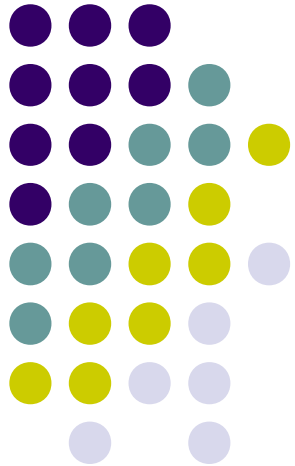
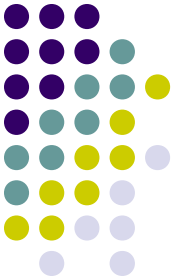
Static-Server 介绍

jamiesun



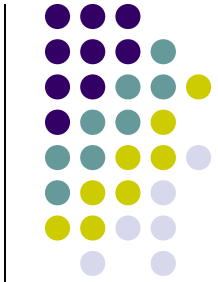


- **Static-Server** 是一个实验性的文件存储服务系统，可以用于多个 web 站点的文件统一存取，可实现分布式的部署，故障转移，同步复制。

应用范围

- 零散的文件，用户上传附件
- 用户上传图片
- 视频服务相关的视频截图。
- 其他文件

客户端支持



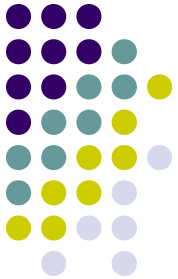
- 支持任意多个 web 站点。
- 通过 java client api 进行文件上传，元数据查询，以及删除。
- 通过 http 的形式读取网络文件。

图片处理功能

- 对于上传类型为图片的，系统会进行缩略图生成处理。
- 对于较大的图片会存储三张不同大小的图片。
- 原图片，未修改的，也是质量最高的。
- 中等大小，相对于 640x480 等比缩放
- 小图片，相对于 120x120 等比缩放。

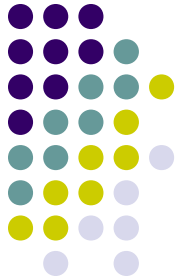
服务端实现

- 基于 java nio 的非阻塞的机制，提供高性能的网络通信。
- 后端采用 oracle Berkeley DB java 版实现存储引擎，存储所有文件资源以及其元数据，提供稳定可靠的存储。
- 基于 tcp 和 http 的方式为所有客户站点提供服务接口。



- 支持各种文件类型
- 海量数据 TB 级支持（硬件配置决定）。
- 支持高并发，高性能。
- 高可用性 / 复制 (HA) — 支持自动系统故障切换和读操作负载平衡，从而帮助维护人员消除单点故障并缩短停机时间

存储引擎 Oracle Berkeley DB Java 版



数据存储

- 本地、进程间数据存储
- 与模式无关的、应用程序本机数据存储
- 按关键字访问的数据检索和顺据数据检索
- 易于使用的 Java 集合 API
- 用于访问 Java 对象的直接持久层 (DPL)
- DPL 类的模式进化
- 单进程、多线程模型
- 用于高并发性的记录级锁定
- 对辅助索引的支持
- 内存中和 / 或磁盘上
- 可配置的后台清理器线程重新组织数据并优化磁盘使用

事务

- 完全符合 ACID
- 可选择的隔离级别和持久性保证，可在每事务基础上进行配置
- 使用 Java 事务 API (JTA) 托管的事务
- 使用 J2EE 连接器体系结构 (JCA) 进行 J2EE 应用服务器集成
- 使用 Java 管理扩展 (JMX) 进行审计、监视和管理
- 灾难和例程故障恢复模式
- 基于超时的死锁检测
- 冷热备份、日志文件压缩和完整的数据库转储

