南京大学

本 科 毕 业 设 计

院　系 软件学院

专　业 软件工程

题　目 基于 SpringMVC 框架的轻量级任务协同系统的微信模块的设计与实现

年　级 2011　学　号 111250196

学生姓名 徐　诺

指导教师 荣国平　职　称 讲师

论文提交日期 2015 年 5 月

南京大学本科生毕业论文（设计）中文摘要

毕业论文题目： 基于 SpringMVC 框架的轻量级任务协同系统的微信模块的设计与实现

软件学院 院系 软件工程 专业 2011 级本科生姓名：徐诺

指导教师（姓名、职称）：荣国平 讲师

摘要：

在社会不断进步、经济快速发展的大背景下，众多企业迅速扩张成多元化公司，这易导致员工以及多方合作之间的管理和沟通效率低下，而现有的大型协同办公系统过于庞大、不灵活且需要安装客户端。随着轻应用盛行，结合微信的兴起和良好的发展势头，轻量级任务协同系统应运而生。本系统在 PC 端提供网页版，方便操作，移动端借助微信实现移动办公，无需安装任何额外的客户端或插件，真正实现了跨平台、高易用性、高效率的轻量级应用。

本系统有四个角色，包括系统管理员、任务管理员、内部员工和外聘人员。系统的 PC 端主要提供给系统管理员和任务管理员使用，主要包括内外员工管理、任务管理、日志管理、资讯管理的功能；系统的移动端即微信模块主要提供给内部员工和外聘人员使用的，其功能主要有绑定账号、任务管理、资讯管理、查看信息、联系同事、日常聊天。

本系统基于 SpringMVC+Hibernate 框架进行开发，总体遵循分层的设计思想，系统架设在 SAE 服务器上，使用的是 MySQL 数据库。本系统前端主要基于 Bootstrap+jQquery 框架搭建，还用到了 jsp、ajax、JavaScript 技术。本系统后台用到了 log4j 框架和缓存技术。另外，微信模块还使用了 Wechat4j 框架封装微信公众平台服务，Ansj 框架封装了分词服务，日常聊天功能中还使用到图灵机器人平台提供的相关服务。

本人在项目中主要负责移动端即微信模块的设计与实现，并参与了项目总体体系结构设计、数据库设计和界面设计。

关键词：任务协同、微信、SpringMVC 框架、Hibernate 框架

南京大学本科生毕业论文（设计）英文摘要

THESIS：The Design and Implementation of the WeChat Module of Light-weight Co-work System Based on SpringMVC

DEPARTMENT： Software Institute

SPECIALIZATION: Software Engineering

UNDERGRADUATE: 2015

MENTOR: Guoping Rong

ABSTRACT：

As rapid development of society and economy, many an enterprise is expanding fast toward a diversified one, potentially leading to the inefficient management and communication between both corporate workers and multilateral cooperation. However, client in need, inflexibility and oversize of the existing cooperative office system has been working against itself. Meanwhile, prevailing light apps joining hands with promising future of WeChat give birth to the light-weight Co-work System. The system provides the convenient web edition for PC users and realizes mobile office through WeChat, without installing any extra apps, finally making the multi-platform, easy-to-use and high efficient lightweight apps become real.

This system includes four roles, namely system administrator, task administrator, internal and external staff. The PC side is mainly for system administrator and task administrator to manage the internal and external staff, the tasks, the logs and the news. The WeChat module is mainly used by both the internal and external staff regarding the binding of accounts, management of the tasks and messages, news viewing, contacting colleagues and daily chat.

Based on SpringMVC+ Hibernate framework, this system adheres to the layered design pattern as a whole. The erection of this system is on the SAM server, uses MySQL data base. The front end is established based on Bootstrap+ jQquery framework by employing jsp, ajax and JavaScript. The back end adopts log4j framework and the caching technology. Besides, the WeChat module applies Wechat4j framework to the encapsulation of WeChat's public information service platform; Ansj framework encapsulates the segmentation service; its daily chat function adopts some services provided by the Turing Robot Platform.

In this project, the author is mainly responsible for the design and realization of the WeChat module and participates in the design for the structure of the system, data base and interface.

KEY WORDS: Co-work, WeChat, SpringMVC, Hibernate

目 录

图目录

表目录

第一章 引言

1.1 项目背景

在社会不断进步、经济快速发展的大背景下，众多企业迅速扩张拓展成多元化公司，然而这样的转型升级必然会带来一定的副作用。其一，企业经营规模不断扩张导致内部员工和组织结构变更频繁，从而导致企业内部在任务分配协调、部门间和员工间沟通协商等方面产生管理上、沟通效率上的问题。其二，企业的多元化理念和跨领域的发展方向将会形成多方合作的业务模式，诸如主营业务上的分模块外包托管、外聘专家顾问等等合作形式，使得公司的组成结构更为复杂，不确定因素大大增加，也相应增加了在人员管理、沟通协作、权限分配等方面的难度；其三，企业的迅猛发展使得企业对员工素质和技能的要求日益严格，且企业常常会需要购买其他企业生产的系统或设备并进行个性化定制，所以许多企业一般都会邀请企业外部人员给员工定期做集中培训辅导。而这种短期合作的项目恰恰需要通过企业内部人员和外聘讲师双方的相互配合协调才能顺利完成。基于以上三方面的因素，多方协同的任务管理迫在眉睫。

目前，企业应对以上需求的方法主要有两种：传统任务管理（手写卡片、纸质文件等）方式和使用企业大型内部系统。传统任务管理方式的局限性：

1. 需要大量人力物力支持；
2. 不易于更改，任务管理汇总效率低下；
3. 信息流通不方便、时效性差。

企业大型内部系统的产生与使用一定程度上弥补了传统任务管理方式的缺陷：在任务管理和企业内部人员管理上更加高效，内部人员之间信息传递更加方便及时。另外，大型内部任务管理系统功能齐全，一些定制的内部系统则更为贴合企业的实际需求，但功能提升的同时也不可避免地带来一些新的问题：

1. 系统过于庞大，导致消耗资源较大、运行速度变慢；
2. 系统运行需要安装客户端，可移植性差、不能跨平台或者要安装不同平台的软件，给用户造成不便；
3. 大多为内部系统，对于内部员工来说离开公司后使用不方便，而对于外聘人员则更为不便；
4. 大型系统功能复杂而强大，但易用性差，导致外聘人员上手难，而且大部分功能是不常用甚至根本不会用到的。

为了满足市场的需求和实际工作的需要，我们决定开发一个基于 SpringMVC

框架的轻量级任务协同系统，使用微信公众平台实现移动办公，整个系统无需安装任何额外的客户端或插件，在任务协同管理的功能方面也进行了精简，只保留最常用的功能，实现跨平台、高易用性、高效率的轻量级应用。

1.2 轻量级任务协同系统的研究现状

1.2.1 任务协同的应用与研究

随着 IT 行业的不断进步，从上级建立项目分配任务、下级部门各自执行任务的断层式、低效的工作模式逐渐向扁平化的企业任务管理体系转变，从传统办公模式逐渐向 OA 系统辅助办公转变，使得企业可以更高效、更顺利地完成任务，降低人工等成本，实现可持续发展[1]。

今年 2 月，CNNIC[1]在京发布的第 35 次《中国互联网络发展状况统计报告》中指出，截至 2014 年 12 月，我国网民规模达到 6.49 亿且近十年来都保持良好的增长势头，并且手机网民规模占比已高达 85.8%[2]。可见，移动端的开发是重要且必要的。

从国内外的协同系统来看，国外的相关技术相对较为成熟[3]，国内的协同系统还有很大进步空间。例如：

思科达协众 OA[2]：完全基于 B/S 架构，移动端为 PC 端的延展，同样借助网页访问，可定制系统个性化功能模块和工作流。

用友致远 A8[3]：完全基于 C/S 架构，通过不同平台的不同客户端使用系统，适用于中大型组织机构实现智能化协同办公。

以上两种是国内比较典型的任务协同系统。B/S 架构均是通过网页操作，没有充分考虑到移动端的特性和局限性。而 C/S 架构在跨平台方面投入成本高，对用户并不友好，不同平台都要下载安装不同的客户端，尤其对于内存容量并不充裕的移动设备来说影响更大。

因此，PC 端应用 B/C 架构，而移动端则应该根据其特性采用不同的方法。

1.2.2 轻应用的应用与研究

百度在 2013 年 8 月份高调提出“轻应用”这一概念，微信、UC、支付宝等

[1] China Internet Network Information Center，中国互联网络信息中心，官网：http://www.cnnic.cn/

[2] 协众软件官网：http://www.cnoa.cn/

[3] 致远 A8 协同管理软件：http://www.ufidaoa.com/OneProduct/24.html

公众平台相继为其摇旗呐喊[4]，于是“轻应用 Light App”逐渐进入人们视野并逐渐为人们所接受。

Light APP 介于 Native App 和 Web APP 之间，是以超级 APP（如：百度、微信、支付宝等）的开放平台为技术支撑的中间态服务产品[5]。于开发者来说，其跨平台开发、维护的成本相对低廉；于用户，免去下载安装不同平台的客户端。轻应用完全符合系统对于移动端的需求。

据企鹅智库[4]发布的《解密微信：微信平台首份数据研究报告》显示，55.2％的用户每天打开微信 10 次以上，近八成用户关注微信公共号用以获取资讯等[6]。迅猛增长的用户量、稳定的日活跃度和高占比的微信公众平台关注者促使微信公众号作为轻应用的一种呈现形式脱颖而出。因此，系统移动端最终决定基于微信公众平台开发。

1.3 论文的主要工作和组织结构

本人在基于 SpringMVC+ Hibernate 框架的轻量级任务协同系统中主要负责任务协同系统的微信模块的相关设计与实现。所以，本文主要介绍了任务协同系统的项目背景、具体架构设计和相关理论支持，基于 Wechat4j 框架的微信模块相关的理论知识、技术细节、模块的详细设计实现和单元测试，以及在实践过程中遇到的问题与解决心得。

第一章：引言部分。本章主要阐述该系统的市场背景和实现的必要性，介绍了任务协同系统的研究现状以及基于微信公众平台的轻应用的发展前景，并概述了本文的主要工作。

第二章：模块技术综述。本章主要介绍基于 Wechat4j 框架的微信模块的实现与开发过程中涉及的相关理论支持和技术实现。

第三章：系统需求分析与概要设计。本章讨论了系统用户角色，概述了系统功能；通过对系统特性等的需求分析从而得出系统的功能性和非功能性需求；基于需求分析结果对系统进行概要设计，描述了系统整体的设计架构、物理架构、子模块划分和数据库设计。

第四章：微信模块的详细设计与实现。本章主要借助微信模块的详细类图、流程图等模块设计图、界面结果展示、关键代码说明来对微信模块的详细设计和实现进行描述。

第五章：总结与展望。本章主要总结了该项目已实现的功能和实现过程中遇到的问题和启发，探讨项目以及开发过程中的缺点与不足，最后展望该项目在功能上和适用范围上的扩展和未来的发展方向。

[4] 腾讯企鹅智库官网：http://re.qq.com

第二章 微信模块主要技术概述

2.1 Wechat4j 框架

2.1.1 应用平台——微信公众平台

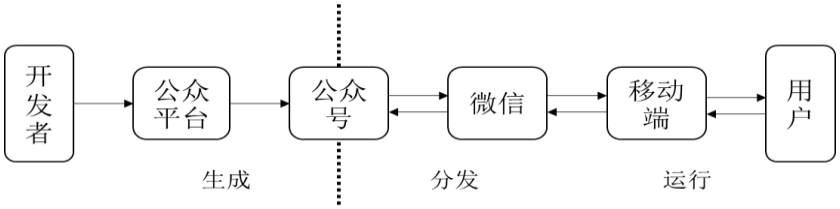

图 2.1 公共号轻应用模式图

腾讯微信公众平台是基于微信的扩展功能模块，是一种基于超级 APP（微信）的轻应用模式（如图 2.1）。开发者利用微信公众平台的开发者模式中提供的接口与自己已有系统接口及其数据库连接，实现微信公众平台和开发者已有系统的数据集成[7] [8]。

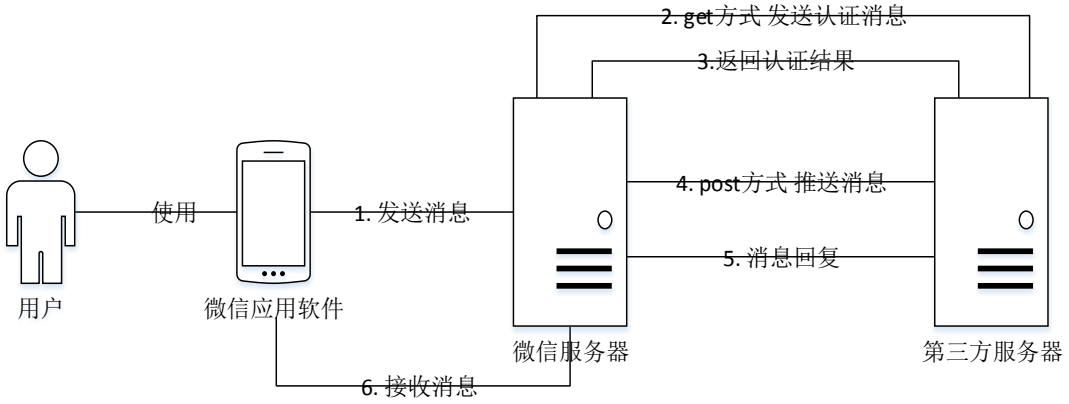

图 2.2 微信公众号的通讯机制图

微信公众平台消息接口为开发者提供了基础接口、收发消息接口、管理接口等等。公众号与用户主要的交互方式是采取主动推送和被动回复消息的形式，其中被动回复的基本通讯机制如图 2.2 所示：由用户发消息给公众号，微信公众平台服务器以 HTTP GET 形式向第三方服务器发送认证信息并获取返回的认证结果，结果正确则以 HTTP POST 形式继续向其发送用户发送的消息并获取其响应的消息，最后由微信服务器解析出回复信息并返回给用户[9]。

2.1.2 框架概述与配置

由于微信公众平台提供的开发文档只有 PHP 接口和示例，给非 PHP 开发的项目带来不便。而 Wechat4j，即 Wechat develop framework for java，是一个开源的 java 版微信公众平台开发框架，代替 JAVA 开发者封装了微信公众平台提供的 PHP 接口、XML 消息模板、access_token 操作等等，是目前最简单易用的微信公众平台的 java 开发框架。

access_token，就是与微信服务器交互过程中的一个凭证，有一定的过期时间且限制每天的获取次数。为了避免低效地重复获取相同的 access_token 和获取次数超出限制而影响提供正常服务，框架中通过 TokenProxy 来获取 access_token，而代理中就封装了 access_token 中控服务器，根据 expires_in 定义的过期时间，定时刷新 access_token，刷新后默认在内存中存储。当然，框架也提供自定义的持久化存储方式，定义在一个继承 CustomerServer 的类中，然后通过配置文件关联该类，如图 2.3。

```
……
#if no this property,then token server is default memery accesstoken server
wechat.accessToken.server.class=自定义的类（继承 CustomerServer）路径
```

图 2.3 wechat4j.properties 的中控服务器配置

2.2 智能聊天机器人技术

2.2.1 发展现状

智能聊天机器人，就是一种基于人工智能原理，通过对用户提出的问题进行分析，根据分析结果到数据库预存的数据中进行匹配，匹配失败则进行搜索，最后组织答案信息，给出应答的一类程序[10]。

随着中文分词技术和词库的不断发展，中文聊天机器人技术也日趋成熟，不仅是一个更形象友好、更符合人际交流习惯的网络搜索引擎，更是一个拥有学习能力、记忆能力且可以自动升级的人工智能产物[11]。例如：众包回复，一种学习如何处理新问题的手段，即聊天机器人将一次失败会话中无法回答的问题抛给另一个或多个用户，分析、学习并记录其反馈的答案[12]。

