安卓平台天津理工大学学生信息查询系

摘要

[中文摘要]

关键词：移动校园 校园信息系统 移动客户端 安卓

Student Information Query System for Tianjin University of Technology on Android

**ABSTRACT**

[English Abstract]

**Key Words**: Mobile Campus, Infromation System in Campus, Mobile Client, Android

目 录

[第1章 绪论 1](#_Toc388085887)

[1.1 课题的意义 1](#_Toc388085888)

[1.2 国内外发展状况 1](#_Toc388085889)

[1.3 课题的研究内容 3](#_Toc388085890)

[1.4 研究方法和研究手段 4](#_Toc388085891)

[1.5 研究步骤 5](#_Toc388085892)

[第2章 业务背景 6](#_Toc388085893)

[2.1 问题领域 6](#_Toc388085894)

[2.1.1 业务概况 6](#_Toc388085895)

[2.1.2 业务目标 6](#_Toc388085896)

[2.2 涉众分析 6](#_Toc388085897)

[2.2.1 涉众概要 6](#_Toc388085898)

[2.2.2 涉众简档 7](#_Toc388085899)

[2.3 规划业务范围 9](#_Toc388085900)

[2.3.1 规划业务目标 9](#_Toc388085901)

[2.3.2 规划涉众期望 9](#_Toc388085902)

[2.4 用户分析 9](#_Toc388085903)

[2.4.1 用户概要 9](#_Toc388085904)

[2.4.2 用户简档 9](#_Toc388085905)

[第3章 获取需求，建立业务模型 11](#_Toc388085906)

[3.1 确认业务边界、参与者和用例 11](#_Toc388085907)

[3.2 业务用例场景及实现 12](#_Toc388085908)

[3.2.1 查阅信息 12](#_Toc388085909)

[3.2.2 维护个人信息 13](#_Toc388085910)

[3.2.3 查课 14](#_Toc388085911)

[3.2.4 查成绩 15](#_Toc388085912)

[3.3 领域建模 16](#_Toc388085913)

[3.3.1 课程 16](#_Toc388085914)

[3.3.2 通知 17](#_Toc388085915)

[3.3.3 个人信息 18](#_Toc388085916)

[3.3.4 业务主线——信息查询 18](#_Toc388085917)

[第4章 系统分析 20](#_Toc388085918)

[4.1 系统用例 20](#_Toc388085919)

[4.1.1 确定系统用例 20](#_Toc388085920)

[4.1.2 用例实现 20](#_Toc388085921)

[4.2 核心功能——网站内容解析 22](#_Toc388085922)

[4.3 组件和部署模型 24](#_Toc388085923)

[第5章 系统设计 26](#_Toc388085924)

[5.1 网站内容解析器 26](#_Toc388085925)

[5.1.1 问题分析 26](#_Toc388085926)

[5.1.2 类模型 27](#_Toc388085927)

[5.1.3 交互模型 28](#_Toc388085928)

[5.1.4 实体类 29](#_Toc388085929)

[5.2 数据库 30](#_Toc388085930)

[5.3 代理服务器 31](#_Toc388085931)

[5.4 手机前端 32](#_Toc388085932)

[第6章 系统实现 33](#_Toc388085933)

# 绪论

## 课题的意义

安卓平台天津理工大学学生信息查询系统,英文全称（Query System for Tianjin University of Technology on Android）。

由于我校（天津理工大学）各部门的网站大多数都没有为手机做优化，学生平时用手机查看信息时，不仅不好看而且很缓慢不稳定，尤其对于常用的师生服务网站、门户网站、教务处网站和学院网站来说，更是不易使用。此外，与我们相关的通知又分布于不同的网站上，不仅难以查找，更容易遗漏通知。

为改善这一情况，帮助同学们更方便高效地利用好我校的信息系统，我们需要一款能够高效抓取并整合学校网站信息的手机应用系统。

从大一起我就与他人合作共同制作了这样的软件，现在由于学校信息系统换代，Android平台夸版本升级，以及使用的云平台基础设施更新等原因，我需要重新构造这一系统。同时新版系统还要改进系统机构，改善系统难以维护的问题。

本系统具有以下优点:

1. 美观。为手机定制，信息有序，方便查找查看。
2. 高效。一方面，本应用仅读取必要的信息，不读取多余的网页、图片等；另一方面，本系统包括一个代理服务模块，它缓存通知等通用信息，手机客户端通过它读取信息更准确快捷。这些机制能减少消耗的手机流量和响应时间，同时减轻学校服务器的负担。
3. 便捷。学生可以从桌面插件方便地查看课程信息，可以使用筛选、查找功能查询通知。
4. 流畅。除了能够高效地抓取信息外，应用还在本地保存抓取到的信息，且在后台自动更新，用户通过应用查询信息会比直接通过网站查询快得多。

## 国内外发展状况

如今基于Web的信息系统已在各高校普及，然而与时代不符的是这些系统绝大多数都没有为手机等移动设备优化，这与其主要用户经常使用的终端类型不符。因此市场上出现了许多课程表类手机应用，实现了部分教务系统的查询功能。也有一些学校率先制作了移动校园手机应用。

从多发面的数据可以看到，最近几年使用智能手机的网民飞速增长，同时使用计算机上网的用户却在下降，移动互联的时代已经到来。根据《第33次中国互联网络发展状况统计报告》[1]的数据显示，截至2013年12月，我国已有5亿手机网民，占所有网民的81%，较2012年底增长8.72%；而通过笔记本电脑和台式电脑上网的用户却都在下降。网民的25.5%都是学生，然而在学校用电脑上网的用户比例大幅下降了28.03%，原因在于学生通过手机接入互联网的比例明显增加。报告也总结到移动互联网行业全面发展，并且加速向日常生活渗透。另一篇针对大学生的调研报告[2]也显示，2012年9月前，大学生中拥有手机的就已达100%，其中拥有具备上网条件的智能手机的达61.2%，经常用手机上网的占比51.7%。此外孙耀庭等人论述了移动校园对大学立体教学的深远助益[3]，它不仅能帮助用户更便利地使用现有系统的功能，更能带来更多现实有用的双向、点对点交流的功能，可以直接为教学工作带来帮助。王韦进一步探讨了移动校园中的商务创新价值[4]，移动校园的市场前景也非常不错。

在搜索引擎和Google Play、小米商店等应用商店搜索后，会发现课程表类的应用已有许多，其中不少做得非常漂亮，如课程格子、超级课程表、掌上课表等。这些应用以课表为核心，向外延伸出一些附加功能，如笔记、校友聊天、空教室查询、考试时间查询、成绩查询等功能。目前这些应用都不支持从我校网站导入课程信息。电子商店上与本系统更为相似的电子校园类应用比较少，现有的有“中華大學 eCampus”。除了公告、课表、成绩外，它还有教材、作业、讨论、发邮件、签到这些功能。华南理工大学为其BBS系统开发了移动客户端[5]。惠州学院教育技术中心的陈泽恩探索了Android移动校园应用的设计与实现[6]，浙江工商大学的谢文焘、董黎刚做了进一步实现[7]。

近几年，美国高校的信息系统对移动设备的支持得到快速发展。根据美国高校信息化项目（Campus Computing Project）十月的调查报告[8]显示，IT管理人员认为平板电脑和智能手机很重要的分别占86%和82%，而认为笔记本电脑很重要的占62%。79%的高校在2013年秋季或下学期开始使用移动应用程序，比上一年的60%增长了31.67%。其中私立大学和公立大学的应用使用率更是高达95%和93%。移动校园应用在美国高校发展得非常不错，美国高校的信息系统建设得比较完善，并且能够健康及时地持续改善与发展。

由于国内高校的现有信息系统一般没有信息发布接口，移动设备客户端程序主要通过分析抓取学校网站的网页来获取信息。这种方式不必修改现有系统，能够较快速地开发新系统。不过新系统依赖旧系统，导致更难维护，更新网站布局可能导致客户端失灵。此外客户端大量扫描网站（如读取通知时）既浪费用户的流量和时间，也会给网站带来过大压力，为解决此问题需要多重缓存机制（代理服务器+本地缓存），这提高了新系统的复杂性。

对基于现有Web系统的移动客户端来说，Web页抓取与解析的实现较为重要。目前已有多个Java开源项目实现了这一功能[9]，其中jsoup较为适用于本项目。它为处理网络上各种HTML变体而设计，包括早期不规范的HTML文档[10]，而我校有些旧网站的不规范代码正是我们面临的难题之一。此外jsoup杰出的CSS选择器特性不仅便于使用[11]，还有较好的健壮性，能够提取现代动态web应用页的信息[12]。

