



上海交通大学
SHANGHAI JIAO TONG UNIVERSITY

淘宝的架构演进

CASE STUDY

沈备军

bjshen@sjtu.edu.cn

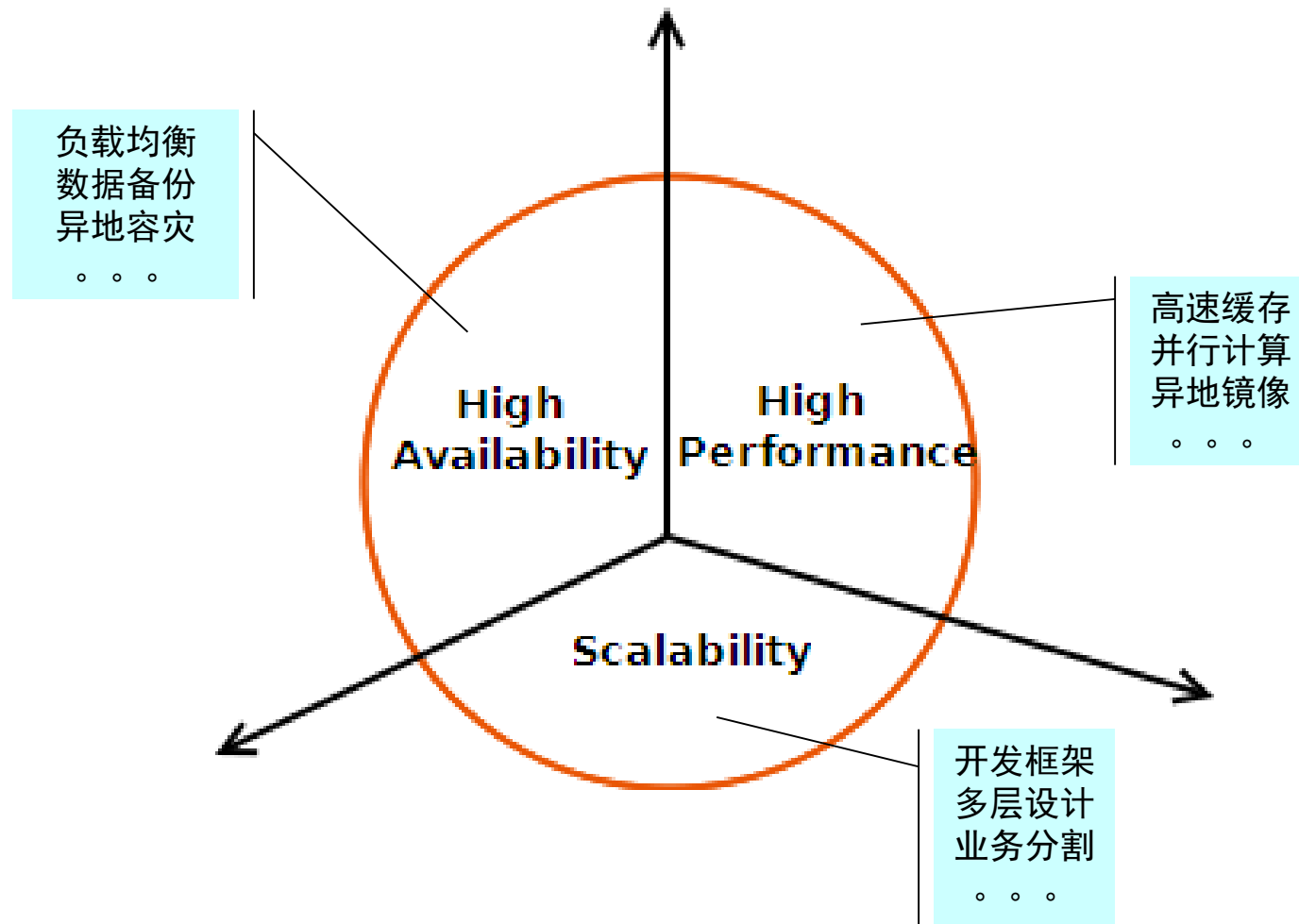


案例研究：淘宝的架构演进

□ 大型网站：日均流量IP > 1,000,000

网站	日均流量[IP/PV]	 Alexa The Web Information Company
www.hao123.com	IP ≈ 5,972,587 PV ≈ 9,376,962	
www.facebook.com	IP ≈ 229,680,000 PV ≈ 2,955,981,600	
www.sina.com.cn	IP ≈ 25,680,000 PV ≈ 222,132,000	
www.tianya.cn	IP ≈ 5,532,000 PV ≈ 25,723,800	
<i>www.pingan.com</i>	<i>IP ≈ 300,000 PV ≈ 747,000</i>	