现今国内比较成熟的聊天机器人（如：上海赢思软件的小 i 机器人[5]，爱博的小 A 等）满足人机交互友好性方面的要求，适用于各类交互型、社交型软件。

[5] 小 i 机器人，官网：http://www.xiaoi.com/；产品体验：http://i.xiaoi.com/

2.2.2 核心思想

从学术角度看，聊天机器人的核心技术是自动问答(Question Answering)。通常我们日常需要的、也是比较难以实现的开放域自动问答系统[6]，主要有三个环节：问题分析，文档、句段（passage）检索和答案抽取、生成[13]。

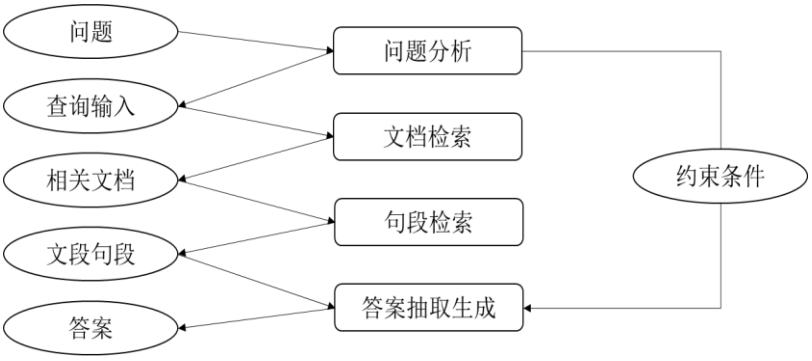

图 2.4 开放域自动问答系统一般的体系结构

如图 2.4 所示，用户向系统提出问题后，进入环节一：问题分析，用以确定问题的类别（按答案类别大致分为：事实型、列举型、定义型、交互型）以及问题关键词与答案之间的约束关系（简单共现、句法依存、浅层语义等）并生成约束条件；环节二：文档、句段检索，从文档中筛选出相关文档的相关文本块；环节三：答案抽取生成，根据环节一中得到的约束条件从相关文本块中提取最终的答案[13]。

2.2.3 应用实践——以图灵机器人平台为例

以图灵机器人平台[7]，一个免费的 API 开放平台为例：图灵机器人平台接入地址为 http://www.tuling123.com/openapi/api，是以 http get 的请求方式、json 数据格式的方式来接入，其具体请求参数[14]如下表 2.1：

表 2.1 图灵机器人接入 API 的请求参数表

参数	是否必须	说明
key	必须	开发者注册帐号激活后，即可获得
info	必须	请求内容（问题），编码方式：UTF-8
userid	上下文必须	此 userid 针对开发者自己的每一个用户

[6] 自动问答在系统功能角度可以分为：开放域、限定域。开放域：不限定问题领域，可以回答所有领域的问题；限定域：限定问题领域，其他领域问题无法回答，如：FAQ 问答系统。

[7] 图灵个性化智能机器人平台，官网：http://www.tuling123.com/openapi/cloud/home.jsp；产品体验链接：http://www.tuling123.com/openapi/cloud/proexp.jsp

loc	非必须	位置信息，编码方式：UTF-8
lon/lat	非必须	经度信息/纬度信息

其消息类型主要有六种：文字类、链接类、新闻、列车、航班、菜谱，所有消息都是以 JSON 数据形式返回，其中 code 状态码和 test 文字内容参数是所有消息公用的返回参数。

关于接入方式，图灵机器人平台为开发者提供了 PHP 和 JAVA 两种语言的示例，由于项目架设于 Java EE 平台上，所以在此介绍一下 JAVA 的平台接入方式[14]，如下图 2.5 所示：APIKEY 即请求参数中的 key，INFO 需要转成 utf-8 编码格式，获取 api 链接并读取响应内容。

```
……
// 获取链接
URL getUrl = new URL(“http://www.tuling123.com/openapi/api” + "?key=" +
APIKEY + "&info=" + INFO);
HttpURLConnection connection = (HttpURLConnection) getUrl.openConnection();
connection.connect();
// 取得输入流，并使用 Reader 读取
BufferedReader reader = new BufferedReader(new
InputStreamReader( connection.getInputStream(), "utf-8"));
……
while ((line = reader.readLine()) != null) { sb.append(line); }
// 断开连接
……
```

图 2.5 调用图灵机器人平台接口 JAVA 调用示例代码

2.3 中文分词和关键字提取技术

2.3.1 中文分词技术概述

不同于英文单词之间以空格区分，中文汉字是以字为最小单位，那么中文分词则需要将连续的字序列按照一定的规范重新形成词序列[15]。中文分词是文本挖掘的基础，也是其它中文信息处理的首要环节，使得电脑可以“理解”（自动识别）中文文本，其根本手段便是分词算法。算法大致可分为四类[16]：

a. 字符匹配（机械分词），基于字典分词，即文本与一个大词库中数据进行匹配来识别词。按文本的扫描顺序分，有正向、逆向和双向三种方式；按匹配原则分，主要有最大匹配、最小匹配、逐词匹配和最佳匹配；

b. 理解法（基于人工智能的分次法），即模拟人对文本的理解方式，在分词过程中进行句法、语义分析并处理歧义现象。以此为基础发展的分词法有：专家

系统分词法、神经网络分词法以及前两者集成分词法；

c. 统计法（无字典分词），词是相对稳定的字序组合，在文本中出现频率越高的字序组合越可能是词。因此，用统计模型（如：n-gram 模型、隐 Markov 模型、MaxEnt 模型等）得出字序组合的频率很大程度上可以作为词的可信度。适用于识别未登录词和学术性领域性较强的大文本。

d. 语义法，更为注重对语言本身所表达的意思方面进行处理，如扩充转移网络法、邻接约束法、后缀分词法、特征词库法、矩阵约束法等。

可是，即便有了许多的分词算法，中文分词依旧存在难以克服的困难：歧义处理、未登录词（实体名、新词热词等）识别、颗粒度界定。可见，中文分词技术想要更高效、更顺利地完成产品化、服务化，还有很长一段路要走。

2.3.2 分词系统——以 NLPIR/ICTCLAS 为例

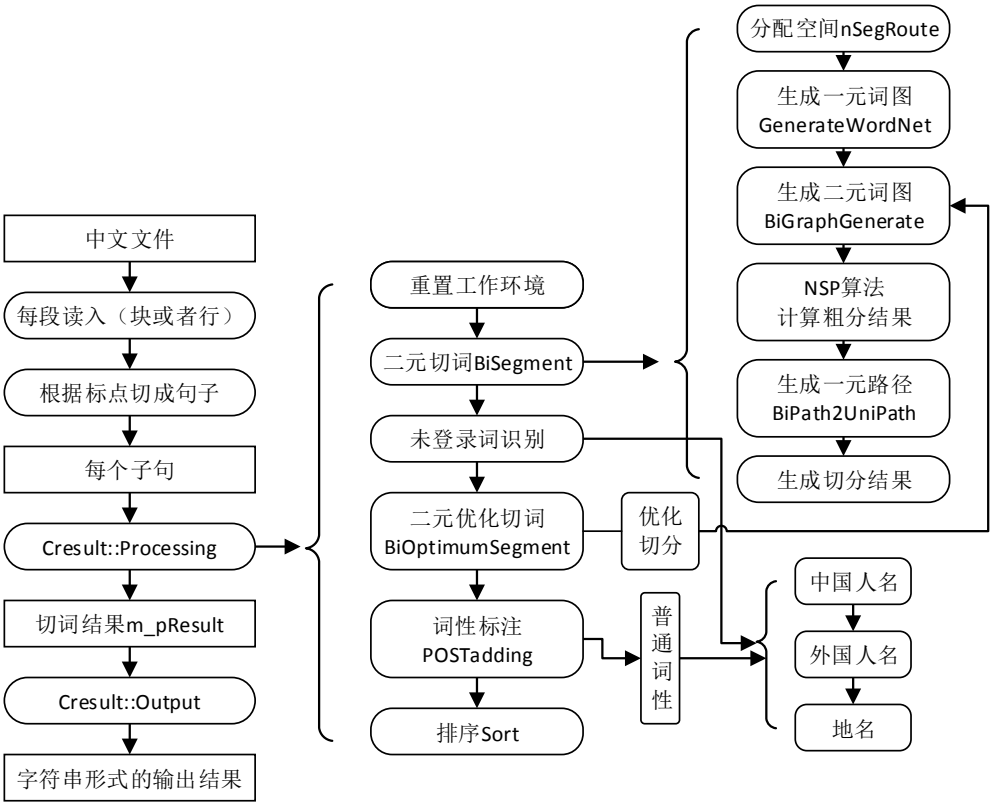

图 2.6 ICTCLAS 简单数据流程图

NLPIR/ICTCLAS[8]分词系统是由中科院计算所张华平博士倾力打造十余年研发而成，基于 CHMM 进行分词，主要功能包括中文分词、关键字提取、命名实体识别、用户词典功能等等，是当前最好的汉语词法分析器[17]。

如图 2.6 所示为一个 ICTCLAS 的简单数据流程图[18]，图中除中文文件、

[8] ICTCLAS，又名 NLPIR，官网：http://ictclas.nlpir.org/

每个句子、字符串形式的输出结果为数据以外，其余部分均表示处理的子流程或者主要函数名。总流程：读取中文文件，导入词典，按标点符号将文本切分成子句，然后对每个子句进行子句处理并获得处理结果，以字符串形式输出。子句处理流程：二元切词，未登录词识别（中国人名→外国人名→地名识别），对识别后的分词结果进行优化切词，基于 CHMM[19]对分词结果进行词性标注并排序，得到最终子句处理结果。二元切词流程：原子切分，得到独立的中文字符、数字+特殊符号串、英文字符串；一元切分，将原子切分结果与核心字典进行最大匹配；二元切分，将一元切分结果与二元词典进行最大匹配；NSP[9]算法计算粗分结果，就是将二元切分结果用基于 Dijkstra 的 N 元最短路径算法得出最短路径及其对应词表[20]。

ICTCLAS 主要是由 C++实现的，现开放接口有 C++，JAVA，Python。而 Ansj[10]则可以算作是 ICTCLAS 的 JAVA 开源版本，其基本原理一致，只是在分词的优化算法上稍作改进[21]。

2.3.3 基于中文分词的关键字提取算法

关键词提取，顾名思义，就是从文本中提取出文本最关键最核心的词。关键字提取分为关键字分配和抽取两种。其中，关键字抽取应用比较广泛，又分为单纯抽取关键词和抽取包含关键词的短语。明显后者更具有应用研究价值，也更难实现。而上文提到的 ICTCLAS 和 Ansj 自带的关键字提取方法就属于这类。

关键字抽取算法五花八门，使用外部知识库的有：TF-IDF 算法、初代 KEA 算法、使用 LDA 或者 word2vec 模型对词聚类抽取等。还有不使用外部知识库而基于中文分词的算法：TextRank（PageRank）算法、ICTCLAS 的关键词左右熵比较高的思想等[22]。

2.4 本章小结

本章重点介绍了开发微信模块所涉及的相关技术中比较陌生的几块，包括 Wechat4j 框架、智能聊天机器人技术以及中文分词和关键字提取技术的概述、核心思想原理和应用实践。

[9] N-最短路径分词算法，具体参见：http://blog.csdn.net/xlvector/article/details/1374949

[10] Ansj 中文分词的开源代码详见：https://github.com/NLPchina/ansj_seg

第三章 任务协同系统需求分析与概要设计

3.1 任务协同系统整体概述

任务协同系统旨在为公司内部短期或常驻外聘人员或者公司员工工作地点不固定、不集中的中大型企业提供任务协同管理的服务，为频繁到其他企业工作的人员提供及时与其他公司联系沟通的渠道。

因此，任务协同系统的主要目标是：提供数据导入，任务分配、跟踪，日志自动生成，资讯实时推送等功能，实现系统功能精简、无需安装客户端、注册即能用。

3.1.1 系统用户角色

本系统的最终用户包括系统管理员、任务管理员（企业中高层）、内部员工（企业基层）和外聘人员，他们的特征如下表 3.1：

表 3.1 涉众及其特征表

<table>
<tr><th colspan="2">涉众</th><th colspan="2">特征</th></tr>
<tr><td colspan="2">系统管理员</td><td colspan="2">当企业初次使用本系统时，系统管理员向系统数据库批量导入现有数据；当企业需要备份、清空等时，他们对数据库进行批量操作。他们主要使用系统 PC 端，且使用并不频繁。</td></tr>
<tr><td colspan="2">任务管理员
（企业中高层）</td><td colspan="2">当企业需要启动一个新项目时，任务管理员发起创建一个新项目，可以在项目下创建（子）任务，并为任务分配相关执行者；他们可以追踪并管理项目的变更日志；他们可以管理资讯并发布，从而将资讯推送到微信公众号。他们主要使用系统 PC 端，是 PC 端主要使用人员。</td></tr>
<tr><td rowspan="2">任务执行者</td><td>内部员工
（企业基层）</td><td>通过关注微信公众号，任务执行者可以绑定账号；查看被分配的任务、修改任务状态；当任务管理员发布资讯时，他们可以查看资讯；当被分配任务出现变更时，他们可以收到变更通知；他们可以使用公众号进行日常聊天，可以发送特定位置来获取到达方式。他们主要使用系统移动端。</td><td>内部员工可以查看系统中所有用户的基本信息；他们可以直接向移动端的其他人员发送文本消息。他们使用频率较高。</td></tr>
<tr><td>外聘人员</td><td>外聘人员可以查看参与项目涉及用户的基本信息；他们可以联系移动端的项目相关的其他人员。当他们有任务时使用频率会变高。</td></tr>
</table>

3.1.2 系统功能概述

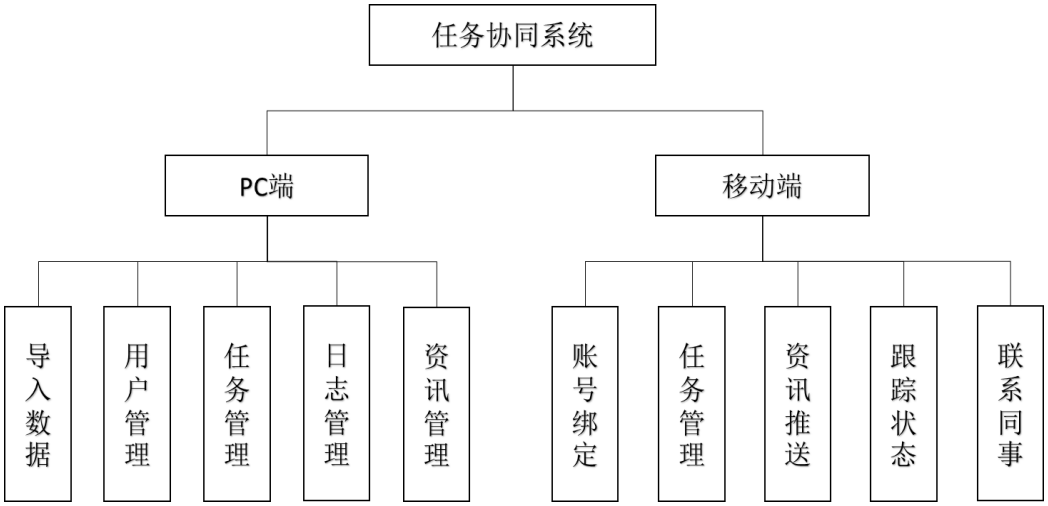

图 3.1 系统功能图

如图 3.1 所示，轻量级任务协同系统基于 SpringMVC + Hibernate 框架开发，主要分为 PC 端和移动端。

PC 端主要是面向系统管理员和公司项目管理员，有易于管理操作、可移植、无需额外搭建环境或安装客户端、无额外消耗的基本需求，所以采用 web 网页的形式，主要功能包括导入数据初始化系统、用户管理、任务管理、日志管理和资讯管理。