为加快对通知等信息的读取速度，减少流量消耗，减轻学校网站的负担，需要部署代理服务程序。综合考虑价格、稳定性、易用性、性能等因素后，发现把它托管到Google App Engine比较合适。在一定配额内，GAE可以免费使用，我其他的一些程序也于GAE持续正常使用多年，GAE支持Java语言，可以部署Servlet程序，配合Hessian协议[13]，可以方便地为移动设备客户端提供远程过程调用（RPC）服务，便于使用的同时，性能也令人满意[14]。

本系统是典型的定制信息系统，用到的技术比较成熟。它具有较高的现实意义，能显著提高现有系统的使用效率，且潜力很大。本系统较明显地受限于现有系统，与现有系统的结合互助是关键问题。

## 课题的研究内容

设计并开发一个Android应用，实现课程信息查看和提示、成绩查看、通知查看、学籍信息查看等功能。额外说明见表1：

表 1.1 基本功能

Table 1.1 Basic Functions

|  |  |  |
| --- | --- | --- |
| 信息对象 | 功能 | 说明 |
| 课程信息 | 查看课程表 | 包含以下视图：本周课表、当前学期课表、下学期已选课程表。允许用户在本地修改、添加课程。 |
| 下节课提示 | 桌面小插件，显示课程名称、时间、地点。 |
| 查看成绩表 | 显示统计信息，如平均绩点等。 |
| 通知 | 筛选和查找 | 根据通知来源（教务处、学院、学生网站等）、通知发布时间、通知标题等筛选，排序，查找。代理服务器缓存通知列表。 |
| 查看通知 | 客户端缓存抓取下来的通知。 |
| 学籍 | 查看学籍 | 包括入学时间、学制、学院、专业、班级等信息。 |

根据功能特征，把系统分成信息抓取、数据存储和GUI等基本模块。具体说明如表2：

表 1.2 基本功能模块

Table 1.2 Basic Modules

|  |  |  |
| --- | --- | --- |
| 功能模块 | 描述 | 涉及的知识 |
| 信息抓取 | 从学校各网站抓取数据。 | 网络操作、网站登录、网页分析、代理技术 |
| 数据存储 | 保存抓取到的数据。 | SQLite数据库、Android Content Provider和Shared Preference等 |
| GUI | 根据用户命令，整理并显示数据。 | Android Activity及其布局XML和App Widget等 |

## 研究方法和研究手段

### 研究方法

本系统采用简化的RUP（Rational统一过程）方法设计开发。RUP与UML师出同源，是一种体系清晰完整的面向对象的软件工程方法。RUP是一种重量级过程，适合开发大型软件。由于本系统规模较小，也没有大型开发团队，开发时间也较短，所以这里使用RUP的简化版本，只进行主干工作，舍去了许多细节和文档工作，一些分析过程也从简。

图1.1展示了RUP的一般过程（科目）和各阶段的迭代。RUP一般有业务建模、识别需求、分析和设计、实现、测试、部署六个工程科目，还有此图没有显示的配置与变更管理、项目管理、环境这三个支持科目。（这张图展示了9个科目的迭代过程：http://commons.wikimedia.org/wiki/File:Rup\_espanol.gif）如图中的时间轴所示，系统开发要经历先启、精化、构建、产品化这些阶段，每个阶段都根据复杂度进行多次迭代，每次迭代都是跨科目的，只不过不同阶段下不同科目的比重不同。与增量式模型类似，它可以提早交付可视化的工件，提早发现问题，渐渐完善系统。

本系统主要进行了前四个工程科目，其它科目略有涉及。



图 1.1 RUP科目和阶段迭代[15]

### 研究手段

我主要使用astah进行业务建模、识别需求、系统分析和系统分析。它是一个小巧但很完整的UML建模工具。UML以外的模型用Visio绘制。

在系统设计和实现等过程中，本系统使用Java作为开发语言，程序有些部分使用XML文件做配置。本系统的客户端在Android平台上，使用Android SDK。代理服务器运行在Google App Engine上，使用Google App Engine SDK for Java。GAE for java使用Servlet框架。

本系统使用JUnit做单元测试。

本系统使用Git做版本控制，使用GitHub做远程仓库。本系统主要靠它进行配置和变更管理。

本系统使用Gradle自动化建构工具。它负责项目的配置和构建，由它自动编译、测试和部署各子项目。本系统可以使用各种兼容Gradle的IDE进行开发维护，例如Eclipse、IntelliJ IDEA、NetBeans等。我在开发过程中使用Eclipse、IntelliJ IDEA和Android Studio。

开发和维护时的系统需求总结在表1.3。目标系统需求总结在表1.4。

表 1.3 开发系统环境需求

Table 1.3 Develop System Requirements

|  |  |
| --- | --- |
| 硬件 | 计算机 |
| Internet连接 |
| 测试用手机 |
| 软件 | JDK 1.7 |
| Git |
| 与Gradle兼容的IDE |
| Android Studio（建议） |
| GAE SDK for Java |
| Android SDK |
| 在本地Maven仓库安装有com.exadel.flamingo.android:flamingo-android-hessian-client:2.2.0（否则无法编译代理服务器客户端） |

表 1.4 目标系统环境需求

Table 1.4 Target System Requirements

|  |  |  |
| --- | --- | --- |
|  | Android客户端 | 代理服务器 |
| 硬件 | HVGA(320x480)及以上规格的触屏 | GAE前端服务器F1 |
| Internet连接 | GAE后端服务器B1 |
| 软件 | Android 2.3(Gingerbread)及以上OS | 无特殊要求 |

# 业务背景

首先需要做一些准备工作。我们需要调研系统所处的环境，包括系统所处领域的知识、涉及到的用户和工作人员、识别确认本系统在整体业务中的范围和位置，最后需把涉众中的用户拿出来做更具体的分析。

## 问题领域

本节总结系统要实现的业务所处领域的相关信息。

### 业务概况

本系统的目标任务是校园给师生的服务业务（business）的一部分，天津理工大学是这一业务的业主，服务系统要帮助学校正常并尽量高效的运转，帮助教工人员高效办公，帮助学生学习和实现其培养计划。

大学有许多管理部门和学院，各部门都有自己的子业务，相应的大部分部门都各自有其信息系统。如教务处、团委、图书馆、理学院、计算机学院等等，各自都有其特殊的业务，也都有自己的网站等信息系统。具体来说，选课、排课、组织校级考试等业务由教务处主导，其网站上也有相应功能和信息；而计算机学院负责了自己的外教课、实验、院级考试的安排，相关信息在其自己的网站上发布。此外有些部门还有学生网站，与学生生活相关的许多信息发布在这里，如理工人、计算机学院学生网站等。

### 业务目标

如果整理总结业务概况，大体可以得到以下业务目标：

* 帮助师生管理个人信息。如查看修改学籍和联系信息、查看课程安排和成绩等。
* 帮助师生完成与其相关的工作。例如选课、报考、查找图书等。
* 帮助师生便捷快速地获取与其相关的通知等消息，帮助部门发布消息。
* 统筹信息，提高工作协调性、准确性和效率。

## 涉众分析

本节总结业务中涉及到的相关人员，并分别对他们调研分析。

### 涉众概要

涉众（Stakeholder），又叫利益相关者。业务的成败与他们息息相关，他们是业务目标的来源，业务的执行也离不开涉众。调查分析业务情况后，总结出的涉众概要信息如表2.1：

表 2.1涉众概要

|  |  |  |  |
| --- | --- | --- | --- |
| 编号 | 涉众名称 | 涉众说明 | 期望 |
| SH1 | 天津理工大学 | 业主。 | 1. 节省成本。 2. 利用现有系统，避免修改现有系统。 |
| SH2 | 信息发布者  工作发起者 | 向师生通告事项。  发起工作活动。 | 1. 尽量通知到更多的人。 2. 信息传播快速。 3. 可以选择针对的部门与人员。 |
| SH3 | 信息接收者  工作参与者 | 查询与自己相关的信息。  完成相关工作。 | 1. 便于查找。 2. 有足够提示和帮助 |
| SH4 | 学生 | 查询个人信息，如学籍、课程表等。录入个人信息。选课。 | 1. 界面美观时尚。 2. 便于查看课表。 |
| SH5 | 教师 | 查询个人信息，如教工信息、上课时间表、班级信息等。登记学生的成绩。 | 1. 高效快捷 2. 简单易用。 |