大型网站架构的目标与挑战



🤔 每个目标后面面临着技术、设计、维护等诸多方面的挑战。
而目标本身的期望值也会根据实际情况进行调整，这也意味着**网站架构建设是个不断调整的过程。**

大型网站架构演变

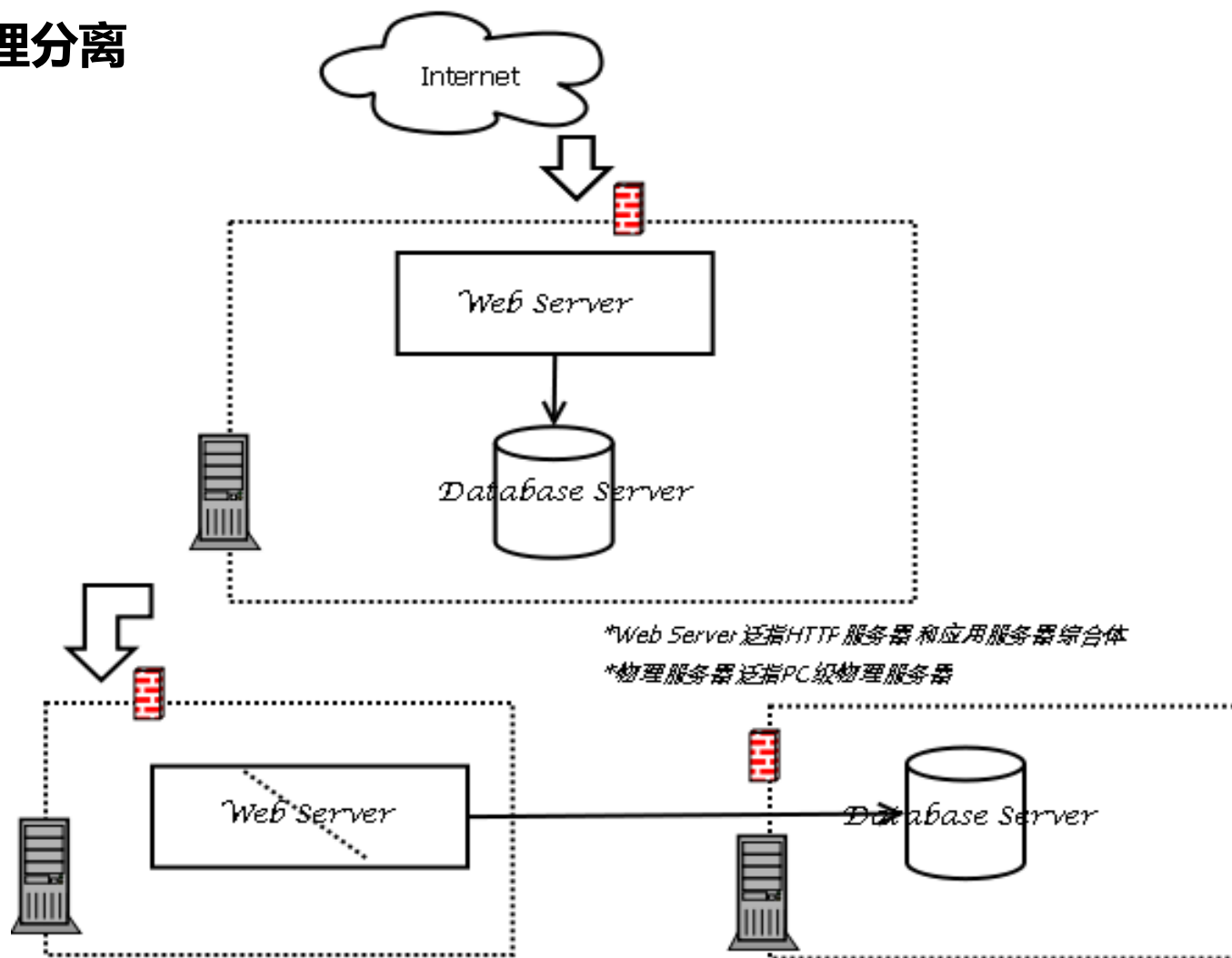
■ [Step1] Web动静静态资源分离及其与DB物理分离

■ 优点：“简单”、安全性提高

■ 缺点：存在单点，谈不上高可用性

■ 技术点：

- ✓ Web Server动/静态资源分离
- ✓ Web Server (Apache\Nginx\IIS\JBoss...)
- ✓ Database Server (Mysql\Oracle\Redis...)



大型网站架构演变

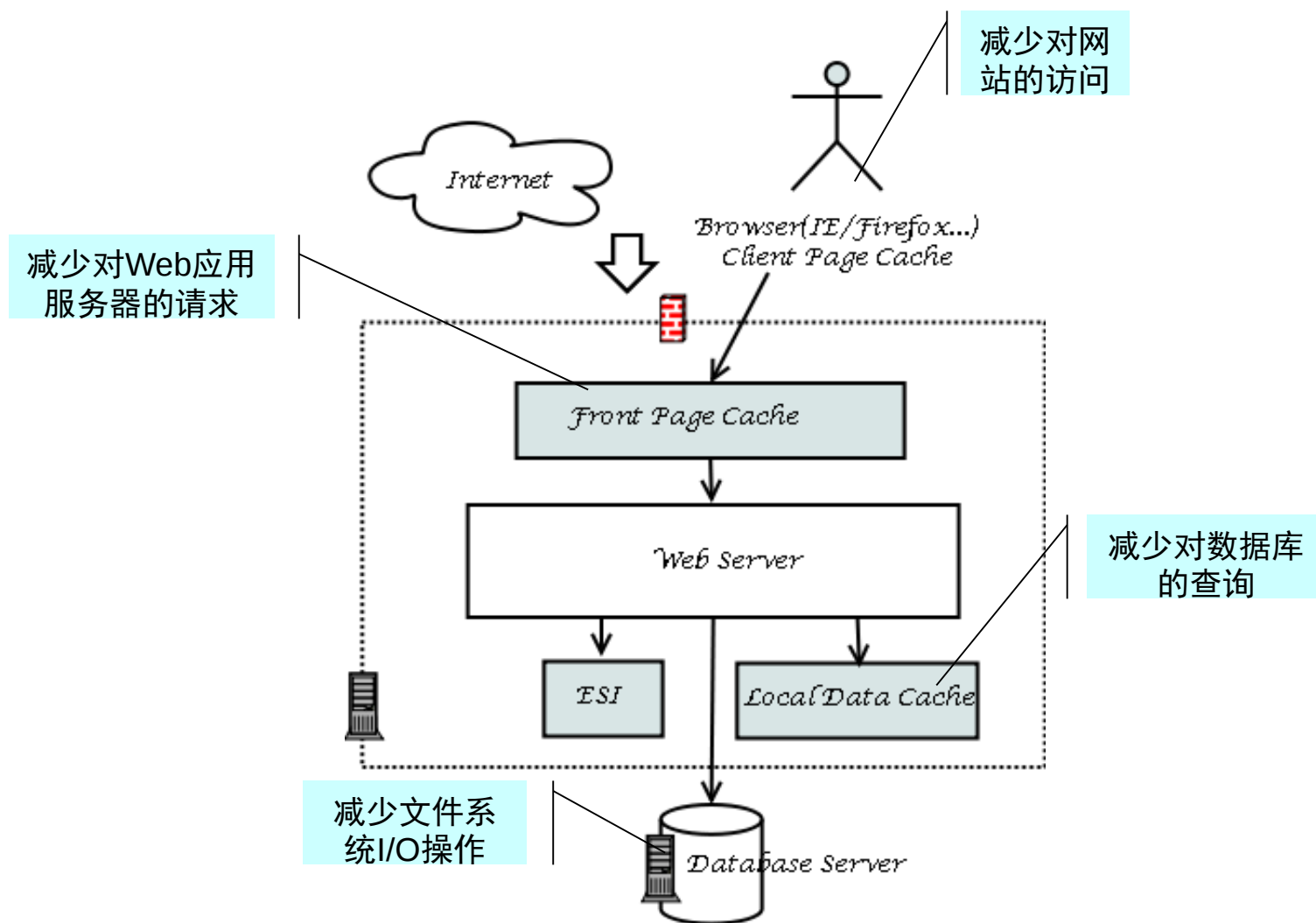
■[Step2]采取缓存处理

■优点：简单有效、维护方便

■缺点：依然存在单点

■技术点：

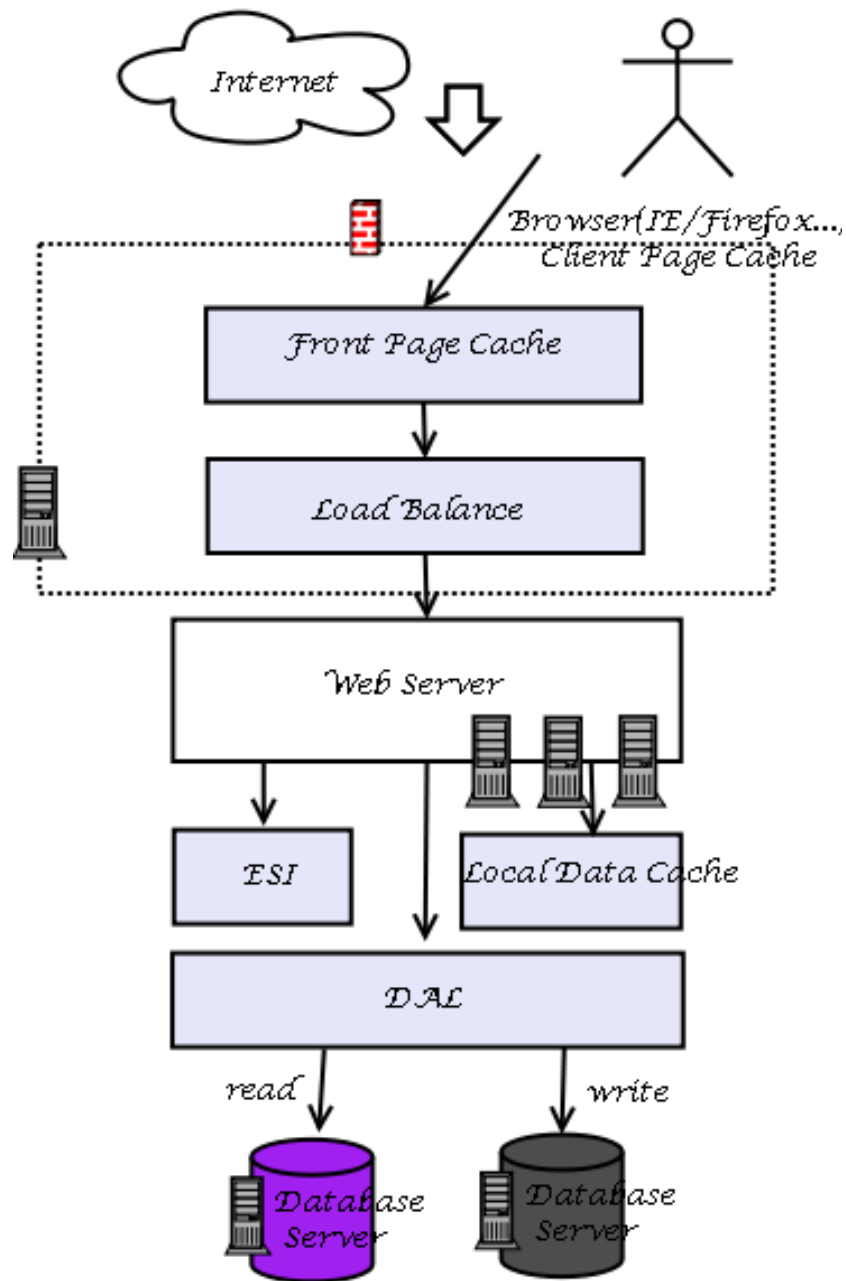
- ✓客户端（浏览器）缓存
- ✓前端页面缓存
- ✓页面片段缓存
- ✓本地数据缓存/数据库缓存



