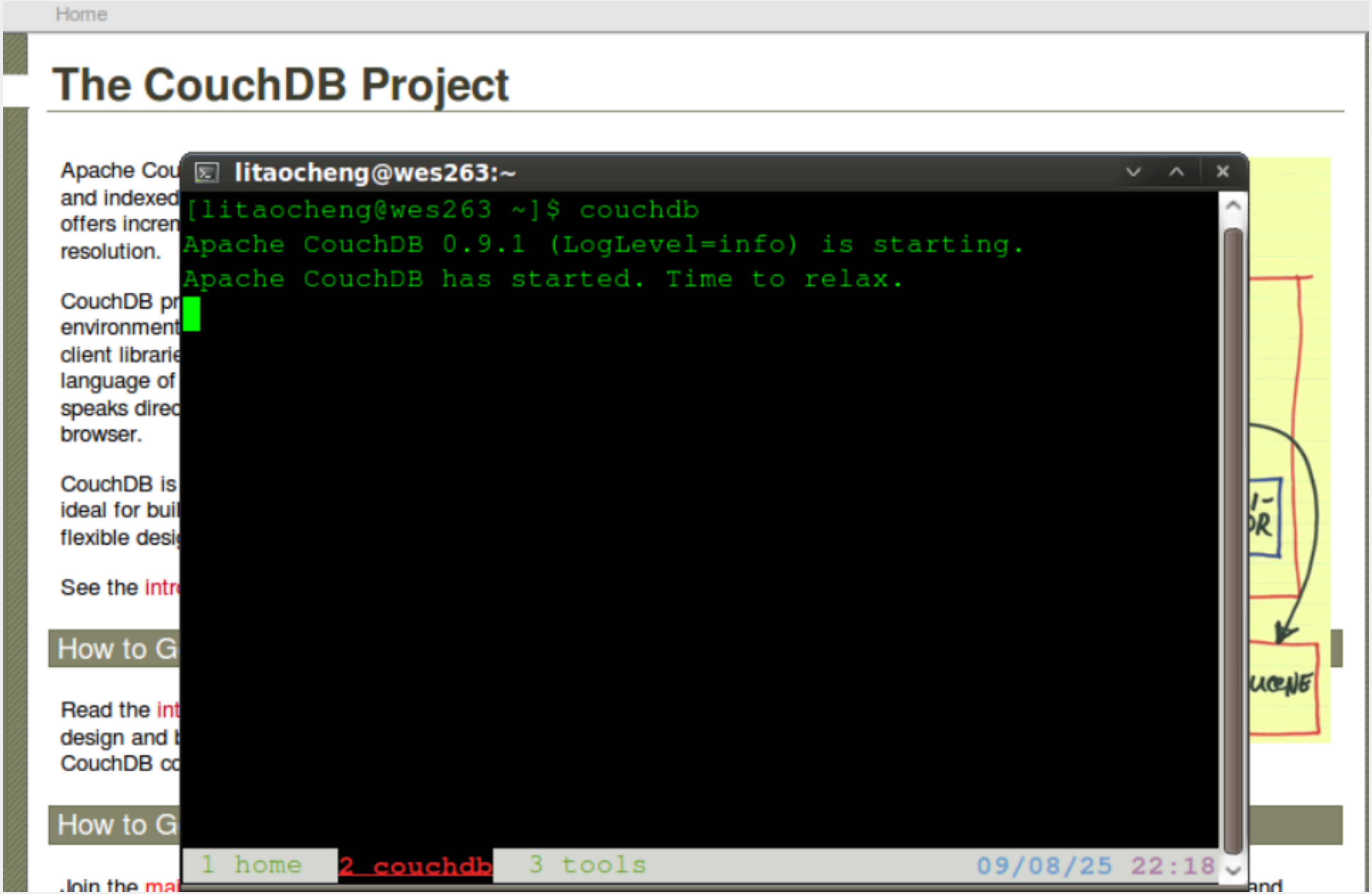
Apache CouchDB

and Erlang

litaocheng@gmail.com

2009.8.29



Apache CouchDB has started.
Time to relax.