移动端，也就是本人主要负责的模块，主要是面向任务执行者，包括公司员工和外聘人员，要求移动端应用为轻量级、便捷易用、跨平台、可移植扩展、无需额外安装 APP 的轻应用，考虑到微信的迅猛发展、庞大的固定用户群和公众平台的良好发展势头，最终系统移动端基于微信公众平台开发，主要功能包括：账号绑定、任务查看和状态修改、任务状态跟踪、资讯定期推送以及查找和联系同事。

3.2 微信模块需求分析

3.2.1 模块需求概述

为了轻量级任务协同系统能够促使企业项目更高效、更及时、更灵活地协作完成，系统对于移动端即微信模块产生了以下需求：

- ✓ 该模块需要提供识别用户并向新用户提供绑定账号、手机的功能；
- ✓ 该模块需要提供关键字提取识别的功能；

- ✓ 该模块需要提供命令匹配功能，如：@XXX <content>，则是发送消息 content 给 XXX；@XXX，则是查询 XXX 基础信息等。
- ✓ 该模块需要分析用户发送的消息并做出针对性处理。如果发送任务相关信息，公众号要回复查看和修改自己任务的链接；如果发送与资讯相关的信息，则需回复最新的资讯列表；如果发送帮助相关信息，则需回复使用说明；如果发送位置信息，则需回复到达方式；如果发送公司相关信息，则需回复公司简介；如果发送命令，则需匹配命令格式执行相关操作并反馈操作结果；如发送日常对话，则回复智能聊天机器人的反馈结果。
- ✓ 该模块需要提供群发消息和向特定用户发送消息的功能，群发接口需要暴露给 PC 端；
- ✓ 该模块需要识别用户类型进行权限管理，区分企业内部员工还是企业外聘人员，后者只能查看系统中项目相关人员的基本信息；
- ✓ 该模块需要尽量实现易用、高效、安全、可移植、可扩展、可维护、跨平台、低消耗的性能质量要求。

该模块涉及的主要用户角色为：内部员工（企业基层）和外聘人员，其各自特征已经在 3.1.1 系统用户角色中进行详细说明。

3.2.2 模块的功能性需求分析

✓ 系统特性 1

特性描述

系统可以提供任务执行者即内部员工（企业基层）和外聘人员绑定用户账号和手机号码的功能。

相关功能需求

表 3.2 绑定账号需求表

需求编号	需求内容	优先级
R1	系统可以根据 openid[11]识别是否是新用户； 系统为新用户提供绑定账号的登陆界面。	10

任务执行者首次关注系统公众号，系统将会发送一个绑定链接，用户点击链接即可跳转到绑定界面，输入手机号和密码即可。如果用户未在首次关注时点击链接绑定，那么当未绑定用户进行需要绑定用户权限的操作（如：查看自己的任

[11] openid 是关注者和公众号按一定算法生成的，也就是说对于同一个公众号来说 openid 是用户的唯一标示符，通过 openid 可以获取用户的基本信息。

务列表、修改任务进度等）时，系统会自动跳转至绑定界面。

✓ 系统特性 2

特性描述

系统可以匹配命令并进入不同的处理流程。

系统可以提供内部员工查看所有人员基本信息的权限，提供外聘人员查看自己参与过项目的相关人员基本信息的权限，并且系统可以识别用户身份从而确定其权限。

系统可以提供任务执行者向其他任务执行者直接发送文本消息。

相关功能需求

表 3.3 命令匹配执行需求表

需求编号	需求内容	优先级
R2	如果命令格式为@XXX，则查询 XXX 基础信息并回复，若查无此人则回复“没有此人”，若该用户没有权限则回复“没有权限查看”。	3
R3	如果命令格式为@ XXX <content>，则发送<content>中的内容给 XXX 并回复“消息已发送”，若查无此人则回复“没有此人”，若发送失败则发送失败原因。	3

系统需要识别出用户发送的消息否为命令、匹配命令格式从而进入不同的处理流程，R2 是延伸需求，所以优先级不高。

✓ 系统特性 3

特性描述

系统可以提取、识别关键字。

相关功能需求

表 3.4 提取识别关键字需求表

需求编号	需求内容	优先级
R4	系统可以识别出给定的关键字	10
R5	系统可以从大文本中提取关键字； 在原文本中标记出提取的关键字。	1

系统需要分析识别用户发送消息中的关键字或其同义词。在查看资讯内容时，在文章中标注出全篇的关键字，方便用户更好地了解资讯的主旨，R5 是 R7 的延伸需求，并不是主体需求，所以优先级较低。

✓ 系统特性 4

特性描述

系统可以根据用户发送消息识别出的不同关键字做出不同处理。

相关功能需求

表 3.5 识别消息并回复需求表

需求编号	需求内容	优先级
R6	如果文本消息关键字识别为“任务”，系统按照用户 ID 生成其任务列表的链接并回复； 用户点击链接查看任务列表，点击具体任务可以查看任务具体信息并可以修改任务完成状态。	10
R7	如果文本消息关键字识别为“资讯”，系统回复最新的 10 条资讯列表； 用户点击具体咨询可以查看资讯内容。	10
R8	如果文本消息关键字识别为“帮助”，系统回复使用说明。	10
R9	如果文本消息关键字识别为“公司”，系统回复公司基本信息。	5
R10	如果文本消息关键字识别为“命令”，系统匹配命令格式并进入不同处理流程。	5
R11	如果文本消息关键字未识别，系统调用聊天机器人 API 并回复反馈的信息。	3
R12	如果消息为位置消息，系统则调用图灵平台 API 并回复如何到达这个位置。	3

系统根据识别出的消息类型或文本消息关键字从而进入不同的处理流程。由于 R8~R11 是为了达到公众号更加人性化、更加实用的目的而扩充的功能需求，所以它们的优先级比较低。

✓ 系统特性 5

特性描述

系统可以提供群发消息接口，并且接口需要提供给 PC 端的任务管理员的资讯管理模块。

相关功能需求

表 3.6 群发消息需求表

需求编号	需求内容	优先级
R13	系统可以获取所有关注者列表并向其推送资讯列表。	10

系统可以获取所有关注者列表并向其推送资讯列表，暴露给任务管理员的资

讯管理模块中的发布资讯功能。

✓ 用例建模

上文中列出了微信模块的系统特性及其相关需求，也提到模块的最终用户为内部员工和外聘人员两种用户角色，并且模块还需要向任务管理员提供一个群发接口。

内部员工和外聘人员除了查看用户信息的权限不同，其余部分相同。内部员工可以查看所有用户的基本信息，而外聘人员只能查看权限内，也就是自己参与项目内人员的基本信息。结合上述需求分析建模，最终得到模块的用例列表，详见表 3.7。其中，UC2~4,10,15 需要验证用户身份，如果用户未绑定账号则会无法完成相关操作，系统会返回“没有权限”或者点击系统回复的链接会直接跳转至绑定账号界面。

表 3.7 用例列表

参与者	**用例**	
	用例编号	**名称**
内部员工	UC1	绑定账号
	UC2	获取任务列表
	UC3	查看任务内容
	UC4	修改任务状态
	UC5	接收资讯推送
	UC6	获取最新资讯
	UC7	查看资讯内容
	UC8	关键字提取和标记
	UC9	关键字识别
	UC10	获取用户信息
	UC11	查看帮助信息
	UC12	查看公司信息
	UC13	日常对话
	UC14	查看地理位置的到达方式
	UC15	向其他用户发送信息
	UC16	获取所有关注者列表
外聘人员	UC1~9,11~16	相同
	UC10	获取权限内的用户信息

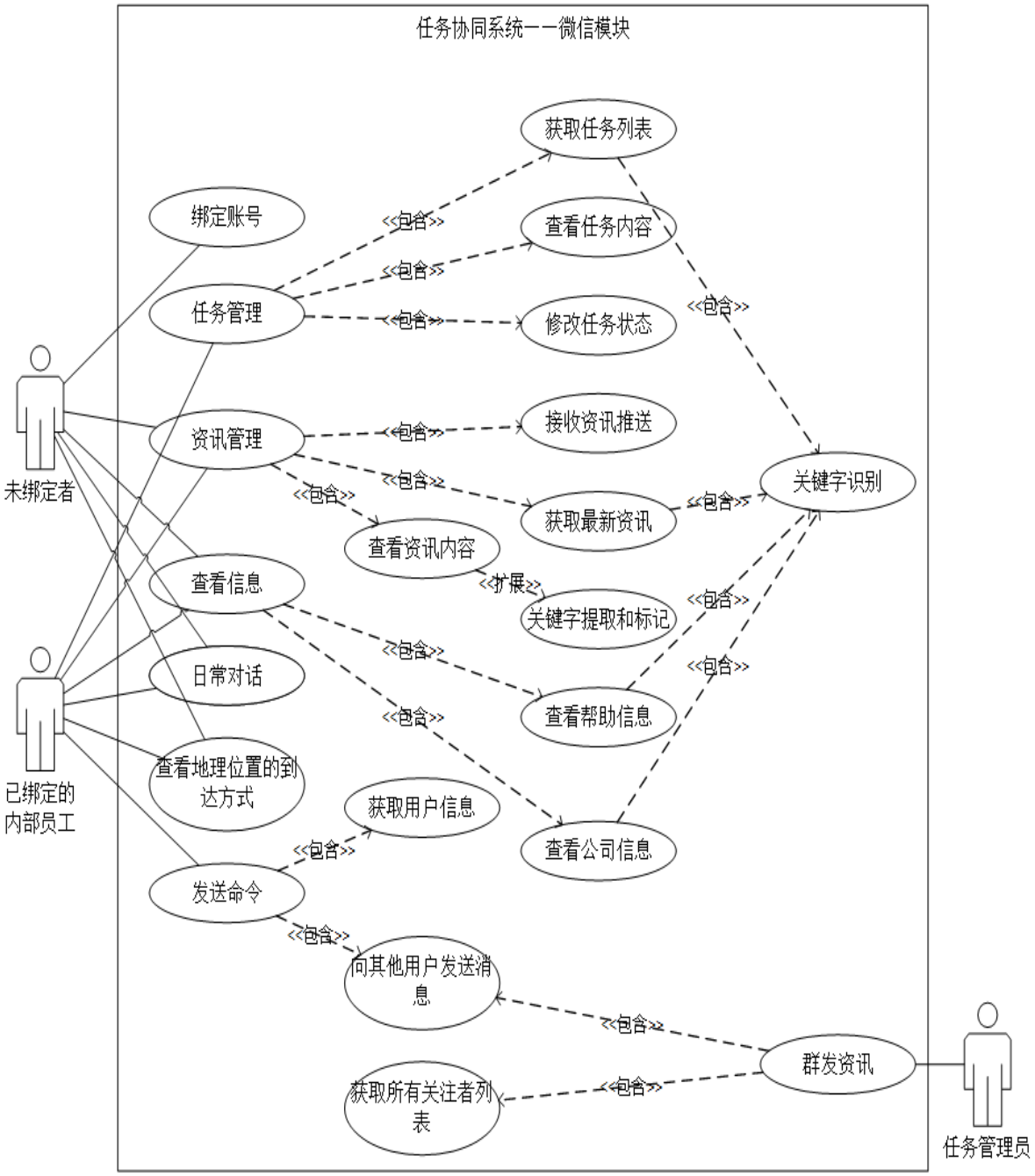

图 3.2 内部员工用例图

结合表 3.7 用例列表，以及对未绑定用户和已绑定内部员工的进一步功能需求分析，我们确定了未绑定用户和已绑定内部员工的权限差别，再加上模块需要对外开放接口，最终获得的内部员工用例图如上图 3.2 所示。

由于外聘人员的功能需求与内部员工仅在 UC10 上有权限方面的差别，所以外聘人员的用例图就是将图 3.2 中的“将获取用户信息”改为“获取权限内的用户信息”，所以在这里就不再展示了。

3.2.3 模块的非功能性需求分析

需求最常见的分类：功能需求、性能需求、质量属性、对外接口、约束[23]。其中，功能需求表现为系统的功能特性，为了达成业务目标而产生的需求，本模块的功能需求已经在上文中详细讨论了。其余四个需求统称为非功能需求，是隐性的系统需求，很容易被忽略却可能会决定系统最终的成败。

根据 3.2.1 中提到易用、高效、安全、可移植、可扩展、可维护、跨平台、低消耗的要求，非功能性需求分析如下：

✓ 性能需求

速度，即系统完成任务的时间[24]。现今社会快节奏的生活使得用户越来越没有耐心，对于软件的响应速度要求越来越高，所以要求系统 90%的操作响应时间不超过 5 秒。

容量，即系统所能储存的数据量[24]。作为目标用户为企业的系统，数据存储量预计会比较大，系统应该可以存储至少 20 万条任务记录。

负载，即系统可以承载的并发工作量[24]。因为目标用户为企业工作人员和外聘人员，用户并发现象比较常见，所以要求系统在 500 个用户并发时不能崩溃。

✓ 质量属性

功能性包括精确性、依从性、互操作性、安全性、合适性。因为模块架设在微信公众平台上，所以要求系统与微信有良好的交互。又因为系统的目标客户为企业，所以系统在安全和权限限制方面需要引起重视。

可靠性包括成熟性、容错性、可恢复性、依从性。由于系统与用户交互性较强，所以对于用户的非法输入或非法操作的防御性非常重要。系统架设在 SAE 服务器上，移动端要依靠微信公众平台，服务的稳定性难移保证，所以系统会缓存操作和数据以防服务出现异常而带来不必要的损失。

易用性包括可理解性、可学习性、可操作性、吸引性、依从性。系统的主要用户定位属于企业员工，所以系统必须让新用户容易上手，操作简单就能满足用户的需求。

可维护性包括可分析性、可改变性、稳定性、可测试性、依从性。由于系统最终需要给企业自己运作，企业需要个性化定制，所以要求修改系统和测试变更的代价尽量低。

可移植性适应性、可安装性、共存性、可替换性、依从性。系统的用户使用环境并不统一，所以为了用户良好的体验，要求系统能适应不同的操作系统和不同的浏览器。

✓ 非功能需求总结

基于以上分析，可以总结出系统的非功能性需求，具体可见下表 3.8 所述。

表 3.8 非功能需求列表

性能需求	PR1：系统 90%的操作响应时间不超过 5 秒。
	PR2：系统应该可以存储至少 20 万条任务记录。
	PR3：系统在 500 个用户并发时不能崩溃。
质量属性	QA1：系统与微信有良好的交互。
	QA2：系统对于每一次需要权限的操作都会进行身份验证。
	QA3：系统对于用户非法输入或操作给予提示，不妨碍系统正常运作。
	QA4：服务器出现异常后重连，系统可以继续正常使用。
	QA5：系统界面友好简约，操作简单易用。
	QA6：修改系统和测试变更的代价尽量低。
	QA7：系统能适应不同的操作系统和不同的浏览器。

3.3 微信模块概要设计

3.3.1 总体架构设计

为了系统日后维护和进一步扩展，也为了系统的层次更加分明，更为符合高内聚低耦合的设计理念，系统需要进行层次划分、模块划分和进一步的子模块划分，这样也更好的满足了系统的非功能需求。

系统总体遵循 MVC（Model-View-Controller）的架构思想，采用 SpringMVC + Hibernate 的框架，降低了系统耦合度，方便小组并行开发，将前端页面显示分离，以隔离界面风格不断变化带来的系统变更。

