**知 识 库 管 理 系 统**

**——软件概要设计说明书**

# 目 录

[第1章 引言 1](#_Toc453409959)

[1.1 编写目的 1](#_Toc453409960)

[1.2 背景 1](#_Toc453409961)

[1.3 定义 1](#_Toc453409962)

[第2章 体系架构 2](#_Toc453409963)

[2.1 软件架构 2](#_Toc453409964)

[2.1.1 客户端 2](#_Toc453409965)

[2.2.2 数据库 4](#_Toc453409966)

[2.2 系统功能模块 5](#_Toc453409967)

[2.3 逻辑架构 6](#_Toc453409968)

[2.4 物理架构 7](#_Toc453409969)

[2.5系统应用技术 7](#_Toc453409970)

[第3章 系统数据结构设计 8](#_Toc453409971)

[3.1 逻辑结构设计要点 8](#_Toc453409972)

[3.2 物理结构设计要点 10](#_Toc453409973)

[第4章 接口设计 11](#_Toc453409974)

[4.1 用户接口 11](#_Toc453409975)

[4.2 外部接口 11](#_Toc453409976)

[4.3 内部接口 11](#_Toc453409977)

# 第1章 引言

## 1.1 编写目的

在完成了软件设计的需求分析的基础上，为了明确软件需求、安排项目规划和进度、组织软件开发和测试，小组成员提出了这份概要设计说明书，主要目的在于设计软件结构，划分出软件的基本组成模块，确定模块间的关系，明确系统的整体框架和接口设计，为下一阶段的详细设计、编码和测试提供参考依据。预期读者为小组开发人员。

## 1.2 背景

在人们日常的生活和工作中，每个人在各自的范围内不断累积，形成不同层次的过程资源财富，涵盖了文本、传真、图形、视频、音频等各个类型的知识财富载体。但是随着知识财富的不断累积，由于知识财富的来源不同、用途不同、载体不同等，经常导致知识财富的时间链条错乱、关联知识无法有效检索、各类知识无法综合运用等问题的出现，伴随着知识财富的持续积累，这种问题愈发突出、迫切需要行之有效的手段来录入、管理、分析、统计知识财富，行之有效的归纳利用知识财富，将助推个人及组织目标的实现。

说明：系统名称：知识库管理系统

任务提出者：中国软件杯组委会

开发者：奔跑的树懒小组

用户：访问知识库管理系统的企业、组织和个人

## 1.3 定义

* Windows 2003 server 64位：服务器操作系统
* MYSQL：数据库管理软件
* JDK1.8：运行环境
* Win7：开发终端操作系统
* Java：软件开发语言
* KBMS：知识库管理系统的简称
* SpringMVC：WEB开发框架