部署

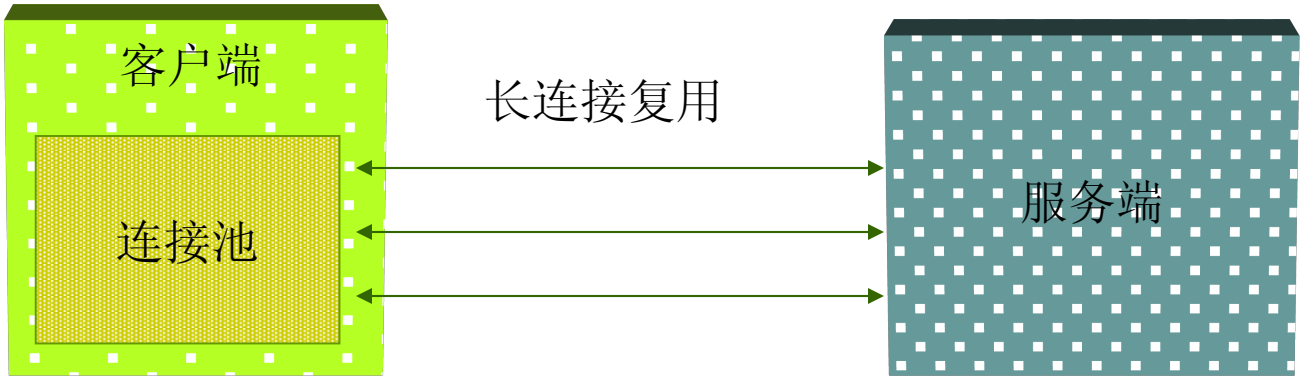
- 100% 的纯 Java，便于移植且易于开发
- 单个 JAR 文件 — 易于安装，与应用程序运行在相同的 JVM 上
- 需要 Java 1.4.2 或更高的标准版 JVM
- 编程管理
- 无需人为管理
- 针对例行管理功能的 API
- 体积小 820KB
- 可扩展到数以 TB 计的数据，数百万条记录
- 包含源代码、测试套件

其他可参考的文件存储引擎

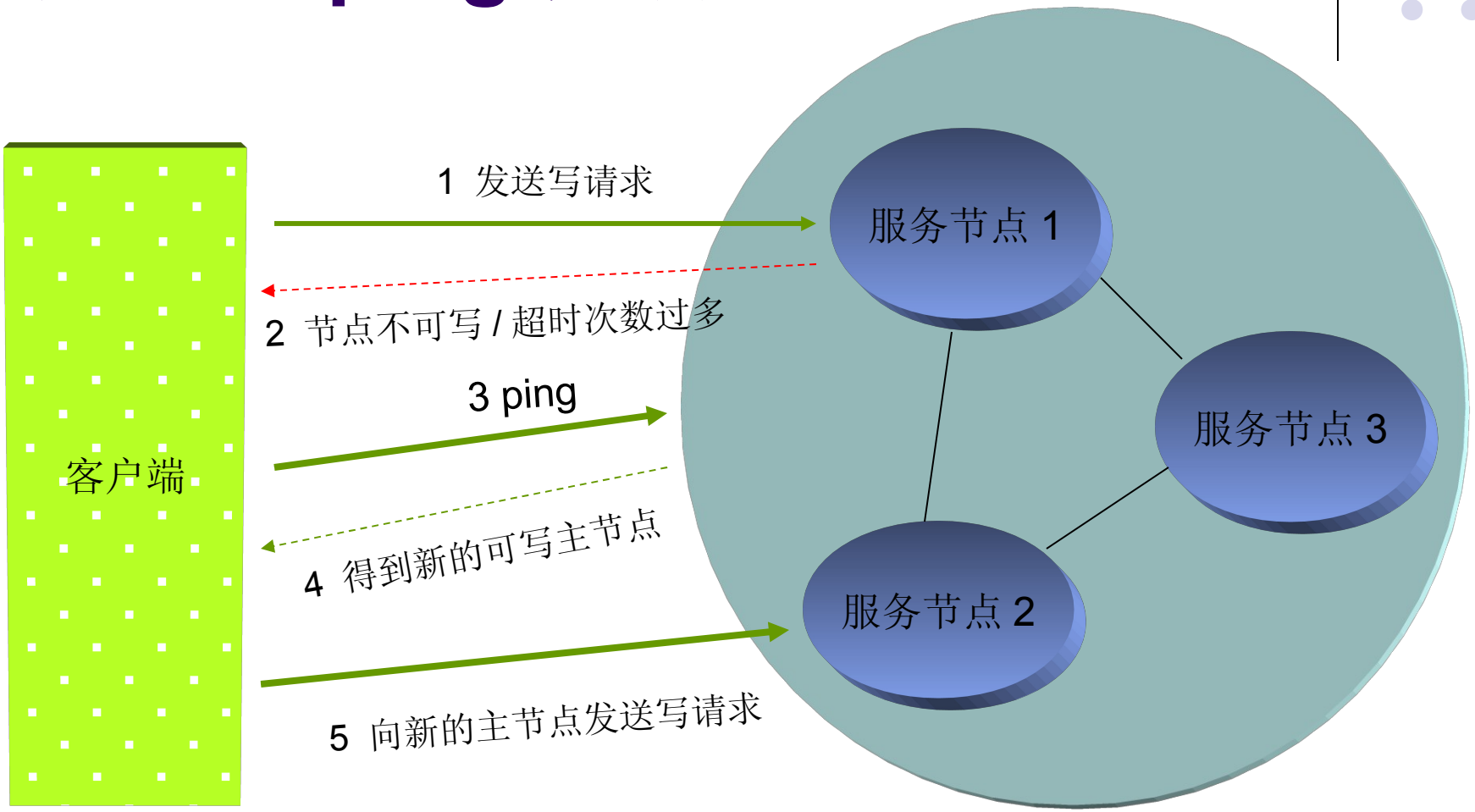
- **mongodb gridfs**：开源 nosql 数据库，可以存储上百万的文件而无需担心扩容性 . 同步复制 , 可以解决分布式文件的备份问题。
- **memcachedb**：开源的 nosql 数据库， memcached 客户端协议 + Berkeley DB 的实现。
- **beansdb**： memcached 客户端协议 +TokyoCabinet （轻量级 KV 数据库）

