**移动应用后端服务框架的设计和实现**

**鞠强**

# 摘要

随着移动互联网技术的崛起，继SaaS（软件即服务：Software as a Service）、IaaS（基础设施即服务：Infrastructure as a Service）和PaaS（平台即服务：Platform as a Service）之后，BaaS（后端即服务：Backend as a Service）生态系统正从一个小众垂直领域迅速成为非常重要的行业环节。本文从理论和实践两个角度出发，研究native的移动开发（以IOS、Android为例）过程中存在的主要问题，学习后端即服务的产业现状和实现技术，并且设计和实现一个简单的移动应用后端服务框架。

本文首先较为详细地介绍了在本系统中运用到的知识背景。接着对Android和IOS应用开发中数据存储传输方面存在的共同问题进行探讨，对其解决之道“后端即服务生态系统”的现状进行研究。而后对设计和实现一个简单的移动应用后端服务框架的可行性进行了分析，同时简要介绍了系统中所用到的开发语言Ruby和数据库Mongodb的特点以及使用方法。对系统的设计思想、设计目标与系统的整体结构进行了明确的规划。最后对框架的设计与实现作了较为详细的讲解。

系统涉及的主要技术有Ruby on Rails和相关的部署技术、Mongodb和相关的分布式数据库技术、Java和Android SDK、Objective-c和Cocoa SDK。其主要功能有：移动应用数据远程存储和本地缓存服务，一站式的移动应用后台搭建，消息推送服务，简单的统计服务。

论文在撰写过程中，力求将理论与实践应用相结合，提高系统的可用性和通用性，希望通过本文的论述能够更充分地体现出后端即服务的设计思想以及上述理论和技术在本系统中的应用。

**关键字**：移动、框架、后端即服务

# Abstract

As the rapid development of mobile Internet,following the SaaS (Software as a Service), IaaS (Infrastructure as a Service) and PaaS (Platform as a Service) BaaS (Backend as a service) ecosystem from a vertical niche areas quickly become a very important industry links. From both theoretical and practical perspectives, we will study the main problems in native mobile development (IOS, Android, for example) process, learning the backend that service the industry status quo and implementation techniques, and design and implement a simple back-end services framework for mobile application.

We will firstly introduce the background knowledge in more detail used in this article. Then we will discuss common problems to Android and IOS application development in data storage transmission to study the status of the "(Backend as a service" of its solution. After that，we’d like to analyze the feasibility of the design and implementation of a simple mobile application framework for back-end services, while brief introduction to the system used in the development of language Ruby and databaseMongodb characteristics and usage. Clear planning system design thinking, design goals and overall system structure. Finally, the design of the framework and made ​​more detailed explanation.

The system involves technologies are Ruby on Rails, Mongodb, Java + AndroidSDK, and Objective-c + CocoaSDK. Its main function: remote data storage and local caching services of mobile applications, building a one-stop mobile application background, message push service, simple statistical services.

We strive to make the system be the combination of theory and practical and hashigh usability and versatility. I hope to be able to more fully reflect the the backend that service design thinking and the theory and technology through the discussion of this paper in the system application

**Keywords**:mobile,framework,backend as a service.

# 目 录

[摘要 - 1 -](#_Toc353378015)

[Abstract - 2 -](#_Toc353378016)

[目录 - 3 -](#_Toc353378017)

[第一章绪论 - 4 -](#_Toc353378018)

[1.1 课题背景 - 4 -](#_Toc353378019)

[1.2移动应用开发技术的发展 - 4 -](#_Toc353378020)

[1.3 开发者面临的问题 - 4 -](#_Toc353378021)

[1.4“后端即服务”所提供的服务和解决方案 - 5 -](#_Toc353378022)

[1.5 本文的组织结构和创新点 - 5 -](#_Toc353378023)

[第二章技术背景及相关知识简介 - 5 -](#_Toc353378024)

[2.1 开发环境 - 5 -](#_Toc353378025)

[2.1.1 Ruby语言简介 - 6 -](#_Toc353378026)

[2.1.2 Mongodb简介 - 6 -](#_Toc353378027)

[2.1.3Android SDK - 7 -](#_Toc353378028)

[2.1.4Cocoa SDK - 7 -](#_Toc353378029)

[第三章移动开发后端即服务现状 - 8 -](#_Toc353378030)

[3.1 移动开发分工细化催生后端即服务 - 8 -](#_Toc353378031)

[3.2 当前后端即服务的服务内容概述 - 8 -](#_Toc353378032)

[3.3 分析几个典型的移动后端服务厂商 - 8 -](#_Toc353378033)

[3.3.1 StackMob - 8 -](#_Toc353378034)

[3.3.2 局部自适应二值化方法 - 12 -](#_Toc353378035)

[3.4 二值化方法小结 - 15 -](#_Toc353378036)

[第四章设计和实现一个移动应用后端服务框架 - 15 -](#_Toc353378037)

[4.1 本章概述 - 15 -](#_Toc353378038)

[4.2 基于背景表面的文本图象二值化方法 - 16 -](#_Toc353378039)

[4.2.1 粗略的估计前景区域 - 17 -](#_Toc353378040)

[4.2.2 获取背景区域[2] - 18 -](#_Toc353378041)

[4.2.3 提取文字部分 - 18 -](#_Toc353378042)

[4.2.4 后期处理[27] - 19 -](#_Toc353378043)

[4.2.5 实验结果 - 19 -](#_Toc353378044)

[4.3 Otsu算法和Bernsen算法相结合的二值化方法 - 20 -](#_Toc353378045)

[4.3.1 超限邻域平均法[28] - 23 -](#_Toc353378046)

[4.3.2 使用Otsu算法进行全局二值化 - 23 -](#_Toc353378047)

[4.3.3 使用Bernsen算法进行局部二值化 - 24 -](#_Toc353378048)

[4.3.4 后期处理 - 24 -](#_Toc353378049)

[4.3.5 实验结果 - 24 -](#_Toc353378050)

[第五章通用性和可用性讨论 - 26 -](#_Toc353378051)

[第六章结论 - 28 -](#_Toc353378052)

[6.1 两种新的文本图像二值化方法总结 - 28 -](#_Toc353378053)

[6.2 文档图像二值化的研究现状及存在的问题 - 28 -](#_Toc353378054)

[6.3 文档图像二值化的发展趋势 - 29 -](#_Toc353378055)

[致谢 - 29 -](#_Toc353378056)

[参考文献 - 30 -](#_Toc353378057)

# 第一章绪论

## 1.1 课题背景

据《中国互联网络发展状况统计报告》显示，到2012年12月底，中国互联网用户规模达到5.64亿，其中，移动互联网用户达到4.2亿。截止2012年底，中国移动互联网用户数量为4.2亿，年增长率为18.1%，增幅超过整体互联网用户增长幅度。在所有互联网用户里，使用移动设备接入网络的用户比例由69.3%提升到74.5%。移动终端无疑已成为互联网入口的第一大终端。

随着移动互联网技术的发展和移动应用开发技术的成熟，移动应用开发过程中分工细化的趋势越来越明显，移动应用后端即服务生态系统正从一个小众垂直领域迅速成为非常重要的行业环节。

## 1.2移动应用开发技术的发展

为了顺应移动互联网的快速发展，移动开发领域也经历了一代又一代的尝试。这些尝试包括最初个人英雄主义的本地应用开发、Flash、html5、widget、移动中间件等。

## 1.3 开发者面临的问题

经过了上述的实验和考证后，人们发现移动开发难度远比PC开发的难度高很多，《神庙逃亡》、《愤怒的小鸟》等神话毕竟是少数，小团队和独立开发者越来越难在移动应用生态系统中获得一席之地。从WAP、原生开发等各式各样的移动开发方式和平台在移动领域无攻而返时，人们开始思考到底什么样的开发方式更能高效、快捷适应移动产业的需要。

当Html5、Widget横空出世火速增援的时候，我们似乎看到了移动蓝海的救星，可是好景不长，Facebook放弃Html5另觅他路，让Html5的弱点暴露无疑，具备一定开发能力的厂商也发现Html5并非为移动开发而生，好多在移动中实现问题还有待解决，Widget也远没有如传说中的那样趟平整个移动领域，移动中间件则经过多年的锤炼一直拥有良好的口碑和解决问题的能力。可是需要学习第三方的语言和开发原理才能应用自如，学习曲线和难度之大使得开发者敬而远之，并且市场上主流的移动中间件产品在技术上采取全面封锁措施，对开发者采取封闭状态，同时更多的软件厂商也不愿把关系产品命脉的赌注，全部押给并不开放的移动中间件的手里，这也是目前移动中间件所面临的四面高歌却孤立冷清的局面。