# 第2章 体系架构

## 2.1 软件架构

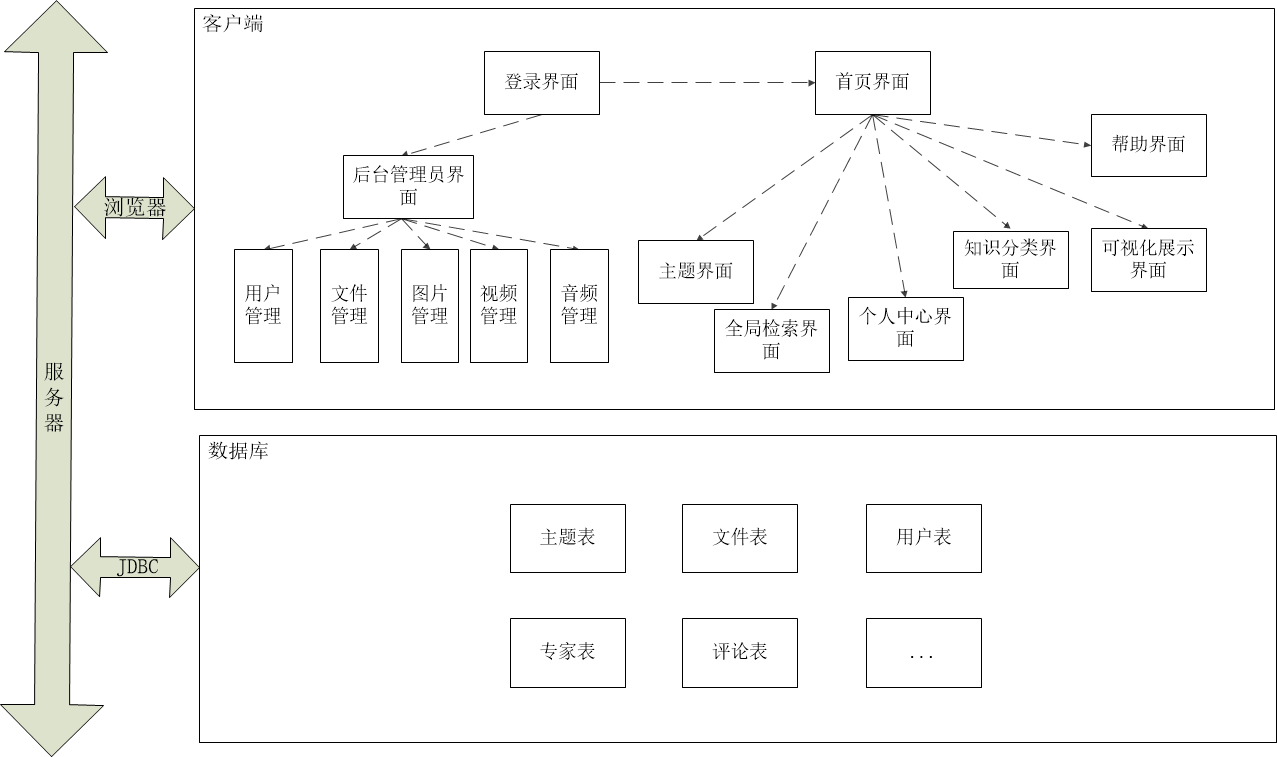


图2-1 系统总体软件结构图

### 2.1.1 客户端

客户端包括多个界面，如图2-1所示，具有登录界面、首页界面、后台管理界面、主题界面、全局检索界面、知识应用界面、可视化界面、用户管理界面、知识管理界面等。

**1.登录界面**

登录界面主要用于用户登录知识库管理系统，没有注册的用户只具有浏览功能，系统的一些功能无法使用，只有注册为用户之后，才可以享受系统的全部功能。

**2.首页界面**

首页界面对系统进行了简要的介绍，并且从该界面可以进入其他的主界面。

**3.主题界面**

主题界面罗列了10个大主题，包括基础科学、农业科技、医药卫生科技、信息科技、哲学与人文科学、经济与管理科学等主题。点击每个大主题，系统会进入该主题下的详细界面，列出其中的每个小领域，以及小领域里的文件、图片、视频和音频。点击每条知识，都可以进入知识详情界面，在里面可以查看知识的详细信息（如大小、作者、上传时间、类别等），并且可以进行评论，在线预览，收藏和下载。

**4.全局检索界面**

全局检索主要是供用户对知识库里的知识进行检索、查找的功能。点击首页的全局检索按钮，进入检索界面，检索分为两种形式，简单检索和高级检索。简单检索是指，用户在搜索框中输入关键字，点击搜索就可以进行查找，这种搜索操作简单，但找到的结果并不精确，系统会返回许多相似的内容。而高级检索则可以非常精确的找到用户希望看到的知识。点击搜索框旁边的高级搜索按钮，在下面可以输入标题、摘要、作者、内容等关键字，还可以选择时间范围，然后选择精确检索还是模糊检索，检索出的结果相对来说更精确。点击检索出来的文件可以进入知识详情界面。

**5.知识分类界面**

知识分类界面将知识分为四种即文件、图片、视频、音频。用户可以分别进入相应的界面进行浏览，检索等操作。

**6.可视化展示界面**

可视化展示界面以图形化的形式对知识进行展示，简单、直观、易懂。主要有主题专家网络、主题图谱等功能。主题专家网络显示的是每个主题中的核心专家，用户输入一个主题，系统会对每个主题下的专家进行计算，分别进行评分，找出其中的核心专家，计算过程通过hadoop集群来计算，将结果以echarts图的形式可视化的展示。主题图谱是指，系统对于10个大的主题分别统计其中的知识总量，并以饼图的形式来展示其所占的比重。

**7.知识问答界面**

知识问答界面可以使用户对于不清楚的难题对系统进行提问，系统会给出一些参考的答复。对于用户提出的问题，系统会首先在自己的系统里进行检索，如果有该问题的答案则返回到界面显示，如果在系统中搜不到，会自动接入互联网，需找答案。

**8.个人中心界面**

只有已经注册的用户才可以看到该界面。进入个人中心界面，用户可以进行上传文件，查看个人资料信息，修个密码，查看自己的收藏，查看自己的检索历史，以及自己的关注。

用户上传：用户可以批量上传、单独上传、拖拽上传。对于上传的文件，用户可以点击编辑按钮，对上传的知识进行编辑，输入文件名，作者，选择所属主题，摘要等信息。点击发布，系统会根据这些信息对文件进行分类，归并到相应的主题下进行显示。