大型网站架构演变

■ [Step3]增加机器做HA、数据库读写分离

- 优点：增加服务器和HA机制，系统性能及可用性得到保证
- 缺点：读写分离，增加程序难度，架构变复杂，维护难度增加
- 技术点：
 - ✓ 负载均衡
 - ✓ DAL
 - ✓ 数据库读写分离



大型网站架构演变

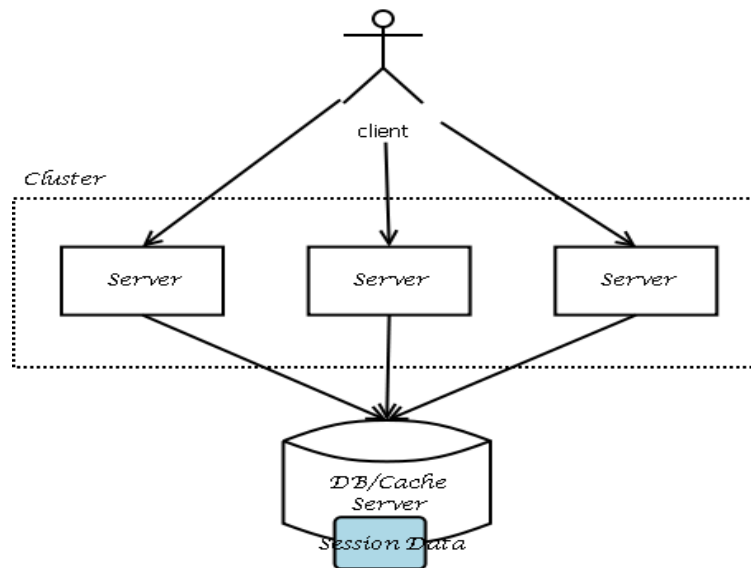
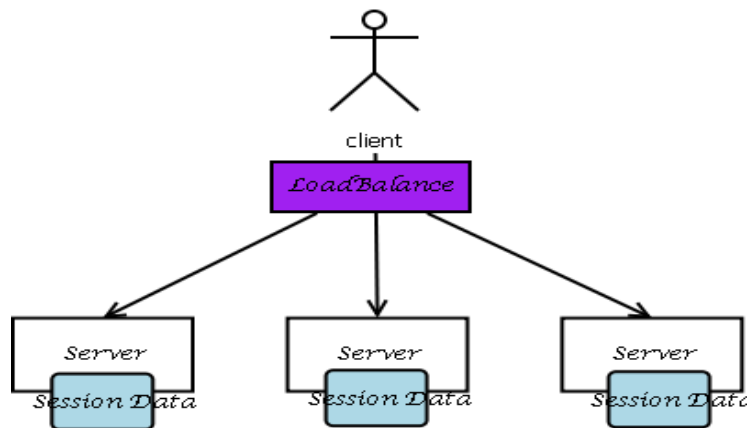
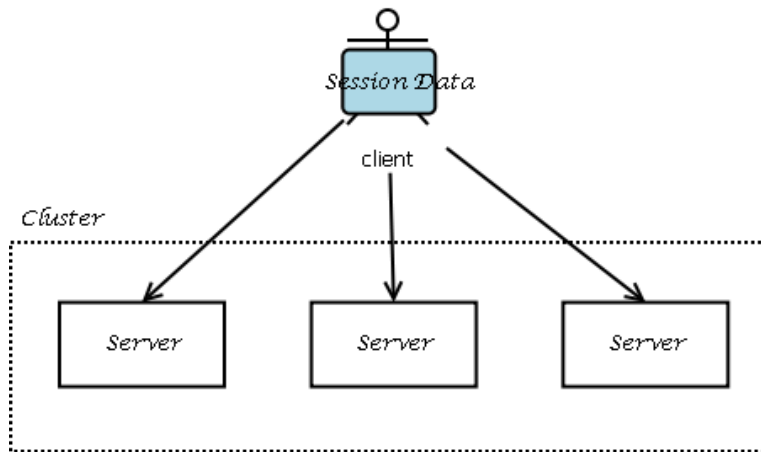
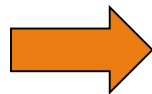
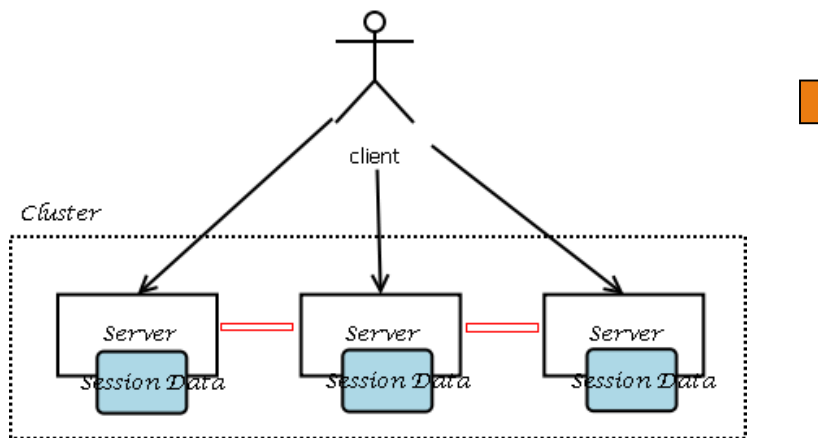
■[Step3]技术点—负载均衡

类型	说明
DNS负载均衡	实现简单，有Cache，缺乏灵活性，但对分区域（如构建CDN方案）访问简单有效
反向代理软件	HAProxy、Nginx、Apache、Lighttpd等
硬件产品	F5、NetScaler等
LVS(Linux Virtual Server)	http://www.linuxvirtualserver.org/
SMART Client	自己写代码某些情况下简单有效