## 1.4“后端即服务”所提供的服务和解决方案

后端即服务的移动解决方案，让繁琐的移动开发过程更简单快捷，也使得移动开发领域分工更加明确，让android/IOS等移动开发者无需掌握任何一门服务器开发语言(JAVA、PHP)，就可以按照提供的android/IOS/WP三个平台的SDK内置的预定义功能接口快速开发移动应用。解决当前原生开发难以解决、耗时耗力的数据存储、文件存储、消息PUSH等工作。

## 1.5 本文的组织结构和创新点

本文共分为六章，具体内容结构安排如下:

第一章绪论部分。介绍论文的研究背景和研究意义，阐述了论文的大体结构。

第二章简单介绍了毕业设计的技术背景及相关知识。

第三章介绍当前后端即服务的发展情况。

第四章设计和实现了一个简单的移动应用后端服务框架

第五章讨论了设计框架的通用性和可用性

第六章总结。

# 第二章 技术背景及相关知识简介

## 2.1 开发环境

* 服务端使用Ruby和Node.js
* Android端使用Java和Android SDK
* IOS端使用Objective-c和Cocoa SDK
* 使用Mongodb进行数据存储
* 使用NetBeans、Xcode、Eclipse作为集成开发环境
* Windows和Mac下进行开发，服务端部署在Ubuntu的Linux上

### Ruby语言简介

Ruby，一种为简单快捷的[面向对象](http://baike.baidu.com/view/125370.htm)编程（[面向对象程序设计](http://baike.baidu.com/view/249254.htm)）而创的[脚本语言](http://baike.baidu.com/view/76320.htm)，在20世纪90年代由日本人[松本行弘](http://baike.baidu.com/view/1438214.htm)（まつもとゆきひろ/Yukihiro Matsumoto）开发，遵守GPL协议和Ruby License。它的灵感与特性来自于 Perl、Smalltalk、Eiffel、Ada 以及 Lisp 语言。由 Ruby 语言本身还发展出了JRuby（Java 平台）、IronRuby（.NET 平台）等其他平台的 Ruby 语言替代品。Ruby的作者于1993年2月24日开始编写Ruby，直至1995年12月才正式[公开发布](http://baike.baidu.com/view/602345.htm)于fj（[新闻组](http://baike.baidu.com/view/834.htm)）。因为Perl发音与6月诞生石pearl（珍珠）相同，因此Ruby以7月诞生石ruby（红宝石）命名。

在开发层面，Ruby具有以下优点：

* 语法简单
* 普通的[面向对象](http://baike.baidu.com/view/125370.htm)功能(类,方法调用等)
* 特殊的[面向对象](http://baike.baidu.com/view/125370.htm)功能(Mixin,特殊方法等)
* [操作符重载](http://baike.baidu.com/view/1033032.htm)
* 错误处理功能
* [迭代器](http://baike.baidu.com/view/925158.htm)和[闭包](http://baike.baidu.com/view/648413.htm)
* [垃圾回收](http://baike.baidu.com/view/159846.htm)
* 动态载入(取决于系统架构)
* 可移植性高.不仅可以运行在多数UNIX上,还可以运行在DOS,Windows,Mac,BeOS等平台上
* 适合于快速开发，一般开发效率是JAVA的5倍

### 2.1.2 Mongodb简介

MongoDB是一个基于分布式文件存储的数据库。由C++语言编写。旨在为WEB应用提供可扩展的高性能数据存储解决方案。

MongoDB是一个介于[关系数据库](http://baike.baidu.com/view/68348.htm" \t "_blank)和非关系数据库之间的产品，是非关系数据库当中功能最丰富，最像关系数据库的。他支持的数据结构非常松散，是类似json的bson格式，因此可以存储比较复杂的数据类型。Mongo最大的特点是他支持的查询语言非常强大，其语法有点类似于面向对象的查询语言，几乎可以实现类似关系数据库单表查询的绝大部分功能，而且还支持对数据建立[索引](http://baike.baidu.com/view/262241.htm" \t "_blank)

MongoDB具有以下特点：

* 面向集合存储，易存储对象类型的数据。
* 模式自由。
* 支持动态查询。
* 支持完全索引，包含内部对象。
* 支持查询。
* 支持复制和故障恢复。
* 使用高效的二进制数据存储，包括大型对象（如视频等）。
* 自动处理碎片，以支持云计算层次的扩展性。
* 支持[RUBY](http://baike.baidu.com/view/45135.htm)，[PYTHON](http://baike.baidu.com/view/21087.htm)，[JAVA](http://baike.baidu.com/view/29.htm)，[C++](http://baike.baidu.com/view/824.htm)，[PHP](http://baike.baidu.com/view/99.htm),[C#](http://baike.baidu.com/view/6590.htm)等多种语言。
* 文件存储格式为BSON（一种JSON的扩展）。
* 可通过[网络](http://baike.baidu.com/view/3487.htm)访问。

### 2.1.3Android SDK

Android开发工具包，为Android手机开发人员提供手机开发的Java接口，从2009年的1.5版本至今，Android SDK已经发布了十数个版本。

### 2.1.4Cocoa SDK

Cocoa是苹果公司为Mac OS X所创建的原生面向对象的[编程环境](http://baike.baidu.com/view/1405990.htm)，是Mac OS X上五大API之一（其它四个是Carbon、POSIX、X11和Java）。

苹果的[面向对象](http://baike.baidu.com/view/125370.htm)开发框架，用来生成 Mac OS X 的应用程序。主要的开发语言为 Objective-c, 一个c 的超集。 Cocoa 开始于1989年9月上市的NeXTSTEP 1.0，当时没有Foundation框架，只有动态[运行库](http://baike.baidu.com/view/1032404.htm)，称为 kit, 最重要的是AppKit. 1993 年NeXTSTEP 3.1 被移植到了 Intel, Sparc, HP 的平台上，Foundation 首次被加入，同时Sun 和 NeXT 合作开发OpenStep也可以运行在Windows 系统上。

Cocoa 应用程序一般在苹果公司的开发工具Xcode（前身为Project Builder）和Interface Builder上用Objective-C写成。

# 第三章 移动开发后端即服务行业现状

## 3.1 移动开发分工细化催生了后端即服务

开发一个需要联网的移动客户端应用是复杂的。开发者不仅需要掌握一门客户端语言（java、objective-c等），了解相应开发工具包（Android SDK、Cocoa SDK等）还需考虑服务器编程和数据库技术；除了需要关注用户交互、产品逻辑，还需分神于数据存储、网络交互、推送服务、数据统计和用户行为分析。随着移动开发分工的细化，以方便移动客户端开发部署为目的的移动应用后端即服务概念应运而生。

## 3.2 当前后端即服务的服务内容概述

当前已知已有二十多家国外公司以及几家国内公司提供移动应用后端服务，综合看来，提供的服务主要包括以下方面：

* 数据存储
* 推送服务
* 地理位置
* 社交整合
* 统计和用户行为分析

## 3.3 分析几个典型的移动后端服务厂商

### 3.3.1 StackMob

StackMob是最早进入移动应用后端服务领域的厂商，成立于2010年。月访问量超过300万，为IOS、Android等应用提供开发SDK和部署支持。同时提供自定义代码接口，以方便开发者进行后端逻辑的扩展。

StackMob提供了较为全面的功能性服务。包括：数据存储、推送服务、地理信息、较色管理等。

以从零开始依靠StackMob开发一个Android应用的角度看去，开发一个Android应用包括以下步骤：

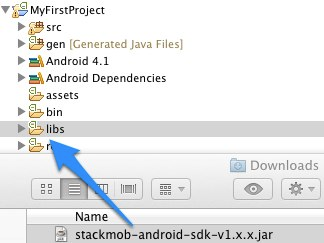
**（1） 安装安卓开发环境**

后端服务不同于html5或者原理类似的应用生成工具，依然需要传统的本地开发环境，从Eclipse官网下载Eclipse，从Android开发者中心下载适应版本的Android SDK。完成后建立本地Android工程。

**（2） 注册StackMob账号和添加其SDK**

在StackMob网站注册账号，获得自己的密钥字符串。此密钥字符串作为服务器本地应用的标识。

从StackMob网站下载jar包，拖入libs文件夹中，android ADT将其自动引入项目。



**（3） 简单的配置**

在AndroidManifest.xml配置文件中设置StackMob必须的网络权限，并在程序启动处加入初始化代码指定应用密钥

**（4） 扩展数据模型基类构建自己的数据模型**

传递给StackMob后端的数据只有被标示为StackMobSDK所定义的数据基类（StackmobModel）的子类才能被接收和识别，通过扩展StackMobModel构建符合自己应用需求的数据模型，这个过程是十分自由的，构建过程如同构建通常的java model类，无需额外配置。