About This Talk

- CouchDB有何异与常人之处
- 了解CouchDB
- Erlang为CouchDB提供了什么帮助
- CouchDB Internal

What's CouchDB

- Damien Katz创立<http://damienkatz.net/>
- Worked on Lotus Notes & MySQL
- 2005开始本项目: C++, XML, on Windows
- 2006基于Erlang进行实现, JSON, POSIX
- 0.7.0 为第一个基于Erlang的发布版本
- 成为Apache基金会正式项目
- 当前版本 0.9.1

CouchDB Next Release(0.10)

- cookie 及 OAuth认证
- Windows发行包
- 连续，可靠，快速的备份同步
- 更加高效的追加式存储
- 更好的HTTP协议兼容性（RFC 2616）
- 后台管理界面改进
- Native Erlang View Server(已经实现)
- Filter API
- Update API

Why CouchDB?

- Document DB，数据存储格式为JSON
- Schema-Free，应用易扩展
- HTTP RESTful 接口，Web开发结合紧密
- CouchDB Replication，开发offline应用
- ACID Storage Engine
- MVCC，性能优越
- Views With MapReduce
- Single Storage File, 便于管理备份
- Fault Tolerant

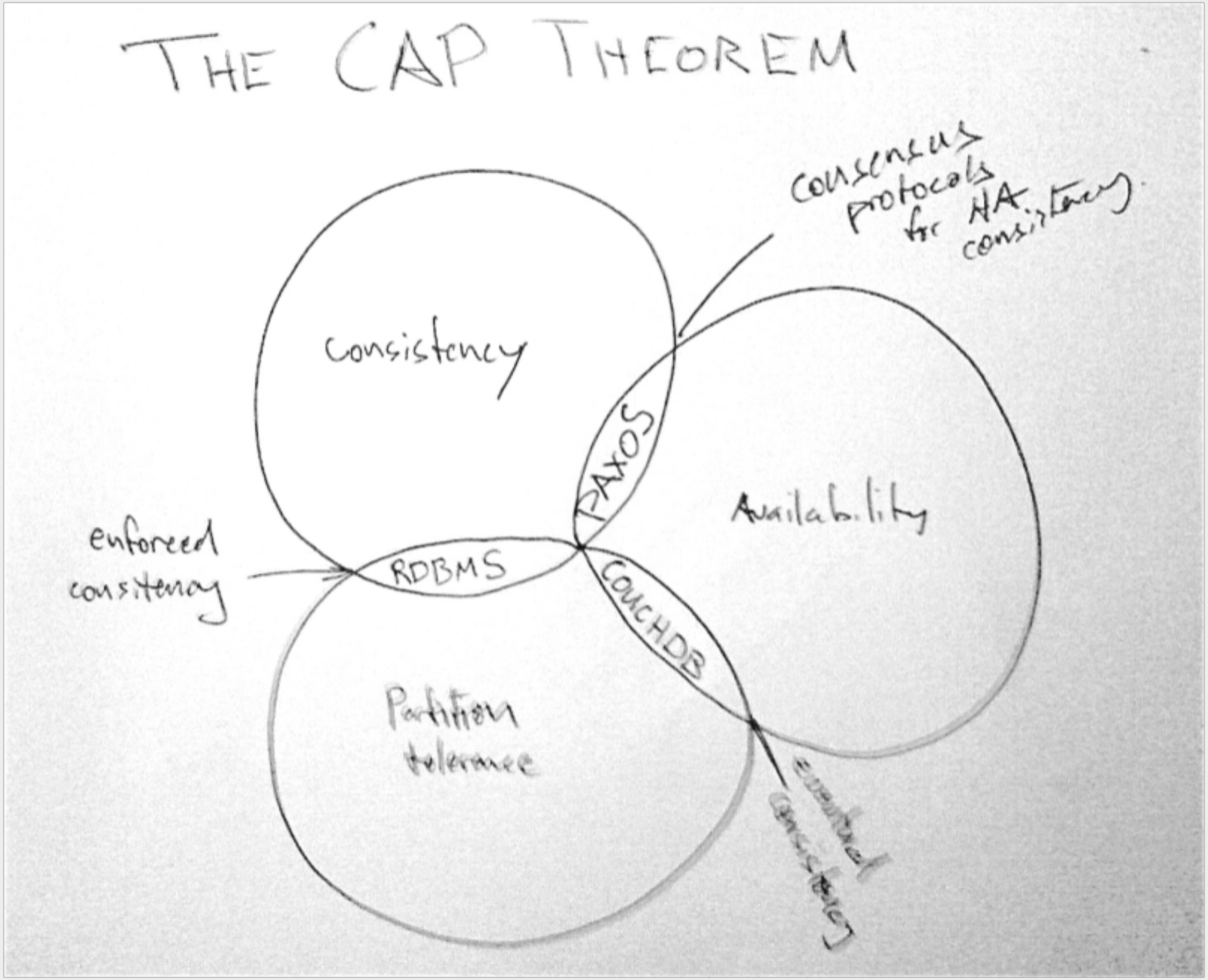
CouchDB系统特性

Standalone (单节点):

- 采用MVCC提高系统能力(并发读，顺序写)
- Append方式更新数据库
- 提供ACID支持

Distributed (分布式)

- CouchDB 基于Document 可以自然的进行Partition
- Replication提高系统Availabity 和 Scalability
- Eventual Consistency by using Incremental Replication


THE CAP THEOREM
consensus protocols for HA consistency
consistency
PAXOS
Availability
enforced consitency
RDBMS
COUCHDB
Partition tolerance
eventual consistency

CouchDB 原则

开始了解CouchDB

API

JSON

Document

View

ACID

MVCC

CouchDB API

HTTP	CouchDB RUD
GET	read
PUT	create or update
DELETE	delete
POST	bulk operation (批量操作)