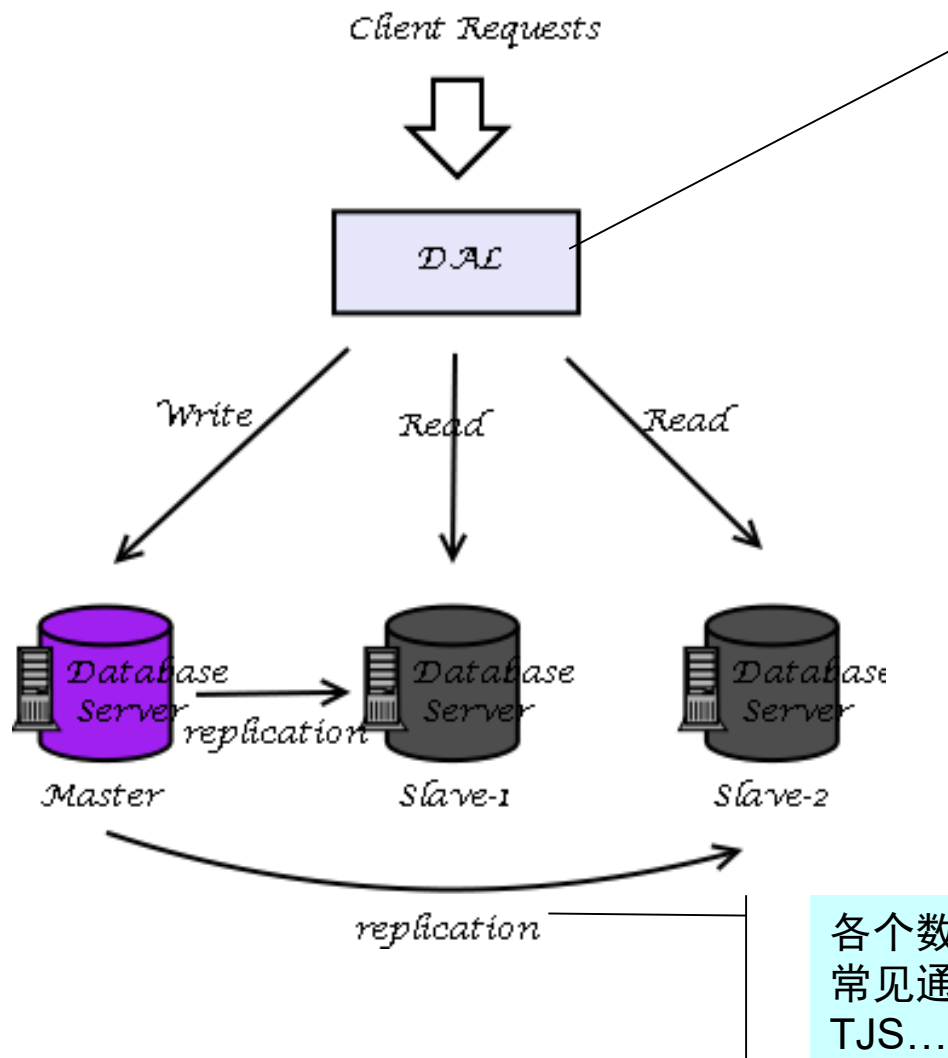
大型网站架构演变

■无共享架构 (Share Nothing Architecture)



大型网站架构演变

■[Step3]技术点—数据库读写分离及DAL



- 读写分离逻辑分批
- 负载均衡
- 失效转移 (failover)
- 数据库分区透明支持
- 两大实现模式：独立Proxy服务器；
单独API库文件

各个数据库厂商都有自己复制方案
常见通用方案：ETL、GoldenGate
TJS...

