项目设计文档

*InSighter*

**更新历史**

|  |  |  |  |
| --- | --- | --- | --- |
| 修改人员 | 日期 | 变更原因 | 版本号 |
| 张诗琦 | 2016-2-28 | 最初草稿 | V1.0 |
| 董轶波 | 2016-2-29 | 加入userbl与userData模块描述 | V1.1 |
| 张诗琦 | 2016-2-29 | 修改repobl与repodata模块描述，插入逻辑包图 | V1.2 |
| 陈之威 | 2016-4-10 | 加入logbl与logdata模块描述 | V1.3 |
| 顾晗 | 2016-4-11 | 加入statisticsbl模块，更新repobl和userbl模块 | V1.4 |
| 顾晗 | 2016-4-16 | 更新开发包图，改错 | V1.4 |

1. **引言** 
   1. **编制目的**

本文档详细完成对GitMining System数据分析系统的概要设计，达到指导开发的目的，同时实现测试人员及用户的沟通。

本文档面向开发人员、测试人员及最终用户编写，是了解系统的导航。

1. **产品描述**

参考GitMining System数据分析系统软件需求规格说明中对产品的概括描述。

1. **逻辑视角**

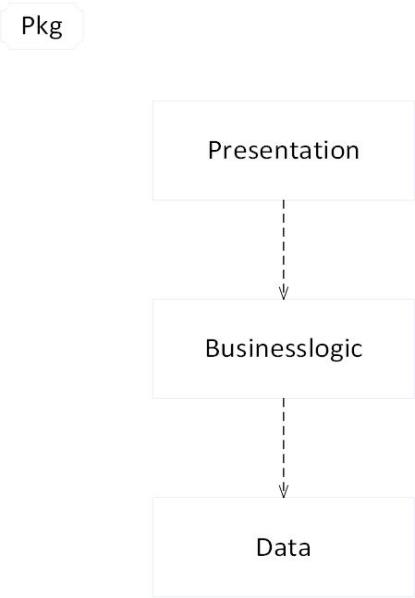
GitMining System数据分析系统中，选择了分层体系结构的风格，将系统分为3层（展示层、业务逻辑层、数据层）能够很好的示意整个高层抽象。展示层包括GUI页面的实现，业务逻辑层包含业务逻辑处理的实现，数据层负责数据的读取。分层体系结构的逻辑视角和逻辑设计方案如图1和图2所示。