如图 3.3 所示的系统总体架构图，任务协同系统共分为五个层次，包括 View 界面层、Controller 控制层、Service 业务逻辑层、DAO 数据访问层和 Entity 实体层。与用户进行交互的是 View 层，View 层再将用户输入和相关数据流传递给 Controller 层，Controller 层接收 View 层数据流并进行集中处理，调用 Service 层提供的业务逻辑接口，所以 Controller 层的耦合度最高，这样的设计是为了降低其他层次的耦合度。DAO 层封装了对 MySQL 数据库的直接操作，利用 Hibernate 框架优化数据库操作。Service 层则封装了对 DAO 层直接操作接口组合出的复杂的业务逻辑处理。Entity 层实体的创建和调用都是通过 Spring 框架来实现的。

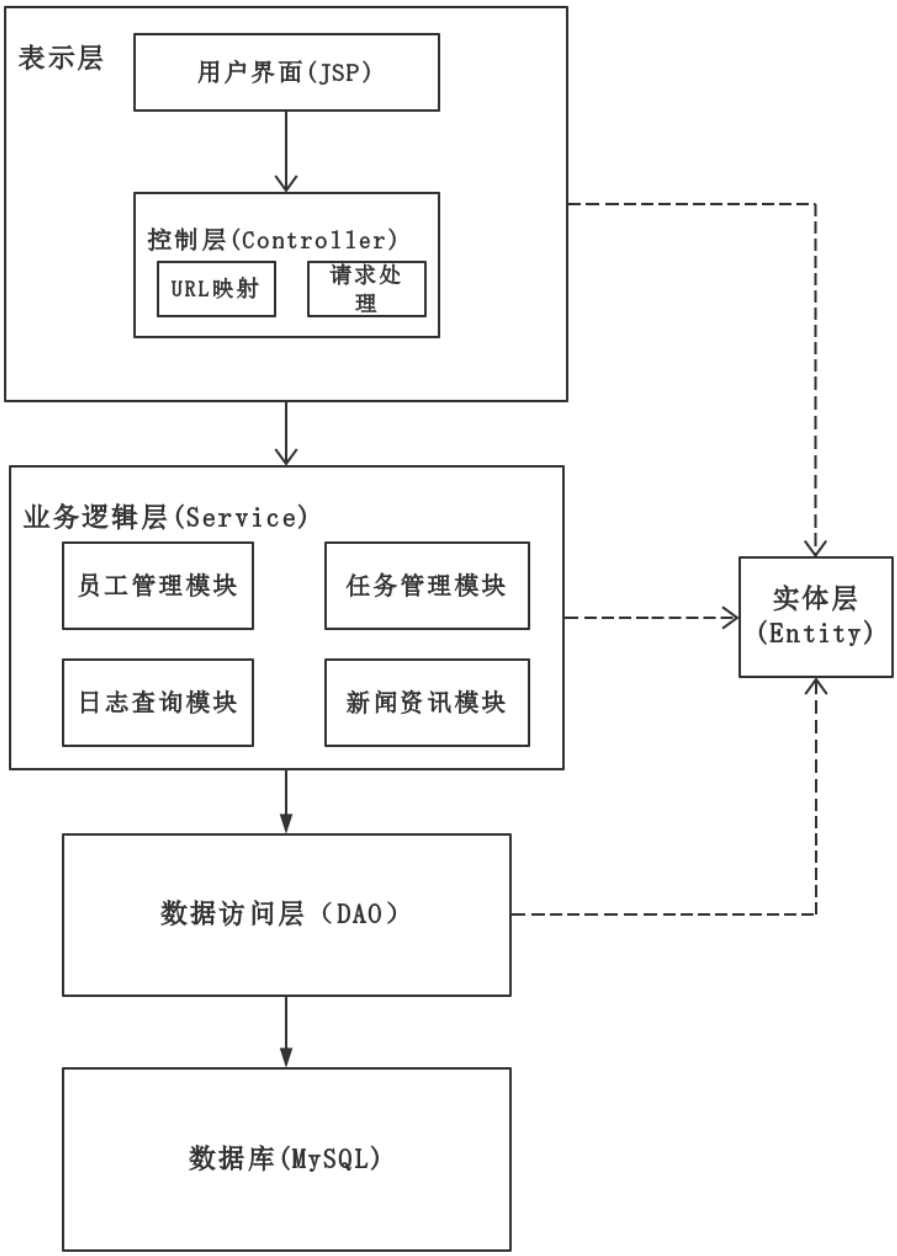

图 3.3 系统总体架构图

3.3.2 模块结构设计

考虑到系统的非功能需求，要求系统可以跨平台兼容、可维护性、易变更性等，又因为微信模块需要基于微信公众平台开发，所以模块在 SpringMVC 的分层框架之上，又添加一个微信相关的接口部分，用来封装模块与微信公众平台之

间的交互操作。

图 3.4 微信模块总体设计构图

如图 3.4 所示的微信模块总体设计图，微信模块主要共分为五层，分别是 View 界面层、Controller 控制层、Service 业务逻辑层、DAO 数据访问层和 Entity 实体层。模块还将外部服务接口的调用单独封装在微信相关接口部分。微信模块的 View 层主要由 JSP 页面构成，主要包括绑定界面、任务界面和资讯界面，以公众号向用户发页面链接的形式展现。实体层也就是数据持久化层，包括内部员工、项目、任务、资讯、公司等实体，需要实现序列化，且需要配置各个实体相应的 XXX.hbm.xml 文件来确保实体属性与数据库字段一一对应。数据访问层用于封装对数据库的直接操作，模块主要涉及查询和更新操作。业务逻辑层中，微信模块主要涉及到用户管理、任务管理、资讯管理三部分。控制层中主要处理界

面层和业务逻辑的数据交互，需要调用微信相关接口。微信相关接口部分封装了外部服务的接口和基于外部服务的框架接口，其中微信公众平台相关的服务封装在 Wechat4j 框架中，ICTCLAS 中文分词相关的服务封装在 Ansj 框架中，图灵智能聊天机器人平台相关的服务则直接调用官方提供的 API。以上涉及的外部服务和技术详细描述请参见上文中第二章微信模块主要技术概述。该部分主要包括接收消息、发送消息、获取关注者列表、关键词、Access_token 中控服务器，主要负责通过微信公众号与用户交互，通过文本信息、图文消息交互。

3.3.3 物理架构图

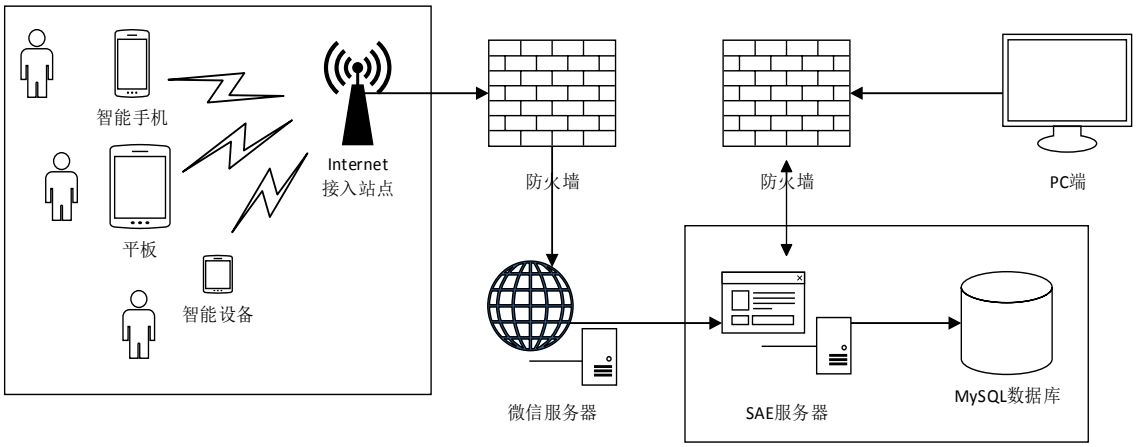

图 3.5 物理架构图

如图 3.5 所示的物理架构图，用户通过移动设备，如：智能手机、平板电脑等智能设备接入 Internet 网络，通过防火墙后接入微信服务器，通过微信公众平台转入程序接口，即 SAE 服务器，通过 SAE 提供的 MySQL 服务与数据库交互。PC 端通过 Web 浏览器访问服务器上的程序接口。

3.3.4 子模块划分

根据 3.1 中项目概述和 3.2 中的需求分析，可以按照使用环境的不同将系统划分为 PC 模块和微信模块两个模块，再按照模块的功能进行模块下的子模块划分。

如图 3.5 所示，微信模块可以按主要功能大致分为：绑定子模块，用来绑定和解绑用户账号和手机号；任务子模块，用于获取任务列表、查看任务信息、更改任务状态；资讯子模块，用于获取最新资讯、查看资讯内容；消息子模块，用于查看帮助信息、公司等信息；联系子模块，用于查看同事信息、发消息给单独的用户；聊天子模块，用于进行日常对话、充当生活小助手等。

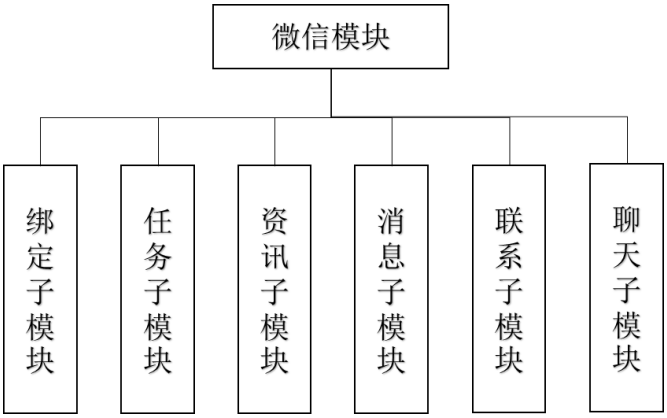

图 3.6 模块功能划分图

3.3.5 数据库设计

系统数据库基于 Hibernate 框架，采用 SAE 服务器提供的 MySQL 的关系型数据库。根据需求分析和建模，可以得出本模块涉及到的实体对象之间的关系如下：

- ✓ 每个公司可以拥有多个内部员工、多个项目、多条资讯，而每个员工、每个项目、每条资讯只能隶属于一个公司，所以公司和内部员工、项目、资讯之间属于一对多的关系；
- ✓ 一个公司可以聘用多名外部员工，而外部员工可以受聘于多家公司，所以公司和外聘人员属于多对多的关系；
- ✓ 一个项目下可以有多个任务，而一个任务只能包含于一个项目，所以项目和任务是多对多关系；
- ✓ 任务可以分配给多名内部或外部员工，而内部或外部员工可以被分配多项任务，所以任务和内部或外部员工属于多对多关系。

由以上分析可得出，本模块主要涉及到的实体类，包括 Company、Member、OutEmployee、Project、Task、News 和关系实体。所以涉及到对应的数据库表，包括 company 公司表、member 内部员工表、out_employee 外聘人员表、project 项目表、task 任务表、news 资讯表和各种关系表。其中，Task 表的数据结构为树形。所有表中字段类型为 varchar 的字段都采用 utf8-general-ci 的排序规则。具体表结构详细信息参见表 3.9~表 3.15。

由上述实体关系分析可以得到微信模块的 ER 实体关系图，具体如下图 3.7 所示：

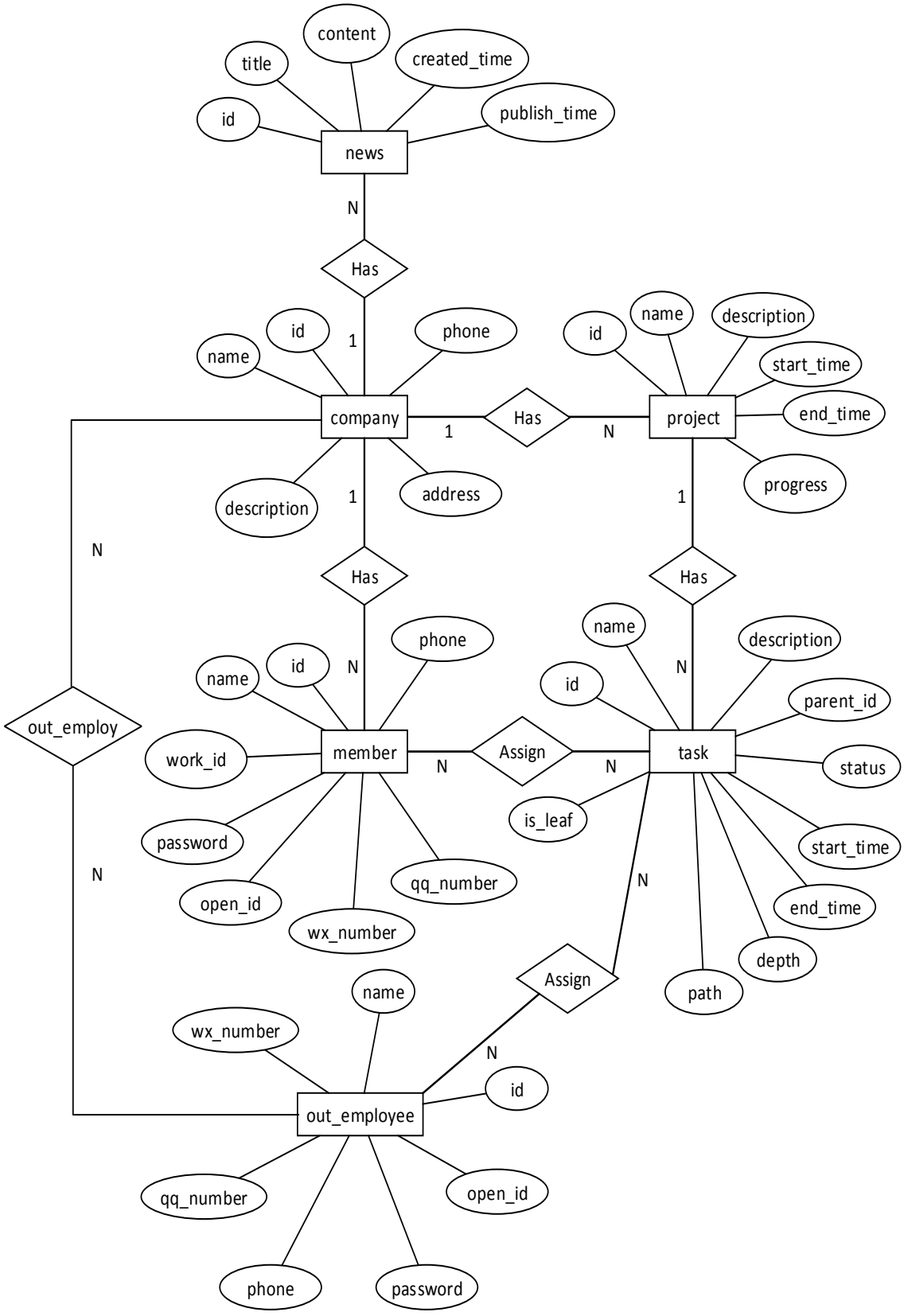
content
title
created_time
id
publish_time
news
N
Has
1
id
phone
name
company
1
Has
N
id
name
description
start_time
project
end_time
progress
description
1
address
1
N
Has
Has
phone
name
id
N
N
name
description
id
parent_id
out_employ
work_id
member
N
Assign
N
task
status
is_leaf
password
start_time
N
qq_number
N
open_id
wx_number
end_time
depth
Assign
path
name
wx_number
N
id
out_employee
qq_number
open_id
phone
password

图 3.7 实体关系图

表 3.9 company 公司表结构定义

字段名称	类型	长度	允 许为空	默认值	说明
id	int	11	否	无	公司 ID，主键，AUTO_INVREMENT
name	varchar	100	否	无	公司名称
description	varchar	500	否	无	公司简介
phone	varchar	20	否	无	公司联系电话
address	varchar	200	否	无	公司地址