知识关联：用户可以对相似的知识进行添加关联的操作，可以单个关联和批量关联，只需要用户选中知识的编号，系统会自动根据唯一的编号进行关联。对于关联成功的知识，将在知识详情界面的下方进行显示。

检索历史：系统使用AOP技术，对用户的行为进行监控，记录用户的行为特征，用户在个人中心里可以查看自己的检索历史，并且可以进行删除操作。

知识收藏：用户在浏览知识的同时，碰到喜欢的，重要的知识，为了便于下次查看，而不用再去主题里查找，可以点击收藏，系统会将该知识发送到收藏文件夹中。另外用户可以对其进行删除。

个人信息管理：用户可以浏览自己的详细资料，并且可以进行修改。可以对密码进行修改操作。

用户推荐：系统会根据用户的行为，对用户进行个性化的推荐。采用基于用户的协同过滤算法实现推荐的功能。

**9.后台管理界面**

后台管理只有管理用有权限进入，普通用户无法进入，管理员界面分为用户管理，文件管理，图片管理，视频管理，音频管理。

用户管理：管理员可以对系统用户进行查询、添加、删除等操作

文件管理：管理员可以对用户上传的文件进行查询、删除等操作

视频管理：管理员可以对用户上传的视频进行查询、删除等操作

图片管理：管理员可以对用户上传的图片进行查询、删除等操作

音频管理：管理员可以对用户上传的音频进行查询、删除等操作

### 2.2.2 数据库

本系统使用MySQL数据库对数据进行存储，对于系统涉及到的数据结构表进行统一管理，包括用户相关的表，主题相关的表，文件相关的表等，具体ER图见第3章。

## 2.2 系统功能模块



图2-2 系统整体功能模图

## 2.3 逻辑架构

用户登录本系统，通过web界面与系统进行交互，向系统发送请求，系统接收到请求，通过业务逻辑进行处理，并连接服务器从数据库中取的所需的数据。另外本系统中涉及到大量的数据生成，如专家图谱中寻找核心专家，向用户进行推荐，都会有数据的计算，所以通过hadoop集群处理大数据量，可以更好地解决关于大数据的处理问题。



图2-3 系统逻辑架构图

## 2.4 物理架构



图2-4 系统物理架构图

## 2.5系统应用技术

* 开发语言Java
* 前台UI采用Metronic，后台UI采用EasyUI
* 前端采用JavaScript、JQuery、Ajax
* 框架采用Spring 3.0、Spring MVC、Mybatis持久层框架
* 数据库采用MySQL 5.5或更高
* Maven构建项目
* Git版本控制系统
* 中间件采用Tomcat

# 第3章 系统数据结构设计

## 3.1 逻辑结构设计要点



图3-1 系统数据库ER图



图3-2 用户实体及属性



图3-3 知识实体及属性



图3-4 主题实体及属性

图3-5 专家实体及属性

## 3.2 物理结构设计要点



图3-6 数据库表结构

# 第4章 接口设计

## 4.1 用户接口

根据需求分析的结果，用户需要一个用户友善、美观、易用的、响应式的界面。在界面设计上，应做到简单明了，易于操作。另外，应该将用户常用的、比较重要的界面进行合理的布局，突出显示。可以使用网页设计工具画出美观、简洁、友善的用户接口。另外对于后台管理人员的界面要做到操作简单，易于管理。同时，运行出错时应以标准的形式给出出错提示。总的来说，用户界面应该做到可靠性、简单性、易学习和使用。

## 4.2 外部接口

* 服务器程序可使用第三方jar包，通过JDBC的形式，对MYSQL数据库进行访问。
* 服务器程序上可使用MySQL的对数据库的备份命令，以做到对数据的保存。
* 采用B/S架构模式，用户无需客户端，只需要有浏览器，能够正常上网就能够使用本平台。
* 支持Windows/Linux的服务器。