大型网站架构演变

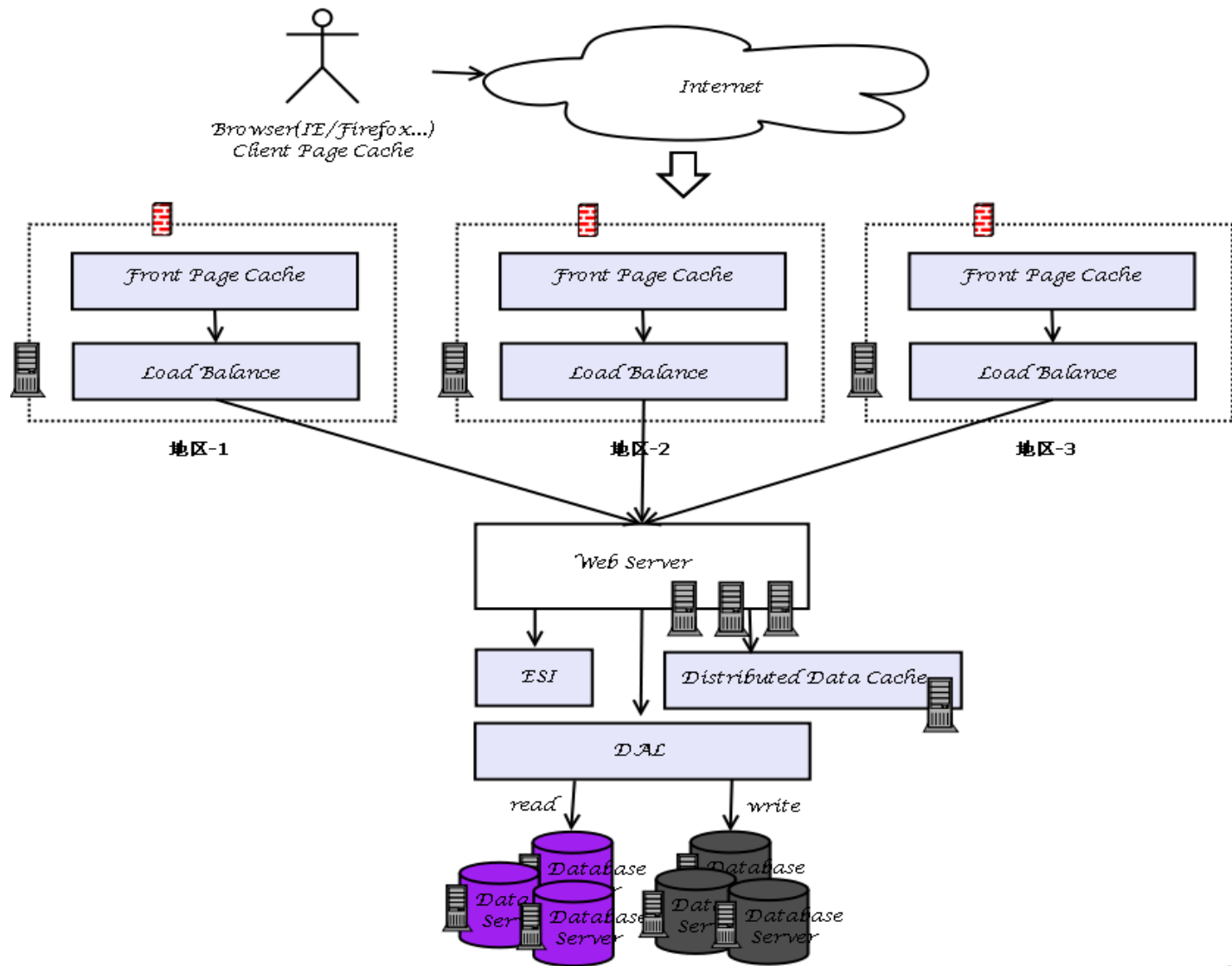
■[Step4]CDN、分布式缓存、分库

■优点：异地缓存有效解决不同地方用户访问过慢的问题；分库策略带来网站性能整体提升

■缺点：成本大幅增加，架构进一步复杂化，也维护难度进一步增大，架构开始臃肿了

■技术点：

- ✓ CDN
- ✓ 分布式缓存
- ✓ Shard分库



大型网站架构演变

■[Step4]技术点—CDN

■CDN(Content Delivery Network)内容分发网络

■将网站的内容分发到最接近用户的网络“边缘”，使用户可以就近获取，从而解决互联网网络拥挤的状况，提高用户访问的响应速度。

■适合静态内容很多（如：静态页面、图片、视频等）及页面内容实时性要求不高的网站，如：新闻类门户网站

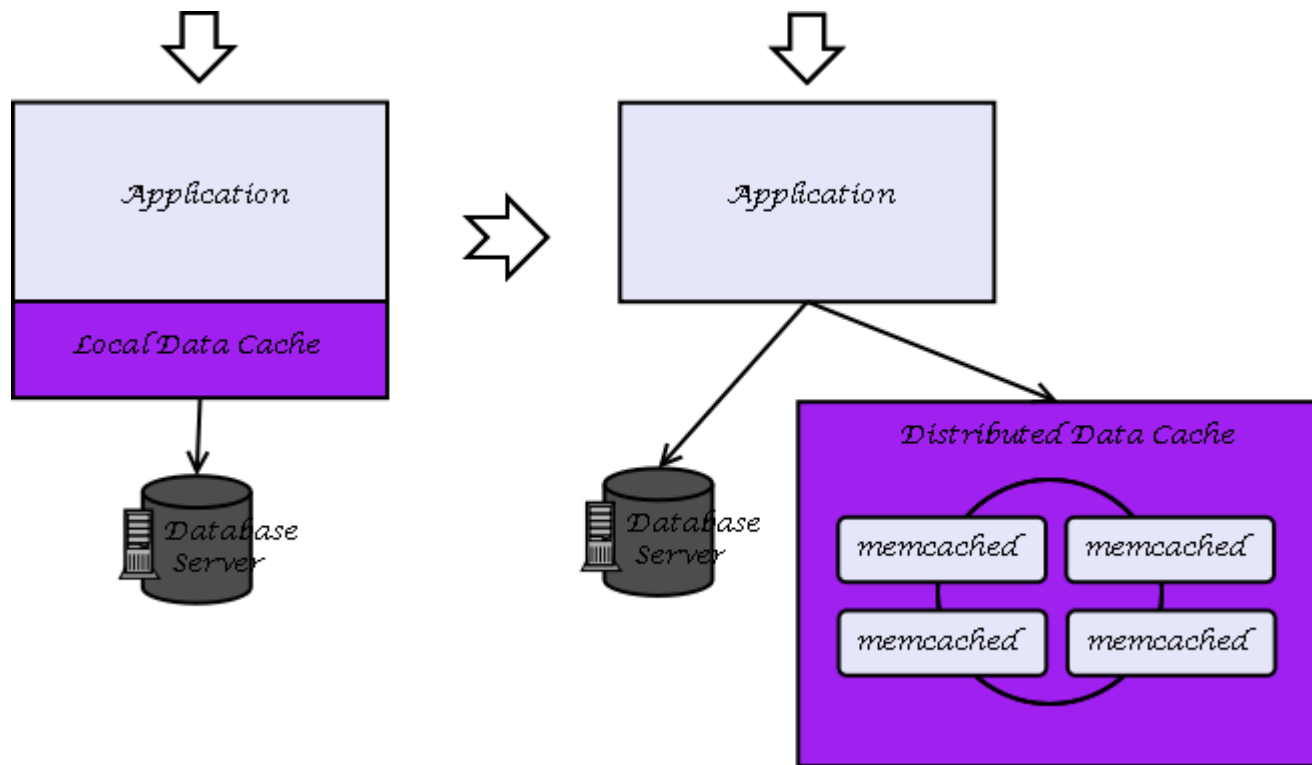
■CDN构建可以做的很简单，也可以很复杂，主要根据自己网站实际情况而定



大型网站架构演变

■[Step4]技术点—分布式缓存

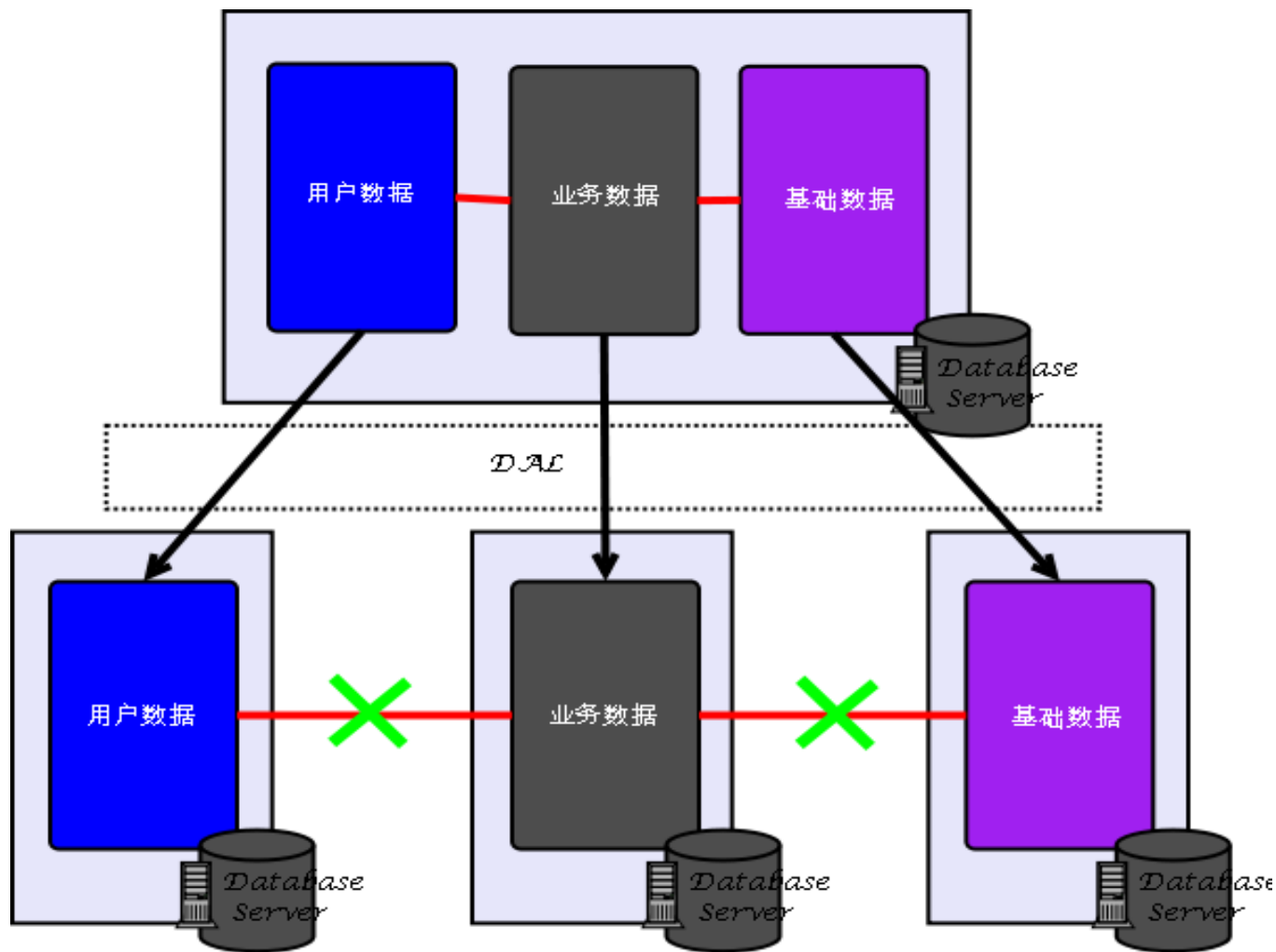
- 本地缓存性能优秀，但容量有限，无伸缩性
- 采用分布式缓存方案突破容量限制，具备良好伸缩性；但分布式涉及远程网络通信消耗其性能本地缓存来得优秀，并可涉及节点状态维护及数据复制问题，其稳定性和可靠性是个挑战。
- 目前流行分布式缓存方案：memcached、membase、redis等，基本上当前的NoSQL方案都可以用来做分布式缓存方案