Java client api

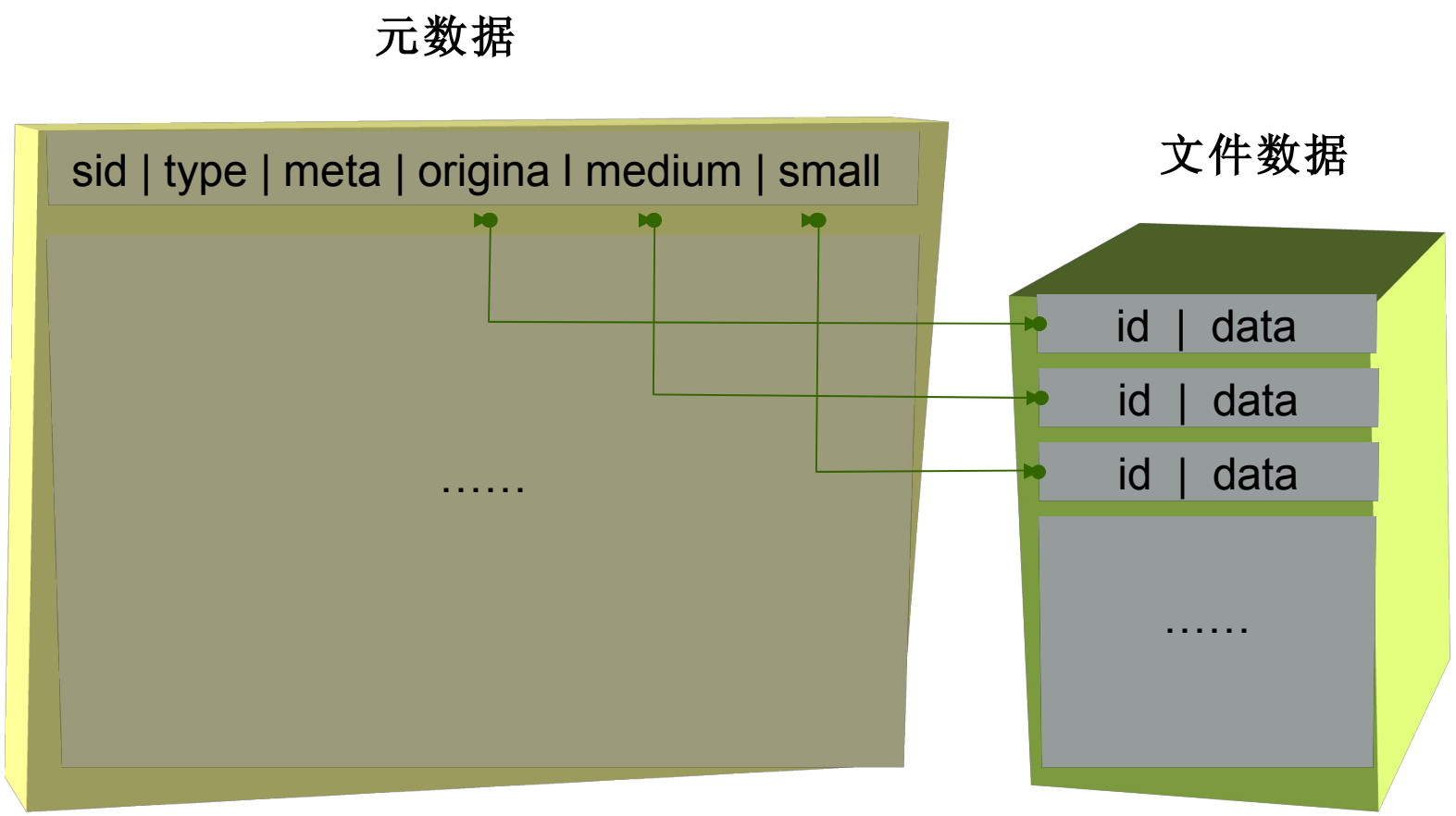
- 客户端使用了连接池来实现 tcp 的连接复用，可调整的参数可以保证一定的伸缩性。
- 使用 ping 的方式来保证总是能快速准确连接到可写的主节点。完成数据提交。