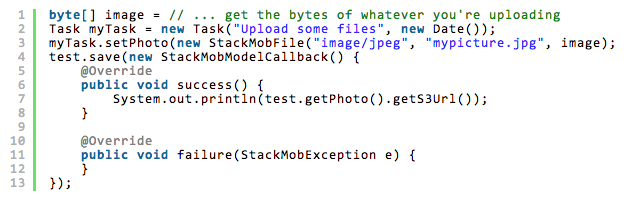
StackMob支持最多三层嵌套的数据模型。

**（5） 存储和使用云端数据**

至此StackMob的简单配置工作已经完成。分别调用model的save与getDataStore方法即可方便地从远端读取数据。这个过程使得服务器端数据库的建表、指定数据类型、建立索引等操作对移动开发者透明。

**（6） 文件存储**

文件存储方面，StackMob封装了亚马逊 S3文件存储服务，开发者可以在StackMob后台配置自己的S3账号，然后即可在应用中使用简单的代码实现文件的云端存储。存储一张图片仅需要编写很少的代码：

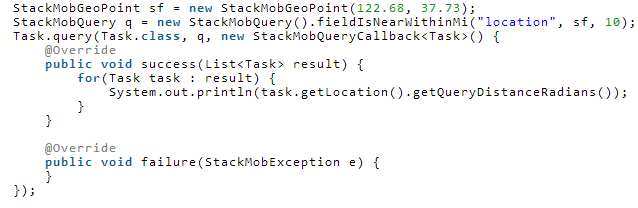


**（7） 消息推送**

类似使用S3实现实现文件存储功能，StackMob直接使用了GCM（Google Cloud Message）实现自身的推送功能。这一方面提高了推送服务的覆盖面和可靠性，另一方面却增加了开发者的关注点和配置的工作量。

**（8） 地理位置服务**

StackMob的地理位置服务是对手机位置信息服务的封装，提供了简单的位置模型StackMobGeoPoint，并在服务端封装和位置相关的算法，寻找十米之内的设备的方法在此框架中可以很容易得通过少量代码写成：



### 3.3.2 Parse

Parse成立于2011年，是国外又一大移动应用后端服务提供商。有2万名开发者使用其平台，40万款应用使用其服务。支持的API有iOS、Android、JavaScript、Windows 8，以及提供了REST API和Cloud Code JS API，作为后来者，Parse提供了与StackMob类似的后端服务，下面从几个方面简单介绍Parse的独特之处。

**（1） 数据存储**

Parse的数据存储模式不如StackMob灵活，所有的存储类型都必须是ParseObject所包含的键值对，并且通过ParseObject中的save、get等方法处理。

这种处理办法可以提高解析效率，也降低了服务端的复杂程度，在开发关于简单数据结构模型的应用时更加方便快捷。但提高了Model和Controller的耦合性，在开发复杂模型相关的应用时，存在着模型管理困难和移植成本高的风险。如此的情境，可能会增加开发对移动应用后端服务提供商的依赖性。

**（2） 自定义后台**

与StackMob一样，Parse提供了对用户自定义后台逻辑的支持。所不同的是，StackMob提供了Java的接口，而Parse用户以JS的形式编写自定义后台函数。例如定义：



则客户端即可调用名为hello的服务端方法，打印“Hello World！”字符串。

**（3） 文件存储和权限管理**

Parse的文件存储并未借用第三方服务，而是使用自己的文件服务器，这从一定程度上增强了基于Parse开发过程的流畅性。开发者可以直接使用统一的后端厂商和数据接口进行开发。

此外访问控制方面Parse相对StackMob也更为灵活，提供了细化到对象的访问控制管理。

### 3.3.3 国内移动后端服务现状

国内目前尚无StackMob或Parse一样提供如此功能覆盖全面、支持终端众多的移动后端服务平台，但是也有一些功能相对简单或更注重领域性的移动后端服务提供商正在发展。

**（1） Bomb**

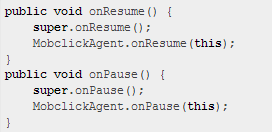
Bomb是广州鹏锐公司的产品，产品诞生于2012年，是国内第一个Baas平台，目前仅支持Android手机应用的后端服务。

Bomb提供了与Parse非常类似的数据存储服务和SDK接口，同样采用名值对的形式将数据包装成BombObject，继承了Parse简单易用优点的同时，也同样存在耦合性高、不易于管理复杂对象和迁移成本大的缺点。

同时作为一款本土的移动后端服务平台，Bomb在SDK中集成了支付功能，目前支持十几种游戏点卡、充值卡的在线支付功能。

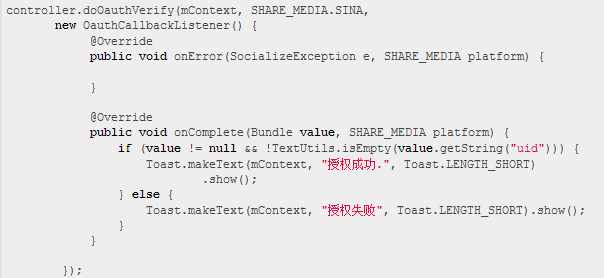
**（2） 友盟**

友盟是一个以应用统计起步的移动后端服务提供商，在2010年推出了第一个友盟统计产品。目前在Android、IOS、Windows三个移动平台提供数据统计、社交网络集成、移动广告集成、用户行为分析等服务。

在上述服务中，友盟最以数据统计服务闻名。仅需在应用工程中添加少量的代码（如右图）即可实现SDK自带的崩溃日志捕获和使用时长统计的功能。同时友盟SDK支持用户自定义事件，分别在后台界面设置事件类型、名称即可在移动端发送相应统计。

除去专业的数据统计分析，友盟的社交网络集成服务也很有特色。目前国内各大主流社交网络都提供了一定程度上的开放接口和SDK供开发者调用，众多的SDK与不同风格的接口代码，使在应用中简单整合社交网络成了一项复杂的工作。应用中很简单的连接一个社交网站，考虑界面、表情包等因素，动辄需要几百行代码。将不同社交网站以统一风格整合在APP中更是一个不小的工作。

友盟所提供的社会化组件很好得解决了这个问题，开发者只需将在各社交网络开放平台取得的API KEY填入友盟的统一后台，即可在应用中很方便地链接对应网站，也无需考虑界面风格不统一、代码风格混乱或API过期的问题。



**（2） 百度开发者中心**

依托自身技术积累，百度为移动开发者提供了较为个性的后端服务，包括：百度个人云存储服务、百度地图服务、百度移动统计服务、社会化网络整合服务等。

上述服务中，在国内最有特色的应属个人云存储服务。百度云为移动端个人云存储提供了易用的REST接口，以上传单个文件为例，客户端通过post方式发送传送请求：

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名称** | **类型** | **是否必需** | **描述** |
| method | string | 是 | 固定值，upload。 |
| access\_token | string | 是 | 开发者准入标识，HTTPS调用时必须使用。 |
| path | string | 是 | 上传文件路径（含上传的文件名称)。 |
| file | char[] | 是 | 上传文件的内容。 |
| ondup | string | 否 | * overwrite：表示覆盖同名文件； * newcopy：表示生成文件副本并进行重命名，命名规则为“文件名\_日期.后缀”。 |

服务器则以JSON形式返回相应结果：

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名称** | **类型** | **UrlEncode** | **描述** |
| path | string | 是 | 该文件的绝对路径。 |
| size | uint64 | 否 | 文件字节大小。 |
| ctime | uint64 | 否 | 文件创建时间。 |
| mtime | uint64 | 否 | 文件修改时间。 |
| md5 | string | 否 | 文件的md5签名。 |
| fs\_id | uint64 | 否 | 文件在PCS的临时唯一标识ID。 |

## 3.4 行业现状小结

综上所述，当前移动应用后端服务行业在国外出现了像StackMob、Parse这一类提供多平台、综合服务的专业服务提供商，而国内Baas服务的实践者多是创业型公司，无论是提供服务的平台普及率或是功能覆盖面都还有较大差距。而国内技术实力较强的大公司，在移动应用后端服务领域多注重自己的专业方面。

综合看来，当前移动应用后端服务所提供的服务，功能上，主要包括数据存储、推送服务、地理位置、社交整合、统计和用户行为分析等；结构上，普遍由一个服务平台以及多个客户端平台的开发工具包构成。更少的代码、更便利的后端服务，使开发者关注于前端业务逻辑和交互，而非后端技术，是各大后端服务产品统一的追求。

# 第四章 设计和实现一个移动应用后端服务框架

## 4.1 本章概述

本章将在上述研究和总结的基础上设计和实现一个简单的移动应用后端服务框架，框架以ruby、Mongodb搭建服务端，且分别提供对Android、IOS两个平台的支持。使开发这在此框架功能下，可以方便地进行移动客户端与后台的数据交互、日志传送、统计分析、消息推送等功能。