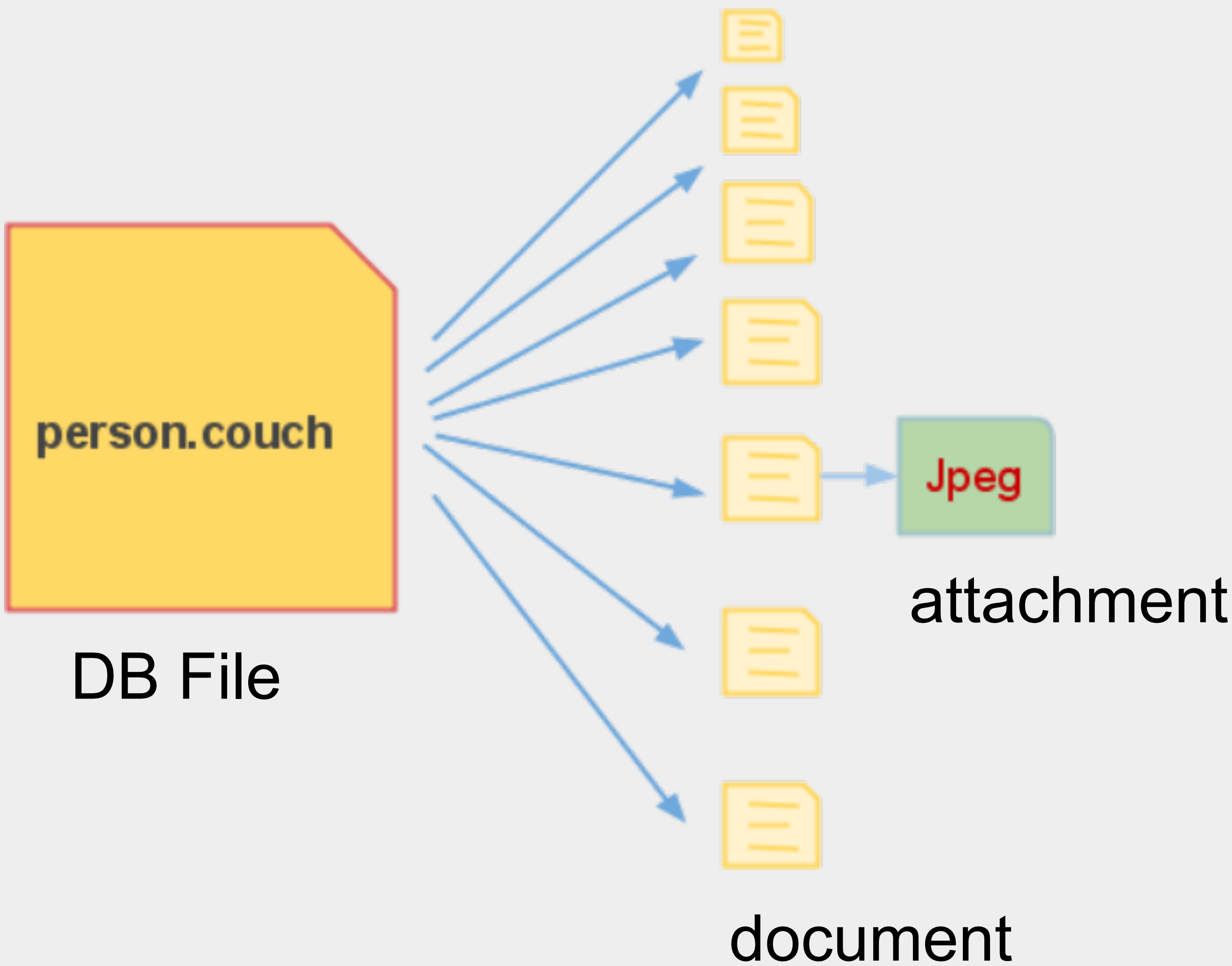
熟悉HTTP，那么你就已经了解CouchDB API ！

JSON Documents

```
{
        "_id"  : "some uuid",
       "_rev"  : "mvcc key",
      "title"  : "a blog post",
     "author"  : "foo",
       "tags"  : ["life", "happy"],
  "post_date"  : "2008-08-30 15:30:26",
    "content"  : "this is an blog...."
}
```

Schema-Free,根据应用调整数据,与JS亲密接触

Document and couch file



View

用户自定义函数，生成展现数据（javascript)
不影响原有的storage数据(.couch)

两种类型:

- 常规view
  design documents 存储view的定义
  view index存储view结果(BTree)
- 临时view
  通过POST DATA提交view定义，返回结果

Querying Options - startkey, endkye, limit ...
