第一章 缓存为王

主讲人：陈向
湖南科技大学
计算机科学与工程学院

什么是缓存?

Wikipedia: 存储在计算机上的一个原始数据复制集，以便于访问。

- CPU的缓存：CPU与内存读写速度不匹配
- Linux 文件缓存：页表(page table)

软件系统中的缓存

所处位置

- 客户端缓存
- 服务器端缓存
- 网络中的缓存

规模和部署方式

- 单体缓存
- 缓存集群
- 分布式缓存

无处不在，缓存为王

为什么使用缓存？

用户体验

- 人机交互技术发展过程中受到相当的重视
- 关注度与传统三大可用性指标（效率、效益和基本主观满意度）
- 影响用户体验的因素：
  - 使用者的状态
  - 环境
  - 系统性能

系统性能

- 响应时间
  - 软件系统所有功能的平均响应时间或者所有功能中的最大响应时间
  - 网络传输时间 + 应用延迟时间
    - 互联网应用中一般单个接口的延迟时间不超过5s
- 吞吐量
  - 系统在单位时间内处理请求的数量
- 并发用户数
  - 同时正常使用系统功能的用户数量
- 资源利用率
  - 一段时间内资源被平均占用的情况

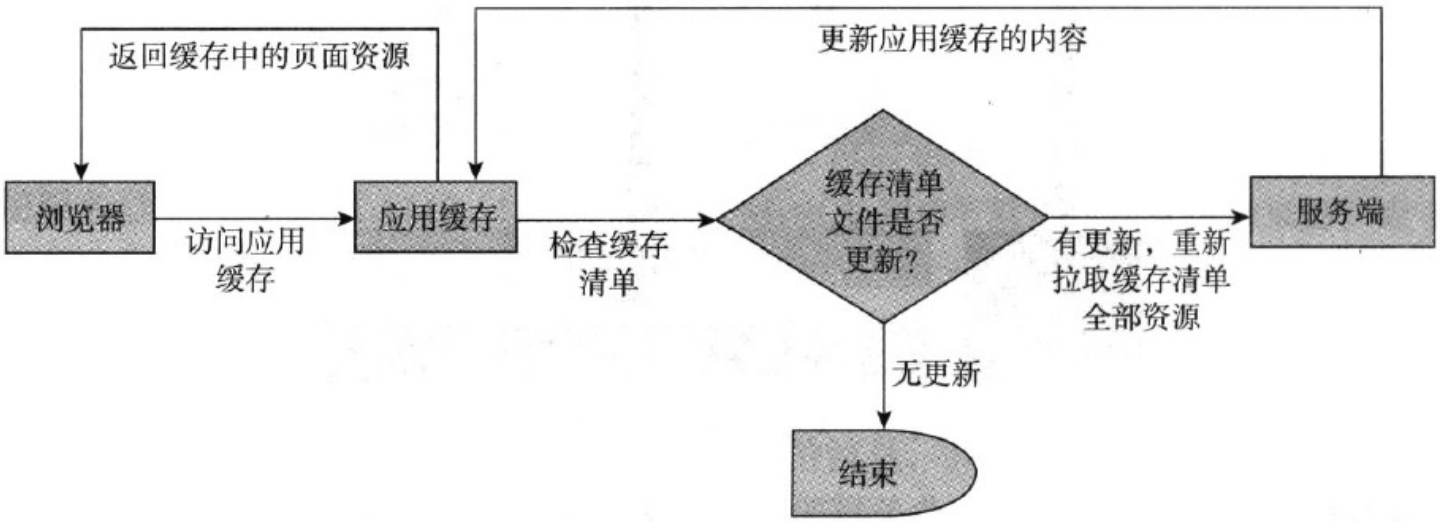
总结

缓存是系统调优时
的常见手段

缓存是软件系统中
关于时间的艺术

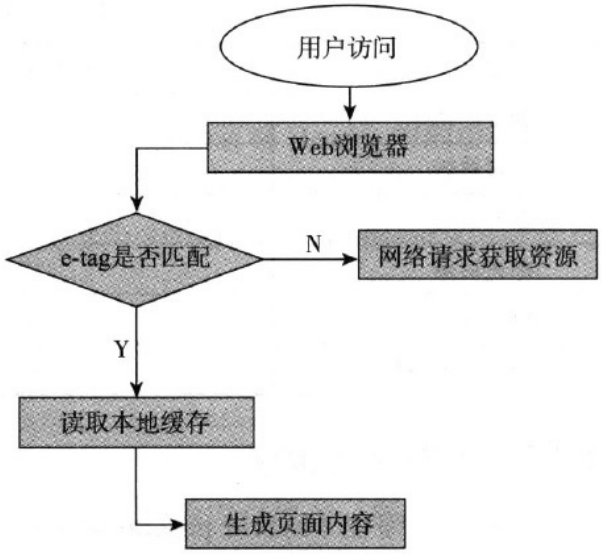
客户端缓存

- 页面缓存
  - 页面自身的缓存或者离线应用缓存
  - HTML5 的离线缓存



客户端缓存

- 浏览器缓存
  - 检查以确保副本是最新的，通常只要一次会话。
  - HTTP 1.0，在服务器端设置 Expires 的 HTTP 头
  - HTTP 1.1，引入实体标签 e-tag

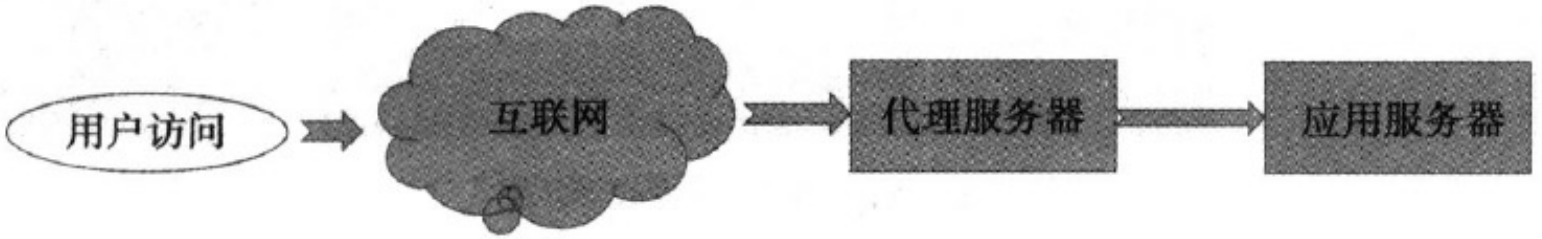


客户端缓存

- App 上的缓存
  - 混合编程（hybrid programming）成为时尚，仍然是原生应用 App 的天下
  - 内容缓存在内存、文件或本地数据库（例如 SQLite）
    - 本地数据库：URL、路径、下载时间、过期时间放在数据库，根据URL先从数据库中查询，未过期根据路径读取本地文件，过期则重读。
    - 文件：文件的最后修改时间
    - e.g. iOS 开发中的图片缓存框架：SDWebImage