大型网站架构演变

■[Step4]技术点—分库

- 读写分离（简单有效，前面已介绍）
- 垂直分区

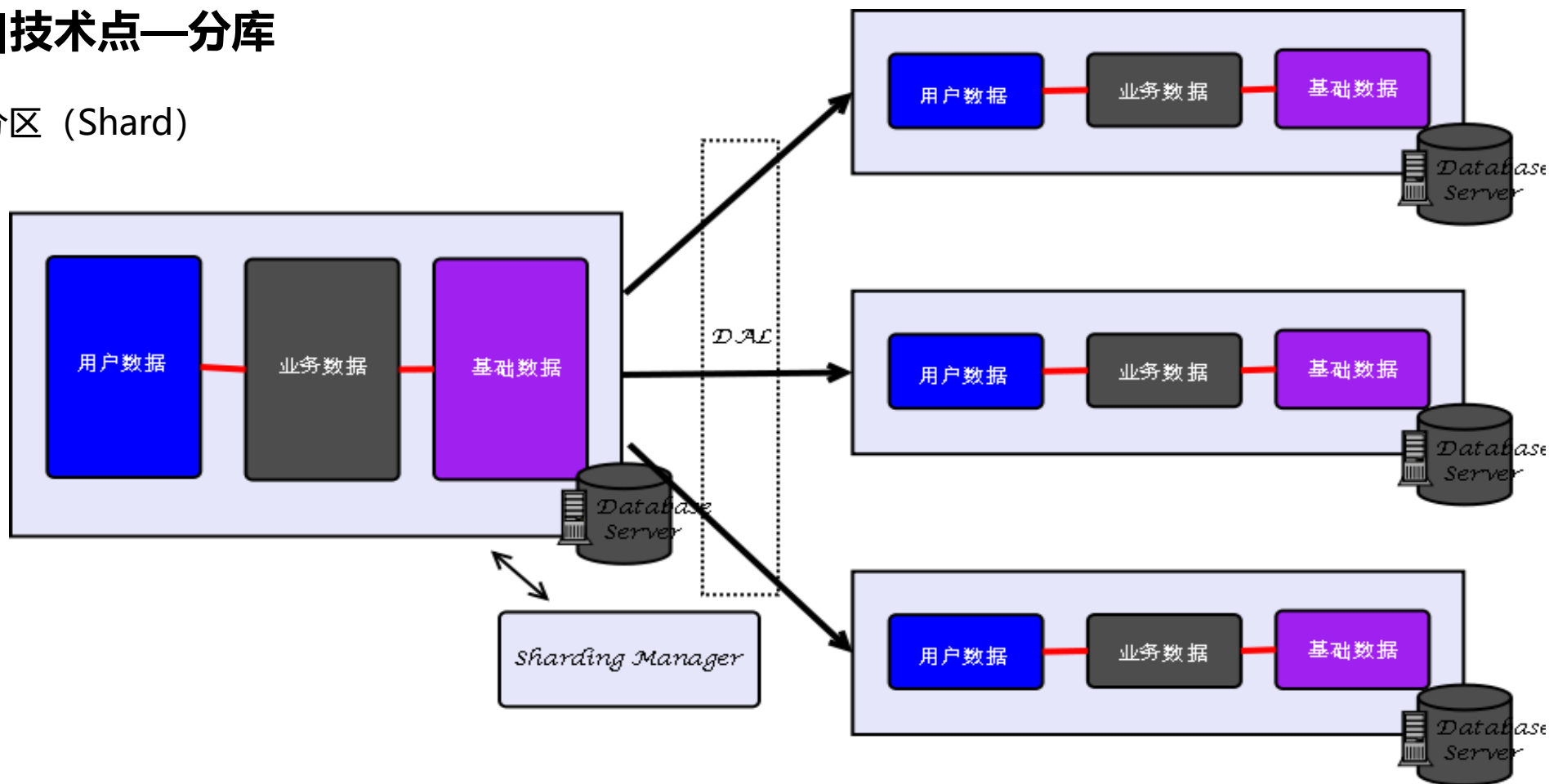


良好的松耦合的模块化设计是垂直分库的前提

大型网站架构演变

■[Step4]技术点一分库

■水平分区 (Shard)