默认language为js
支持erlang native view定义

View example(js)

```
map fun:
function(doc) {
  emit(doc.post_date, doc.title);
}

result(BTree, sorted):

|     key                   |    value         |
-------------------------------------------
| "2008-08-30 15:30:06"  | "a blog post" |
| "2008-08-31 20:31:24", | "second post" |
| "2008-09-01 10:10:21", | "third post"  |
```

方便检索“一个星期内”，“某年某月”发布的blog post

ACID

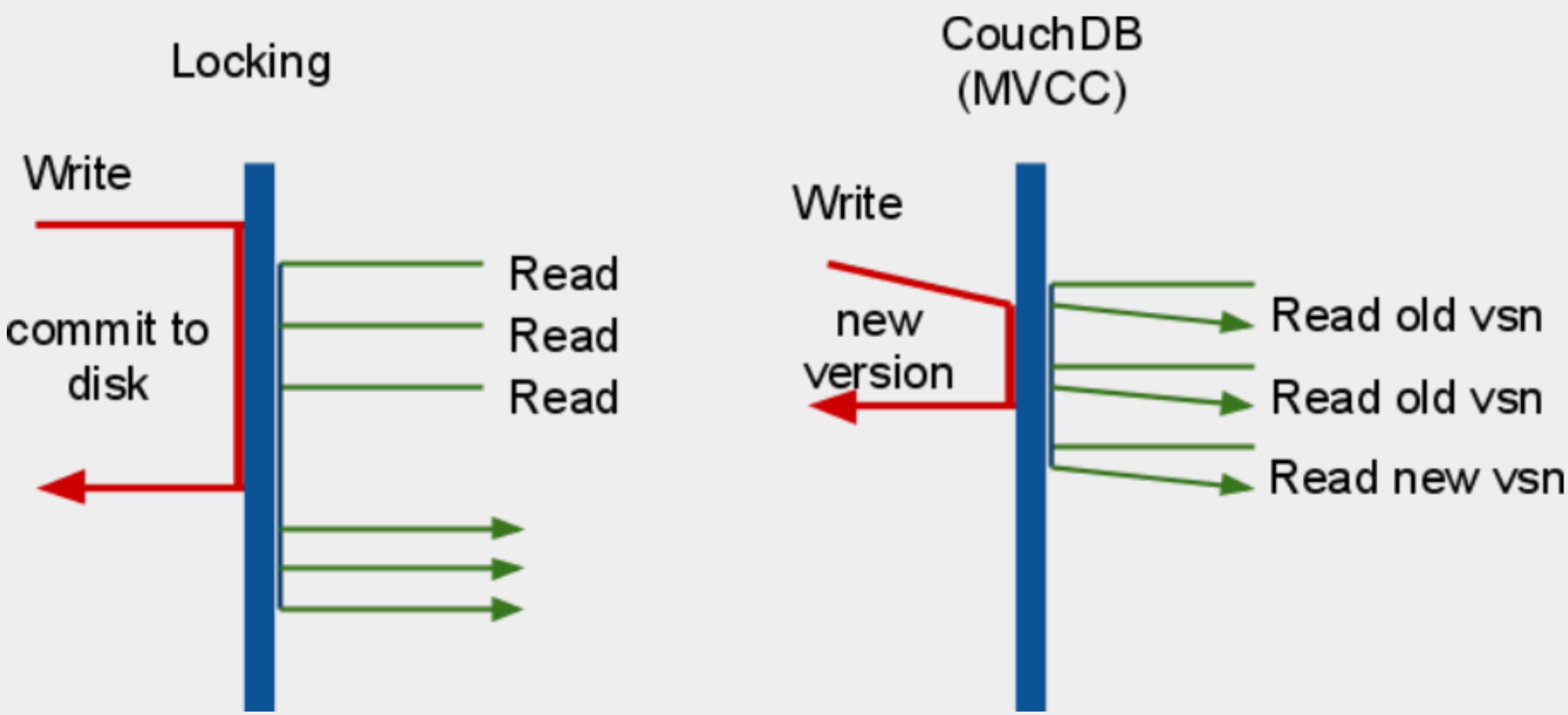
Storage文件布局 (double header)
Append方式更新，无overwirte,无commit log
crash-only，数据保持一致性
MVCC, read（并发)不加锁,write为顺序化执行
包含2个BTree，以docid和seqid为key