业主做出系统的出资者和所有者，希望系统性价比高，并且尽量稳定长久。

信息发布者/工作发起者希望能够快速高效地通知相关人员完成相关工作。

信息接收者/工作参与者希望不错过与自己相关的信息，并能够清除明白地完成相关工作。

学生希望界面时尚美观（新），便于查看常用信息。

教师希望系统简单易用、使用顺手并且高效。

### 涉众简档

分别对各涉众做更具体的分析后，总结得到以下简档。

表2.2总结了各涉众的身份、职责、其工作的成功标准、如何参与业务实现，及其需交付的工件。

表 2.2涉众简档

|  |  |
| --- | --- |
| 涉众 | SH1 天津理工大学 |
| 特点 | 系统的所有者。 |
| 职责 | 提供基础设施。 |
| 成功标准 | 为师生提供房屋、设备等支持。 |
| 参与 | 不参与系统建设。 |
| 可交付工件 | 无 |

|  |  |
| --- | --- |
| 涉众 | SH2 信息发布者/工作发起者 |
| 特点 | 系统的主要使用者之一。 |
| 职责 | 1. 发布通知。 2. 发起工作活动。 |
| 成功标准 | 1. 按指定格式和内容发布通知。 2. 按规范发起并管理活动。 |
| 参与 | 业务需求的主要提供者。 |
| 可交付工件 | 通知文档、活动说明 |

|  |  |
| --- | --- |
| 涉众 | SH3信息接收者/工作参与者 |
| 特点 | 系统的主要使用者之一。 |
| 职责 | 1. 查阅通知。 2. 执行活动。 |
| 成功标准 | 按通知行事。 |
| 参与 | 业务需求的主要提供者。 |
| 可交付工件 | 执行报告 |

|  |  |
| --- | --- |
| 涉众 | SH4 学生 |
| 特点 | 系统的主要使用者之一。一般具有一定智能设备操作水平。 |
| 职责 | 1. 录入并更新个人信息。 2. 选课。 3. 确认选课结果和课程安排。 4. 查阅个人信息。 |
| 成功标准 | 1. 按规定录入个人信息。 2. 及时更新个人信息。 3. 按规定选课。 4. 按时上课。 |
| 参与 | 业务需求的主要提供者。 |
| 可交付工件 | 个人信息表、选课表 |

|  |  |
| --- | --- |
| 涉众 | SH5 教师 |
| 特点 | 系统的主要使用者之一。老教师可能不擅长计算机操作。 |
| 职责 | 1. 录入并更新教工信息。 2. 登录教学任务。 3. 上课。 4. 确认学生成绩。 |
| 成功标准 | 1. 按规定录入教工信息。 2. 及时更新教工信息。 3. 确认教学任务。 4. 完成教学任务。 5. 按规定录入学生成绩。 |
| 参与 | 业务需求的主要提供者。 |
| 可交付工件 | 教学任务书、学生成绩单、学生成绩修正申请 等 |

## 规划业务范围

### 规划业务目标

本移动客户端系统处在业务前端的位置，它是业务实现和用户间的接口。也因此本系统不涉及“统筹信息，提高工作协调性、准确性和效率”这一业务目标的实现，这是后端的工作。

### 规划涉众期望

在开发本系统的过程中，由于没有教师（SH5）的权限，无法做相应分析和测试，且本项目的主要目标用户是学生，而教师一般作为业务工人（business worker）参与活动，所以本系统不准备实现与教师相关的业务。基于同样的原因，本系统也不涉及SH2信息发布者/工作发起者。

由于本项目没有选课系统测试权限或接口，无法实现选课功能，对项目取消涉众SH4学生的职责②选课。

此外，由于无法修改现有系统，而且咨询教务处后，其回应“为保证系统的安全性，学校暂不对学生提供应用程序开发的接口”，所以本系统目前只能通过现有网站获取通知、课程等信息。

## 用户分析

经过以上分析调整，本系统的主要目标用户为学生，对其做单独分析总结。

### 用户概要

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 编号 | 用户名称 | 用户概况和特点 | 使用系统方式 | 代表涉众 |
| US1 | 学生 | 学生需要查询其课程信息、成绩和与其相关的通知。  大学学生绝大多数都有智能手机，拥有一定手机操作能力。 | 直接与系统交互。 | SH3  SH4 |

### 用户简档

|  |  |
| --- | --- |
| 用户 | US1 学生 |
| 说明 | 学生可以通过手机应用查询信息，也可以通过网站查询。 |
| 特点 | 应用的直接使用者。一般具有一定智能设备操作水平。 |
| 职责 | 1. 查询个人信息。 2. 查询选课结果和课程安排。 3. 关注与自己相关的通知等信息。 |
| 成功标准 | 1. 方便地查阅学籍信息。 2. 便利地查看上课时间和地点，按时上课。 3. 及时收取通知。 |
| 参与 | 界面设计。 |
| 可交付工件 | 无 |

# 识别需求

在调研收集项目所处业务的领域知识后，我们需要从中抽象出业务需求，还要整理业务条理结构，为其建立业务模型，作为后续需求分析的信息来源。

上一章得到的领域知识是我们的出发点，涉众和用户分析报告为我们指明了方向。本章据此分析业务边界和业务参与者，并建立业务用例，此外本章还为关键概念建立了领域模型。

由于我们要实现的目标系统只是整个业务中很小的一部分，在本章的分析中我们重点围绕本系统要实现的业务目标及本系统要服务的用户展开。

## 确认业务边界、参与者和用例

以业务目标和涉众分析为依据，识别确认业务边界为天津理工大学。

在此业务中，业务主角（Actor，参与者）有信息发布者、信息接收者、教师和学生，其中信息发布者、教师属于业务工人，学生、信息接收者是主要服务对象。



图 3.1业务用例

业务用例如图3.1所示。教师和学生都有可能作为信息接受者查询信息。信息发布者发布信息，发起工作。教师为学生打分，学生应当及时更新自己的个人信息（校区、电话、邮箱等），并且按时选课，上课并确认成绩

## 业务用例场景及实现

一般的业务用例只需要一种实现方式，不过根据需要有些业务用例也可以有多种实现途径。例如发布信息这一用例，就可以通过网站和短信两种途径实现，如用例图3.2。



图 3.2发布通知用例实现

根据要通知事项的重要性和紧急性，通知可以通过短信发布，不过无论是否使用短信发通知，网站上都会发布所有信息，如活动图3.3所示。



图 3.3发布通知活动图

通过网站发布和通过短信发布是并发进行的，其先后顺序没有特殊要求。

下边分别展示本系统涉及到的用例实现。

### 查阅信息

查阅信息也有多种实现，如用例图3.4。



图 3.4查阅信息用例实现

除了从相应网站获取信息外，我们也可以从同学那得到通知，有些活动安排也会展示在通告栏上。很显然，本客户端系统只负责通过网站查询信息这种方式。



图 3.5在网站查询信息

如活动图3.5所示，从网站获取信息时，我们可能需要同时在多个网站查阅，如我院的同学经常查看教务处网站、学院网站和计算机学院学生网站的通知。

### 维护个人信息

我们可以在师生服务网站查询修改个人信息，也可以通过学工管理信息系统维护，如用例图3.5所示。

以在师生服务网站维护个人信息为例，维护个人信息的业务流程如活动图3.7所示，登录师生服务网站后，选用个人信息功能页后，就可以看到自己的信息，学生也可以更新自己的信息，然后提交修改。

|  |  |
| --- | --- |
| 图 3.6维护个人信息用例实现 | 图 3.7维护个人信息活动图 |

### 查课

我们目前基本都在师生服务网站查询已选课程信息，包括课程性质、学分、任课教师、上课时间和地点等信息，如用例图3.6所示。

查询课程业务过程如图3.7所示，同学登录师生服务网站后，选用查课功能，选择查询学期后，网站便会显示选课结果。

|  |  |
| --- | --- |
| 图 3.8查课用例实现 | 图 3.9查课活动图 |

### 查成绩

目前我们可以在师生服务网站、教务处网站查询成绩，此外还可以在教务处打印成绩单，如用例图3.8所示。



图 3.10查成绩用例实现

查询成绩的业务过程如活动图3.9所示，学生登录师生服务网站后，选用查询成绩功能，选定学年学期后便可以得到相应的成绩信息。



图 3.11查成绩活动图

## 领域建模

从上边的业务模型可以看出，业务围绕着课程、通知、个人信息这些业务实体展开。



图 3.12业务实体

下边分别分析对这三种业务实体。

### 课程

虽然查课和查成绩被分为了两个用例，且它们有不同的业务流程，但其实它们都是在查询课程信息，只不过视图不同，两种查询的结果也有重回的部分。把两种查询得到的信息进行合并整理，就可以得到实际的课程结构，如类图3.13所示。



图 3.13课程业务类图

一个课程除了有编码、名称、学年学期等基本信息外，还包含任意多个上课时间地点信息，院排课程在师生服务网站中没有时间地点信息。

### 通知

用户可以通过网站查询信息，然而不同网站的信息格式略有不同。分别总结通知列表和通知详情在不同网站上的格式如表3.1、表3.2所示。

这两个表格对比总结了不同网站上通知的内容。

表 3.1通知列表在不同网站的格式

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **门户网站** | **教务处网站** | **计算机学院** | **计算机学院学生网站** |
| **标题** | 有 | 有 | 有 | 有 |
| **发布人** | 有 | 无 | 有 | 无 |
| **时间\*** | YY-MM-DD | YYYY-MM-DD | YYYY-M-D | YYYY-MM-DD |
| **通知类型** | 无 | 无 | 有 | 无 |
| **通知对象** | 无 | 无 | 有 | 无 |