本章从：系统设计原则、总体结构介绍、关键技术讨论、系统使用和部署流程四个方面展开叙述。

## 4.2 系统设计原则

本系统纵向程度较大，为使系统的设计和实现过程是可以被把握的，在设计之初确定以下三个简单的设计原则，这三个简单的原则将贯彻体现在下文中：

1. 少做重复工作。积极复用久经考验的框架、技术和协议来保证开发效率和系统稳定，积极使用最新技术提供的新特性来提高系统的运行效率和性能。
2. 用合适的工具和方法。用合适的工具和方法做特定的事，综合利用各种技术解决问题。避免出现“如果你有的只是锤子，你会把所有问题看成钉子”这样的情况出现。
3. 保持干净和简单。避免过度设计，复杂过程透明，使系统对开发人员友好，便于部署、易于使用。

## 4.3 总体结构介绍

系统面向移动应用开发者，提供数据存储、统计分析、通用逻辑处理（用户、位置信息、文件IO）和推送服务

### 4.3.1 数据存储服务总体结构

数据存储服务依托文档型数据库Mongodb，总体为数据库-服务器-客户端三层结构。服务器是一个RESTFUL的Ruby on Rails模块，客户端和服务器通过REST协议进行通信。

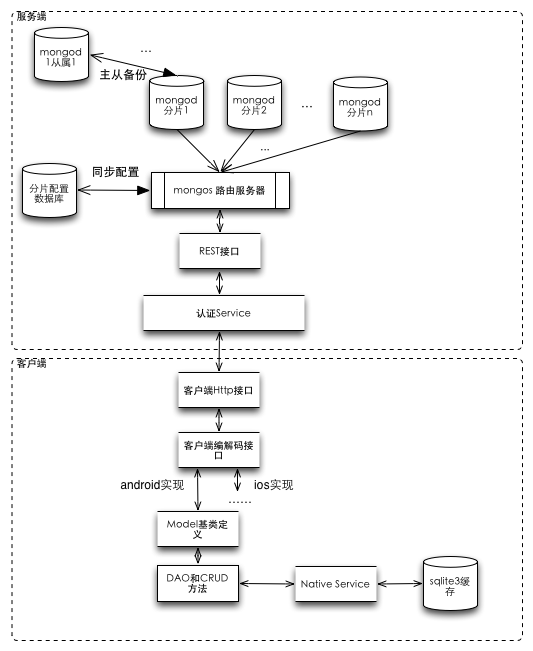
为做到安全和效率的平衡，REST协议通过HMAC方法进行身份认证，HAMC的方法对REST进行认证是很多通过REST接口向外提供服务的后端厂商的选择，例如Amazon的S3云存储服务，就使用了这种方法。

为提高存储效率和提升数据安全稳定性，Mongdb使用数据库分片技术进行分布式部署，使用主从备份的方式进行安全保证。分片集群的规模和备份策略的选取被封装成简单和可配置的。

在客户端，定义了对象序列化反序列化接口、http方法、对REST API封装的接口，然后分别根据不同客户端平台特征进行了各自实现，这三部分构成了客户端SDK数据存储方面的底层。

面向开发者，客户端SDK提供便于扩展的Model基类和通用的Data Access Object（DAO）及其相关方法，DAO提供Remote和native的两套实现，使得开发者可以在本地缓存和远端存储之间进行选择。

数据存储服务的总体结构示意图如下：



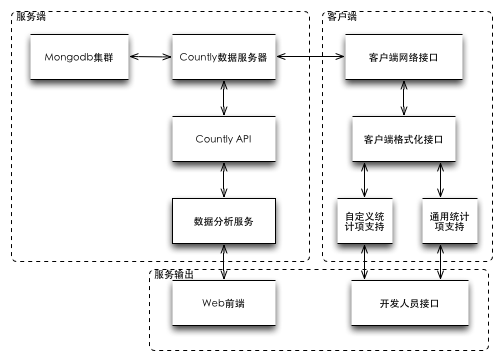
### 4.3.2 统计服务

统计服务围绕Countly进行整合和开发，Countly是目前最强大的开源移动分析应用。统计服务在Countly-server的基础上构建统计服务的监听器和前端，使用统一的数据库进行数据存储。

在客户端，根据安卓和IOS的不同特性，编写各自的统计SDK，对移动应用的生命周期、使用时长、日活率、版本分布、用户分布等通用信息统计项进行默认支持，同时，为用户提供自定义事件统计的相关支持。

在服务端，围绕Countly统计数据，在Countly API的基础上进行开发，构建可视化的配置界面和Web前端。

统计服务的系统结构示意图如下：



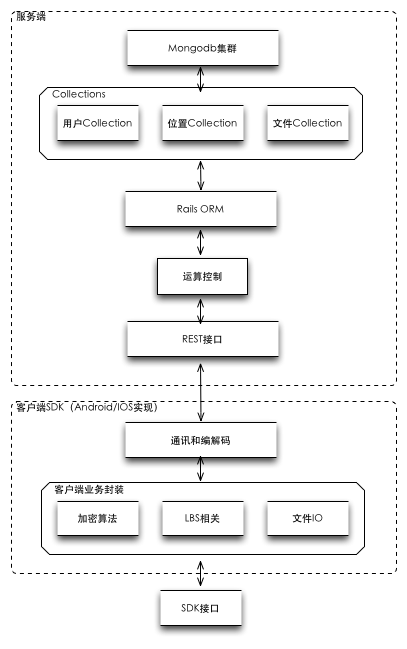
### 4.3.3 通用逻辑处理

这一层服务主要针对移动应用开发过程中的常见功能和常用处理逻辑，使得简单应用开发不必进行重复劳动。

值得注意的是，成熟的后端服务提供商往往提供更为复杂的处理服务（例如根据地理信息推荐周围商家），或者提供支持开发者自定义服务端逻辑的中间语言接口（如Parse提供的JS接口），而笔者不具备精力和能力把握这些，故这一层功能支持相较较弱，只提供了简单而基本的服务，扩展服务端逻辑也须在服务器代码的基础上进行Ruby on Rails开发。

本系统中，通用逻辑处理提供三种系统内置对象的相关服务：用户、位置和文件。结构上，所有的服务端存储工作依旧基于如4.3.1中所描述的分布式Mongodb数据库群，在此基础上使用Ruby on Rails + MongoMapper 实现了运算逻辑并将计算和资源以RESTFUL的形式暴露。客户端分别根据Android、IOS提供的LBS、IO、网络接口对客户端运算进行了封装，根据前文所述通用REST接口与服务器进行通讯。

通用服务的系统结构示意图如下：



## 4.4 关键技术讨论

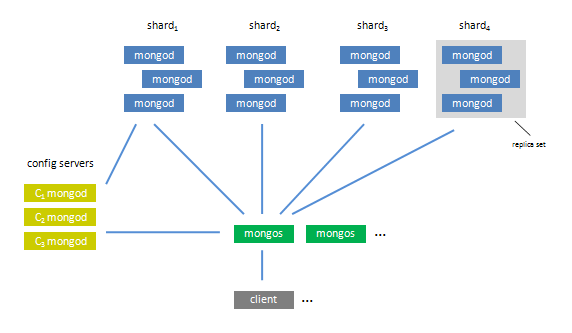
### 4.4.1 搭建分布式数据库集群

使用分布式数据库集群的结构实现本系统的存储主要有性能和成本两方面的考虑。

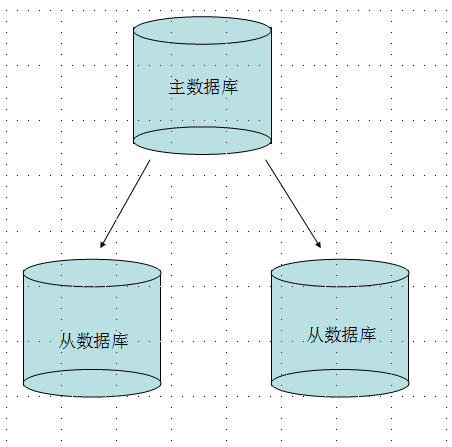
一方面，分布式系统更经济，可用性、可靠性好。分布式系统比集中式系统具有更高的可靠性和更好的可用性。如由于数据分布在多个场地并有许多复制数据，在个别场地或个别通信链路发生故障时，不致于导致整个系统的崩溃，而且系统的局部故障不会引起全局失控。另一方面，分布式系统可扩展性好，易于集成现有系统，也易于扩充。对于一个企业或组织，可以采用分布式数据库技术在以建立的若干数据库的基础上开发全局应用，对原有的局部数据库系统作某些改动，形成一个分布式系统。这比重建一个大型数据库系统要简单，既省时间，又省财力、物力。也可以通过增加场地数的办法，迅速扩充已有的分布式数据库系统。

Mongodb便捷的部署方式和天生具有的自动分片机制，决定了十分符合本系统对数据库层的需要。

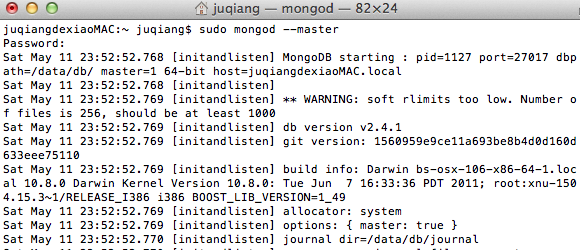
一个完整的Mongodb数据库集群包括一个mongos路由器，若干mongod分片存储服务器，以及若干配置服务器。结构如图：