表 3.10 member 内部员工表结构定义

字段名称	类型	长度	允 许为空	默认值	备注说明
id	int	11	否	无	员工 ID，主键，AUTO_INVREMENT
name	varchar	100	否	无	员工姓名
company_id	int	11	否	无	公司 ID，外键，对应 company 公司表中的 id 字段
work_id	varchar	200	否	无	员工工号
password	varchar	500	否	无	员工密码
wx_number	varchar	100	是	NULL	员工微信号
qq_number	varchar	100	是	NULL	员工 qq 号
phone	varchar	50	是	NULL	员工电话号码
open_id	varchar	200	是	NULL	员工微信号对本系统公众号的唯一标识

表 3.11 out_employee 外聘人员表结构定义

字段名称	类型	长度	允 许为空	默认值	备注说明
id	int	11	否	无	外聘人员 ID，主键，AUTO_INVREMENT
name	varchar	100	否	无	外聘人员姓名
wx_number	varchar	100	是	NULL	外聘人员微信号
qq_number	varchar	100	是	NULL	外聘人员 qq 号

phone	varchar	50	是	NULL	外聘人员电话号码
password	varchar	500	是	NULL	外聘人员密码
open_id	varchar	200	是	NULL	外聘人员微信号对本系统公众号的唯一标识

表 3.12 company_out_employee 关系表结构定义

字段名称	类型	长度	允 许为空	默 认值	备注说明
id	int	11	否	无	主键，AUTO_INVREMENT
company_id	int	11	否	无	公司 ID，外键，对应 company 公司表中的 id 字段
out_employee_id	int	11	否	无	外聘人员 ID，外键，对应 out_employee 外聘人员表中的 id 字段

表 3.13 Project 结构定义

字段名称	类型	长度	允 许为空	默认值	备注说明
id	int	11	否	无	项目 ID，主键，AUTO_INVREMENT
company_id	int	11	否	无	公司 ID，外键连接到 company 公司表
name	varchar	200	否	无	项目名称
description	varchar	500	否	无	项目简介
start_time	timestamp		否	CURRENT_TIMESTAMP	项目开始时间，默认为当前时间，ON UPDATE CURRENT_TIMESTAMP
end_time	timestamp		否	0000-00-00 00:00:00	项目结束时间
progress	double		否	0	项目进度，根据任务节点的进度计算而得

表 3.14 task 任务表结构定义

字段名称	类型	长度	允 许	默认值	备注说明

			为空		
id	int	11	否	无	主键，AUTO_INVREMENT
project_id	int	11	否	无	项目 ID，外键连接到 project 项目表
name	varchar	200	否	无	任务名称
description	varchar	500	否	无	任务简介
parent_id	int	11	否	0	子任务的父节点 ID，默认父节点为 0，即为直接在项目下的任务
status_id	int	11	否	1	任务完成状态 ID
depth				1	任务处于任务树的层数，默认在 1 层，即为直接在项目下的任务
start_time	timestamp		否	CURRENT_TIMESTAMP	任务开始时间，默认为当前时间，ON UPDATE CURRENT_TIMESTAMP
end_time	timestamp		否	0000-00-00 00:00:00	任务结束时间
path	varchar	255	是	NULL	记录子树添加路径，用于提高删除子任务树的效率
is_leaf	tinyint	4	否	1	是否是叶节点，当任务是叶节点才可以分配该任务给他人

表 3.15 task_assign 关系表结构定义

字段名称	类型	长度	允许为空	默认值	备注说明
id	int	11	否	无	主键，AUTO_INVREMENT
task_id	int	11	否	无	任务 ID，外键，对应 task 任务表中的 id 字段
member_id	int	11	是	NULL	内部员工 ID，外键，对应 member 内部员工表中的 id 字段
out_employee_id	int	11	是	NULL	外聘人员 ID，外键，对应 out_employee 外聘人员表中的 id 字段

表 3.15 task_status 任务状态表结构定义

字段名称	类型	长度	允许为空	默认值	备注说明
id	int	11	否	无	主键，AUTO_INVREMENT
name	varchar	200	否	无	任务状态名称

3.4 本章小结

本章主要对系统总体和微信模块进行了需求分析与概要设计：

第一部分：需求分析。首先，明确系统总体目标，通过涉众分析确定了系统的用户角色及其特征，并进行了功能概述。其次，依照软件工程需求分析的步骤，根据功能概述得到系统特性，通过用例建模分析出功能性需求。最后根据系统的客户和市场定位得出非功能性需求。

第二部分：概要设计。根据需求分析得到的需求，按照自上而下逐步求精的设计原则进行概要设计。首先，通过总体架构图、模块结构图和物理架构图概述了系统模块的高层设计和层次划分。其次，对系统按照功能进行模块划分和子模块划分。最后，分析了实体之间的关系并依此建立了系统的关系型数据库。

本章为下一章系统的详细设计与实现打下夯实的基础。

第四章 任务协同系统微信模块的详细设计与实现

4.1 微信模块概述

微信模块负责实现系统移动端的功能，包括绑定用户账号和手机号，获取任务列表、查看任务信息、更改任务状态，获取最新资讯、查看资讯内容，查看帮助信息、查看公司信息，查看同事信息、发消息给单独的用户，查看同事信息、发消息给单独的用户，进行日常对话、充当生活小助手等。微信模块的有些功能需要外部服务接口的支持，比如封装在 Wechat4j 中的微信公众平台 API，基于 ICTCLAS 的 Ansj，图灵机器人平台 API。

4.2 微信模块的详细设计

根据 3.3 中的系统总体设计以及微信模块的模块结构设计，模块划分为表示层（界面层+控制层）、业务逻辑层、数据访问层、实体层。其中，涉及界面的功能有绑定账号、任务管理、资讯管理，所以在控制层有相应的 controller，controller 集中调用业务逻辑层中提供的业务逻辑接口。模块中的 Service 层主要涉及对任务、资讯、内部员工、外聘人员、公司等 DAO 提供接口的调用，从而实现较为复杂的业务逻辑。实体层主要包括任务、资讯、公司、外聘、内部员工和它们之间的关系实体，与数据库表字段一一对应，方便数据库操作和不同层之间的调用依赖。除此之外，还有微信服务相关接口以及通用工具类来封装常用操作和外部接口服务。微信相关服务接口负责调用微信公众平台提供的 API，完成通过公众号与用户交互的操作，包括响应用户关注公众号事件、获取并回复用户发送的消息、向用户主动推送资讯和通知等。通用工具类提供常用的工具类和常量类，其中还封装了调用外部分词服务的关键字处理操作。

基于分层式、模块化的设计思想和自下向上的设计原则，在详细设计与实现时，按照实体层、数据访问层、业务逻辑层、控制层、界面层的顺序进行，并在其过程中抽取出常量、经常重复、相对独立的、易变的逻辑部分单独封装，放置在通用工具类中，涉及微信公众平台交互的逻辑放置在微信处理类中。

在微信模块中，流程比较复杂且功能也比较重要的部分包括公众号消息交互、账号绑定、任务管理和资讯管理，将在 4.2.1-4.2.4 给出业务流程图、详细类图等详细设计，其中根据以上对层级结构和各部分主要职责的分析讨论得到以下详

细设计中的详细类图大致架构下图 4.1 所示：

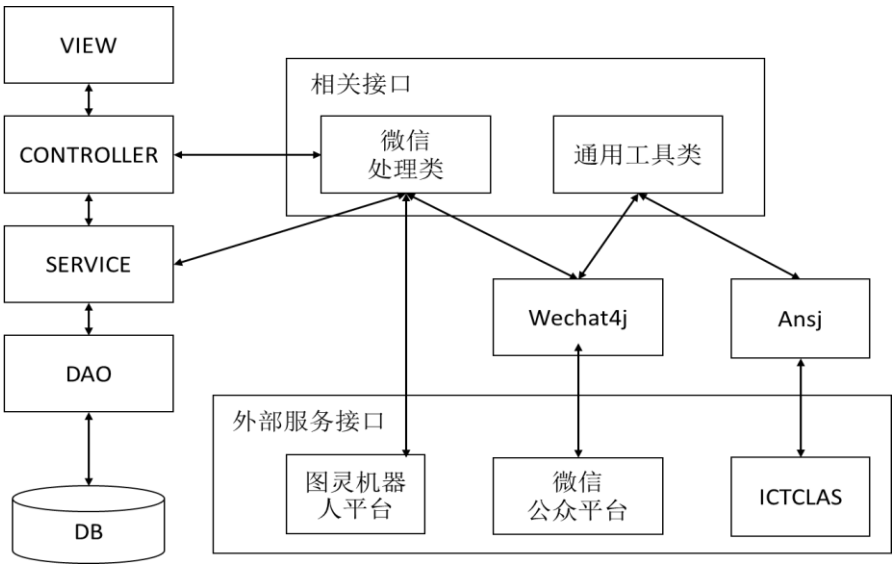

图 4.1 详细设计架构图

4.2.1 公众号消息交互的设计

用户与公众号通过消息传递的方式交互是微信模块的主线功能，串联了不同的功能子模块，实现更为友好的用户体验。如图 4.2 所示的消息交互流程，微信公众平台通过服务器向微信模块程序接口转发接收到的用户消息，模块接收到用户信息后进行处理解析、转入不同功能子模块并格式化获取的返回参数，通过微信服务器转发回复用户。