客户端 ping 机制


客户端
1 发送写请求
2 节点不可写 / 超时次数过多
3 ping
4 得到新的可写主节点
5 向新的主节点发送写请求
服务节点 1
服务节点 3
服务节点 2

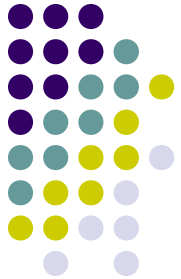
数据存储结构


元数据
sid | type | meta | origina l medium | small
……
文件数据
id | data
id | data
id | data
……

文件访问

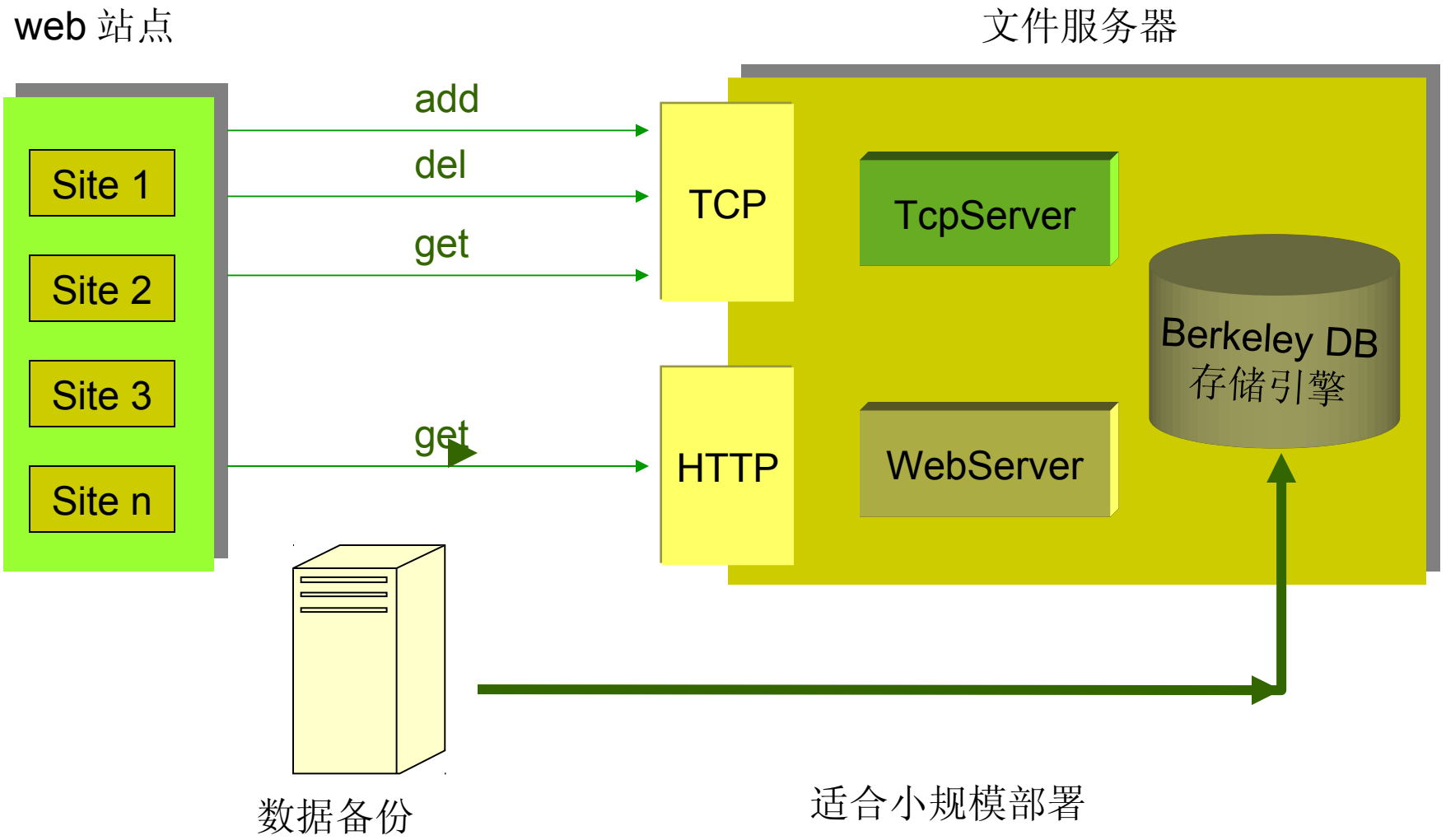
- 可通过 ngnix 作为前端反向代理服务器来访问文件。
- **文件 url 规则：**
- **http://url/{sid} 原始文件**
- **http://url/{sid}/s 小缩略图**
- **http://url/{sid}/m 中等质量缩略图**