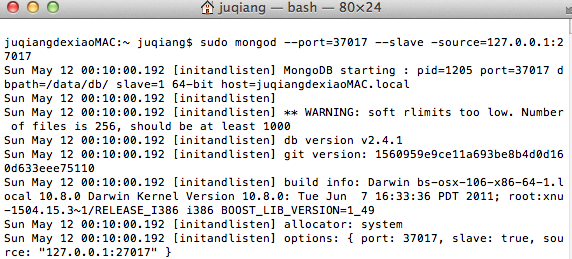
**Shard：**

图中Shard为一组mongod，存储一组分片数据。一组Mongod之间实现主从备份。在mongodb中，主从备份的配置是十分便捷的，其数据结构如图： 

首先使—master参数指定shard中的主数据库如图：



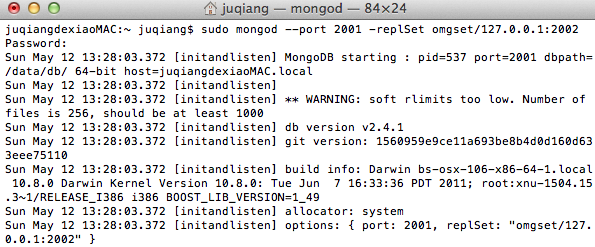
添加一个从属服务器，此处使用本地其他分区mongodb模拟多台机器，开辟新的端口并指定主服务器为mongod master，mongodb默认使用27017端口，此处从属服务器使用37017端口，如图：



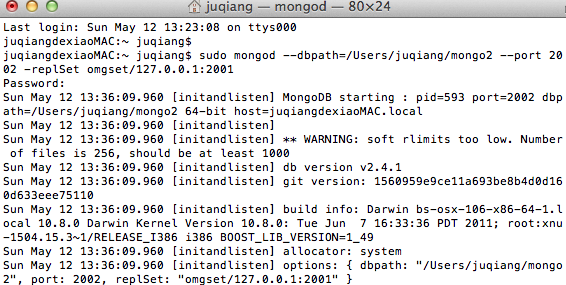
mongodb从属数据库默认10s从主数据库同步数据，同步数据通过同步主数据库的Oplog实现。

对于更高要求的数据安全以及系统故障自动恢复要求，也可以使用副本集的结构代替以上的主从备份方式，副本集的特点是没有如上结构所说特定的master数据库，在当前数据库故障的情况下，副本集会使用另外的备份节点代替故障数据库，从而使系统具备了一定的自动恢复能力。

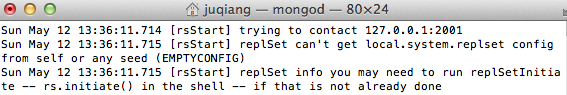
这里依然采用在一台机器上通过多个数据库进程模拟多台数据库服务器的方式。首先指定副本集中第一个副本的端口和副本集名称（sudo mongod --port 2001 -replSet omgset/127.0.0.1:2002），建立我们的副本集omgset，如图



随后，如上命令所说我们在2002端口初始化副本集的第二个副本，并指向omgset(sudo mongod --dbpath=/Users/juqiang/mongo2 --port 2002 -replSet omgset/127.0.0.1:2001)，如图，其中—dbpath所指的，为我们模拟另一个数据库服务器的mongodb的存储路径



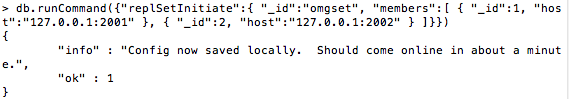
此刻，副本集的连接工作基本完成，若需要在副本集中继续添加副本服务器，重复上述步骤即可，观察此刻mongodb系统日志如图



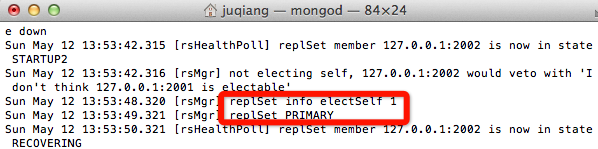
表明副本集并未开始工作，此时需要进入任意一个mongodb副本进行副本集的初始化。例如，进入2001端口的副本（mongo 127.0.0.1:2001/admin），在此处我们初始化副本集信息：

1. > db.runCommand({"replSetInitiate":{
2. ... "\_id":"omgset",
3. ... "members":[
4. ... {
5. ... "\_id":1,
6. ... "host":"127.0.0.1:2001"
7. ... },
8. ... {
9. ... "\_id":2,
10. ... "host":"127.0.0.1:2002"
11. ... }
12. ... ]}})

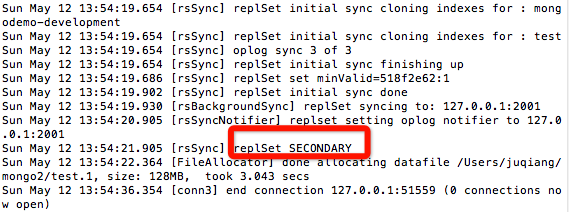
如图：



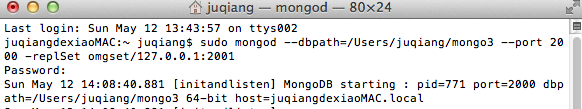
此时我们进入了omgset副本集的控制区，可以看到2001端口的副本已成为主数据库：



而2002端口的副本，已成为从数据库：



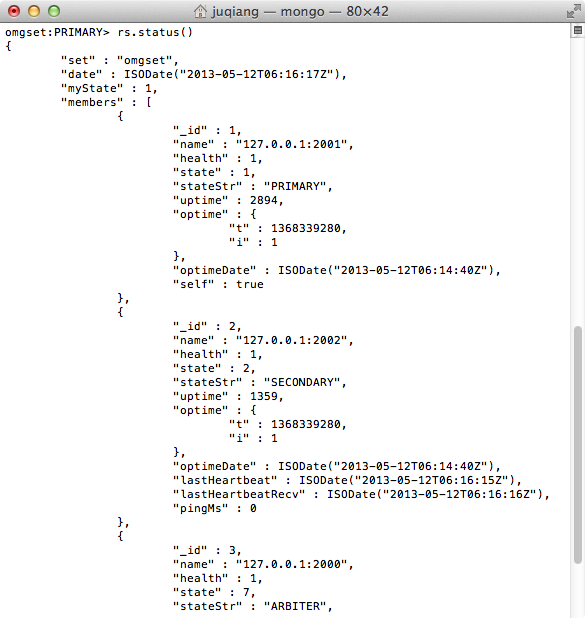
此外，一个完成的副本集还应包括仲裁服务器，仲裁服务器只参与投票不参与数据存储查询。使用在2000端口的mongodb作为仲裁服务器,可任意指定它连接到omgset副本集中的任意服务器，如在此连接到2001端口的服务器，如图



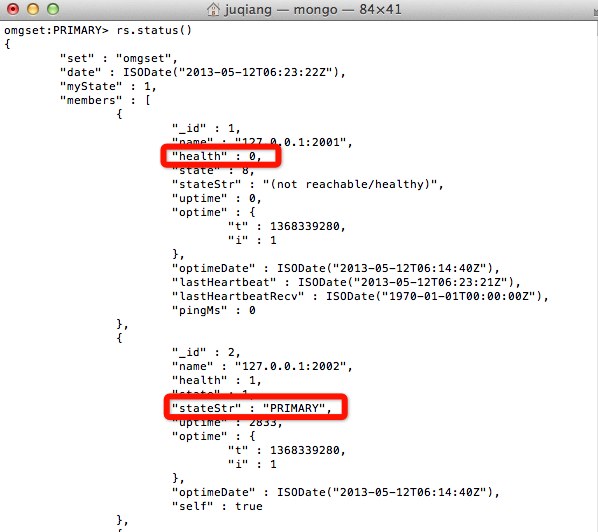
然后在2001服务器上，可以添加2000端口的mongo3为仲裁服务器。



此时，使用rs.status()命令查看omgset副本集状态，我们可以看到，PRIMARY（主）、SECONDARY（从）、ARBITER（仲裁）三个服务器已经连接就绪。



这个副本集已经具备了初步的备份和自我恢复能力。例如，我们把2001数据库进程关闭，



可以清楚得看到，当2001主数据库不可用（health==0）的时候，副本集自动推举2002副本为主数据库（PRIMARY）。

**Mongos：**

Mongos是数据库集群的路由器和控制中心，mongos处理客户端发来的请求并路由到Shards中，整理Shards返回的数据或结果传递给客户端。以Mongos为界，数据库的分布式结构对客户端是透明的。这使得整个分布式数据库系统十分容易在不影响线上产品的前提下进行维护和扩展。