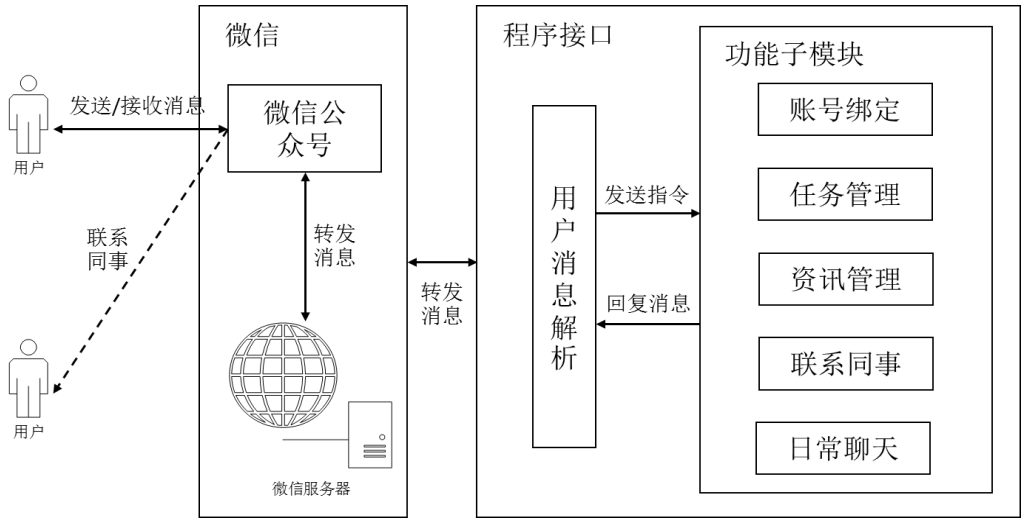

图 4.2 用户与公众号交互模式图

由于微信公众平台只提供了消息传递的形式直接交互，所以对于用户消息进行分析处理是很重要的一个步骤，可以确定程序进入的流程分支以及最终回复用

户的信息。

用户向微信公众号发送消息，系统接收到微信 web 服务器转发的用户消息后触发交互流程。其具体处理流程如下图 4.3 所示：

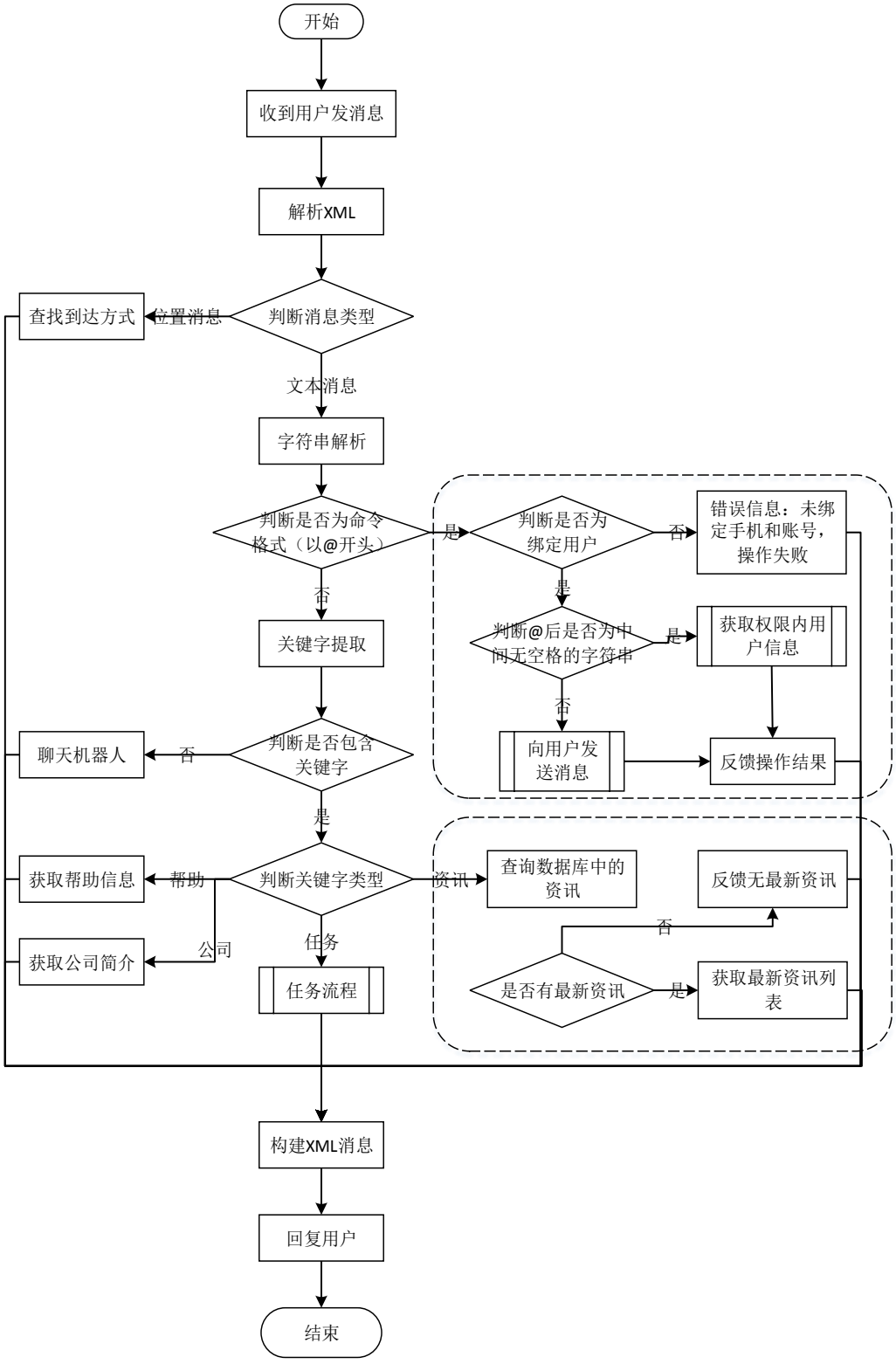

图 4.3 处理用户消息流程

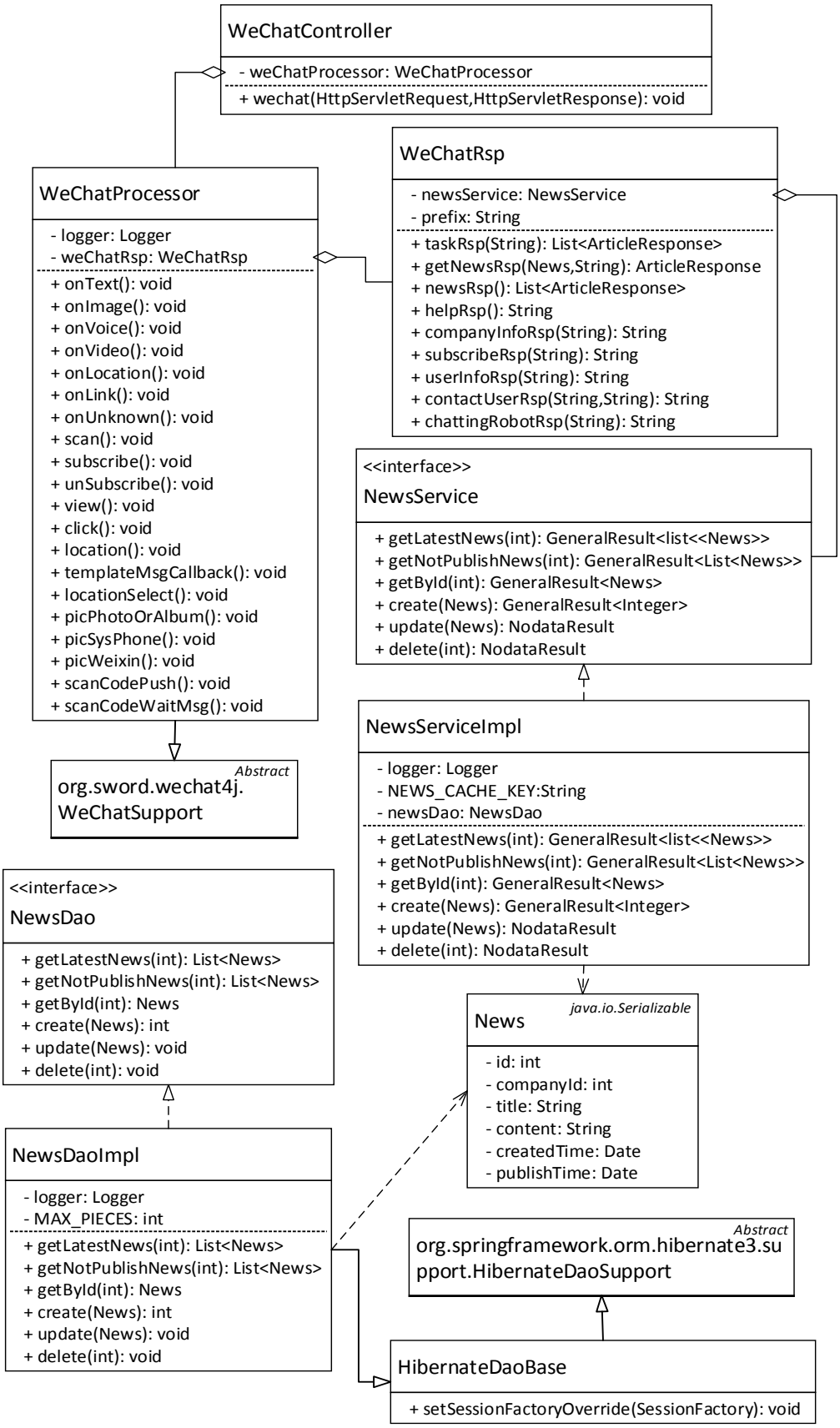

图 4.4 处理用户消息详细类图

系统解析接收到的 XML 形式的用户信息，获取 MsgType[12]，模块暂时可以处理的类型为文本消息和位置消息，其他类型的消息会在下一版本中给出处理方案。如果是位置类型的消息则回复到达方式，如果是文本类型消息则进行字符串解析。如果字符串为“@”开头则表明后面的信息为一条命令，需要进入命令处理流程（即第一个虚线框中的步骤）。由于只有绑定过的用户才有权限使用命令功能，所以要先进行身份验证，验证通过后进行命令匹配，否则发送错误信息。现有的命令格式有两种：“@”后格式为“XXX”则执行获取 XXX 用户信息流程，“@”后为“XXX YYY”则执行向 XXX 用户发 YYY 消息流程，操作完成后返回操作结果。如果不是命令格式，则进行关键字比对匹配，按不同关键字进入不同流程并返回结果。其中，绑定账号、任务管理、资讯管理流程在下文中会进行详细描述。若果关键字比对没有匹配项，则进入聊天机器人流程，返回图灵机器人 API 返回的参数。流程最后，根据各个返回信息构建符合平台约定格式的 XML 形式消息并回复给用户。

图 4.4 是处理用户消息的详细类图，如图所示：控制层为 WeChatController 类，聚合了继承 Wechat4j 包中 WeChatSupport 抽象类上文 WeChatProcessor 类。WeChatProcessor 类实现了对用户触发事件的响应处理的实现。其中从该类中抽取出 WeChatRsp 类负责构建各类事件流程处理的回复消息，隔离了复杂的、易变的逻辑。由于流程涉及资讯列表回复，所以详细类图中还关联了资讯相关的业务逻辑、数据访问接口和实现类。

4.2.2 绑定账号的设计

模块提供的有些功能（具体功能详见 3.2.2 需求分析中的用例列表）需要验证身份，确定有使用权限方可继续，但是微信公众平台现在已经不提供关注者的微信号列表了，公众号开发者只能获取 openid。Openid 是通过一定算法处理而得的类似于乱码的字符串，关注者自己都不知道自己对这个公众号的 openid 是什么，所以系统只能采用初次使用时，绑定用户的手机号码和 openid 的方式来关联公司导入的用户数据。

在用户关注该公众号的时候，系统会提供绑定账号的链接，用户只需点击链接就可进入绑定界面，输入手机号码和密码并提交，如果检验传递的参数齐全有效，则访问系统数据库并查找是否存在有这样手机号和密码的用户，如果存在则将其 openid 更新为传递过来的 openid 参数并跳转至个人任务界面。如果出现参数缺失或任何错误异常，则跳转错误界面或者返回错误信息。绑定账号的具体流

[12] 消息类型，详见微信公众平台开发者文档，

http://mp.weixin.qq.com/wiki/10/79502792eef98d6e0c6e1739da387346.html

程图如下图 4.5 所示：

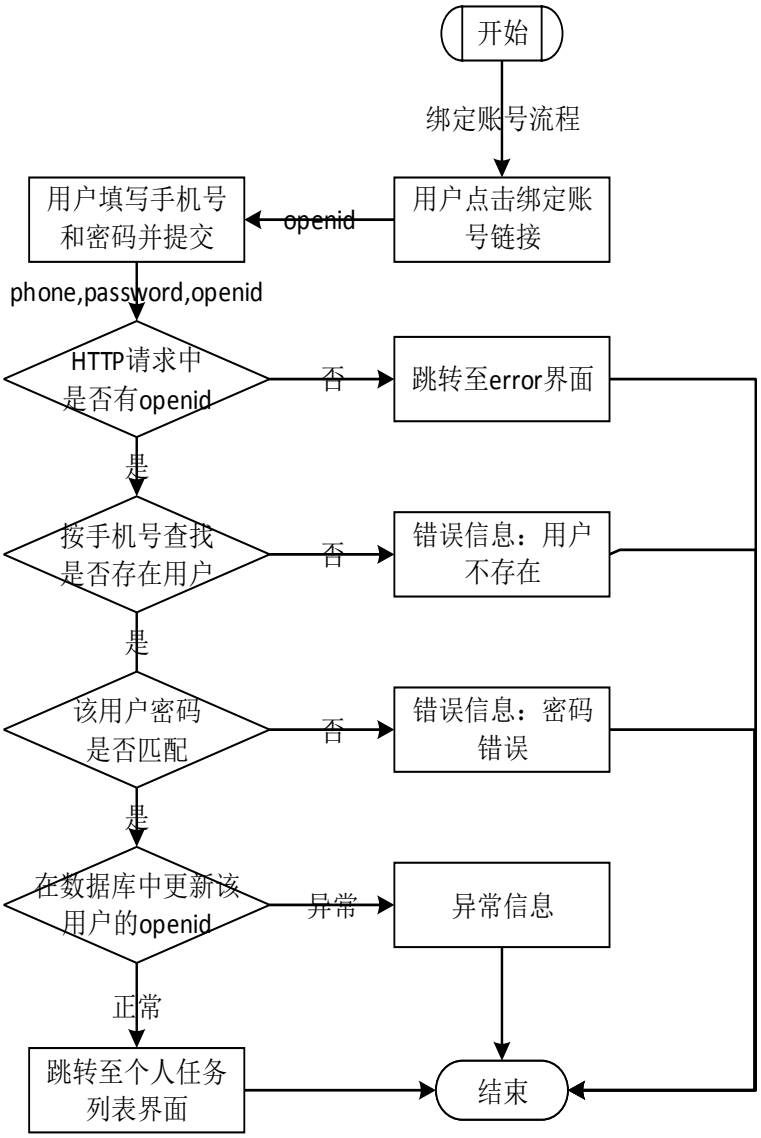

图 4.5 绑定账号流程图

图 4.6 是绑定账号的详细类图，如图所示：控制层为 WeChatLoginController 类，聚合了业务逻辑接口类 MemberService、OutEmployeeService。MemberServiceImpl、OutEmployeeServiceImpl 为内部员工和外聘人员业务逻辑接口的实现类。实现类中聚合了数据访问层的相应 DAO 接口。从数据访问层的实现类中抽取出 Hibernate 框架中的重复的基本操作处理并封装成一个单独的类，HibernateBaseDao 类。该类继承 Hibernate 框架提供的 HibernateDaoSupport 抽象类。流程涉及的实体类有 Member 和 OutEmployee 类，因为模块涉及内部和外部员工两种用户角色，所以访问两种角色的数据表。具体详细类图如下：

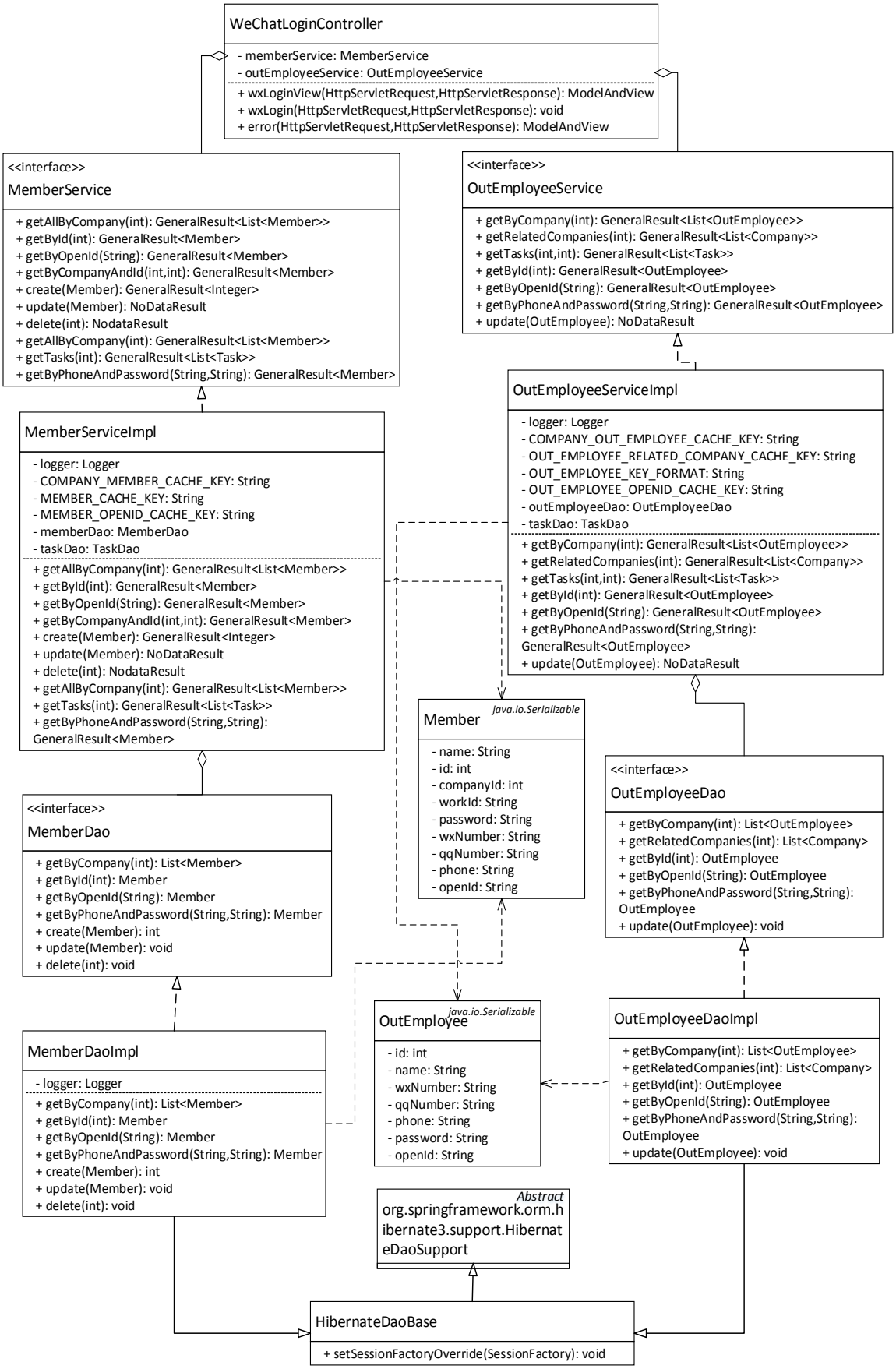

图 4.6 绑定账号详细类图

4.2.3 任务管理的设计

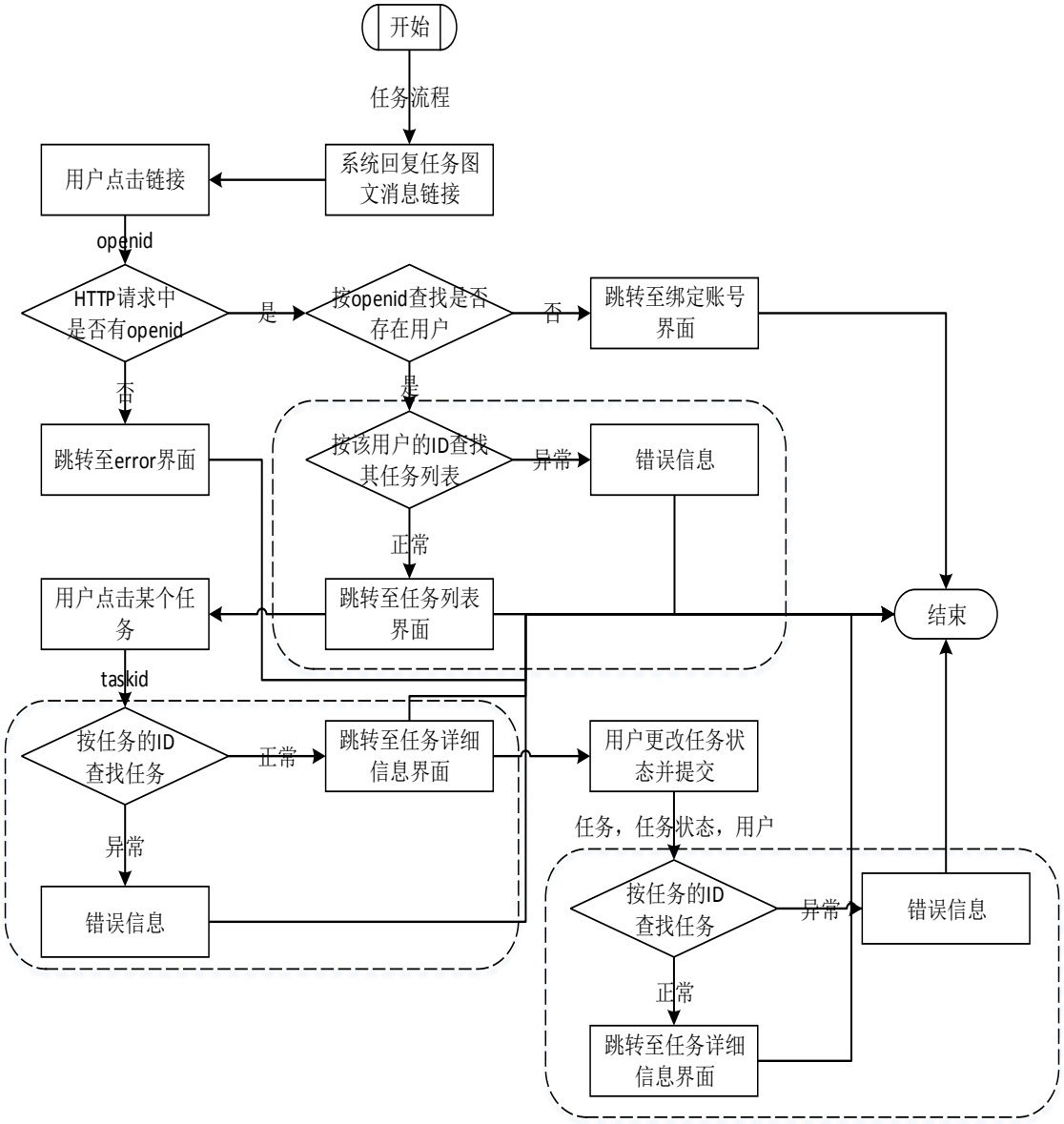

图 4.7 任务管理流程图

如图 4.7 所示的任务管理流程图，任务管理模块包括获取个人任务列表、查看任务信息和更改任务状态三个主要功能，分别对应图中从上到下的那三个虚线框所框出的步骤。

用户发送包含任务相关的消息，系统返回个人任务列表的图文链接，用户通过点击链接来查看。该功能需要身份认证，所以在界面跳转请求发送后，系统需要检验该用户的 openid 是否被绑定到具体的手机号码，如未绑定用户则跳转至绑定界面；如已绑定则显示个人任务列表界面。用户点击相应任务则会跳转至任务详细信息界面，可以更改任务状态一栏并提交来实现任务状态更改。期间有任何异常或错误，则返回错误异常信息并刚结束流程。