更新步骤:

1. 数据追加到storage文件中
2. 更新两个连续的header信息(4K)

以上任意步骤发生异常，均保证数据consistency

Locking
vs
Multi **V**ersion **C**oncurrency **C**ontrol

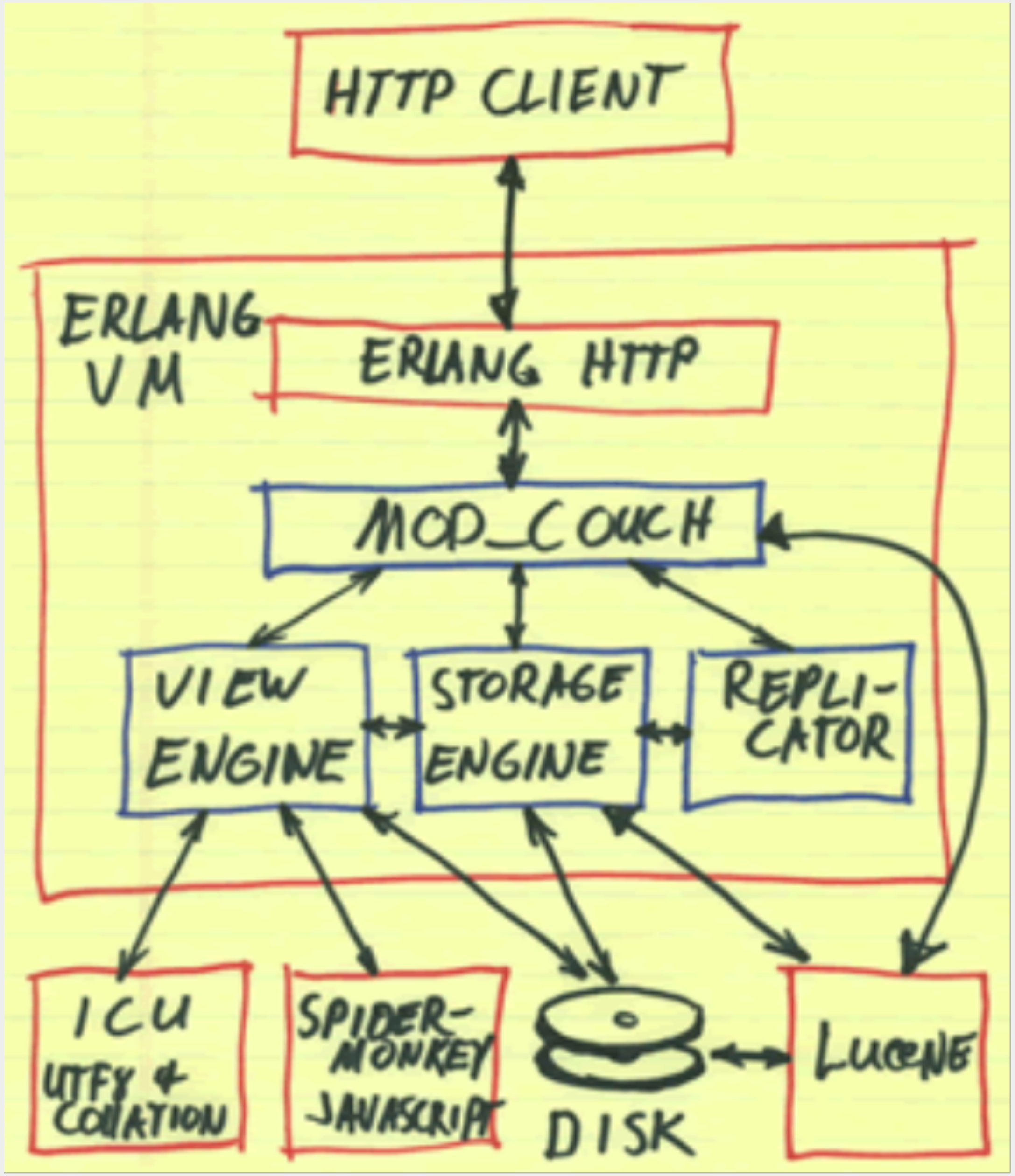


Attachment

- Inline Attachments
  附件内嵌在document中
  以base64进行编码

- Standalone Attachments
  附件不是以text形式写入document
  保存于storage文件中
  Create － HTTP PUT
  Change － HTTP PUT携带rev信息
  Delete - HTTP DELETE 携带rev信息

CouchDB Internal



CouchDB Internal - Storage

- 数据库文件的后缀为.couch，由Header和Body组成
- magic code:$g, $m, $k, 0
- 头部长度为2048 ＊ 2，由两个完全相同的header信息组成
- header主要包含root节点信息
- 数据是以B＋Tree的方式进行存储
- 分为kv node和kp node
- 对某个数据进行修改后，从被修改数据开始，向root节点遍历，将所有更新的数据append到couch文件中

CouchDB Internal - compaction

- 由于 采用Append的方式, 以磁盘空间换取性能
- couch文件大小会迅速增长， 需要进行compaction
- 创建dbname. compact文件， 进行db初始化
- 从当前db中拷贝最新数据到compact文件
- CouchDB依旧可用
- 占用较多空间， 需要在系统负载较低时进行