## 4.3 内部接口

内部接口方面，各模块之间采用函数调用、参数传递、返回值的方式进行信息传递。具体如下。

**模块类说明**

表4-1 模型类表

|  |  |  |
| --- | --- | --- |
| **序号** | **模块类名** | **类说明** |
| 1 | annotation | Mybtais接口标注标签 |
| 2 | autoanswer | 自动问答模块 |
| 3 | autocomplete | 输入字符自动补全 |
| 4 | aop | 用户检索足迹记录 |
| 5 | constant | 字符串常量定义 |
| 6 | dict | 系统词库及更新机制 |
| 7 | index | 索引定时更新模块 |
| 8 | intelligence | 系统情报模块 |
| 9 | literature | 文献模块 |
| 10 | lucene | Lucene检索模块 |
| 11 | page | 分页类 |
| 12 | plugin | 拼写转换类、分页拦截器 |
| 13 | relation | 关联模块 |
| 14 | search | 检索服务 |
| 15 | security | 系统管理员模块 |
| 16 | theme | 系统主题模块 |
| 17 | userCenter | 用户个人中心模块 |
| 18 | Upload | 上传模块 |

**userCenter类接口详细说明**

表4-2 Expert类接口说明表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **序号** | **接口函数** | **函数类型** | **参数名称** | **参数类型** | **函数描述** |
|
|
| 1 | void updatePassword(User user) | void | user | User | 更新密码 |
| 2 | void  updateUserInfo(User user); | void | user | User | 更新用户信息 |
| 3 | List<History> getUserHistoryListPage(History history) | List<History> | history | History | 获得浏览历史 |
| 4 | List<General> getUserGeneralListPage(General general) | List<General> | general | General | 获得常用文档 |
| 5 | void  delUserHistory(int id，int userId) | void | Id,userId | int | 删除历史 |
| 6 | void  delGeneral(int id，int userId); | void | Id,userId | int | 删除收藏 |

**literature类接口详细说明**

表4-3 literature类接口说明表

| **序号** | **接口函数** | **函数类型** | **参数名称** | **参数类型** | **函数描述** |
| --- | --- | --- | --- | --- | --- |
| 1 | getAllLiterature(Page page) | List<Literature> | page | page | 获取所有文件列表 |
| 2 | getTotalRecord() | int | 无 | 无 | 获得文件数量 |
| 3 | getTotalRecordByThemeId(int themeId) | int | themeId | int | 通过同一主题文件数量 |
| 4 | getLiteratureByThemeId(int themeId,Page page) | List<Literature> | themeId | int | 通过同一主题的所有文件内容 |
| page | page |
| 5 | getLiteratureById(int id) | Literature | id | int | 通过文件id获得文件内容 |
| 6 | void setScore(int fileId,int userId,int score) | void | userId,fileId,score | int | 用户给文件打分 |
| 7 | void setTotleScore(int fileId,int score) | void | fileId,score | int | 计算文件平均分 |
| 8 | List<RecommendedItem> recommend(int userId)throws IOException, TasteException | List<RecommendedItem> | userId | int | 文件推荐支持类 |
| 9 | void relationFile(int relationId,int id) | void | relationId,id | int | 关联文件 |
| 10 | int fileUniqueId(String themeName,String type,String fileId) | int | themeName,type,fileId | string | 生成唯一编号 |

**search类服务层接口详细说明**

表4-4 search类接口说明

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **序号** | **接口函数** | **函数类型** | **参数名称** | **参数类型** | **函数描述** |
| 1 | SearchResponse<Picture> searchPicture(SearchRequest request); | SearchResponse<Picture> | request | SearchRequest | 图片检索 |
| 2 | searchLiterature(SearchRequest request) | SearchResponse<Expert> | request | SearchRequest | 文献检索 |
| 3 | SearchResponse<Video> searchVideo(SearchRequest request) | SearchResponse<Patent> | request | SearchRequest | 视频检索 |
| 4 | SearchResponse<Audio> searchAudio(SearchRequest request) | SearchResponse<Audio> | request | SearchRequest | 音频检索 |
| 5 | searchGlobal(SearchRequest request) | SearchResponse<Intelligence> | request | SearchRequest | 全网检索 |
| 6 | SearchResponse<Literature> searchAll(SearchRequest request) | SearchResponse<Literature> | request | SearchRequest | 全类型检索 |

**theme类接口详细说明**

表4-5 theme类接口说明表

| **序号** | **接口函数** | **函数类型** | **参数名称** | **参数类型** | **函数描述** |
| --- | --- | --- | --- | --- | --- |
| 1 | getTopTheme（） | List<Theme> | 无参数 | 无 | 返回热门行业列表 |
| 2 | updateThemeHotScore（） | void | 无参数 | 无 | 定时更新行业热度 |
| 3 | getThemeTree() | List<ThemeTree> | 无 | 无 | 获得主题树 |
| 4 | getThemeIdByThemeName(String themeName) | int | themeName | String | 通过主题名获得主题id |