缓存支持

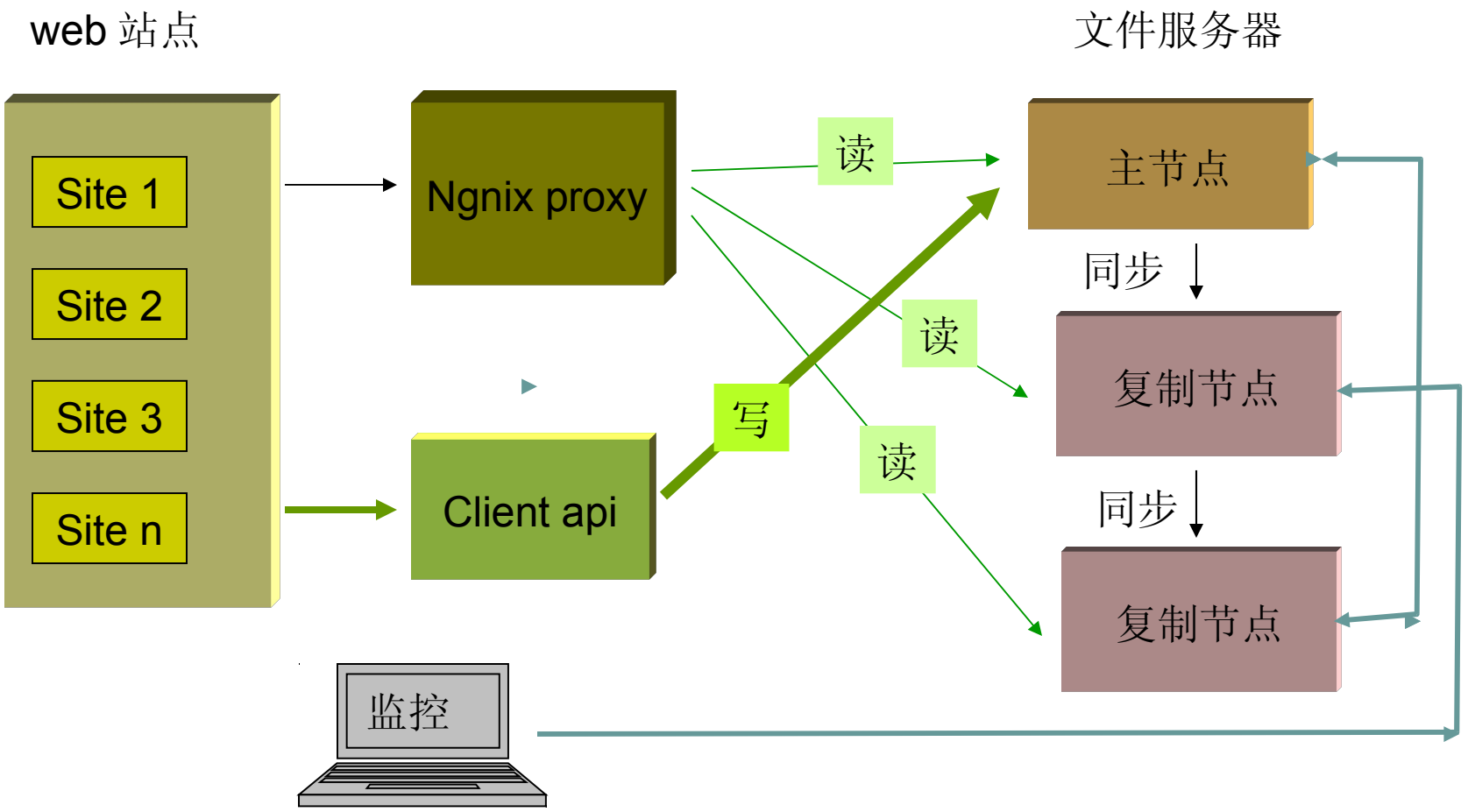


- 支持 Etag 头标识，可利用浏览器客户端缓存，根据实际要求可以进一步扩展。
- 配合 ngnix 的缓存模块可达到更好效果

单节点部署模型

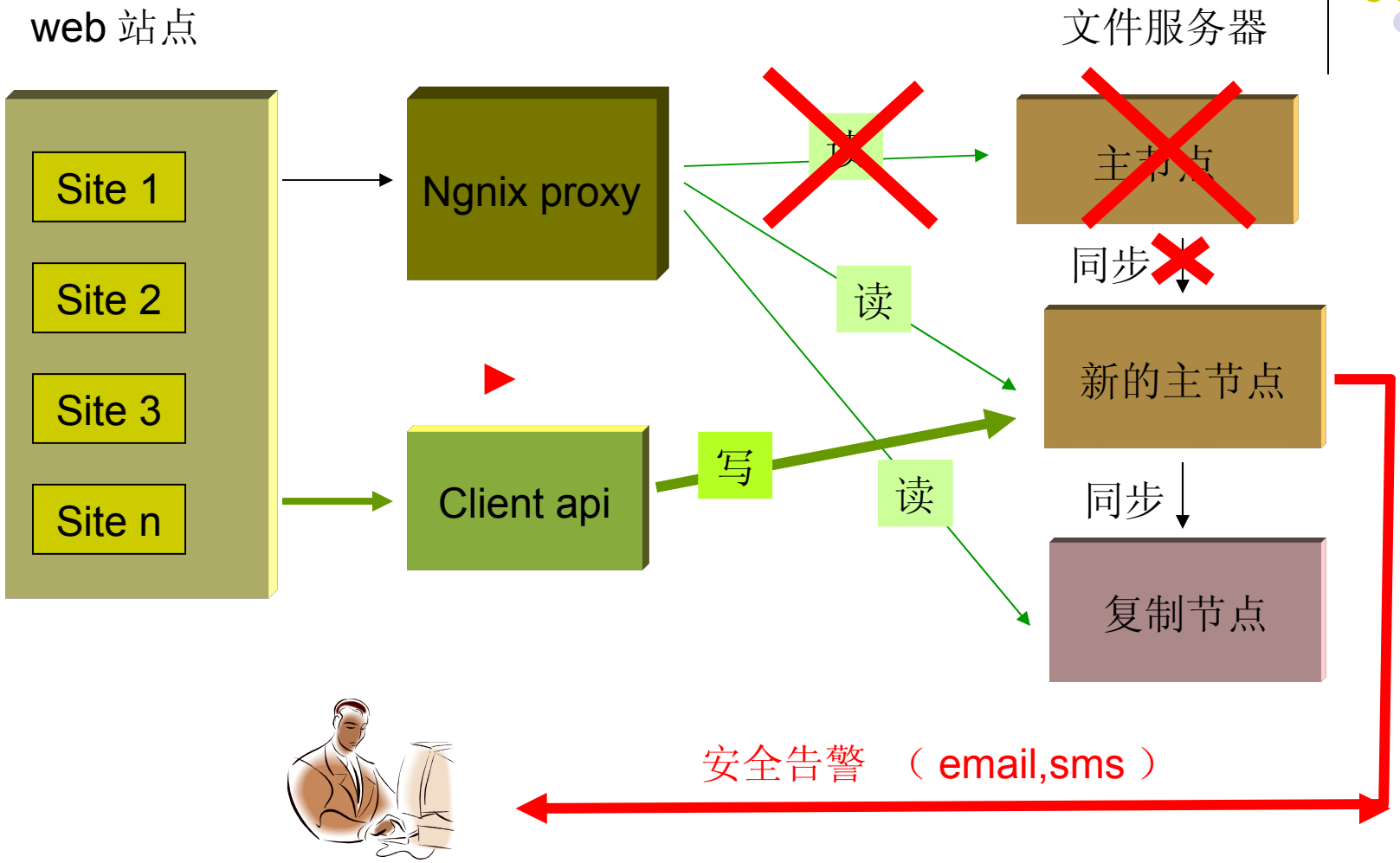

web 站点
文件服务器
Site 1
Site 2
Site 3
Site n
add
del
get
get
TCP
HTTP
TcpServer
WebServer
Berkeley DB
存储引擎
数据备份
适合小规模部署

多节点部署模型



通过软硬件的配置优化，可支持更高并发的读写请求 .

故障转移



当主机当掉后，由系统算法自动选举新主节点，对客户请求重新路由

示例：同步方式新增

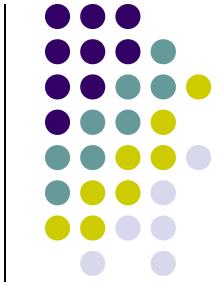
```
StaticClient client = new StaticClient(new String[]
{"127.0.0.1:8000","127.0.0.1:8001","127.0.0.1:8002"},60,200);