**Config Server：**

Config Server是分片配置服务器，它是一个Mongodb数据库或如上所述的一个Mongo数据库副本集，Config Server保存Shards集群的分片信息，若为副本集，则每个副本中保存相同的数据，确保分片信息完全一致。

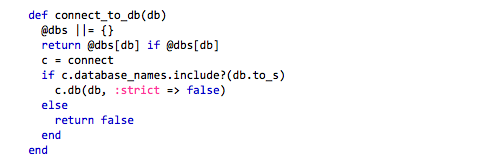
### 4.4.2 构建RESTFUL的Mongodb接口

REST 描述了一个架构样式的互联系统。REST 约束条件作为一个整体应用时，将生成一个简单、可扩展、有效、安全、可靠的架构。系统中使用REST的方式构建数据访问的服务层，正是基于它简便、轻量级以及通过 HTTP 直接传输数据特性的考虑。

传统上，位于移动应用与数据库服务器之间的中间层，往往包含两个两个功能，一个是向下实现对数据库服务器的抽象，将数据库中的数据转换为结构化的数据模型，一个是向上对客户端层提供数据传输和数据操作的接口。在这样的结构中，数据模型的改变一般至少需要进行：修改数据库表结构、修改中间层数据模型、修改应用程序数据模型等步骤才能完成。这样复杂的步骤无疑增加了开发成本，也对后端框架的通用性造成了很大的局限。

Mongodb作为非关系型文档数据库，无表结构的特点非常适合解决上述问题。而Ruby作为动态的服务端脚本语言，其编码的灵活性使其成为构建Mongodb RESTFUL 接口的良好选择。

我们修改Ruby访问Mongodb的工作模式为非严格模式，以此获得更灵活的数据访问功能。如图：

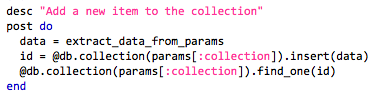


在集合不存在的情况下自动建立集合和文档格式，从而在通用性的基础上最大程度的降低了前端移动开发人员对后端的关注度。

基于HTTP的GET、POST、PUT、DELETE四种方法，分别实现对数据的访问和操作：

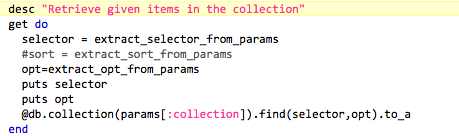
**POST方法**

Post方法是最基本的方法，通过post /db/collection的形式，将json格式的文档数据存储至服务器，若成功则返回201以及插入的文档。



**GET方法**

通过HTTP的GET方法实现对文档资源的展现，使用GET /db/colleciton的形式，讲查询条件和选项参数传递给服务器，返回相关查询结果



其中，最寻Mongodb的JSON查询条件，Selector可以构建类似关系型数据库的多种形式的查询，例如：

selector={“name”:”juqiang”} //查询name是juqiang的文档记录

selector={“name”:”juqiang”,”age”:23} //查询name是juqiang，且age为23的文档记录

selector={“name”:”juqiang”,”age”:{“$exists”:true}} //查询name是鞠强，且age存在age属性的文档记录

selector={“age”:{“$lt”:30}} //查询所有age属性小于30的文档记录

selector={“age”:{“$lt”:30},“age”:{“$gte”:10}} //查询所有age大于等于10小于等于30的文档记录

同时，为更好得满足查询需求，除selector之外，一条GET请求还可单独附加选项参数，以实现对查询行为的调控，目前支持的参数有：

fields(Array)：筛选文档的指定字段集合的查询结果返回

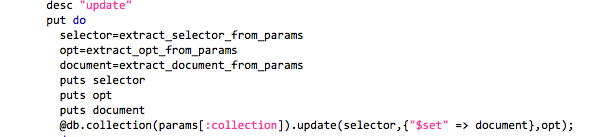
skip(Integer)：跳过某几条记录查询其后的记录

limit(Integer)：指定返回文档记录结果的最大数目

sort(key-values): 指定返回结果的排列顺序，且排列规则的优先级按key-values的先后顺序决定，如[["age","ASC"],["grade","DESC"]]会返回先按年龄升序排列，相同年龄按年级降序排列的文档队列

**PUT方法**

HTTP PUT方法映射文档资源的UPDATE方法，使用PUT db/collection的形式，使用更新条件、目标内容和操作参数控制文档的更新行为



其中，selector，如同上述GET方法中的查询参数，支持=、lt、gt、lte、gte、exists等参数及组合式的查询条件。

目标文档此处定义为需要更新的字段列表。或许在某些情况下，类似delete + insert操作的update操作是更合适的，但在移动客户端开发的过程中，我认为更新字段或字段列表才是更加经常的业务需求。

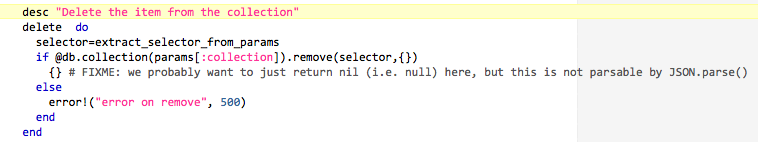
除了selector和document，update操作还支持两个可选参数

upsert(Boolen):即update + insert，默认情况下若update操作未发现符合selector条件的文档时，update操作以失败结束，若开启upsert选项，则在查询无果的情况下，执行insert操作，讲目标文档作为一条新的文档数据插入collection中

multi(Boolen):默认情况下，update操作更新collection中第一个符合selector的文档为目标文档，若开启multi选项，则update操作会对所有符合selector的文档进行更新

**DELETE方法**

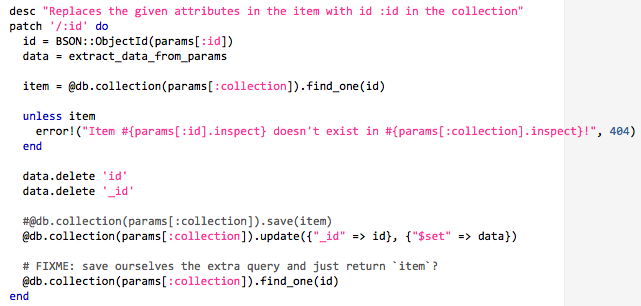
HTTP DELETE方法映射文档资源的REMOVE方法，使用DELETE db/collection的形式，进行文档删除操作



DELETE方法与UPDATE方法极为类似，删除collection中符合selector描述形式的文档，同样的，selector支持=、lt、lte、gt、gte、exists等操作。

通过上述讨论和介绍可以看出，本系统中RESTFUL API操作的最小粒度是Collection而非文档，这看似违反了REST规则中为所有资源确定唯一路径的要求。

这个问题可以从两个方面来看。一方面，本系统的REST API的底层实际上也包含了对文档资源的REST封装，如图是对指定id的文档的修改操作：



这些接口暂时没有暴露，若开发扩展需要，可以很方便的修改至可用状态。另一方面，这么做的主要因素，不向移动端提供对文档的REST接口，是为了弱化关系型数据库中主键的概念，继而弱化表结构的概念，减少配置和改动带来的成本。

具体说来，Mongodb文档的物理主键是文档插入时形成的id字符串，形如“4deeb1d9349c85523b000001”等，具有唯一性，这是REST uri的直接选择。但此id对于前端业务来说是没有意义的，加之REST是无状态的协议，使用此id进行对资源的操作必然增大了开发过程的复杂性。解决之道看似很简单，即使用逻辑主键，类似关系数据库中我们定义的主键。但是一方面文档型数据库mongodb不支持逻辑主键的定义，一方面定义逻辑主键再次把我们拉回了数据库定义-服务器模型-客户端模型的开发过程，违背了系统便捷轻巧的初衷。综合考虑我决定将这个功能对客户端层面舍去，并采用增强GET的方式作为补偿。

### 4.4.3 使用HMAC方式提供高效率的认证

通过构建RESTFUL的mongodb接口，实现了对文档数据的CRUD操作，REST是无状态的协议，若通过Cookie的方式进行身份验证，则破坏了无状态性，这一方面加大了服务器的压力，一方面增加了用户使用成本。

参考微软、谷歌、亚马逊等公司的做法，这个问题可以使用HMAC的解决办法解决。HMAC全称Hash-based Message Authentication Code. HMAC保证消息的合法来源，而不对消息加密，使用轻量级的方法保证消息安全性。

加入HMAC的REST API工作过程大致如下：

1. 客户端与服务端事先约定用户名和密钥并各自存储
2. 客户端向服务端发送消息时，通过HASH算法以用户名和密钥+请求资源的唯一路径为输入，输出一个散列值
3. 客户端将用户名和散列值附加于HTTP请求的HTTP头中，通过4.4.3中所述REST API与服务器交流
4. 服务器通过用户名查询约定密钥，并通过与客户端相同的HASH算法对用户名、密钥、请求资源URI进行散列，输入出散列值。
5. 若服务端散列值匹配HTTP头中获取的散列值，则认证成功，执行相关请求，反之返回失败信号。