💡 分片Key识别（划分检索依据）是关键

💡 是否还有其它招？用NoSql数据库部分替换关系数据库



大型网站架构演变

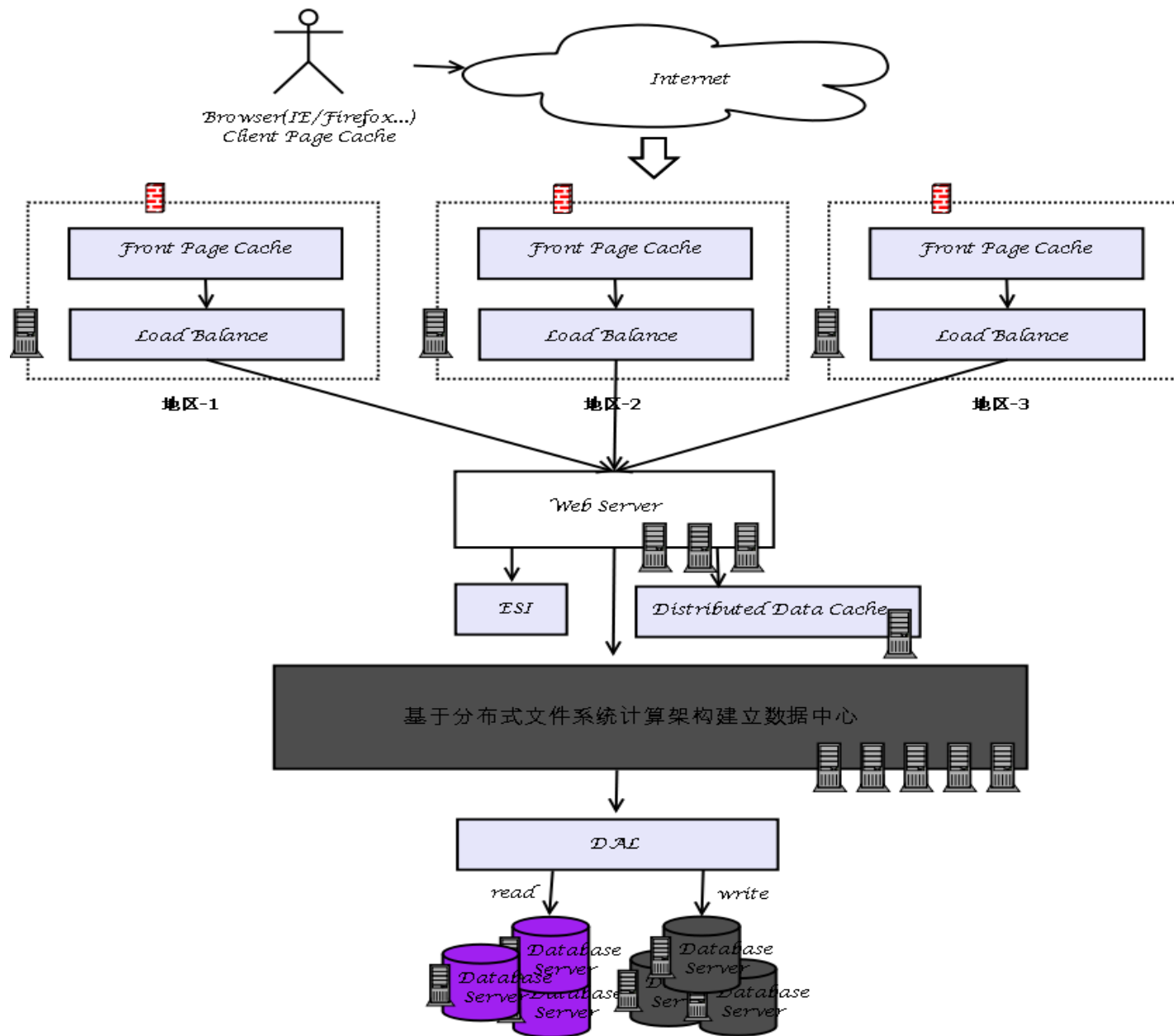
■[Step5]多个数据中心（异地多活）， 向分布式存储和计算的架构体系迈进

■优点：多数据中心，带来更高质量区域服务体验；分布式存储及计算架构有效解决pb级数据量存储、检索及计算性能问题

■缺点：架构复杂、数据同步、一致性及系统维护、技能要求等成本十分高

■技术点：

- ✓分布式文件系统
- ✓Map/Reduce
- ✓Key-Value存储



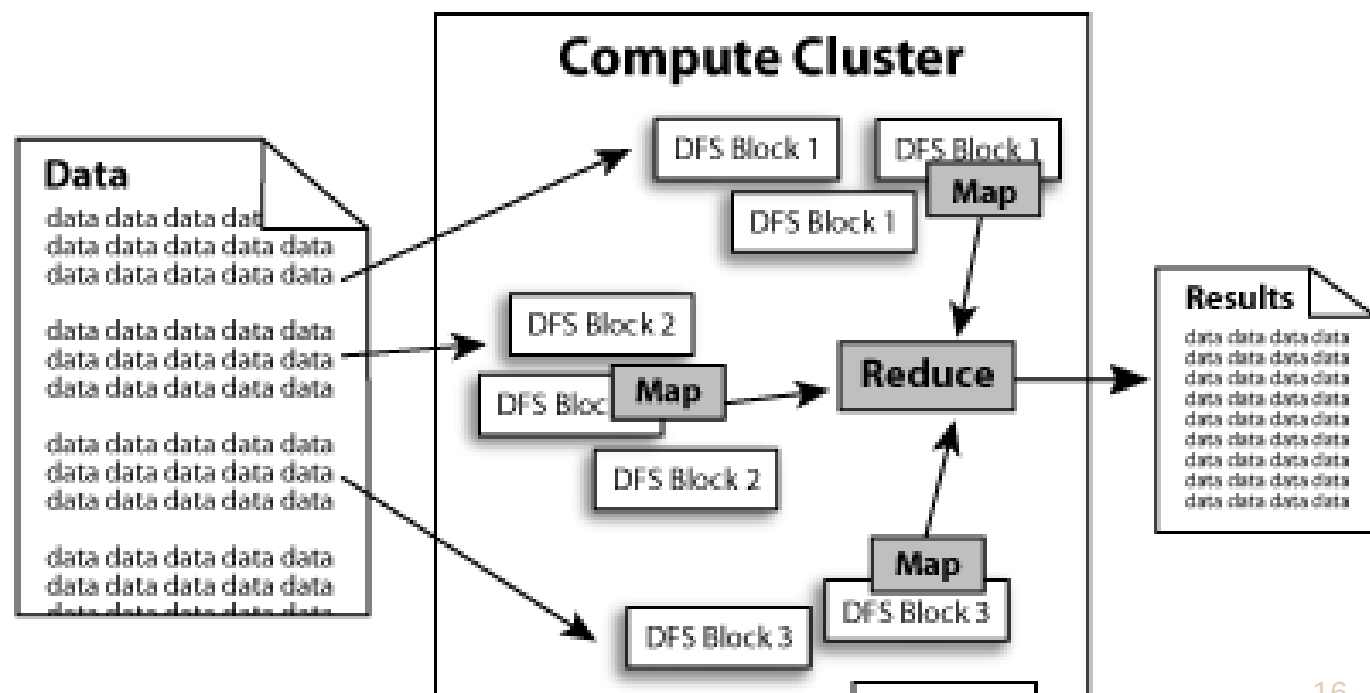
大型网站架构演变

■[Step5]技术点—向分布式存储计算解决方案[DFS、Map/Reduce、Key-Value DB]

- DFS分布式文件系统，如：Lustre\HDFS\GFS\TFS\FreeNas等
- Map/Reduce算法（计算框架），基本上现有NoSQL数据库中都支持此算法。
- Key-Value DB，也作为NoSQL解决方案，如：BigTable\Tair\Hbase\ HyperTable等
- 提供完整解决方案：

Google(GFS|Map/Reduce|BigTable)