图1逻辑视角

# 4.组合视角

# 4.1.开发包图

GitMining数据分析系统的最终开发包设计如图3和表1所示。

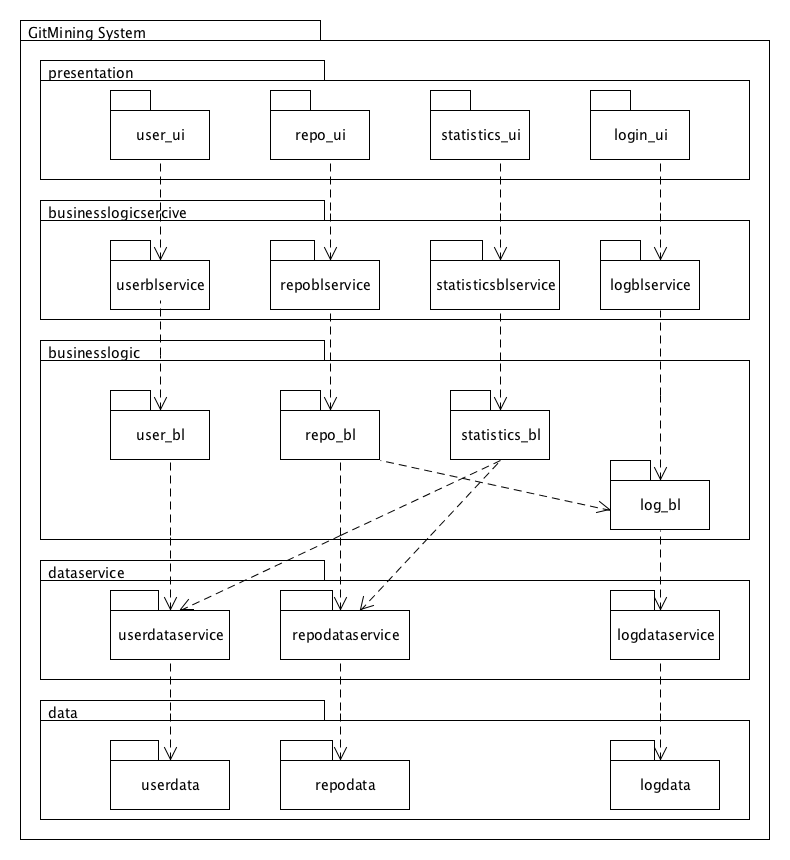


图3:开发包图

**表1层之间调用的接口**

|  |  |  |
| --- | --- | --- |
| 接口 | 服务调用方 | 服务提供方 |
| userblservice  repoblservice  logblservice  statisticblservice | 展示层 | 业务逻辑层 |
| userdataservice  repodataservice  logdataservice | 业务逻辑层 | 数据层 |

# 5.接口视角

## 5.1.业务逻辑层的分解

### 5.1.1业务逻辑层模块的职责

|  |  |
| --- | --- |
| 模块 | 职责 |
| userbl | 负责实现查询Github用户信息界面所需要的服务 |
| repobl | 负责实现查询Github项目信息界面所需要的服务 |
| statisticsbl | 负责实现界面展示统计图表所需要的服务 |
| logbl | 负责实现为当前用户提供个性化的服务 |

### 5.1.2业务逻辑模块的接口模范

**repobl接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| repobl.showRepoByName | 语法 | public RepoVO showRepoByName(String repoName) |
| 前置条件 | 系统根据关键字查询列出相关项目列表 |
| 后置条件 | 根据名字查找具体项目，返回项目详细信息 |
| repobl.searchVague | 语法 | public void searchVague(String keyword) |
| 前置条件 | keywords确实存在 |
| 后置条件 | 在RepoBL中设置searchlist |
| repobl.searchPrecise | 语法 | public RepoVO searchPrecise(String repo, String user) |
| 前置条件 | 系统处于某一用户详细信息的展示界面 |
| 后置条件 | 返回该用户的某一具体项目的详细信息并显示 |
| repobl.filterRepo | 语法 | public ArrayList<SortVO> filterRepo(CategoryOfRepo cate, Language lan, String year, boolean isSearch) |
| 前置条件 | 用户通过分类检索或关键词查找之后进行筛选 |
| 后置条件 | 根据分类或筛选指标显示筛选结果 |
| repobl.recommendRepos | 语法 | public ArrayList<String> recommendRepos(String full\_name) |
| 前置条件 | 用户点击项目链接查看详细信息 |
| 后置条件 | 在推荐项目栏显示推荐项目名称 |
| repobl.sort | 语法 | public ArrayList<SortVO> sort(ArrayList<String> list) |
| 前置条件 | 系统已经通过模糊查询获得相关的项目信息序列 |
| 后置条件 | 根据排序类型对项目信息序列进行排序，并且返回已经排好序的项目信息序列 |
| 需要的服务（需接口） | | |
| 服务名 | | 服务 |
| repodataservice.getAllRepos() | | 返回所有项目的PO |
| repodataservice.searchVague(String keyword) | | 根据关键字得到相关项目列表 |
| repodataservice.searchPrecise(String repo, String User) | | 根据项目和用户名字得到项目信息 |