### 4.4.4 服务器的选择和使用反向代理的方式部署应用服务器

在Ruby的服务器部署方面，系统选择使用Nginx和Passenger的组合。实际上，对于Ruby的应用服务器，我们可以有多种选择，可能是 FastCGI, 或者是 Mongrel，Thin, Passenger 等。然后我们通过 Apache，Nginx，Lighttpd，HAProxy 之类的 Web 服务器访问我们的应用服务器，这个访问可以直接通过 HTTP 协议，也可以是 FastCGI，或者是自定义的其它协议，如此这般，我们便可以通过我们制定的访问协议访问我们的服务器程序了。

对于前端的 Web Server，Apache 是事实上的工业标准，在 Web 服务器市场，是占有率最高的，全球大量的网站采用 Apache 来部署他们的应用，Apache 是一款成熟稳定的 Web 服务器，功能非常强大，提供对几乎所有 Web 开发语言和框架的扩展支持，在对 Rails 框架的支持上，我们可以采用 mod\_fcgid 模块，通过 FastCGI 协议与 Rails 进程通讯。或者利用 mod\_proxy\_balancer 对后端的独立的 Rails 服务器如 Thin Cluster，Mongrel Cluster 或者 Apach/Nginx+Passenger 进行 HTTP 分发。但 Apache 作为一个通用的服务器，在性能上和一些轻量型的 Web 服务器相差甚远。Apache 的 mod\_proxy\_balancer 模块的分发性能不高，比 Nginx 或者 HAproxy 都相差很多，另外，Apache 目前并不支持 Event（事件）模型，它仅支持 Prefork（进程）模式和 Worker（线程）模式，每处理一个链接，就需要创建一个进程或线程，而一些轻量级 Web 服务器如 Nginx 和 Lighttpd，则都很好地利用内核的事件机制提高性能，极大减少线程或进程数量，降低系统负载。

Nginx 是一个轻量级的高效快速的 Web 服务器，它作为 HTTP 服务器和反向代理服务器时都具有很高的性能。Nginx 可以在大多数 Unix like OS 上编译运行，并有 Windows 移植版。 Nginx 选择了 Epoll 和 Kqueue 作为开发模型，它能够支持高达 50,000 个并发连接数的响应，可以在内部直接支持 Rails 和 PHP 程序对外进行服务。另外，Nginx 作为负载均衡服务器，也可以支持作为 HTTP 代理服务器对外进行服务 , Nginx 不论是系统资源开销还是 CPU 使用效率都比 Apache 要好很多。Nginx 本身就是一个安装非常简单，配置文件非常简洁，甚至可以在配置文件中使用 Perl 语法。

在处理静态文件上， Apache 和 Nginx 都可以胜任。但对于应用服务器或者是前端的负载均衡服务器， Nginx 是我们更好的选择。

对于应用服务器，Mongrel 一度是最流行的部署方式，它的 HTTP 协议的解析部分是用 C 语言编写的，效率上有所保证。Mongrel 使用了 Ruby 的用户线程机制来实现多线程并发 , 但是 Ruby 并不是本地线程 ,Rails 也不是线程安全的，因此 Mongrel 在执行 Rails 代码的过程中，完全是加锁的状态，那和单进程其实也没有太大差别。所有我们在使用 Mongrel 来部署 Rails 应用程序时，一般是在后台启动一个 mongrel\_cluster 来启动多个 Mongrel 进程。

Passenger 是类似于 mod\_php 的 Rails 运行环境，而不是 Mongrel 那样是独立的 Http 服务器。Passenger 对目前主流的 apache 和 Nginx 两大 Web 服务器都有支持。Passenger 使用起来极其方便，而且它具有较高的性能，从 Passenger 官方网站公布的测试结果来看，Passenger 的性能要优于 Mongrel 服务器，目前来说，Passenger 无疑是最好的选择。Passenger 继承了 Rails"不重复自己"的惯例，通过 Passenger 部署应用程序，仅仅需要将 Rails 项目程序文件上传到目标服务器，甚至都不需要重启服务器，非常简单。

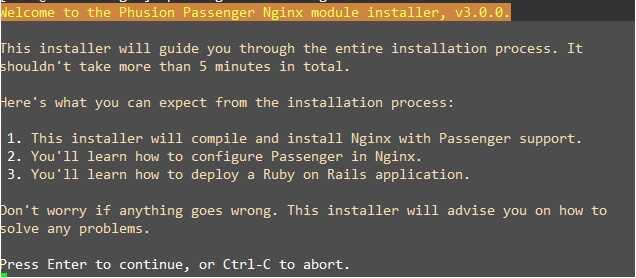
基于上述讨论，从性能和灵活性两方面综合考量，系统使用Nginx作为HTTP服务器，使用Passenger作为应用服务器。

由于Nginx是不支持加载插件的，使用Passenger需要重新编译Nginx，使编译产出带有Passenger的Nginx服务器。使用如下命令进行上述过程：

gem install passenger

Passenger-install-nginx-module

如图



在Nginx配置文件中插入如下代码，连接Nginx和后端Passenger服务

http { ...

server { listen 80;

server\_name www.omg.com;

root /var/www/omg/public;

passenger\_enabled on; }

... }

此处使用反向代理的好处是显而易见的，反向代理充分利用了Nginx处理静态数据的性能优势，可以将负载均衡和代理服务器的高速缓存技术结合在一起，将静态请求交付Nginx处理，将动态请求代理给Passenger处理，提高了系统性能。同时，反向代理为系统后端的扩展和负载均衡的优化提供了可能，具备额外的安全性，外部客户不能直接访问真实的服务器。将负载可以非常均衡的分给内部服务器，不会出现负载集中到某个服务器的偶然现象。

### 4.4.5 基于Countly-server构建统计服务器

### 4.4.6 Mobile First的移动端数据接口设计

|  |  |
| --- | --- |
| 1232  a | bernser1  b |
| otsu1  c | otsubernser1  d |

图表 4 a为原灰度图像，b为只使用Bernsen算法处理得到的二值化图像，c为只使用Otsu算法处理得到的二值化图像，d为使用Otsu算法和Bernsen算法相结合的方法得到的二值化图像。原图像的大小为：396\*407。

通过这组图像的对比，我们可以看出，结合后的方法较只使用Bernsen算法有了很大的改进，但是与只使用Otsu算法的区别不是特别明显，所以我们通过局部放大来观察结合后的算法较Otsu算法在消除断笔和伪影现象以及减少噪声方面的优势：

|  |  |
| --- | --- |
| otsuduanbi  a | obduanbi  b |
| otsuzaosheng  c | obzaosheng  d |
| otsuweiying  e | obweiying  f |

图表 5 a为经过Otsu算法得到的二值化图像的文字部分出现断笔现象，b为使用Otsu算法和Bernsen算法相结合的方法得到的二值化图像相应区域削除了断笔现象，c为经过Otsu算法得到的二值化图像中存在的噪声，d为使用Otsu算法和Bernsen算法相结合的方法得到的二值化图像相应区域削除了该噪声，e为经过Otsu算法得到的二值化图像存在伪影现象，f为使用Otsu算法和Bernsen算法相结合的方法得到的二值化图像相应区域削除了伪影现象。

通过几幅放大后的图像的对比，我们可以看出结合后的二值化方法较单一的Otsu算法有效地消除了断笔和伪影现象，同时减少了噪声。和Bernsen算法相比，综合算法在大大提高运算速度的同时，明显地减少了伪影，断笔现象也大大减少。综合算法在三种方法中，实验效果最佳。

# 第五章通用性和可用性讨论

这一章我们对比几种常用的文本图像二值化方法和本文中提到的两种二值化方法，比较它们的处理效果。

|  |  |
| --- | --- |
| wenben  a | niblak  b |
| niblack40  c | otsu  d |
| bernser  e | obgaidong  f |
| postprocessing  g |  |

图表 6 a为源图像，b为经过Niblack算法处理得到的二值化图像，其中窗口大小为w=15，c为经过Niblack算法处理得到的二值化图像，其中窗口大小为w=40，d为经过Otsu算法处理得到的二值化图像，e经过Bernsen算法处理得到的二值化图像，f经过Otsu算法和Bernsen算法相结合的方法处理得到的二值化图像，g经过基于背景表面的文本图像二值化方法处理得到的二值化图像。