\* Y表示年，M表示月，D表示日期。YY表示年后两位，MM和DD表示各位数有前导0。

表 3.2通知详情在不同网站的格式

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **门户网站** | **教务处网站** | **计算机学院** | **计算机学院学生网站** |
| **标题** | 有 | 有 | 有 | 有 |
| **内容** | 有 | 有 | 有 | 有 |
| **发布时间** | 有 | 有 | 有 | 有 |
| **发布人** | 有 | 无 | 有 | 无 |
| **发布部门** | 有 | 无 | 无 | 无 |
| **附件** | 有 | 有（内容内） | 有 | 有（内容内） |

### 个人信息

个人信息有基本信息、学籍信息、入学信息、毕业信息、联系方式这几种类别，如类图3.14。



图 3.14个人信息业务实体类图

每一类别下都有一些键值对，例如基本信息中就有[学号:20106173]、[姓名:柏杰]、[性别:男]等键值对。

### 业务主线——信息查询

经过分析总结业务模型中的这些需求，我们发现本客户端系统本质上是一个信息收集与展示系统，它代理了用户的那些繁琐的查询操作。

为此本节进一步分析信息查询这一业务主线，综合考虑业务行为，组织业务结构，并为之建立模型。

综合已获取到的业务活动后，我们可以得到图3.15这样的活动图。



图 3.15综合业务活动图

信息查阅者可以在师生服务网站查看已选课程信息、结课成绩和个人信息，也可以在不同网站浏览通知事项。其实查询课程和个人信息时，师生服务网站也并不是是唯一的选择，如学工管理信息系统也有个人信息，教务处网站也有些课程信息。

这类业务流程可以进一步整理抽象为活动图3.16。信息查阅者进入网站后，网站展示出它可提供的服务，查阅者选中希望查找的信息后，网站给出应答，最后查阅者退出网站。其中查阅者选择功能，网站给出结果的交互过程可以重复多次。



图 3.16业务主线活动图

这个业务流程中比较复杂的地方有两点：

* 信息分布在不同的网站上，不同网站的访问方式可能不一样，有些需要登录，有些页面必须先建立会话才能访问。
* 不同信息类型、不同网站的页面格式不同，交互流程也可能有些差异。
* 有些信息不只一页，如通知列表、课程成绩（学期不同页面不同）等。

# 系统分析

经过之前的调研分析，我们用业务模型描述了系统所参与的业务流程和系统用到的业务实体，我们还进一步抽象了业务流程的本质，并注意到了一些难点问题。

接下来我们开始考虑如何用计算机系统实现业务。

[概述]

## 系统用例

### 确定系统用例

我们可以从业务用例提炼精化系统用例。我们的客户端程序不涉及发布信息、教师打分业务，在此系统的上下文中信息接收者就是学生，所以简化合并了这两种参与者。再进行些整理可以得到系统用例图4.1。



图 4.1系统用例图

学生可以查看课程信息，为方便使用，可以以课程表的形式显示，并自动显示当日课程。学生也应当可以查看成绩信息，包括通过率、平均绩点等统计数据。学生还可以查看自己的个人信息以及与自己相关的通知。

### 用例实现

课程查询涉及课程查询边界类、查询流程控制类、课程实体类和数据库管理器。它们的依赖关系如类图4.2所示。



图 4.2课程查询类图

查课界面类依赖于查课控制类，控制类又依赖于数据库管理器类，它们发消息时会用到课程实体类。它们的交互过程如顺序图4.3所示。



图 4.3课程查询顺序图

界面收到学生发出的查课请求后，向查询控制对象发查询消息，查课控制对象通过数据库管理器得到数据库中的课程信息，然后返回界面对象请求的课程信息，最后界面给用户展示课表。

如果用户发出了更新课程信息命令，界面让控制对象执行更新操作，控制对象需要登录师生服务网站，它可能会通过界面向用户询问账号和密码，收到用户提供的账号密码后，依次执行登录操作、已选课程网页读取操作和解析操作，然后通过数据库管理器保存解析结果，最后界面更新课程信息。

也有从数据库查不到课程信息的情况，这时就需要先更新课程信息再显示课表，如活动图4.4所示。



图 4.4课程查询活动图

如果本地数据库中无缓存的课程信息，要先更新课程信息再显示课表。用户也可以主动发出更新课程信息命令。

查成绩、通知和个人信息的用例实现与查课表类似，且成绩查询的控制类与课表查询的相同。

## 核心功能——网站内容解析

我们在3.3.4节总结到通过网站查找信息是业务主线，并注意到了其中的难点，与之对应，对本系统来说，信息的解析与更新属于核心功能。它的行为模式如活动图4.5所示。



图 4.5网站内容解析活动图

程序需要代替用户登录网站，然后根据用户命令读取需要的网页，再解析网页内容，保存解析结果，供后续功能使用。

另外，3.4.4节中提到复杂点同样也是网站内容解析的难点。

1. 信息分布在不同的网站上，不同网站的访问方式可能不一样，有些需要登录，有些页面必须先建立会话才能访问。
2. 不同信息类型、不同网站的页面格式不同，交互流程也可能有些差异。
3. 有些信息不只一页，如通知列表、课程成绩（学期不同页面不同）等。

第一点提到网站可能有特殊的访问方式限制，因此上边的“登录网站”活动不只是平常所说的账号登录，还包括建立会话的活动，例如读取计算机学院网站的通知列表时，就必须先从学院网站主页获取会话信息。一般的登录活动如图4.6所示。



图 4.6登录网站活动图

系统先从用户那获取账号ID和密码，然后向合适的URL发送登录请求，等得到应答后，抽取并保存下登录凭证和会话信息。账号ID和密码是可有可无的，须根据将读取的网页而定。比如读取成绩页面需要账号；而读取学院网站通知列表不需要账号ID和密码，直接从主页获取会话信息即可。

由于存在1、2点所述的系统和数据异构性，需要为各个网站设计一系列连接代理类，并为各个类型的网页设计一系列解析器。

第1、3点也提到，像通知这样的信息分布在不同的网站上，而且每个网站的通知事项列表都不只一页，必须顺序地一页页读取。如果每次更新都把各网站的各页面扫一遍，即使根据日期筛掉许多页面后，对手机用户来说也很费时间和流量；对网站来说，每个客户端都扫描带来的压力也很大。为解决这些问题我们还需要部署代理服务程序，让代理服务器定时从网站更新信息，然后客户端从代理服务器更新解析后的“干货”。

## 组件和部署模型

根据以上分析可知，本系统至少需要网站内容解析、代理服务和手机前端组件，如组件图4.7所示。



图 4.7组件模型

网站内容解析组件负责各网站信息的读取和解析，它还包含实体类包。代理服务组件包括代理服务的接口、代理服务器端程序等。手机前端主要包括应用界面、流程控制等功能。



图 4.8部署模型

这些组件被部署在手机和服务器上，如部署图4.8图所示。网站内容解析服务同时被部署在手机和代理服务器上，代理服务端程序部署在服务器上，手机上需要有代理服务应用接口，手机前端在手机上运行。

本系统为这三个组件创建了三个项目，它们之间的依赖关系如图4.9所示。



图 4.9项目依赖关系图

图中下边的三个是是外部库。网站内容解析时先用jsoup做低层的解析，读取并把HTML文档转换为类似DOM的文档对象，再根据网页内容使用不同Parser解析为业务实体对象。手机通过Hessian协议与代理服务器通信。Hessian是一种二进制Web服务协议，它不仅很简单易用而且性能出众[14]。

# 系统设计

## 网站内容解析器

### 问题分析

我们在4.2节分析到，由于信息分布在不同网站上，不同网站的访问方式可能不太一样，而且不同信息类型、不同网站的网页信息格式不同，这种系统异构性，要求我们为各个网站设计不同的连接代理类，这种数据异构性要求我们为各个类型的信息设计不同的解析器。

之前也提到解析分成了两步，1)把HTML页面标准化并解析为文档对象，2)根据内容把文档对象解析为合适的实体类对象，如课程、通知对象等。

第一步是通用的，已经有不少很不错的类库能完成这一任务，jsoup就是其中很出色的一个。正如其名，它能标准化网上广泛存在的非标准HTML文档，这对本系统来说非常合适，我校有些网站比较旧，许多网页都不太符合W3C标准。

本系统使用jsoup读取网页并解析得到org.jsoup.nodes.Document对象。但4.2也提到过，根据要读取的内容和访问的网站，在读取目标网页（含有真正需要的信息的网页）前，可能需要做些准备工作，如登录、获取会话cookie等工作。这些工作并不统一，对不同网站有不同的实现方式，为此我们需要设计一系列ConnectionAgent类。ConnectionAgent除了代理org.jsoup.Connection的功能外，还要封装读取前的准备工作，对不同网站有不同的实现。