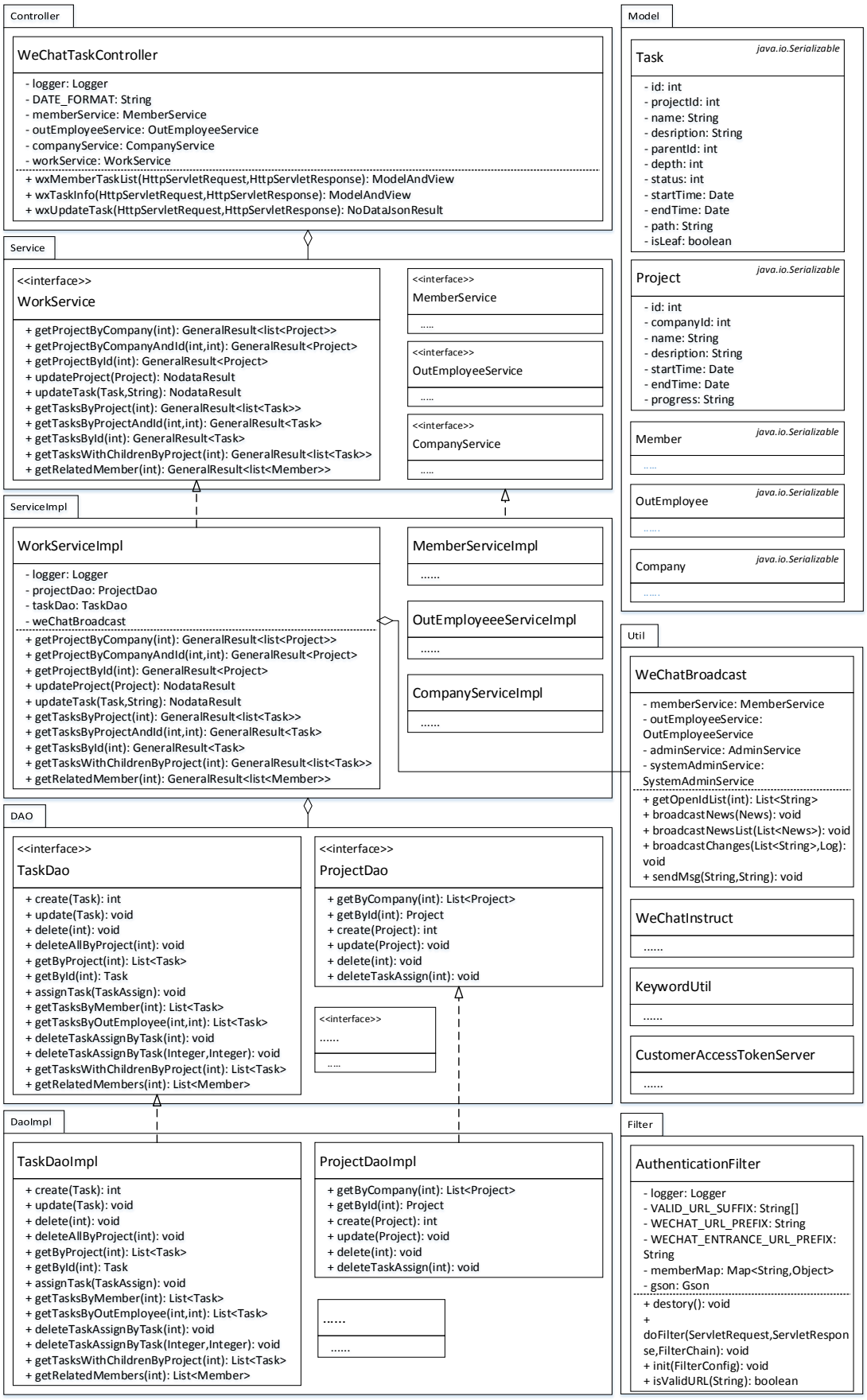

图 4.8 任务管理详细类图

图 4.8 为任务管理的详细类图，采用与图 4.1 详细设计架构相对应的分层模式和分块结构来分包，包括 Controller 包、Service 包、ServiceImpl 包、DAO 包、DaoImlp 包、Model 包、Util/WeChat 包、Filter 包。图中部分类因为上文图中已经详细给出其属性和方法或者不是非常重要，并且还因为空间有限，所以在此省略。Util/WeChat 包中的 WeChatBroadcast 类负责主动向用户发消息，包括发送文本和图文消息，发送对象包括制定对象发送和列表群发。其中，群发图文消息列表接口暴露给任务管理员的资讯发布功能。

4.2.4 资讯管理的设计

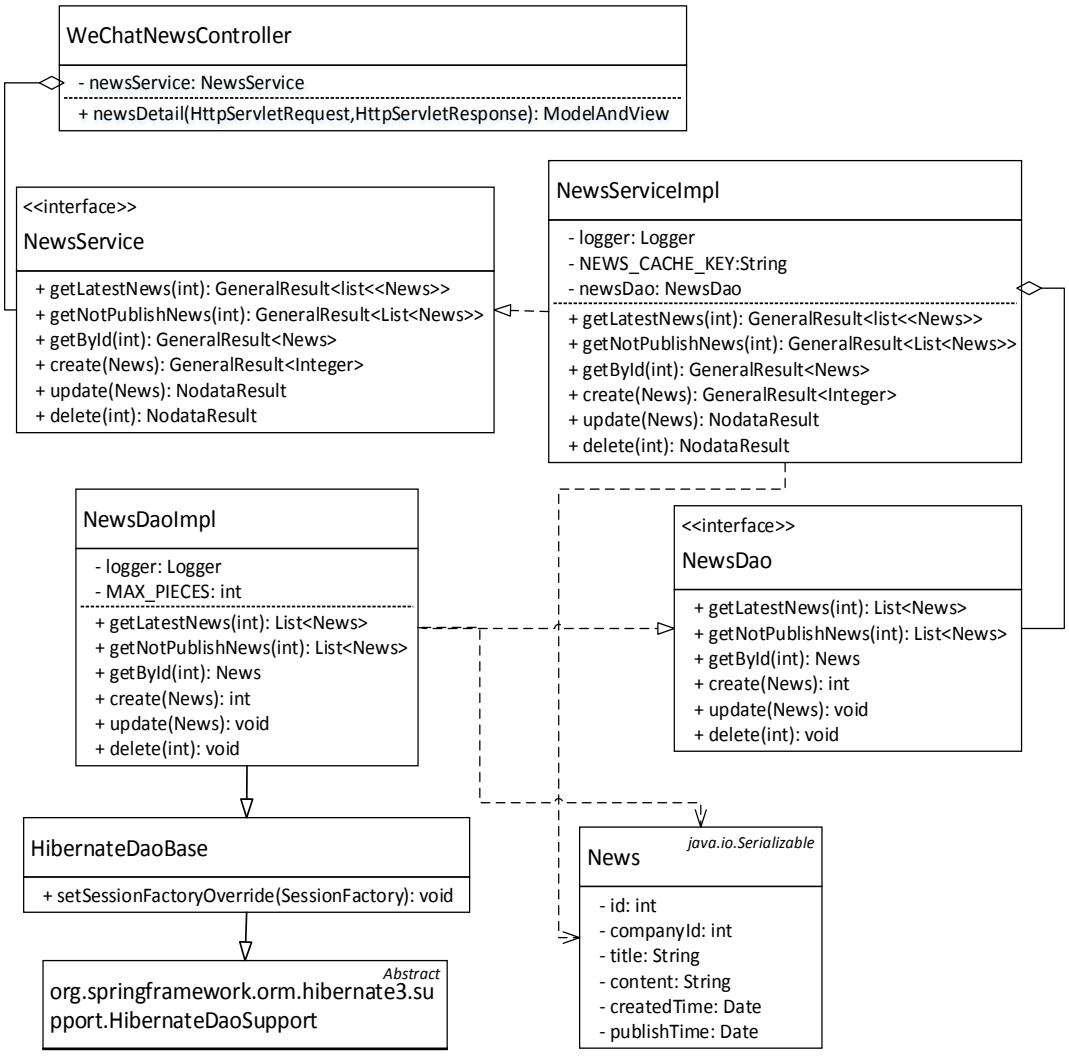

图 4.9 资讯管理详细类图

资讯管理包括获取在最新资讯列表和查看资讯信息两个功能点。其具体流程图已经包括在上文图 4.3 中，资讯相关的详细类设计在 4.2.1 中提及，故在此不再赘述。

4.3 微信模块的实现

4.3.1 相关参数配置的实现

系统是通过Wechat4j框架提供的wechat4j.properties配置文件（如图4.10）来关联微信公众号。配置文件的相关配置数据中，大部分配置项可以在微信公众平台的开发者中心找到。配置文件的最后两项是用来配置 acces_token 和 jsapi_ticket 的中控服务器类路径，自定义其存储和获取方式，自定义类必须要继承 Wechat4j 提供的 CustomerServer 接口类，具体代码如图 4.11 所示。

```
#url  服务器地址
wechat.url= http://1.njucowork.sinaapp.com/weixin
#token 申请公众号时自己设置的令牌
wechat.token=***
#encodingaeskey 消息加解密密钥，用于消息加解密过程
wechat.encodingaeskey=n8lWPHGneefcA7HjYIYZ36uoNggs9kkkF**********

#wechat appid 应用 ID
wechat.appid=wx4f10cb**********
#wechat app secret  应用密钥
wechat.appsecret=c965f2e7c3b973278e8b80**********

#wechat access token server ,when you save in db,must implement you server class
#this class must extend org.sword.wechat4j.token.server.CustomerServer
#if no this property,then token server is default memery accesstoken server
wechat.accessToken.server.class=edu.nju.software.wechat.CustomerAccessTokenSer
ver

#jsapi_ticket customer server class name,
#this class must extend  org.sword.wechat4j.token.server.CustomerServer
#if no this property,then ticket server is default memery ticket server
wechat.ticket.jsapi.server.class=edu.nju.software.wechat.JsApiTicketCustomerServer
```

图 4.10 配置文件 wechat4j.properties

```
public class CustomerAccessTokenServer extends CustomerServer {
    /**
     * 获取 accessToken
     */
    @Override
    public String find() {
        String accessToken = null;
        //执行数据库操作
```

```
            ……
            return accessToken;
        }
        /**
         * 存储 accessToken
         */
        @Override
        public boolean save(Token accessToken) {
            //如果没有需要插入，如果有的就更新，假设已经有了数据库配置项
            ……
            return true;
        }
}
```

图 4.11 CustomerAccessTokenServer 类代码

4.3.2 用户消息处理的实现

WeChatProcessor 继承了 Wechat4j 框架提供的 WechatSupport 支持类，集中响应用户触发的各种事件。onText()用来响应用户发送的文本消息，按照 4.2.1 中消息交互的详细设计流程来实现，具体回复消息抽出来封装成了 WeChatRsp 类中。其中，“@Component”和“@Autowired”是 SpringMVC 框架的注释，分别表示组件和依赖注入。具体代码如图 4.12 所示：

```
@Component
public class WeChatProcessor extends WechatSupport {
    @Autowired
    private WeChatRsp weChatRsp;
    ……
    /**
     * 文本消息
     */
    @Override
    protected void onText() {
        String content = wechatRequest.getContent().trim(); //用户发送的消息
        String openId = wechatRequest.getFromUserName(); //用户的 openid
        // 命令匹配和处理
        ……
        // 回复任务相关图文链接
        if (KeywordHandler.isTask(content)) {
            List<ArticleResponse> tasksRsp = weChatRsp.tasksRsp(openId);
            responseNews(tasksRsp);
        }
        // 回复资讯
```

```
        else if (KeywordHandler.isNews(content)) {
            List<ArticleResponse> news = weChatRsp.newsRsp();
            ……
        }
        // 回复帮助信息
        else if (KeywordHandler.isHelp(content)) {
            responseText(weChatRsp.helpRsp());
        }
        ……
        // 聊天机器人
        else {
            responseText(weChatRsp.chattingRobotRsp(content));
        }
    }
    /**
     * 订阅事件
     */
    @Override
    protected void subscribe() {
        String openId = wechatRequest.getFromUserName();
        responseText(weChatRsp.subscribeRsp(openId));
    }
    ……
}
```

图 4.12 WeChatProcessor 类代码

WeChatRsp 类是从 WeChatProcessor 类中抽取出来的逻辑代码，封装了对用户事件尤其是用户发送文本消息事件的处理逻辑，返回处理过的返回消息，包括资讯和任务需要返回的图文消息和其他文本消息等，具体代码如图 4.13 所示：

```
@Component
public class WeChatRsp {
    @Autowired
    private NewsService newsService;
    ……
    // 返回任务相关图文链接
    public List<ArticleResponse> tasksRsp(String openId) {
        String parameter = "?openId=" + openId;
        List<ArticleResponse> tasksRsp = new ArrayList<ArticleResponse>();
        ArticleResponse viewTasksRsp = new ArticleResponse();
        viewTasksRsp.setTitle("查看任务");
        viewTasksRsp.setDescription("查看任务");
        viewTasksRsp.setUrl(prefix + "/wechat/MyTasks" + parameter);
        viewTasksRsp
```

```
                    .setPicUrl("http://njucowork-pic.stor.sinaapp.com/assign.png");
            tasksRsp.add(viewTasksRsp);
            return tasksRsp;
        }
        // 按照参数构建图文消息
        public static ArticleResponse getNewsRsp(News news, String picURL) {
            ……
        }
        // 返回资讯
        public List<ArticleResponse> newsRsp() {
            GeneralResult<List<News>> result = newsService.getLatestNews(1);
            if (result == null || result.getData() == null
                    || result.getData().isEmpty()) {
                return null;
            } else {
                List<News> newsList = result.getData();
                List<ArticleResponse> newsRsps = new ArrayList<ArticleResponse>();
                int index = 1;
                for (News news : newsList) {
                    String picURL = "http://njucowork-pic.stor.sinaapp.com/number"
                            + index + ".png";
                    ArticleResponse newsRsp = WeChatRsp.getNewsRsp(news,
picURL);
                    if(null == newsRsp){
                        continue;
                    }
                    newsRsps.add(newsRsp);
                    index++;
                }
                return newsRsps;
            }
        }
        // 返回聊天机器人回复
        ……
}
```