- compaction完毕后， 数据切换

CouchDB Internal - Replication

- document and design can replicate
- incremental replication，只有更新的内容进行同步
- 支持自定义函数进行replication过滤，获取某个数据子集
- 通过seq num获取最新数据
- continues replaction，监视"_changes" API，所有数据更新进行即是更新

CouchDB Internal - View

- Default View Language : Javascript
- Js engine : Mozila SpiderMonkey
- couchjs /server/main.js 启动view server
- db更新时，通过couch_view_updater更新view
- view 初次请求时，比较耗时，需要声称view index

Erlang介绍

- Ericsson创建, 最初用于电信系统开发
- 函数式编程语言(FP)
- 面向并发(OC),基于消息
- 内建分布式系统支持
- SMP支持, 充分利用多核特性
- 提供OTP, 简化规范应用开发
- 成熟, 稳定, 具有20多年历史
- 适于电信系统, 分布式系统, 高并发服务器
- Open Source, 跨平台

Erlang带来的惊喜

- 短短几行代码开发一个分布式应用
- 不用为内存苦恼（泄露，越界，非法访问），GC自动回收内存
- 利用OTP模型, 书写必要的callback函数, 构建一个服务
- 面向并发，轻量级进程大大简化编程复杂度
- 同样的代码（bytecode),运行于多种平台, 轻松拥有多核优势
- 代码热替换，可以在不停止服务的情况下进行更新
- 丰富的测试，诊断，监控工具，完全了解系统的内部状态
- 开源, 任何时候出现问题，去代码中grep
- ...