第二步要解析Document对象，从中抽取真正实体数据，并转换为合适的实体对象。不同类型的信息、不同的网页的解析方式不同，因此需要一系列Parser实现。

例如解析已选课程信息时，有活动图5.1这样的过程，师生服务网站的ConnectionAgent先登录网站，再把目标网页解析为文档对象，最后由已选课程解析器进而解析为课程列表。

图 5.1已选课程解析活动图

### 类模型

根据以上思想设计出的类图见附录图1，这里的图5.2是其缩略图。

可以通过ConnectionAgent.url(url:String)方法指定要读取的网页，然后使用get或post方法向之发送get或post HTTP请求。

这些类支持链式调用。所有的设置方法（setters）都会返回目标对象本身（this）这一引用，所以可以连续调用方法，例如像这样：connectionAgent.url(url).get()。

LoginConnectionAgent接口添加的setAccount方法可以设置用户名和密码，login操作可以触发登录过程。

可以用getConnection和getLoginConnection取得用于读取目标网页和用于登录的org.jsoup.Connection，子类可以根据需要配置它们，例如配置cookie、HTTP请求附带数据等。

抽象类AbstractConnectionAgent和AbstractLoginConnectionAgent实现了这两个接口，它们实现了一些通用功能，例如简单的代理功能，把收到的请求转发给内部的Connection对象。这里有两个有意思的功能值得一提：

AbstractConnectionAgent代理网络操作时简单处理了下IOException。网络操作难免遇到超时等IO异常，然而本系统有不少时候要连续进行网络操作（如扫描新通知），如果遇到异常就放弃或重头重试的话，之前解析的信息就都浪费了，重试也会费更多时间和流量，更大的麻烦是随着网络操作数的增加，每次操作都成功的概率会明显下降，也就是说出异常的概率可能变得很高。因此AbstractConnectionAgent代理网络操作时如果遇到了异常，会自动重试几遍，这样成功率就能提高许多。举个例子，假设每次网络操作遇到异常的概率是3%，连续读10个网页不遇到异常的概率为(1-3%)10 = 73.742413%；如果每次网络操作遇到异常再自动重复4次（AbstractConnectionAgent就是这么做的），即每个网络操作最多尝试5次的话，单个网络操作失败的概率为0.035，连续10个网络操作都不遇到异常的概率则为(1-0.035)10 = 99.999976%，成功率提高了许多！

图 5.2解析器设计类图

在让AbstractLoginConnectionAgent读取目标网页前，它会自动检测之前是否成功登录了网站，如果没登录或距上次成功登录过了很久，AbstractLoginConnectionAgent会自动执行login操作，这样在使用时设好目标网页和账号后就可以直接读取网页了，例如像这样使用：new SSFWWebsiteConnectionAgent().setAccount(用户名, 密码).url(url).get()。

针对不同的网站，有不同的ConnectionAgent具体类，它们实现了其特殊的网页读取方式，例如计算机学院网站的SCCEConnectionAgent会提前从网站主页得到会话Cookie，又例如师生服务网站的SSFWConnectionAgent收到login消息时会先带着用户名密码从门户网站得到认证信息，然后到师生服务网站再次验证，获取师生服务网站的认证信息。

有了这些ConnectionAgent后，Parser就可以从网站解析需要的信息。

抽象类AbstractParser实现了Parser接口并实现了通用的基本操作，如setConnectionAgent方法。

根据要读取的信息类型，衍生出了课程解析器、通知解析器和个人信息解析器。

BaseCourseParser抽象了已选课程和成绩查询的共性，它们依赖于SSFWConnectionAgent，BaseCourseParser实现了些通用方法，如课程表格的读取，表格字段的分拣分类等。SelectedCourseParser和ScoreParser具体化了已选课程解析和成绩解析的实现细节。它们的解析结果是List<Course>。

BasePostParser实现了通知解析的通用功能，如对时间、最大条目数、信息类别过滤器的设置。教务处网站通知解析器、计算机学院通知解析器、计算机学院学生网站通知解析器能够解析相应网站的各类信息，如通知、规定及表格、培养计划、公告、招聘快讯等等，它们依赖于对应网站的ConnectionAgent。它们的解析结果是List<Post>。

PersonalInformationParser解析器依赖于SSFWConnectionAgent，它可以从师生服务网站上读取基本信息、学籍信息、入学信息、毕业信息和联系方式等个人信息。由于这些信息的内容都是键值对，所以SSFWConnectionAgent的解析结果是Map<String, Map<String, String>>类型的信息，这是个双层映射，外层是信息类别到其内容的映射，内层是属性名到属性值的映射，例如用我的账号解析得到结果result后，result.get("基本信息").get("姓名")就是“柏杰”。

### 交互模型

这里以解析已选课程为例，看看这些类间的交互过程。

解析已选课程时的交互过程如顺序图5.3所示。解析器的用户创建SSFWConnectionAgent、SelectedCourseParser对象并告诉SSFWConnectionAgent账号和密码后，再把SSFWConnectionAgent对象传给SelectedCourseParser解析器对象，就可以让SelectedCourseParser对象进行解析工作。SelectedCourseParser让SSFWConnectionAgent读取已选课程网页，SSFWConnectionAgent发现还没有登录，自动登录师生服务网站，然后读取指定网页，给SelectedCourseParser返回文档对象。最后SelectedCourseParser解析得到的文档对象，得到List<Course>结果并返回给用户。



图 5.3已选课程解析顺序图

### 实体类

为了表示和返回解析结果，这里引入了本系统用到的实体类。

系统实体类的设计直接由3.3节总结的业务实体导出，如类图5.4所示。

Course（课程）实体类与3.3节中的完全一致，Post（通知）实体包含3.3中通知列表和通知详情的所有信息。个人信息比较特殊，本系统没有专门的个人信息实体，正如5.1.2节最后看到的，个人信息实际上是键值对集合，用Map映射对象表示不仅满足使用需求，也简单易用，而且更灵活，可扩展性好，增加属性很方便。

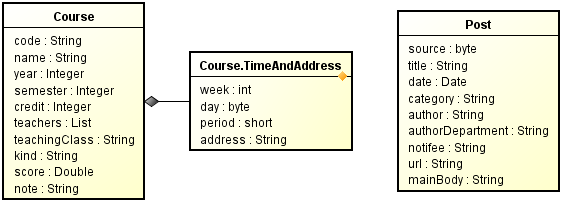


图 5.4系统实体类图

## 数据库

Android前端和代理服务器需要持久化实体对象，可以使用数据库实现这一需求。我们直接根据实体类结构映射出数据库模式，如模式图5.5所示。

每个课程实体对象，都有其编码、名称、学年学期等基本信息，此外还可能有任意多个时间地点对象，课程和其上课时间地点是组合关系，时间地点只属于一个课程，课程消失其时间地点也就没意义了。因此有course和time\_and\_address表，它们在Course和TimeAndAddress类的基础上加上id字段，另外time\_and\_address还有个对course.id的外键course\_id。

通知类的持久化比较简单，加上id字段后就得到了与其对应的post表。

个人信息是Map映射，我们把它持久化到EAV（Entity-attribute-value，实体-属性-值）表中。个人信息是双层映射，外层映射在EAV表中对应的实体名为PersonalInformation，其属性为类别名，如“基本信息”、“学籍信息”等，值都为null；内层映射在EAV表中的实体名为“基本信息”这样的类别名，属性为“姓名”这样的属性名，值为“柏杰”这样的映射值。

EAV表的实体和属性名字段不能为null，而且有UNIQUE约束(entity, attribute)，即没有重复的实体属性，每个实体的每种属性是唯一的。



图 5.5数据库模式图

## 代理服务器

如4.2节最后提到的，为了1)提高客户端更新通知的速度与效率，2）避免给学校服务器带来过大压力，本系统需要通知解析代理服务器。

代理服务器有API接口、服务器端、客户端三部分，其中服务器端有为客户端提供数据的前端和定时更新通知的后端两部分。

本系统使用Hessian协议实现客户端与服务器端的通信。使用Hessian能很方便地实现远程过程调用（Remote Procedure Call，RPC）。服务器端和客户端共用API，客户端通过API知道服务器提供了怎样的功能，以及如何使用功能。客户端实现了Hessian协议客户端，服务器端实现了API并通过Hessian为客户端提供服务。服务器端使用Servlet实现。

本系统使用GAE（Google App Engine）作为Servlet容器。GAE要求前端在收到查询请求后1分钟内给出结果，而计划任务可以运行10分钟。GAE为前端、后端模块提供不同服务器，每类服务器都进一步划分了不同性能的服务器，每个模块都可以独立选择服务器类别和性能。本系统大部分时间在进行IO操作，计算量不是很大，前端后端都使用了性能等级最低的服务器F1、B1，这样在免费配额下总可运行时间最长。

