# 海量小文件存储服务的开发

张凯朝

2014,04,30

# 内容

- 需求
- 选型
- 设计
- 实现
- 问题
- 待续

# 需求

- 单个文件小于 2MB 的小文件，海量为 100W 级别
- 支持分布式存储
- HTTP 的文件读取 API
- 上传 API
- 监控 API，查询存储服务工作状态和剩余存储空间
- benchmark

# 选型

- 存储引擎： Beansdb 及其代理 Beanseye
- 存储服务器连接客户端： python-memcached
- web 框架： Morepath
- web 服务器：各种 WSGI 服务器即可，比如 gevent 的 WSGIServer, Tornado, meinheld, gunicorn, Apache 的 mod_wsgi, 等等，开发的时候使用标准库 wsgiref （ morepath 默认）

# 存储引擎

- 为什么是 Beansdb
  * 基于 memcached 文本协议，接口简单
  * 配置简单，运行维护简单
  * 官方豆瓣网成功案例，性能可以接受
  * 问题：弱一致性、手动同步、等
  * ……
- 为什么使用了 Beanseye 作为存储代理
  * 同样基于 memcached 文本协议
  * 同样配置简单，运行简单
  * Beansdb 中的 Python 客户端不完整
  * 后台存储节点配置存储在代理中，客户端不需要依赖于此
  * ……

# Beansdb/Beanseye 测试

- 配置 Beansdb 节点
- 配置 Beanseye 代理
- 使用 memcached 客户端
  * 10,000 个 1.5MB - 2MB (6 nodes)
  * 1,000,000 个 5KB - 10KB (6 nodes)
  * ……

# Web 框架

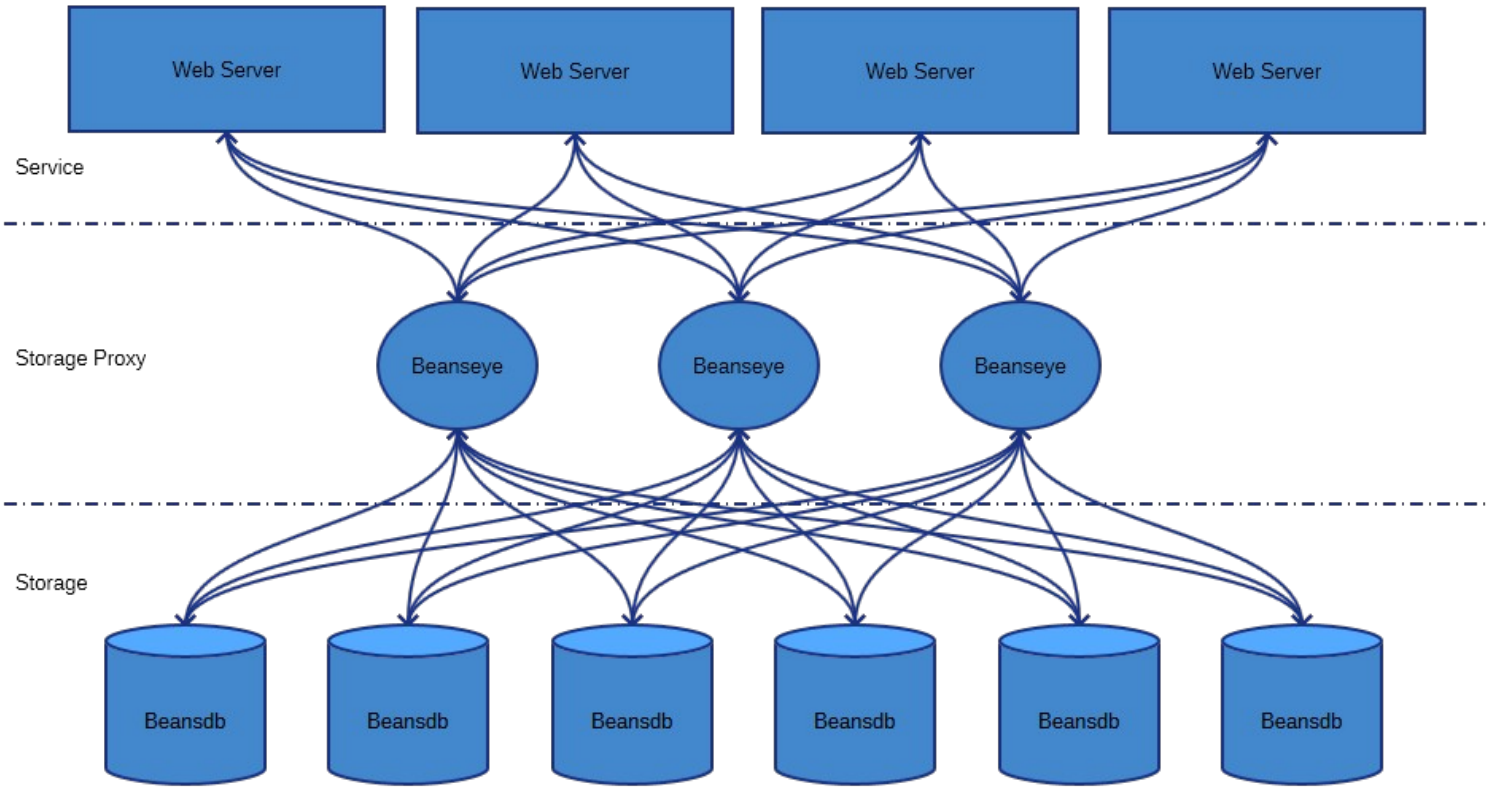
- 为什么采用 Morepath
  * 设计上的亮点，项目维护
  * 应该够用，功能、性能
- 存在问题
  * 当时还不支持 Python 3
  * 后来反应几个问题
    * RESTful URL 风格 bug
    * 无法限制请求大小
    * ……