**userbl接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| userbl.showUserByName | 语法 | public UserVO showUserByName(String userName) |
| 前置条件 | 系统根据关键字查询列出相关用户列表 |
| 后置条件 | 根据名字查找具体项目，返回用户详细信息 |
| userbl.searchVague | 语法 | public ArrayList<String> searchVague(String keyword) |
| 前置条件 | keywords确实存在 |
| 后置条件 | 返回与keyword相关的用户信息列表，按照默认方式排序显示 |
| userbl.searchPrecise | 语法 | public UserVO searchPrecise(String repo, String user) |
| 前置条件 | 系统处于某一项目详细信息的展示界面 |
| 后置条件 | 返回该项目的某一合作者的详细信息并显示 |
| userbl.filterByFollowers | 语法 | public ArrayList<String> filterByFollowers(int low, int high) |
| 前置条件 | 用户在进行关键词查找用户后设定followers筛选区间 |
| 后置条件 | 返回符合条件的用户名称列表 |
| userbl.sort | 语法 | public ArrayList<String> sort(ArrayList<String> list) |
| 前置条件 | 系统已经通过模糊查询获得相关的用户信息序列 |
| 后置条件 | 根据排序类型对用户信息序列进行排序，并且返回已经排好序的用户信息序列 |
| 需要的服务（需接口） | | |
| 服务名 | | 服务 |
| userdataservice.getAllUsers() | | 返回所有User的PO |
| userdataservice.searchVagus(String keyword) | | 根据关键字得到相关用户列表 |
| userdataservice.searchPrecise(String repo, String User) | | 根据项目和用户名字得到项目信息 |

Statisticsbl 接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| userstatisticsbl.getUserAssessment | 语法 | public UserAssessmentVO getUserAssessment(String user) |
| 前置条件 | 用户存在于可获得的数据中 |
| 后置条件 | 返回当前用户的评价指数 |
| userstatisticsbl.getUserActivity | 语法 | public HashMap<MyDate, Integer> getUserActivity(String user) |
| 前置条件 | 用户存在于可获得的数据中 |
| 后置条件 | 返回当前用户最近一周内的commit次数按天统计 |
| userstatisticsbl.getOwnerType | 语法 | public HashMap<OwnerType, Integer> getOwnerType() |
| 前置条件 | 本地数据已经加载完毕 |
| 后置条件 | 返回每种OwnerType对应的User数量 |
| userstatisticsbl.getCreated\_at | 语法 | public HashMap<Integer, Integer> getCreated\_at() |
| 前置条件 | 本地数据已经加载完毕 |
| 后置条件 | 返回创建于不同日期的User数量 |
| repostatisticsbl.getRadar | 语法 | public RepoRadarVO getRadar(String Repo) |
| 前置条件 | 本地数据已经加载完毕 |
| 后置条件 | 返回对应项目的雷达图参数 |
| repostatisticsbl.getRepoActivity | 语法 | public HashMap<MyDate, Integer> getRepoActivity(String Repo) |
| 前置条件 | 本地数据已经加载完毕 |
| 后置条件 | 返回项目近期活动数量 |
| repostatisticsbl.getLanguage | 语法 | public HashMap<Integer, Integer> getLanguage() |
| 前置条件 | 本地数据已经加载完毕 |
| 后置条件 | 返回每种语言对应的项目数 |
| repostatisticsbl.getCreateTime | 语法 | public HashMap<Integer, Integer> getCreateTime() |
| 前置条件 | 本地数据已经加载完毕 |
| 后置条件 | 按年份统计项目的创建时间 |
| repostatisticsbl.getRepoFork | 语法 | public HashMap<Integer, Integer> getRepoFork(int interval) |
| 前置条件 | 本地数据已经加载完毕 |
| 后置条件 | 按fork数区间统计项目数 |
| repostatisticsbl.getRepoStar | 语法 | public HashMap<Integer, Integer> getRepoStar(int interval) |
| 前置条件 | 本地数据已经加载完毕 |
| 后置条件 | 按star数区间统计项目数 |
| repostatisticsbl.getLanguage | 语法 | public HashMap<Integer, Integer> getLanguage(Language language) |
| 前置条件 | 本地数据已经加载完毕 |
| 后置条件 | 返回特定语言的年变化量 |
| 需要的服务（需接口） | | |
| 服务名 | | 服务 |
| userdataservice.searchPrecise(String keyword) | | 根据关键字返回对应用户 |
| userdataservice.getAllUsers() | | 提供所有可获得的用户信息 |
| repodataservice.getAllRepos() | | 返回所有项目的PO |
| repodataservice.searchPrecise(String repo, String User) | | 根据项目和用户名字得到项目信息 |