服务器系统的模块结构如图5.6所示。

计划任务按时触发后端程序更新通知，更新要用到上一节设计的解析器，解析结果存入App Engine Datastore。Datastore是Google的一个分布式的无模式对象数据库，它不是关系关系数据可，其结构类似带索引的映射，程序可以根据主键取得Datastore上的对象，也可以根据属性查询对象，但是查询条件用到的属性必须在索引中。



图 5.6服务器端结构

前端使用官方Hessian库实现服务器API，内部实现会从Datastore中读取之前解析到的通知对象。

客户端扩展了第三方Hessian Android库，以适应本系统的需求。代理服务用户通过客户端使用API提供的服务。

## 手机客户端

手机客户端主要负责展示解析结果。为避免重复的解析过程，减少较费时的网络操作，手机客户端也会在其本地数据库缓存数据，来提高对用户的相应速度。下边分别介绍数据存储和GUI的设计。

### 数据存储

本项目使用Android系统自带的SQLite数据库，数据库模式与5.2节的设计一致。SQLite数据库有个比较方便的特性，虽然建立数据库表时可以指定每个字段的优选数据类新，但实际使用时可以在任意字段（INTEGER PRIMARY KEY字段除外）存任意类型的数据。这让我们用到的EAV表更加灵活方便，EAV表中value字段大部分情况是字符串，所以其优选类型为VARCHAR，然而有时也会遇到其他类型的数据，例如photo（头像）属性的值是照片，对数据库来说是BLOB型的，SQLite允许实际数据类型与优选类型不同，所以基本信息的照片属性也可以存到EAV表中。

在数据库用户和系统数据库操作接口之间，有一个简单的数据库管理器，它封装了一些常用操作，方便上层使用，也方便数据库操作的维护与优化。它的主要工作是持久化和反持久化课程等实体对象，即把Course对象存入数据库或根据数据库的数据返回Course对象。

此外本项目也实现了部分Content Provider的功能。它是Android应用四大组件之一，它提供了一个统一的内容访问接口。利用它可以便利高效的实现数据和UI的绑定，即UI上显示的数据会与Content Provider内部的数据保持同步，及时更新。像通知列表这样包含许多条目的界面，利用它可以非常明显地提高用户响应速度。

这里还有一个较具体的问题涉及到数据库。由于现有通知有很多条，第一次更新通知时会从代理服务器下几千条通知，这给代理服务器带来过大压力。为解决此问题，程序应当预置现有的通知数据。

程序的设置信息保存在[SharedPreferences](http://developer.android.com/guide/topics/data/data-storage.html#pref)中，这也是Android应用程序的一般做法。SharedPreferences对象为Android应用提供了一种便利地持久化键值对的方法，Android系统会把用其存储的数据以XML文档的形式保存在程序目录中。

### GUI

手机前端的主要任务是呈现解析到的信息，因此GUI的层次结构也围绕其要呈现的内容设计。图5.7展示了本应用的导航模式和各界面的结构。

主界面列出了本应用的功能，例如本周课程表、成绩单、通知等。点击这些按钮后会跳转入相应内容的展示界面。

因为个人信息实际上是键值对集合，所以个人信息界面以列表的形式显示这些信息。

课程表界面以类似纸质课程表的形式显示课程。它有两种变体，一种本周课表，一种总课表。左右滑动可以切换星期，也可以点击上方的Tab切换。默认选中当前星期，例如星期二打开应用时会自动选中星期二。

成绩单界面展示了各学期的成绩。按学期分页，每页中有选中学期各科的课程名称、学分、考试成绩、绩点等信息，最后还有些统计数据，例如平均绩点、通过率。

通知列表界面展示了从各网站解析到的通知。按通知来源分页，每页都列出了相应网站的所有通知，每一项都包括通知标题、内容类别、发布时间、发布人等信息。

在课程表和成绩单界面点击任意课程都可以进入课程详情界面，来显示选中课程的详细信息。这个界面列出了解析到的此课程的所有信息。这个界面也允许修改课程信息，另外手动添加课程的功能也是由此界面实现的。

在通知类表界面点击任意通知项后跳转入通知内容界面。此界面展示相应通知的所有信息，包括通知的正文。第一次打开通知时，会自动从网站上下载通知正文，并保存到数据库。此界面还有个在浏览器中显示的功能，用户可以通过此功能查看此通知的原始网页。



图 5.7 GUI层次结构图

这些界面中的大部分都有刷新按钮，用于手动更新相应信息。例如在通知界面点刷新可以立即检查是否有新通知，在成绩单界面点刷新可以更新成绩信息。

从所有界面都可以跳转入设置界面。在设置界面可以设置当前周次、查询账号和密码、通知更新频率等信息。

# 系统实现

系统设计确定了系统的大体结构、以及各部分间的协作关系，系统实现在其指导下解决具体的实现细节问题。本章主要介绍本系统涉及的较独特的实现问题。

根据上一章的设计，本系统分成三个子项目：WebsiteParserForTJUT（网站内容解析器）、WebsiteParserForTJUT-proxy-GAE（代理服务器）、移动校园Android客户端。下边分别介绍它们，然后总结它们之间依赖关系。

## 网站内容解析器

### 包结构

网站内容解析器有四个包：org.orange.parser.util、org.orange.parser.entity、org.orange.parser.connection和org.orange.parser.parser。其结构如图6.1所示。



图 6.1包结构图

org.orange.parser.util包括一些工具类。BitOperate有一些位操作工具方法，主要用于操作Course.TimeAndAddress里的时间，详见6.1.2节。StringUtils包括一些字符串工具方法，例如删除空白字符方法。

org.orange.parser.entity包中有Course和Post实体类。

org.orange.parser.connection包中有一些列ConnectionAgent和Cookie等类。

org.orange.parser.parser包有一些类Parser还有Constant常量类、ParseListener等辅助类。

下边几节分别介绍下entity、connection和parser包。

### 实体类包

实体类图见5.1.4节的图5.4。实体类的大部分方法都是setter和getter方法。对于集合类型的属性，例如任课教师、上课时间地点，还有些addXXX快捷方法，可以直接向集合添加元素。

对于Course.TimeAndAddress和Course的组合关系，由于很少根据TimeAndAddress找其所在Course，我们可以直接用List<TimeAndAddress>类型的Course.timeAndAddress属性实现。

为了便于持久化Course.TimeAndAddress、便于运算、节省资源，我们用位状态表示周次、星期和节次。例如对于TimeAndAddress.week:int周次属性，其0-20位的1/0表示是/否有这一周。如16进制0x00 00 01 02（即二进制0b 0001 0000 0010，省略前导0）表示1、8周有课。TimeAndAddress依赖org.orange.parser.util.BitOperate工具类来执行位操作。

### Connection代理包

Connection代理和解析器的类图见附件1或5.1.2节中的图5.2。

AbstractConnectionAgent实现了ConnectionAgent，它提供了接口的基本实现。它添加了个beforeExecute方法，在实际执行网络操作get或post之前被调用，子类可以在此检查状态。AbstractConnectionAgent本身在这检查是否设置过url，如果在get或post前没有调用url，则在标准错误终端显示警告信息。

如5.1.2节所述，AbstractConnectionAgent代理get和post操作时，遇到IOException会自动重试几次，可以通过retryCount(int numberOfRetries)方法设置重试次数。

AbstractLoginConnectionAgent继承了AbstractConnectionAgent并实现了LoginConnectionAgent接口。AbstractLoginConnectionAgent在beforeExecute方法中检查登录状态，如果执行post或get操作前还没登录，或自从上次成功登录已经过了较长的时间，它会尝试自动登录。

类似AbstractConnectionAgent，AbstractLoginConnectionAgent执行login操作遇到IOException时会自动重试几次，重试次数与AbstractConnectionAgent一样（都是AbstractConnectionAgent.mRetryCount）。

用了AbstractConnectionAgent和AbstractLoginConnectionAgent之后，子类只需添加与网站相关的特殊实现就可以了。例如SSFWWebsiteConnectionAgent继承了AbstractLoginConnectionAgent。它自己只添加了login的执行部分，即获取门户网站的登录凭证Cookie和师生服务网站的会话Cookie，仅40多行代码。

### 解析器包

AbstractParser实现了Parser接口。具体来说它实现了setConnectionAgent和setParseListener方法。此外，在执行parse前，如果没有执行过setConnectionAgent，它会抛出IllegalStateException异常。

根据解析内容的不同，AbstractParser进一步特化为PersonalInformationParser、BaseCourseParser、BasePostParser类。