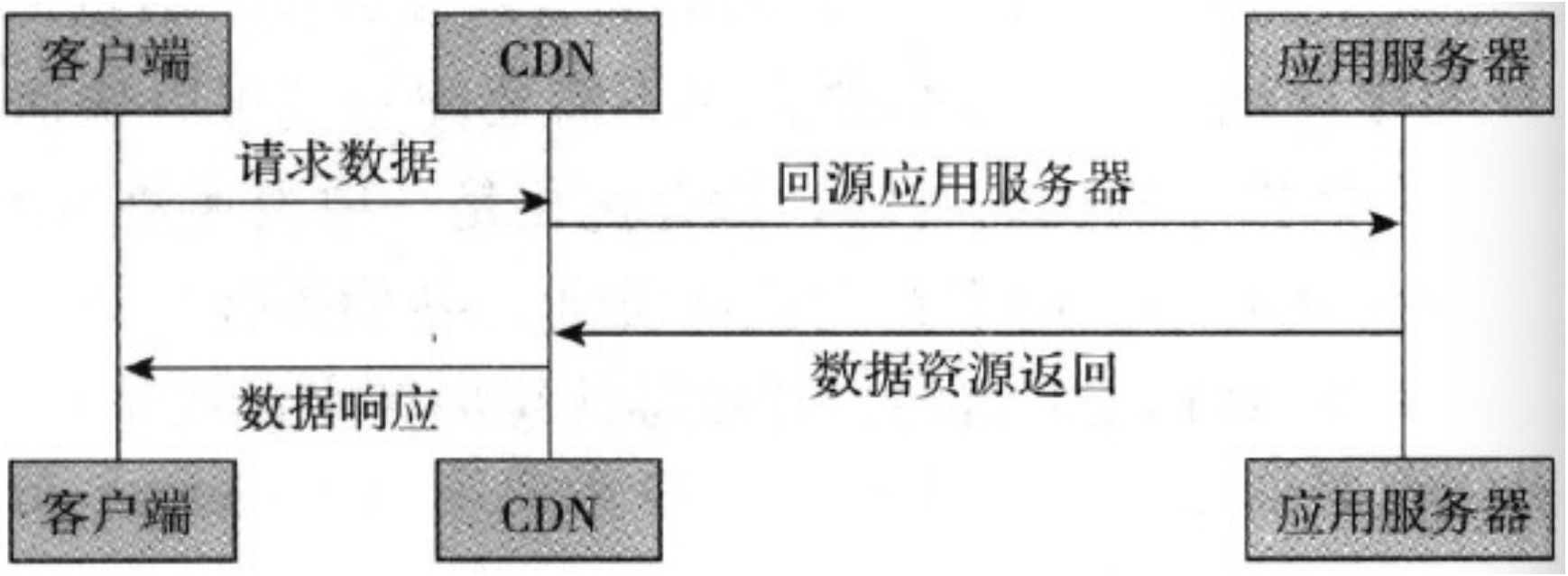
网络中的缓存

- Web 代理缓存
  - 正向代理，反向代理，和透明代理，一般指正向代理



- 边缘缓存
  - CDN （AWS的CloudFront，CloudFlare，ChinaCache）

CDN



- 边缘节点的缓存策略因服务商不同而有所变化
- 服务商会提供基于文件后缀、目录等多个维度指定在CDN上的缓存时间
- 对开发者透明

服务端缓存

- 数据库缓存
  - 数据库属于IO密集型应用，负责数据管理及存储
  - MySQL select查询的ResultSet 由 Query cache 缓存
- 平台级缓存
  - 带有缓存特性的应用框架，或者用于缓存功能的专用库
  - Java：Ehcache，Cacheonix，等；
  - PHP：Smarty 模板库
- 应用级缓存

应用级缓存

- 平台级缓存不能满足系统性能要求
- 通过代码来实现缓存机制，NoSQL 技术的战场
  - Redis
  - MongoDB
  - Memcached

  定时主动更新缓存，或者热数据变化时更新缓存

面向 Redis 的缓存应用

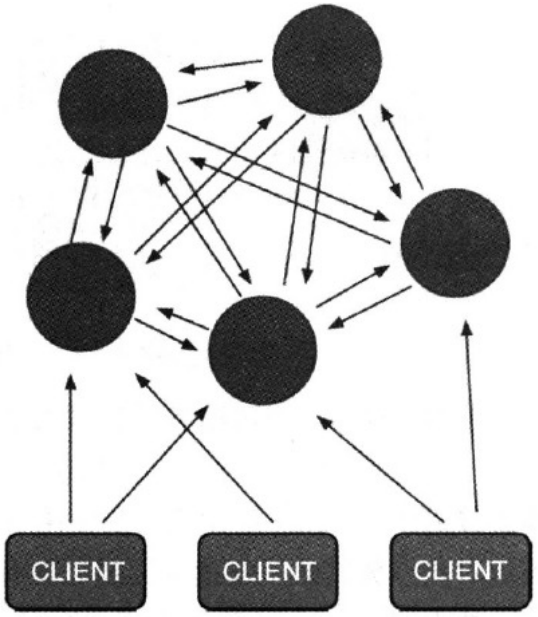
Redis is an open source (BSD licensed), in-memory data structure store, used as a database, cache, and message broker. Redis provides data structures such as strings, hashes, lists, sets, sorted sets with range queries, bitmaps, hyperloglogs, geospatial indexes, and streams. Redis has built-in replication, Lua scripting, LRU eviction, transactions, and different levels of on-disk persistence, and provides high availability via Redis Sentinel and automatic partitioning with Redis Cluster.