//参数（主机，超时时间，连接池大小）
Map meta = new HashMap();
meta.put("uuCode", "110168");
String sid = ..
byte [] data = ..
StaticObject sto = new StaticObject(sid, "image/jpg", meta, data);
resultCode = client.add(sto);
```

示例： 异步方式新增

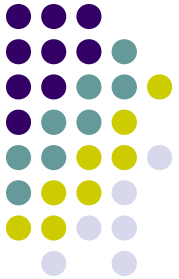
```
Map meta = new HashMap();
meta.put("uuCode", "110168");
String sid = ..
byte [] data = ..
StaticObject sto = new StaticObject(sid, "image/jpg", meta, data);
client.add(sto, new ClientHandler(){
    public void onResponse(int status){
        System.out.println(status);
    }
});
```

示例： 同步删除

```
Client.del(sid);
```



示例： 同步获取



```
StaticObject obj =
  client.get(sid,StaticClient.QUALITY_SMALL);
```

简单性能测试—同步新增

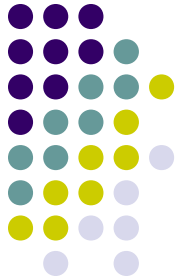
- 在一台双核普通 hp541 笔记本 windows 系统上的测试，服务器和客户端都跑在一个机器上。
- 上传图片文件大小 280kb 。文件服务器生成缩略图后共存储 600K
- **测试并发总数： 50**
- **测试消息总数： 10000**
- **测试耗时： 1539750 毫秒**
- **每秒处理请求数： 6**
- **平均每个请求耗时： 153 毫秒**
- **事务全部成功，零失败**
- **在实际的服务器环境中，性能应该会有更好的表现：**
- **更多的 cpu 核心支持，更大的内存支持， linux 下的 epoll 支持，更大的网络带宽，都可以带来性能的显著提升。**