PersonalInformationParser获取个人信息网页后，通过Document. select("#form1 .tableGroup")得到基本信息、学籍信息等个人信息类别组。以每组的标题作为键，每组的内容作为值，存入HashMap。内容使用readKeyValues方法读取，其参数为组中的第一个table HTML元素，返回Map<String, String>，其键类似“姓名”、“学号”，其值类似“柏杰”、“20106173”。

BaseCourseParser是SelectedCourseParser和ScoreParser的抽象，它包含一些解析课程信息用到的通用方法。例如readCourseTable方法遍历课程信息表格，对每行调用readCourse方法读取并保存，在此之前它会调用readHeading方法读取表头信息。readCourse根据表头信息，把表格行参数的各列元素传给合适的方法进一步解析。这些具体列元素的解析方法，通用的由BaseCourseParser实现，如parseStringProperty；特殊的由子类实现，如parseTimeAndAddress由SelectedCourseParser实现。SelectedCourseParser和ScoreParser除了负责这些具体单元格的解析外，还负责相应网页的获取，如ScoreParser要依次获取各学期的成绩网页。

BasePostParser主要负责设置过滤器，例如setFilterMinDate方法设置了需解析通知的最早时间，即只需解析此时间以后的通知项；而setFilterCategories方法设置了需解析通知的类别，例如Post.CATEGORYS.SCCE\_STUDENT\_NEWS（即字符串"新闻中心"）、"教学建设文件"等。AcademicAffairsPostParser、SCCEPostParser、SCCEStudentPostParser分别负责教务处、计算机学院网站、计算机学院学生网站的具体解析工作，例如SCCEStudentPostParser对每种需解析的通知类别，解析其相应通知列表网页各页的信息。因为通知列表是按时间倒序排列的，所以解析到早于MinDate以前的通知时就可以停止了。

解析器包中的Constant常量类集中了用到的网页的URL，便于统一管理和维护。

解析器包中还有个ParseListener监听器类，可以用它监视解析进度，各解析器也会向它发送日志信息。

### 解析器的使用

以解析个人信息和已选课程为例：

LoginConnectionAgent connectionAgent = new

SSFWWebsiteConnectionAgent().setAccount("20106173", "20106173");

Map<String, Map<String, String>> personalInformation =

new PersonalInformationParser().setConnectionAgent(connectionAgent).parse();

List<Course> courses = new

SelectedCourseParser().setConnectionAgent(connectionAgent).parse();

如果用System.out.println显示出来，结果类似这样：

-------------------- 个人信息 --------------------

{联系方式={家长电话=13081106100, 个人邮箱=baijie1991@gmail.com, 个人手机=15620906177, 校区=2, 家长工作单位=部队, 家庭邮编=050207, 家庭电话=0311-87973959, 家长关系=01, 家长姓名=柏俊泽}, 基本信息={姓名=柏杰, 民族=23, 证件号码=130103199103082112, 出生日期=1991-03-08, 政治面貌=3, 学号=20106173, 证件类型=1, 姓名拼音=Bo Jie, 籍贯=130400, 国家地区=01, 性别=1}, 学籍信息={班级=0607201006, 是否在籍=1, 年级=2010, 学生类别=1, 是否在校=1, 院系=06, 学制=4, 专业=0607, 学籍状态=1}, 入学信息={招生季度=01, 考生号=10130103150617, 入学年级=2010, 入学日期=2010-09-12, 学习形式=01, 生源地=130100, 培养层次=01, 外语种类=01}}

-------------------- 已选课程 --------------------

[0686046 毕业设计（论文） null 宁红云 15 null null 2013 2 null null

null, 0666136 计算机结构 null null 3 null null 2012 1 null null

null, 0666166 面向对象的程序设计 null null 3 null null 2012 1 null null

null, 0666196 运筹学 null null 3 null null 2012 1 null null

null, 0666206 网络体系结构与程序设计 null null 3 null null 2012 1 null null

null, 0666226 数据库与功能分析 null null 3 null null 2012 1 null null

null, 0666336 多媒体：概念与方法 null null 3 null null 2012 1 null null

null, 0668036 操作系统 null null 4 null null 2012 1 null null

null, 1190076 毛泽东思想和中国特色社会主义理论体系概论 null null 6 null null 2012 1 null null

null, 0662016 计算机网络 null null 3 null null 2012 2 null null

null, 0666026 会计学基础概念 null null 3 null null 2012 2 null null

null, 0666076 软件工程：软件构件与软件重用 null null 3 null null 2012 2 null null

null, 0666096 信息和电子商务技术 null null 3 null null 2012 2 null null

null, 0666146 对象设计与建模 null null 3 null null 2012 2 null null

null, 0666236 高级数据库 null null 3 null null 2012 2 null null

null, 0686016 企划与项目开发 null null 3 null null 2012 2 null null

null, 0666036 财务管理 null 郭力杰 3 null null 2013 1 null null

1-5,8-14 周 星期三 3-4 节 北楼3 3-0201

1-5,8-14 周 星期一 7-8 节 北楼3 3-0201, 0666106 计算机项目管理 null 夏承遗 3 null null 2013 1 null null

1-5,8-14 周 星期五 3-4 节 北楼1 1-0202

1,3,5,9,11 周 星期二 7-8 节 北楼1 1-0202, 0666176 工业计算机系统 null 杨鹏 3 null null 2013 1 null null

null, 0666266 分布式计算机系统 null 蔡靖 3 null null 2013 1 null null

null, 0666276 微机硬件组成原理 null 韩盛磊 3 null null 2013 1 null null

1-5,8-14 周 星期五 1-2 节 北楼1 1-0202

1-5,8-15 周 星期二 1-2 节 北楼1 1-0202, 0686026 市场调研 null 李双喜 2 null null 2013 1 null null

null, 0686416 数据库课程设计 null 董晨 2 null null 2013 1 null null

null]

已选课程只有2012年以后的信息且2012年的课程没有上课时间地点，是因为师生服务网站最近两年刚上线，但是ScoreParser可以解析到所以课程的成绩。

以上示例代码和结果摘自网站内容解析器的继承测试类org.orange.parser.integration.ParserTest。

## 代理服务器

根据5.3节代理服务器的设计，WebsiteParserForTJUT-proxy-GAE项目又分为api、client、server三个子项目，server包括server:shared、server:appengine-modules-default、server:appengine-modules-scanner、server:appengine-modules-ear四个子模块。WebsiteParserForTJUT-proxy-GAE根目录下的settings.gradle已经配置好了子项目信息，Gradle可以自动对各项目进行编译、测试、部署等任务，各项目/模块间的依赖也使用Gradle管理。

### API

API项目定义了Server为Client提供的服务接口，Server和Client都依赖于它。

具体来说此项目包含有GetterInterface接口，它有两个方法：String echo(String in)、List<Post> getPosts(Date start, Date end, int max)。

echo回传方法用于测试服务器连通性。

getPosts方法用于从代理服务器获取通知。服务器应当返回[start, end)时间区间内的通知，包括start，不包括end，start设为null表示-∞，end设为null表示+∞。最多只返回max条通知，max小于-0时，表示无max限制。

由于API用到了Post，它依赖于WebsiteParserForTJUT项目。

### Server

Server定时更新通知，并通过实现API的接口为Client提供服务。

Server使用了Servlet框架，此项目部署在GAE容器中。

GAE支持多模块，可以分别为每个做单独配置，例如选用不同性能的服务器。GAE前端和后端服务器独立计算配额。

appengine-modules-default是前端模块，它对Client来说也是默认模块。appengine-modules-scanner是后端模块，由它来完成通过定时更新任务。这两个模块打包为war，可以单独部署。shared共享模块包含前两者共用的数据存取工具，它不可以被单独部署。appengine-modules-ear模块把default和scanner模块打包为ear，并做些全局配置，例如本appengine-application的ID。

default模块的Getter类继承了Hessian库的HessianServlet并实现了API项目的GetterInterface接口。通信过程由Hessian库处理，Getter只需实现GetterInterface接口即可。Getter类使用shared模块的数据存取工具，查询并返回符合条件的通知。

scanner模块的UpdateFromSchoolWebpage类，先查询最新通知的时间，再用网站内容解析器解析此时间以后的通知，最后再使用shared模块的数据存取工具保存解析结果。scanner模块设置了cron计划任务，会在每天的0、6、12、18点更新通知。

shared模块主要负责数据的存取。本项目使用GAE的Datastore存储服务，它是一种无模式的分布式的NoSQL存储服务，它不是关系数据库，更像是带有索引的映射的集合。

在一定配额内Datastore服务是免费的。目前免费用户最多可存储1GB的数据，最多200个索引，每日最多读/写/小操作的次数为50,000次。

存一个对象要用的写操作次数与其使用的索引个数有关。而且没有加入索引的属性，在查询时是查不到的，所以索引的设置比较重要。我们只需存储Post对象，查询Post时只会用到date属性，所以只为date建立索引。这样设置后每天可以免费存取上万条Posts.