# 设计

- 三层结构
  * 存储服务器节点
  * 存储服务器代理
  * Web 服务器

  每一层都应该是可水平扩展的，无状态的。
- 服务接口

# 三层结构


Web Server
Web Server
Web Server
Web Server
Service
Storage Proxy
Beanseye
Beanseye
Beanseye
Storage
Beansdb
Beansdb
Beansdb
Beansdb
Beansdb
Beansdb

# 服务接口设计

- 参考 Dropbox 的 Core API
- 草稿：见 README.rst
  * PUT /files?path=abc/def/ghi.ext&param=val...
    200 OK
  * POST /files?path=abc/def/ghi.ext&param=val...
    201 Created
  * GET /file?path=abc/def/ghi.ext&param=val...
    200 OK
    ...
  * GET /status?param=val...
    200 OK
    ...

# 实现

- 满足 Morepath 的 Model/Path/File 设计
- 满足 memcached 文本协议的文件名编解码
- 重写

# 满足 Morepath 设计

- Model
  * Filestorage
  * File
- Path
  * /files?path=path/to/file
  * /file?path=path/to/file
- View
  * GET -> File
  * POST -> Filestorage
  * PUT -> File

# 满足 memcached 文本协议设计

- 限制： key 字符，长度
- 对应处理
  * 使用 UTF-8 编码
  * 使用 base64 编码

# 重写

- 为什么重写
  * Morepath 不支持 Python 3 ，缺少功能
    改用直接手写纯 WSGI 应用
    （ Morepath 0.2 开始支持 Python 3 了，再来？暂不。）
  * python-memcached 不支持 Python 3
    改用 aiomemcache
- 为什么切换到 Python 3
  * 新项目
  * asyncio
- 服务接口变更
  * 上传文件接口请求体变化
    类 Dropbox

# asyncio

- 优势
  * 在 Python 3.4 开始成为标准库
  * 有测试表明性能非常好
  * 精确的编程模型，而不是像 gevent 那样通过 monkey patch 隐藏异步细节
- 存在问题
  * 一旦采用，到处需要使用相同的异步风格比如 yield from

# 重写时优化

- 文件上传时候文件流限制
  * 如果请求中有 Content-Length 头部
  * 每小块每小块接收，超过大小返回 416

# 服务接口的客户端

- 使用 aiohttp 的 HTTP client
  * 统一使用，也可用 urllib2/requests 等

# 问题

- 成了 memcached 协议，败也可能是 memcached 协议。每次都是一次性把整个 value 放在内存中，不知道二进制协议有没有支持 streaming ？
- ……

# 待续

- 重写之去掉 Beanseye 存储服务代理层
  * Beanseye -> 基于 Memcached 的代理
  * sfss -> 基于 HTTP 的代理？
  * 使用 Beansdb 的 Python 客户端（ fnv1a ）和 aiomemcache
- 单元测试
- Benchmark
  * 使用服务接口的客户端
  * 多线程
- 慢慢演化……

# 参考


- Beansdb 开发日记之卷土重来
  http://www.douban.com/note/122507891/
- Beansdb 旧项目地址
- http://code.google.com/p/beansdb/
- Beansdb

  https://github.com/douban/beansdb
- Beanseye

  https://github.com/douban/beanseye
- Morepath
  https://github.com/morepath/morepath
- Morepath design
  http://morepath.readthedocs.org/en/latest/design.html
- Morepath 作者博客

  http://blog.startifact.com/
- Morepath Python 3 support
  http://blog.startifact.com/posts/morepath-python-3-support.html
- Asyncio

  http://docs.python.org/3.4/library/asyncio.html
- aiohttp
  https://github.com/KeepSafe/aiohttp/
- Aiomemcache

  https://github.com/KeepSafe/aiomemcache
- ……