图 4.13 WeChatRsp 类代码

4.3.3 推送消息的实现

WeChatBroadcast 类负责推送消息，包括发送图文消息和文本消息，提供发给个人或是群发消息接口。如图 4.14 所示，broadcastNewsList 方法用来向用户推送资讯列表，该方法暴露给任务管理员的资讯发布功能。方法主要通过

Wechat4j 框架封装的客服发送消息接口实现群发推送功能。

```
@Component
public class WeChatBroadcast {
    ……
    // 群发资讯列表
    public void broadcastNewsList(List<News> newsList) {
        List<ArticleResponse> newsRsps = new ArrayList<ArticleResponse>();
        int index = 1;
        for (News news : newsList) {
            String picURL = "http://njucowork-pic.stor.sinaapp.com/number"
                    + index + ".png";
            ArticleResponse newsRsp = WeChatRsp.getNewsRsp(news, picURL);
            if (null == newsRsp) {
                continue;
            }
            newsRsps.add(newsRsp);
            index++;
        }
        List<String> openIdList = getOpenIdList(companyId);
        for (String openId : openIdList) {
            CustomerMsg msg = new CustomerMsg(openId);

            msg.sendNews(newsRsps);
        }
    }
    // 群发资讯
    // 获取用户列表
    // 任务状态修改推送
}
```

图 4.14 WeChatBroadcast 类代码

如图 4.15 所示，broadcastChanges 方法是 WeChatBroadcast 类中负责通过 Log 记录构建任务状态改动通知并推送给该任务相关的用户，需要通过记录中的信息组合成用户友好的文本消息。该接口在修改任务状态的业务逻辑接口调用，以便及时将任务的动态推送给相关人员。

```
// 任务状态修改推送
public void broadcastChanges(List<String> openIDList, Log change) {
    for (Object openId : openIDList) {
        String openIdString = String.valueOf(openId);
        CustomerMsg customerMsg = new CustomerMsg(openIdString);
        String name = "";
        SimpleDateFormat dateFormat = new SimpleDateFormat(
            "yyyy-MM-dd HH:mm:ss");
```

```
            // Member   内部员工
            if (change.getCreatorType() == EmployeeType.MEMBER) {
                    GeneralResult<Member> memberResult = memberService
                            .getById(change.getCreatorId());
                    Member member = memberResult.getData();
                    if (null != member) {
                        name = name + "员工" + member.getName();
                    } else {
                        name += "某员工";
                    }
            // OutEmployee   外聘人员
            } else if (change.getCreatorType() == EmployeeType.OUT_EMPLOYEE) {
                ……
            // Admin   任务管理员
            } else if (change.getCreatorType() == EmployeeType.ADMIN) {
                ……
            }
            String message = "任务状态修改提示：您好，您所参与的"
                    + change.getProject().getName() + "-"
                    + change.getTask().getName() + "任务状态已经于"
                    + dateFormat.format(change.getCreatedTime()) + "被" + name
                    + "由“" + TaskStatus.getStatusStr(change.getOriginStatus())
                    + "”改为“"
                    + TaskStatus.getStatusStr(change.getCurrentStatus())
                    + "”，敬请关注。";
            customerMsg.sendText(message);
        }
}
```

图 4.15 broadcastChanges 方法代码

4.4 运行界面

图 4.16 是用户关注公众账号后，公众号返回的欢迎语和帮助信息，并附带绑定账号的链接。关注公众号后，用户可以与公众号进行交互。

用户点击绑定账号链接后，未绑定账号的用户则会进入绑定账号界面，如图 4.17 所示。用户需要填写自己的手机号码和密码并点击登录来完成账号绑定。在后续版本中，将会考虑将密码填写改为获取动态口令的方式，使功能更加人性化，、更加用户友好。

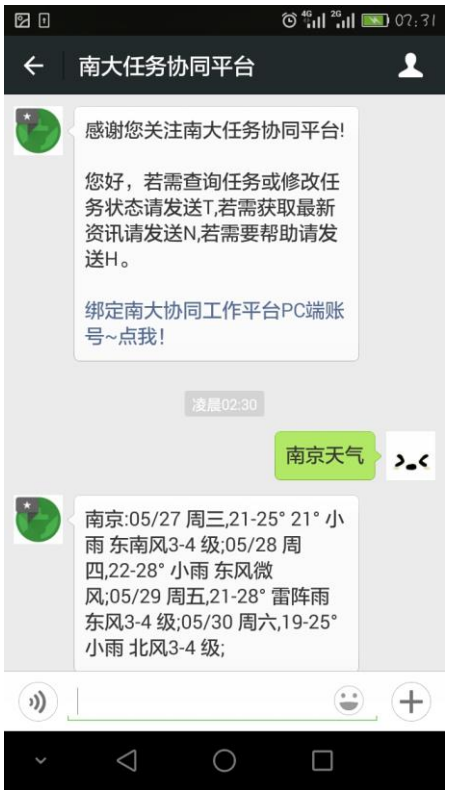

图 4.16 用户关注公众号界面

图 4.17 绑定账号界面

图 4.18 获取任务链接

图 4.19 查看任务列表

图 4.18 是用户发送任务相关的文本信息（如：task、TasK、任务、我的任务、T、t 等）之后，公众号会回复任务链接。用户点击链接后，绑定用户会跳转

至任务列表界面，如图 4.19 所示；如若用户未绑定则跳转至账号绑定界面，如图 4.17 所示。

图 4.20 是用户发送资讯相关的文本信息（如：news、NeWs、资讯、最新资讯、N、n 等）之后，公众号会回复资讯的图文消息列表。用户点击任意一个资讯链接之后，用户就会跳转至查看资讯界面，如图 4.21 所示。注意界面的左上角，我特意为任务协同平台资讯取名为最资讯并设计了一个 LOGO，使之更具有标识性。

图 4.20 获取资讯界面

图 4.21 查看资讯界面

图 4.22 Co-Work 最资讯 LOGO

图 4.23 是用户发送位置消息类型后，公众号会回复到达该位置方式的文本信息。图 4.24 是用户与聊天机器人进行日常对话，除此之外，还可以查询天气、

词条百科、让它讲个笑话等等。

图 4.23 发送位置信息界面

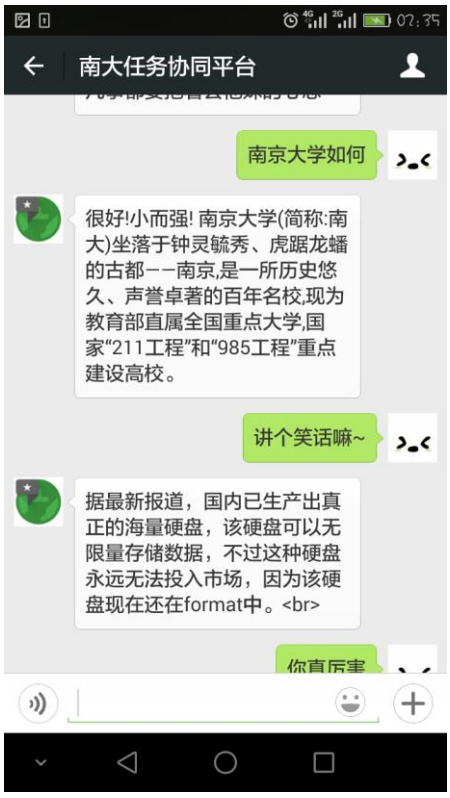

图 4.24 日常聊天界面

4.5 本章小结

本章主要介绍了微信模块的详细设计与实现。首先，在模块的详细设计部分介绍了模块的详细设计架构图，再分别详述了主要功能点的流程图和详细类图。其次，在模块的实现部分，对关键代码做出了解释说明。最后，展示了微信模块的部分运行界面并做出简要说明。

第五章 总结与展望

5.1 总结

由于协同办公系统的发展现状和轻应用的盛行，本文研究了轻量级任务协同系统的市场需求背景与开发的必要性，再结合微信的流行及其良好的发展势头，最终基于 SpringMVC 框架和微信公众平台设计并实现一个轻量级任务协同系统。

系统开发主要包括需求获取、需求分析、系统和模块的概要设计、模块的详细设计、具体实现和测试这几个阶段，是对软件工程理论一次成功的实践。本人在此次项目开发中主要负责系统移动端也就是微信模块的设计与实现。

本文主要介绍了以下几个方面：

1. 介绍了市场背景和系统实现的必要性，阐述了任务协同系统的研究现状以及基于微信公众平台的轻应用的发展前景。
2. 解释了系统的微信模块的实现与开发过程中涉及的比较陌生的技术理论：Wechat4j 框架、智能聊天机器人技术、中文分词和关键字提取技术，为更好地理解系统的实现打下基础。
3. 对系统进行需求分析，先讨论系统用户角色，通过对系统特性分析总结出相关功能需求，并进行了用例建模，同时从系统定位和实际使用的角度出发得出了其非功能需求。
4. 基于系统需求进行概要设计，系统整体分为 PC 端和移动端，总体设计是基于分层式设计理念，从而得到系统总体设计架构、物理架构以及按功能点的子模块划分。根据以上分析设计，确定实体对象以及之间的关系，从而完成关系型数据库的设计。
5. 根据微信模块的主要功能涉及的详细类图、流程图对模块进行详细设计并实现相应功能，通过运行界面截图、关键代码说明来展示模块的完成情况。

开发中主要遇到的问题是，项目架设在 SAE 服务器上，但是 SAE、Java EE、微信公众平台三者并不是很契合，期间引起了许多问题，增加了开发负担。

5.2 展望

在现阶段，本项目只是完成了基本功能。虽然用户已经可以开始使用系统协同办公了，但是由于时间上的缺乏，系统在设计与实现上还有很大改进空间。例如：系统界面设计不够人性化、系统的安全性设计不足、微信端不支持图片音频

等类型的消息、不支持定制开发等等，这都会影响系统的用户体验和性能质量。所以，我们将在后续版本中，主要着眼于以下几个方面：

系统整体的界面设计上要更加符合人机交互方面的理论，使得界面能够更加人性化、更加用户友好；进一步增强系统安全性、健壮性。

PC 端要提高运行和操作响应速度，优化数据库操作和业务逻辑算法；增加自动爬取资讯的功能，实现自动化办公，大大地减轻了任务管理员的负担。

微信模块目前只支持文本和位置消息类型，大大削弱了用户体验，所以需要增加对图片、音频等多媒体消息类型的支持；绑定账号时，将密码改成手机动态口令，可以免去忘记密码的烦恼，使系统更加人性化；现在的版本只支持“@”命令，命令样式及其功能过于单一，所以需要总结出更多种命令格式和操作。

参考文献

[1] 杨思源.中冶建设员工任务管理系统的设计与实现.硕士论文，山东大学，2013年

[2] 中国互联网络信息中心（CNNIC）.第35次中国互联网络发展状况统计报告，2015-01

[3] 郑辉平.协同任务管理协同研究.硕士论文，福建师范大学，2010年

[4] Keyla.突然冒出来的轻应用，都长什么样.2013-11，http://tech.sina.com.cn/zl/post/detail/i/2013-11-12/pid_8436971.htm

[5] 白明凤，匡惠华.高校图书馆移动信息服务中轻应用模式的应用及其借鉴——基于高校图书馆微信公众号的分析.情报资料工作.2014(06):78-79

[6] 腾讯企鹅智库，解密微信：微信平台首份数据研究报告，2015-01

[7] Wang Baocheng, Deng Yu. Study on the Application of Public Platforms of WeChat in Chinese Library Services [J].Library and Information Service, 2013, 57(20): 82-85, 91

[8] 李丹，李娟. 微信和图书馆业务及应用系统整合研究.现代图书情报技术，2014，253（12）：97

[9] 张德申，秦红亮. 微信公众平台开发——订阅号功能开发研究. 电子技术与软件工程，2013（19）：66-67

[10] huj690（博客园），智能聊天机器人小黄鸡及其制作方法，2013-03，http://www.cnblogs.com/huj690/archive/2013/01/24/2875114.html

[11] 宁长英.智能聊天机器人的关键技术研究，杭州电子科技大学. 硕士论文，2011

[12] ESTHER INGLIS-ARKELL. Minstrelboy. 聊天机器人伪装成真人的10种伎俩，http://www.guokr.com/article/245733/

[13] Mjs（360电商技术），聊天机器人与自动问答技术，2015-01，http://www.open-open.com/lib/view/open1421917537250.html

[14] 图灵智能聊天机器人接入平台，http://www.tuling123.com/openapi/cloud/access_api.jsp

[15] 曹卫峰. 中文分词关键技术研究.硕士，南京理工大学，2009

[16] 龙树全，赵正文，唐华.中文分词算法概述.电脑知识与技术，2009（10）：2605-2606

[17] 俞鸿魁. 基于层次隐马尔可夫模型的汉语词法分析和命名实体识别技术. 硕士论文，北京化工大学，2004年

[18] 黄瑾. ICTCLAS 代码学习笔记. 中科院计算技术研究所多语言交互技术评测实验室，2006.

[19]刘群，张华平，俞鸿魁，程学旗，基于层叠隐马模型的汉语词法分析，计算机研究与发展 2004（08）

[20]中科院，ICTCLAS 源码解析——总体架构，http://www.docin.com/p-822394603.html

[21]开源 Java 中文分词器 Ansj 作者孙健专访，http://blog.csdn.net/blogdevteam/article/details/8148451

[22]关键词抽取简介，http://ling0322.info/2014/04/08/introduction-to-keyphrase-extraction.html

[23]罗新星，李柱辉，赵玉洁.软件非功能需求国内外研究综述.计算机应用研究，2015，32（04）

[24]骆斌，丁二玉.需求工程——软件建模与分析，北京市，高等教育出版社，2009，28-25

致谢

经历了两个多月，轻量级协同系统的开发和论文的撰写终于要画上句号了。在此，请允许我向在项目期间所有给予过我帮助和支持人表示由衷的谢意！

首先，我尤其要感谢这段时间一直帮助、督促我进行毕业设计的荣国平老师。在项目设计方面，荣老师给了我们很大的发挥空间；在项目实际开发过程中，荣老师通过组会、微信、邮件多种方式与我们交流、提供帮助；最后在论文修改方面，荣老师也提出了不少宝贵意见。

其次，我非常感谢与我一起开发完成这个项目的组员，他们是徐思敏、金程炜、董一韬。虽然在项目的设计与实现的讨论中，我们产生过分歧，也进行过激烈的争论，但是这反而更加深了我们对项目的认识，也拉近了团队成员间的距离，使得我们可以更好地进行团队合作并完成整个项目。

另外，我要感谢院里所有的老师、教务员、辅导员，是你们对我毫无保留地教诲、付出，使我在获得知识的同时也懂得了许多书本中无法获得的知识，使我受益终身。

同时，我还要感谢一直默默支持帮助我的家人，你们是我无所顾忌、一路向前的最大支撑和动力；感谢陪我一起哭过笑过、共同成长的朋友和同学们，愿友谊地久天长！

最后，我要向所有参与论文评审和答辩的老师们鞠个躬，以此表达我的最诚挚的谢意！