本项目没有直接使用Datastore的低层API接口，而是选用较高层的JDO（Java Data Objects）对象持久化框架，一方面便于使用，另一方面也降低了对GAE的依赖，方便以后把Server端部署到其他Servlet容器上。

在实际实现时遇到了个麻烦。要存储的Post类是在WebsiteParserForTJUT项目中定义的，对shared模块来说Post在外部库生成的jar包中，这带来了些麻烦。首先我们不能使用@Persistent这类的Java注解，因为shared无法修改源代码，不过我们可以改用独立的XML文件来配置，shared/src/main/resources/META-INF/package.jdo就是此配置文件。更大的麻烦是GAE用到的JDO实现使用了class增强技术，它根据package.jdo中的信息，为Post.class添加JDO存取方法或属性，但是它不能很好的处理jar库中的class文件。让shared去修改所有依赖于它的模块的jar文件，是不现实的。shared模块的解决方案是把Post.class复制出来，然后对其进行增强处理，把进过增强的Post.class作为shared模块的制品再发布。依赖于shared的项目，应当把shared.jar的优先级提高，classpath中shared.jar应该在WebsiteParserForTJUT的jar的前面。就default和scanner模块来说，它们在appengine-web.xml配置文件中，提高了shared.jar的优先级。上边说到的再发布过程，已经编写在shared目录下的build.gradle文件中了，shared Gradle子项目的classes任务依赖于这个增强再发布任务（注：依赖于shared的其他子项目也依赖于shared:classes），Gradle会自动执行这些工作。

### Client

其实正常情况下代理服务的用户直接使用API接口和Hessian客户端就可以了，不过由于我们都知道的原因，与Google服务进行网络连接时需做些迂回工作。Client在Hessian连接器的基础上扩展添加了这些特殊工作。

Client代理了客户端与GAE代理服务器间的低层网络操作。它把向GAE服务器发送的HTTP请求发给了www.google.com.hk，使用SSLSocket。它实际上只直接与www.google.com.hk通信，Google的内部网络应该会根据HTTP消息的Host头把请求发至恰当的服务器。

## 手机客户端

如上一章所述，手机客户端只需负责信息的存储和展示。它是个普通的用于显示信息的Android小应用，其大部分组件都是普通的Activities，它们的结构关系符合5.4.2节的设计。本节对这些组件做下一般性介绍并简述了本应用特有的每日课表生成算法。

手机客户端项目也使用Gradle进行配置管理，它使用了最新的Android构建系统，建议配合使用Canary通道的最新版Android Studio IDE进行开发和维护。

本客户端的许多界面都需要用列表的形式展示信息，它们都使用android.widget包的ListView来实现，并通过ListAdapter把数据项适配到ListView中。ListView根据需要触发ListAdapter.getView(int position, View view, ViewGroup parent)方法，把各数据项对应的View放入ListView中。ListView不会一次对所有位置（position）调用getView方法，它每次只获取几个需要的位置的View，例如向下滚动时后续位置的数据项才被绑定到View上。而且getView返回的View可以被重用，其第二个参数就是以前getView返回的而现在不再使用的View，可以直接修改里边的数据，而不重新创建View。本应用的课程表、成绩单、个人信息界面都继承自android.widget.BaseAdapter，使用自定义的ListAdapter。而通知列表界面继承自widget包的CursorAdapter。

如5.1.4节所述的，通知列表界面的后台数据是Content Provider提供的。配合Loader和Content Provider，即使有上千条的通知，通知列表界面也不会让用户感到有延迟时间。android.content.Loader负责与Content Provider交互，它自动异步读取数据，监视数据的变更，并能够尽量重用数据，如横竖屏切换重启Activity时它自动重连cursor而不用重新执行查询操作。android.app.LoaderManager负责管理Loader，例如管理Loader的创建、重置和结束。通知界面通过LoaderManager.LoaderCallbacks得知Loader的状态信息并获取Loader加载到的最新数据。具体来说是在LoaderCallbacks.onLoadFinished(Loader loader, Cursor data)得到最新的data，我们在此更新通知界面的ListAdapter。类似数据库，Content Provider也是以Cursor的形式返回查询结果，所以上边的data是Cursor型的，而通知列表界面使用的ListAdapter继承自CursorAdapter，这个自定义ListAdapter只需实现newView:View和bindView(View, Context, Cursor)方法就可以。

本Android客户端没有涉及到太难的算法问题，相对较复杂的计算问题就是传统课程表的生成，这里简述下其思路。这里使用二位链表数组实现，二维表的格式类似课程表，行表示不同节次，列表示周一至周日。因为可能有星期或节次未知的情况，多加一行放未知节次的课，多加一列放未知星期的课。由于有星期和节次都相同但周次不同的课，所以二维表的元素不是课程而是课程链表。有了这个容器后，我们只需遍历每个Course的每个TimeAndAddress，根据TimeAndAddress把课程放到二维表合适位置，最后只需遍历一遍，就可以得到排好序的课表。本应用在实际实现时以小节为粒度排序，能正常处理类似2、3节这样的时间，在显示前会自动合并连续出现的相同课程，例如1-4节都上A课，则A只显示一遍。

程序界面截图见附录4。

## 项目依赖关系

从5.3节的图5.6可以情况的看到，WebsiteParserForTJUT-proxy-GAE依赖于WebsiteParserForTJUT，而手机客户端同时依赖于前两者（客户端解析通知以外的信息时依赖于WebsiteParserForTJUT，图5.6省略了这个依赖）。此外它们各自还依赖于一些第三方库。这些依赖关系已经在各项目的build.gradle文件中配置好了，可以用gradle(w)或支持Gradle的IDE直接进行编译、测试或部署等活动，Gradle会自动从远程仓库下载需要的库。不过这里有个例外，由于代理服务器客户端用到的com.exadel.flamingo.android:flamingo-android-hessian-client:2.2.0，没有发布到远程仓库上，所以需要手动下载并安装到本地Maven仓库：根据[Maven官网的指导](http://maven.apache.org/download.cgi)下载并安装Maven，然后运行exadel-flamingo-2.2.0\bin\ android-flamingoinstall即可。

参考文献

[1] 中国互联网络信息中心. 第33次中国互联网络发展状况统计报告（2012年12月）[R/OL]. 中国互联网络信息中心（CNNIC），http://www.cnnic.net.cn/hlwfzyj/hlwxzbg/hlwtjbg/201401/t20140116\_43824.htm. 2014年01月16日 10:55

[2] 沈野萤. 大学生手机使用情况调查研究[J]. 今传媒. 2012(11)

[3] 孙耀庭,陈信. 开放大学“移动校园”构建的探索[J]. 中国教育信息化. 2007(19)

[4] 王韦. 基于移动校园平台的移动商务创新价值浅析[J]. 科技信息. 2010(25)

[5] 李涛,陈瑛,黎志生. 木棉BBS客户端开启校园移动应用[J]. 中国教育网络. 2012(02)

[6] 陈泽恩. Android校园网移动客户端的设计及实现探索[J]. 中国新通信. 2013(12)

[7] 谢文焘,董黎刚. 基于Android的校园网移动客户端设计与实现[A]. 浙江省电子学会2012学术年会论文集[C]. 2012

[8] Campus Computing Project. The National Survey of Computing and Information Technology[R/OL]. Campus Computing Project, http://www.campuscomputing.net/item/2013-campus-computing-survey-0. Oct 17 2013

[9] Open Source HTML Parsers in Java[Z/OL]. Java-Source.net, http://java-source.net/open-source/html-parsers. 2013

[10] Jonathan Hedley. jsoup: Java HTML Parser[Z/OL]. jsoup, http://jsoup.org/. 2013

[11] Bauke Scholtz, et al. The Answers of "What are the pros and cons of the leading Java HTML parsers?"[Z/OL]. Stack Overflow, http://stackoverflow.com/questions/3152138/what-are-the-pros-and-cons-of-the-leading-java-html-parsers/3154281#3154281. Aug 28 2012 11:24

[12] Krishna Prasad. HtmlUnit vs JSoup: HTML Parsing in Java[Z/OL]. Krishna's Blog, http://krishnasblog.com/2012/12/19/htmlunit-vs-jsoup/. Dec 19 2012

[13] Scott Ferguson, Emil Ong. Hessian 2.0 Web Services Protocol[R/OL]. Caucho, http://hessian.caucho.com/doc/hessian-ws.html. August 2007

[14] Daniel Gredler. Java Remoting: Protocol Benchmarks[R/OL]. http://daniel.gredler.net/2008/01/07/java-remoting-protocol-benchmarkb/. Jan 7 2008 18:15

[15] Dutchguilder. Iterative development illustration[OL]. http://en.wikipedia.org/wiki/File:Development-iterative.gif. Oct 16 2007