开源、基于BSD许可、高级键值对缓存和存储系统。https://redis.io/

Redis 客户端支持的编程语言

ActionScript	Bash	C	C#	C++	Clojure
Common Lisp	Crystal	D	Dart	Delphi	Elixir
emacs lisp	Erlang	Fancy	gawk	GNU Prolog	Go
Haskell	Haxe	Io	Java	Julia	Lasso
Lua	Matlab	mruby	Nim	Node.js	Objective-C
OCaml	Pascal	Perl	PHP	Pure Data	Python
R	Racket	Rebol	Ruby	Rust	Scala
Scheme	Smalltalk	Swift	Tcl	VB	VCL

Redis 集群



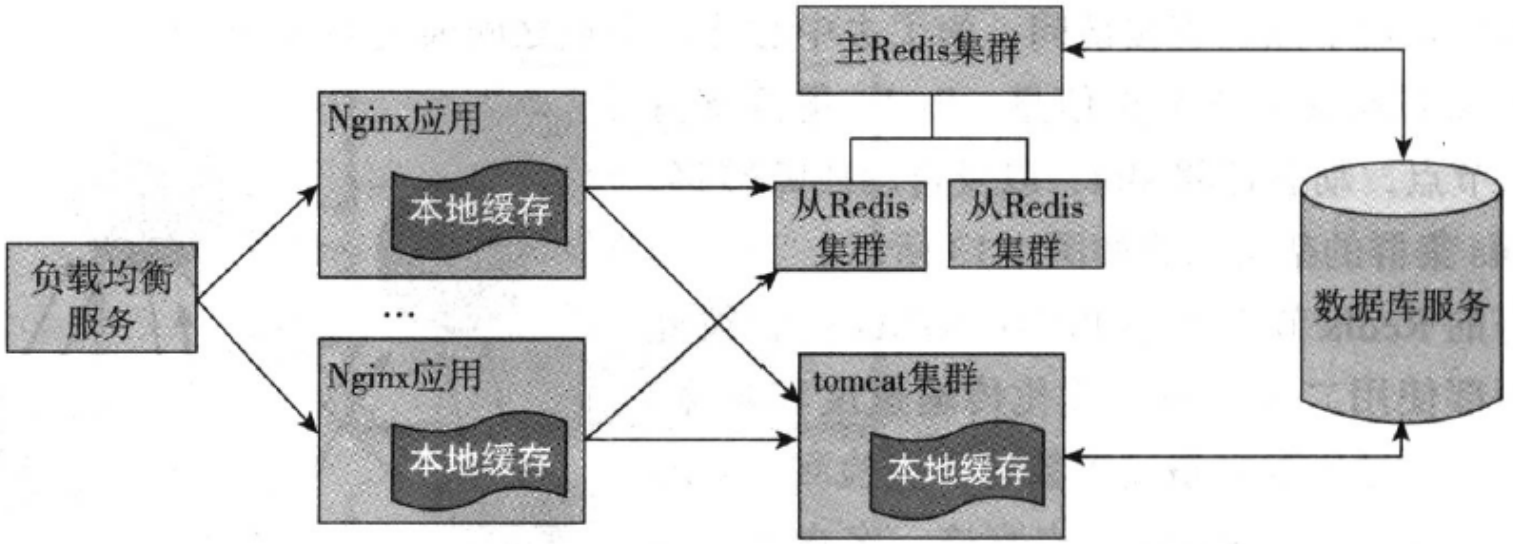
Gossip 协议交换相互的状态，探测新加入的节点信息

无中心节点，无需proxy代理

支持动态加入节点，动态迁移slot，以及自动故障转移。

Redis 3.0 加入了 cluster 功能，解决了Redis单点无法横向扩展的问题。

多级缓存实例



- Nginx：本地缓存解决了热点数据的缓存；
- Redis分布式缓存 集群：减少了访问回源率；
- Tomcat 应用集群：平台级缓存，防止相关缓存失效/崩溃之后的冲击；
- 数据库缓存：提升数据查询时候的效率。

缓存算法

- 缓存术语
  - 缓存命中
  - 没有命中：cache miss
  - 存储成本
  - 缓存失效
  - 替代策略：缓存没有命中，并且缓存容量已经满了，需要在缓存中去除一条旧数据，并加入一条新数据，到底去除哪一条旧数据？采取什么策略？

主流缓存算法

- Least-Recently-Used （LRU）
- Least-Frequently-Used （LFU）
- Least-Recently-Used 2（LRU2）
- Two Queues （2Q）
- SIZE
- LRU-Threshold

使用公有云的缓存服务

- 动态扩容
- 数据多备
- 自动容灾
- 成本较低