通过图表6各种二值化方法对相同文档图像的处理效果的比较，我们可以看出：Niblack局部二值化方法受窗口大小的影响比较大，当窗口较小时会在背景区域产生连续大块的非目标黑色区域；当使用较大的窗口时非目标黑色区域减少了，但是对于光照明暗的变化变得不那么有效了，并且会出现不一致的笔划宽度，噪声仍会遗留在处理后的图像中。使用全局Otsu算法得到的结果中明显有断笔现象。而Bernsen算法的结果中则伪影现象比较严重，并且也存在断笔现象。Otsu算法和Bernsen算法相结合的二值化方法有效地消除了断笔和伪影现象，同时减少了噪声，与Bernsen算法相比，综合算法也大大提高了运算速度。基于背景表面的文本图象二值化方法尽量消除背景区域被二值化为笔划的伪影现象、较好地保留了有效的笔划像素、抗噪声能力比较强，实现速度也较快。

# 第六章结论

## 6.1 两种新的文本图像二值化方法总结

本文通过对以往常用的图像二值化方法的研究，以及对大量文献的学习，根据文档图像自身的特点，结合背景表面方法的处理流程，提出了一种新的基于背景表面的文本图像的二值化方法。该方法能够消除伪影现象、较好地保留了有效的笔划像素、抗噪声能力比较强，实现速度也较快。另外，本文又综合了全局二值化算法与局部二值化算法的思想提出了一种新的综合文档图像二值化算法。该方法有效地消除了断笔和伪影现象，同时减少了噪声，与Bernsen算法相比，综合算法也大大提高了运算速度。

## 6.2 文档图像二值化的研究现状及存在的问题

从二十世纪六十年代初期，人们提出了第一种图像二值化算法，迄今为止，人们已经提出了上千种算法。如此繁多的算法，是人们在不同的研究和工程应用中提出的。因此很难找到一种较好的标准，将所有的图像二值化算法进行合理的归类。本报告主要根据阈值所作用的范围将文档图像二值化算法大体分为两类：全局二值化和局部自适应二值化来介绍的。

但是到目前为止还不存在一个文档图像二值化的通用方法，也不存在一个判断分割是否成功的客观标准。阈值选的过低，会造成字符的笔画断裂；选的过高，图像中的背景不能被去除。这些将严重影响后期的文字识别系统性能。怎样才能兼顾算法的效果与执行的速度两方面，是一个值得深入研究的领域，这些都可以作为我们以后研究的方向。

## 6.3 文档图像二值化的发展趋势

我们必须建立行之有效的一些文档图像二值化质量评价的方法，结合原有的二值化方法，发展更多处理效果良好的新方法。近年来已有一些新的研究手段被引入到阈值选取中，比如人工智能[29]、神经网络[30]、数学形态学[31]、小波分析与变换[32]等等，伴随着文档图像越来越多的进入人们的生活，文档图像二值化会有更广阔的发展空间。

# 致 谢

经过了大概三个月的时间，通过对二值化技术的学习、参考相关书籍和网上信息、资料的查找，以及在老师和同学们的指导和帮助下，我得以顺利的完成了本科毕业设计。在此表示我由衷的谢意。

首先，我要感谢山东大学的指导老师阎中敏，感谢阎老师在本课题的开题、设计和撰写论文期间，给予我的宝贵意见和指导，以及为我提供的良好的实践环境。阎老师渊博的学识、严谨的治学态度、平易近人的风格给我留下了极其深刻的印象。她在我遇到困难的时候鼓励我、教导我要保持良好的心态去克服困难，以积极的心态和严谨的态度面对以后的研究，她的教导使我受益终生。

同时我要感谢中科院自动化所的肖柏华老师和我的导师戴汝为院士，他们在百忙之中抽出时间对我在毕业设计中遇到的难题进行细致的分析与指导，并且在指导的过程中悉心负责，在毕业设计的开始阶段，他们为我的课题选择进行了严谨的分析，为我的毕业设计指明了方向，引导我的设计思路，关注设计的进度，改进程序的功能，使我无论从理论研究还是从实践动手两个方面都受益匪浅。他们渊博的专业知识、严谨的教学态度、敬业的工作态度都给予了我极大的鼓励。

其次，我要感谢我的同学，在我开发过程中，遇到不少难题，是他们不厌其烦地为我解答，给我提供的帮助。

最后，我还要感谢所有指导、帮助我完成毕业设计的专业课老师、教务员老师、辅导员老师，他们的帮助，使我的毕业设计得以顺利完成。

# 参考文献

[1] N. George, “Twenty years of document image analysis in PAMI”, IEEE Trans. On PatternAnalysis and Machine Intelligence, Page(s): 38-62, 2000

[2] B. Gatos, I. Pratikakis, S.J. Perantonis，Adaptive degraded document image binarization，Pattern Recognition, Vol.39, pp317-327.

[3] 刘平，图像分割阈值选取技术综述，中科院成都计算所，2004.2.26

[4] 胡家忠，计算机文字识别技米[M]，北京气象出版社，1994.12一13.

[5] 章毓晋，图像分割评价技术分类和比较，中国图象图形学报，1(2):151一158. 1996.

[6] BarrosoJ,RafaelA,et a1.Number plate reading using computer vision.Proc.EEEInternationarSymposiumonIndustrialE lectionics,Portuga1,1997

[7] 何斌等，Visual C++ 数字图像处理(第二版)，人民邮电出版社2004，人民邮电出版社

[8] 夏良正，数字图像处理，东南大学出版社，南京，1999

[9] Doyle W. Operations Useful for Similarity-invariant Patern Recognition. JACM. 9(2):259-267. 1962.

[10] Lee S , Chung S. A Comparative Performance Study ofSeveral Global Thresholding Techniques for Segmentation[J ] . Computer Vision , Graphics , and Image Processing ,1990 , 52 : 171 - 190.

[11] Prewit J. M. S, Mendelsohn M. L. The Analysis of Cell Images. Ann N YAcad Sci. 128:1035-1053. 1966.

[12] 于新文，几种图像分割算法在棉铃虫图像处理中的应用，中国农业大学学报 ，2001，6

[13] 张旭丽，两种基于空间域聚类分析的彩色图像分割方法比较，贵州工业大学学报（自然科学版）, 第30卷，第2 期（总第109期），第75－79页，2001.4

[14] N. Otsu ,A threshold selection method from gray-level histograms ,IEEE Trans. Systems Man Cybernet. 9(1) (1979) 62-66.

[15] Otsu N. Discriminant and least square threshold selection. In: Proc 4IJCPR, 1978:592-596.

[16] Kapur J N ,Sahoo P K, Wong A K C. A new methodfor gray-level picture thresholding using the entropy ofthe histogram. Computer Vision , Graphics and ImageProcessing , 1985 , 29 (2) : 273～285

[17] Abutaleb A. S. Thresholding of Gray-level Pictures Using Two-dimensionEntropy. CVGIP. 47:22-32. 1989.

[18] PK.Sahoo, S.Soltani, A.K.C. Wong, and Y.C.Chen, "A Survey of ThresholdingTechniques", Computer Graphics,Visionand Image Processing 1988 (41),pp.233-260

[19] 阎之烨，基于计算机视觉的苹果颜色分级系统的研究，南京农业大学，2003-6-1

[20] Kittler J, Illingworth J. Minimum error thresholding. Pattern Recognition, 1986; 19(1): 41-47.

[21] J. Bernsen, Dynamic Thresholding of Grey-level Images, *Proceedings of theEighth International Conference on Pattern Recognition*, pp. 1251-1255, Paris, France,1986

[22] Qivind D, Torfinn T. Evaluation of binazimtion methods for document images[J].IEEE Trans PAMI,1995,17(3):312-315

[23] W. Niblack ,A Introduction to Digital Image Processing , Prentice Hall , Englewood Cliffs ,NJ ,1986 pp.115-116.

[24]L. Eikvil, T. Taxt and K. Moen, A Fast Adaptive Method for BinarizationofDocument Images, *Proceedings of the First International Conference on DocumentAnalysis and Recognition*, pp. 435-443, 1991

[25] J. Sauvola ,M. Pietikainen ,Adaptive document image binarization ,Pattern Recognition 33 (2000) 255-236.

[26] J.M. White and G.D. Rohrer, Image Thresholding for OpticalCharacter Recognition and Other Applications Requiring Character Image Extraction,*IBM J. Research and Development*, vol. 27, no. 4, pp. 400-411, 1989.7

[27]R.J. Schilling, Fundamentals of Robotics Analysis and Control, Prentice-Hall, Englewood Cliffs, NJ, 1990.

[28] 杨淑莹,VC++图像处理程序设计,清华大学出版社/北方交通大学出版社,2003.11

[29] 梁栋，李新华，一种基于人工智能的阈值自动选取方法，微电子学与计算机，1999.1

[30] 叶锋，基于神经网络的综合集成车牌识别技术的研究及其应用，2002.4 p22-26

[31] 彭进业等，分形维数在灰度图像二值化中的应用，小型微型计算机, 2001, 22(8): 961-963.

[32] 汪巧萍，一种基于小波分析的纹理背景下的物体分割算法，数据采集与处理，1998，3