Erlang带来的一些"抱怨"

- FP函数式编程，与OO背道而驰，形同陌路
- Single Assign单次赋值，匪夷所思
- Record语法丑陋无比
- string使用list进行表现，缺乏binary的优势
- ...

所有这些“抱怨”大部分是停留在语法层面
看到她的 脸，以为读懂她的心

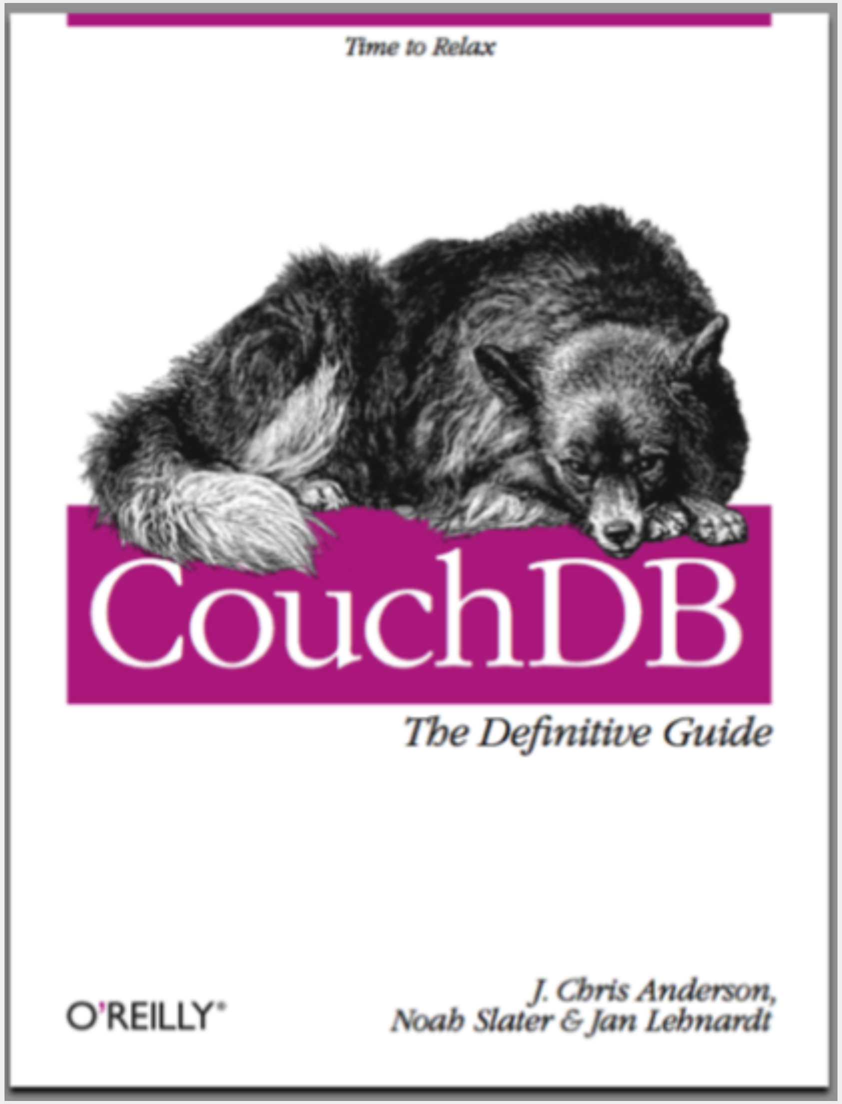
Erlang前面的惊喜才是真正有意义的东西.

资源推荐

blog:
http://mryufeng.javaeye.com
http://www.joeandmotorboat.com/
http://damienkatz.net/
http://jchrisa.net/
http://jan.prima.de/plok/
http://spawnlink.com/
http://userprimary.net/user/

website:
http://www.erlang.org
http://www.erlang-china.org
http://erlang-factory.com/
http://www.erlangplanet.org/
http://wiki.apache.org/couchdb/

Books



http://books.couchdb.org/relax/

CouchDB & Erlang

Relax!

Thanks everyone!