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| logbl.loginAnonymously | 语法 | Public void loginAnonymously() |
| 前置条件 | 系统初始化完成并已连接网络 |
| 后置条件 | 以游客身份访问系统 |
| logbl.login | 语法 | public boolean login(String login, String password) |
| 前置条件 | 系统初始化完成并已连接网络 |
| 后置条件 | 以用户身份访问系统 |
| logbl.logout | 语法 | public void logout() |
| 前置条件 | 系统已经登录 |
| 后置条件 | 退出登陆，保存操作信息 |
| logbl.recommendRepo | 语法 | public ArrayList<String> recommendRepo() |
| 前置条件 | 系统已登录 |
| 后置条件 | 根据当前使用者操作记录返回个性化项目推荐列表 |
| 需要的服务（需接口） | | |
| 服务名 | | 服务 |
| logdataservice.loginAnonymously() | | 获取匿名登录权限 |
| logdataservice.login(String login, String password) | | 获取用户登录权限 |
| logdataservice.logout() | | 保存操作信息 |

## 5.2.数据层的分解

数据层主要给业务逻辑层提供数据访问服务，主要是读取txt格式持久化的项目和用户数据。

### 5.2.1数据层模块的职责

数据层模块职责如下表所示

**数据层模块职责**

|  |  |
| --- | --- |
| 模块 | 职责 |
| userdata | 负责读取用户信息，包括用户基本信息，用户参与项目，用户创建项目。 |
| repodata | 负责读取项目信息，包括git项目的基本信息，使用语言，以及贡献者、合作者、fork的数量。 |
| logdata | 负责读取使用者操作信息记录，包括用户名，密码，访问次数较多项目。 |

### 5.2数据层模块的接口规范

数据层模块的接口规范如下表

**数据层模块的接口规范**

repodata**接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| repodata.getAllRepos | 语法 | public Iterator<RepoPO> getAllRepos() |
| 前置条件 | 无 |
| 后置条件 | 返回所有项目PO的迭代器 |
| repodata.searchVague | 语法 | public ArrayList<RepoPO> searchVague (String keyword) |
| 前置条件 | keyword不为null |
| 后置条件 | 根据关键字返回相关项目信息的列表，如果不存在则提示之 |
| repodata.searchPrecise | 语法 | public RepoPO searchPrecise(String repo, String user) |
| 前置条件 | Repo, user参数合法 |
| 后置条件 | 返回符合该参数的项目的的详细信息 |

userdata**接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| userdata.getAllUsers | 语法 | public Iterator<UserPO> getAllUsers() |
| 前置条件 | 无 |
| 后置条件 | 返回所有用户PO的迭代器 |
| userdata.searchVague | 语法 | public ArrayList<UserPO> searchVague (String keyword) |
| 前置条件 | keyword不为null |
| 后置条件 | 根据关键字返回相关用户信息的列表，如果不存在则提示之 |
| userdata.searchPrecise | 语法 | public UserPO searchPrecise(String repo, String user) |
| 前置条件 | Repo, user参数合法 |
| 后置条件 | 返回符合该参数的用户的的详细信息 |

logdata**接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| logdata.loginAnonoymously | 语法 | public void loginAnonymously() |
| 前置条件 | 系统已连接网络 |
| 后置条件 | 赋予使用者游客操作权限 |
| logdata.login | 语法 | public boolean login(String login, String password) |
| 前置条件 | 系统已连接网络且账号密码输入正确 |
| 后置条件 | 赋予使用者用户操作权限 |
| logdata.logout | 语法 | public void logout() |
| 前置条件 | 使用者已登录 |
| 后置条件 | 系统收回权限 |

### 