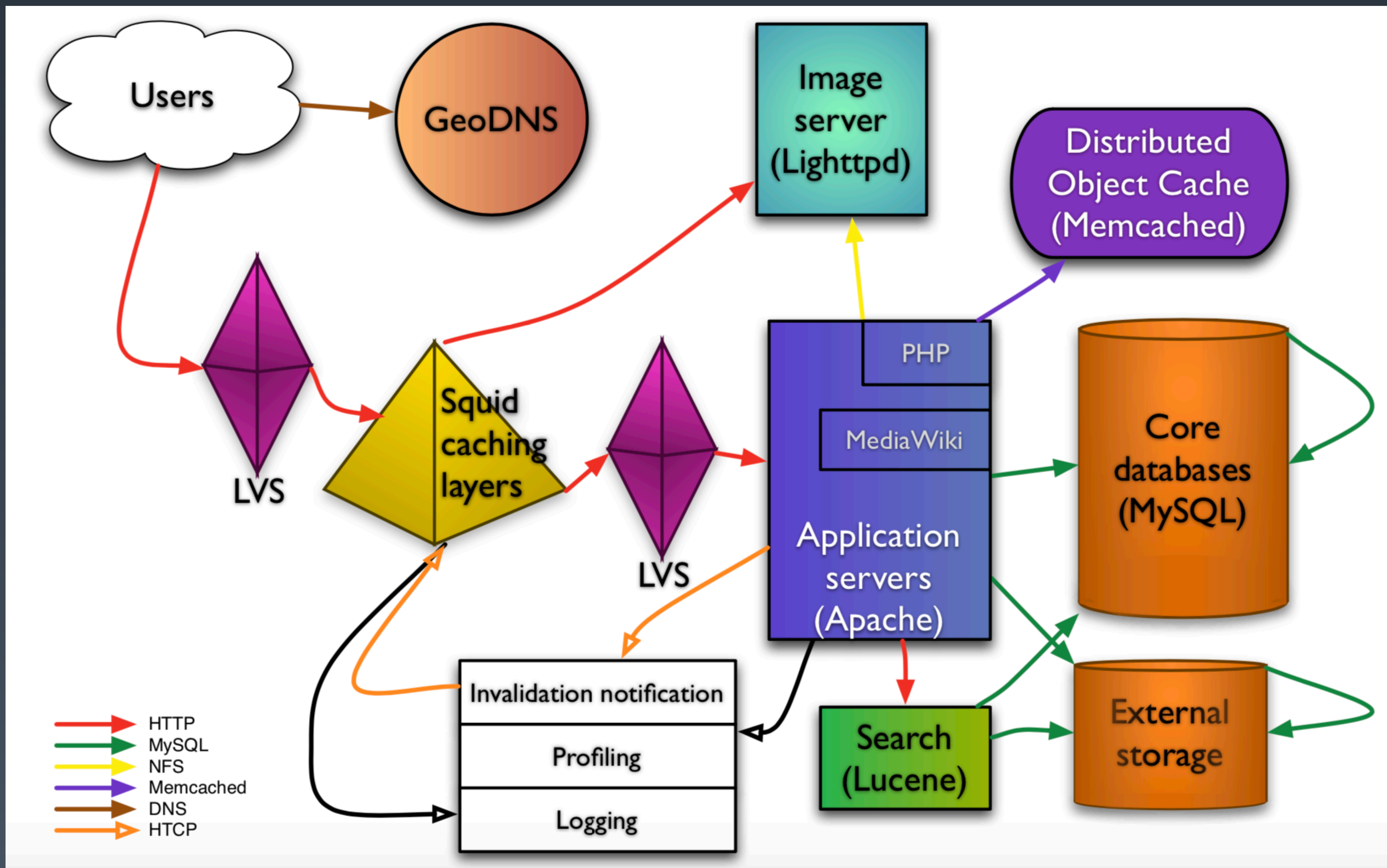
THANKS! |  极客大学



# 极客大学架构师训练营

李智慧

# 维基百科技术架构



THANKS! |  极客大学



## 《架构师训练营》