简单性能测试— http 读取

- 在一台双核普通 hp541 笔记本 windows 系统上的测试，服务器和客户端都跑在一个机器上。
- 测试工具： apache ab
- 测试参数： ab –c 1000 –n 10000 http://127.0.0.1:9000/a61c7d3e-8df9-4d01-9dd0-

（读取缩略图 73.8kb ）

读取性能结果数据

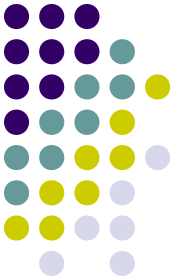


- 内容长度： 75654 bytes
- 并发数： 1000
- 总耗时： 77.625 seconds
- 完成请求： 10000
- 失败请求： 3
- (Connect: 3, Receive: 0, Length: 0, Exceptions: 0)
- 总共传输： 758860000 bytes
- HTML 传输： 756540000 bytes
- 每秒请求数： 128.82 [#/sec] (mean)
- Time per request: 7762.500 [ms] (mean)
- Time per request: 7.763 [ms] (mean, across all concurrent requests)
- 平均每秒传输： 9546.85 [Kbytes/sec] received

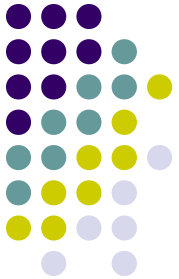
- **连接时间统计 (ms)**

	min	mean[+/-sd]	median	max
Connect:	0	6 50.9	0	516
Processing:	156	7460 1412.7	7813	8859
Waiting:	16	3850 2148.5	3781	8031
Total:	172	7467 1412.8	7813	8859

同步复制

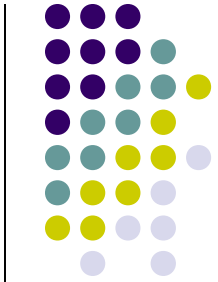


- 支持跨多个系统的复制，从而使应用程序能够以较短的时间进行大规模扩展并为高可用性解决方案提供容错。
- 该技术的工作机制是让所有更新进入一个指定的主节点中，这个主节点自动将更改分发到一组副本。读负载可以跨这些副本分散，而新的副本可以随时加入该组来扩展此系统。如果任何副本失败，其余的副本可以取代它。
- 如果主节点发生故障，副本将进行选举，然后指定一个新的主节点。选定了新的主节点后，所有副本与新的主节点同步并在不中止服务的情况下继续进行正常处理。主 - 故障切换过程通常不到一秒，故障切换期间由副本提供读请求，从而确保不会停机。



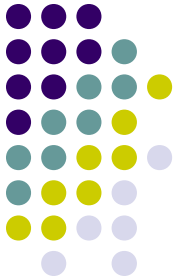
系统中用到了几个第三方组件

ImageMagick



- ImageMagick，图形组件库，主要用来处理高质量的图片缩放， java 本身提供了图形处理库，但是在图片缩放处理上存在一些问题，比如内存占用过大甚至导致溢出，速度慢，而且生成的缩略图质量不高，故放弃。
- JMagick, ImageMagick 的 jni 扩展。
- http://downloads.jmagick.org/

gson



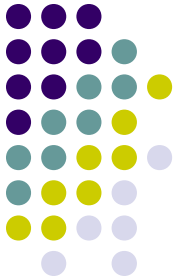
- 一个 json 编解码组件，用来处理元数据的的处理。

Berkeley DB Java

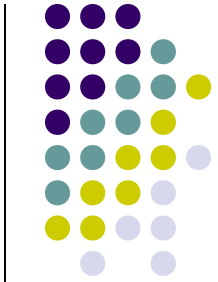
- 文件服务器所用到的核心存储引擎。
- http://www.oracle.com/technology/global/cn/produ

log4j 日志组件

- 系统日志处理



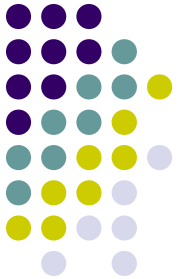
picocontainer



一个简单的依赖注入管理组件。系统中主要用到了其依赖注入管理和组件生命周期管理功能

xSocket， xlightweb

- xSocket，一个轻量级高性能的基于 java nio 的网络通讯组件，系统中主要用来实现文件的网络传输。
- Xlightweb，基于 xSocket 的 web 服务器实现



Thank you ！