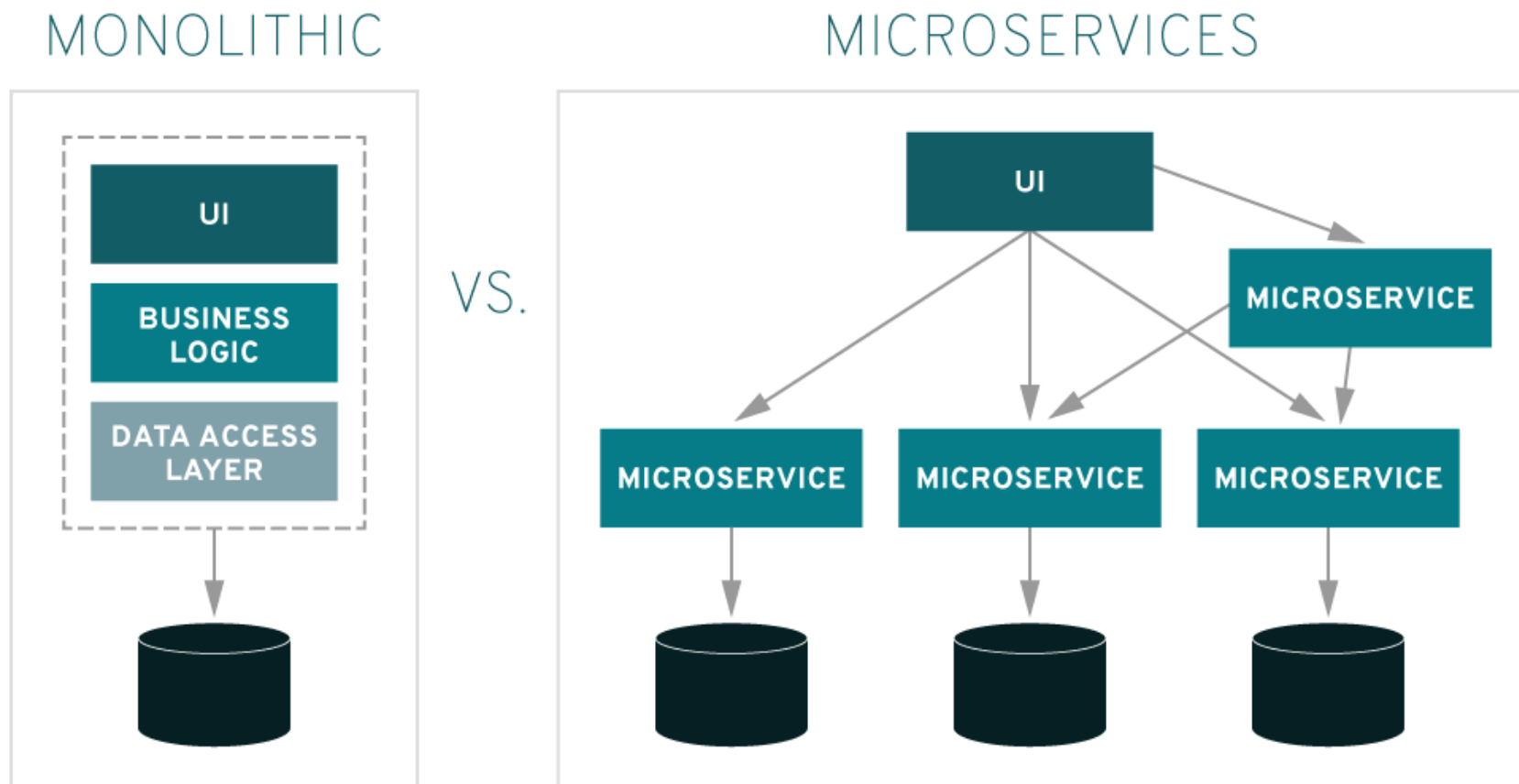
Apache Hadoop(HDFS|Map/Reduce|HBase)



大型网站架构演变

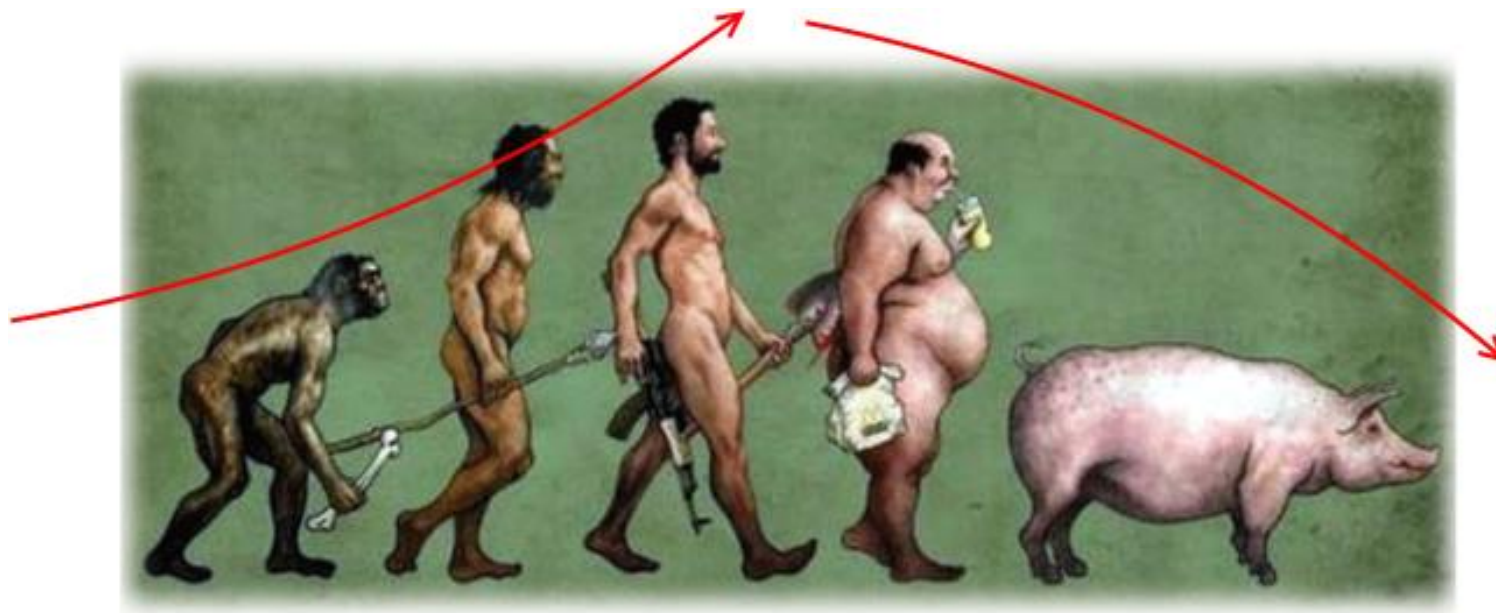
■ [Step6] 微服务架构和容器

- 微服务架构，易于敏捷的开发、更新与部署，所建系统具有高度可扩展和可靠性
- 容器技术，实现运行环境隔离与动态服务管理



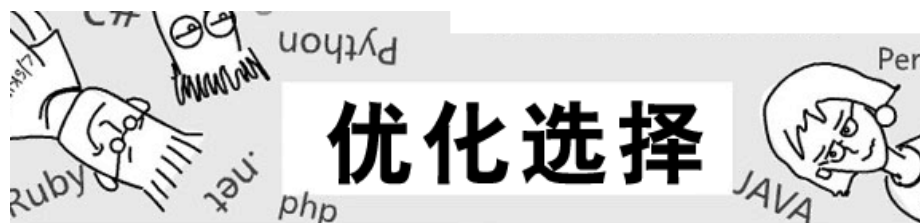
大型网站架构演变

■ 架构进化与退化--奥卡姆剃刀原理



- 进化—寻找最适合的；退化—简化不必要的
- 简单就好，慎防过渡设计

大型网站架构演变



优化选择

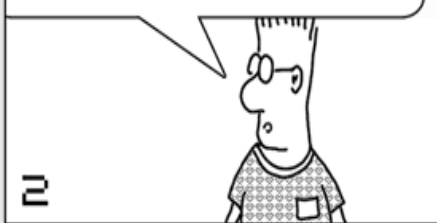
这个性能问题必须要优化了，现在硬盘寻道时间需要60ms，已经是最大的瓶颈了。



我们可以给现有文件系统加上缓存，让热点内容在缓存中读出至少能好20%。



这个优化效率太低了，还不如我们做一种新的分布式文件系统，把读取压力分散到多台机器上，能快个几十倍。



你起码得花6个月，还得增加许多台服务器的预算



■ 考量成本，先硬后软原则

眼下这情况，采用更好的硬件所花的钱，比团队6个月的工资少得多



把硬盘换成SSD，寻道时间1ms以下，效率提高上百倍问题早就解决了



作为程序员，你会面临2种选择
可以花6个月写个复杂程序
把单机系统变成分布的。



也可以休假6个月睡大觉。
因为等你一觉醒来，
让你程序运行得更快的硬件
已经出现了.....

