**NBA信息查询平台**

**项目设计文档**

**（迭代一适用）**

**（V1.0）**

**南京大学软件学院**

**羽见青柠工作组（15）**

金翠 黄涵倩 王宁 (79)严顺宽

**2015年3月11日**

**目录**

更新历史……………………………………………………………… ……………3

[1、引言](#_引言)………………………………………………………………………………3

1.1编制目的……………………………………………………………………3

1.2词汇表………………………………………………………………………3

1.3参考目录……………………………………………………………………3

[2、逻辑视角](#_3、逻辑视角)…………………………………………………………………………3

[3、组合视角](#_4、组合视角)…………………………………………………………………………5

3.1开发包图……………………………………………………………………5

3.2运行时进程…………………………………………………………………7

3.3物理部署……………………………………………………………………7

[4、接口视角](#_5、接口视角)…………………………………………………………………………8

4.1模块的职责…………………………………………………………………8

4.2用户界面层的分解…………………………………………………………10

4.2.1用户界面层模块的职责……………………………………………11

4.2.2用户界面层模块的接口规范………………………………………11

4.2.3用户界面层模块设计原理…………………………………………12

4..3业务逻辑层的分解…………………………………………………………12

4.3.1业务逻辑层模块的职责……………………………………………12

4.3.2业务逻辑层模块的接口规范………………………………………12

4.4数据层的分解………………………………………………………………19

4.4.1数据层模块的职责…………………………………………………21

4.4.2数据层模块的接口规范……………………………………………22

4.5客户端启动模块的分解……………………………………………………27

4.5.1客户端启动模块的职责……………………………………………28

4.5.2客户端启动模块的接口规范………………………………………28

4.5.3客户端启动模块设计原理…………………………………………28

4.6服务器端启动模块的分解………………………………………………29

4.6.1服务器端启动模块的职责………………………………………29

4.6.2服务器端启动模块的接口规范……………………………………29

4.6.3服务器端启动模块设计原理………………………………………29

4.7客户端网络模块的分解……………………………………………………29

4.8服务器端网络模块的分解…………………………………………………30

[5、信息视角](#信息视角)…………………………………………………………………………31

5.1数据持久化对象…………………………………………………………31

5.2序列化文件表……………………………………………………………………33

# 更新历史

|  |  |  |  |
| --- | --- | --- | --- |
| **修改人员** | **日期** | **变更原因** | **版本号** |
| 金翠 | 2015/3/11 | 文档编写 | V1.0 |

# 1、引言

**.1.1 编制目的**

本报告详细完成对NBA信息查询平台的项目设计，达到指导后续软件构造的目的，同时实现和测试人员及用户的沟通。

本报告面向开发人员、测试人员及最终用户而编写，是了解系统的导航。

**1.2 词汇表**

|  |  |  |
| --- | --- | --- |
| **词汇名称** | **词汇含义** | **备注** |
| bl | 业务逻辑（business logic） | 无 |
|  |  |  |

**1.3 参考资料**

骆斌，丁二玉，刘钦.软件工程与计算（卷二）——软件开发的技术基础.北京：机械工程出版社，2012.12;

骆宾，刘嘉，张瑾玉，黄蕾，软件工程与计算(卷三)——团队与软件开发实践，北京：机械工业出版社，2012.12

NBA信息查询平台软件需求规格说明文档。

# 2、逻辑视角

在NBA信息查询平台中，选择了分层体系结构风格，将系统分为3层（展示层、业务逻辑层、数据层）能够很好地展示整个高层抽象。展示层包含GUI页面的实现，业务逻辑层包含业务逻辑处理的实现，数据层负责数据的持久化和访问。分层体系结构的逻辑视角如图1所示。



**图1 参照体系结构风格的包图表达逻辑视角**

# 3、组合视角

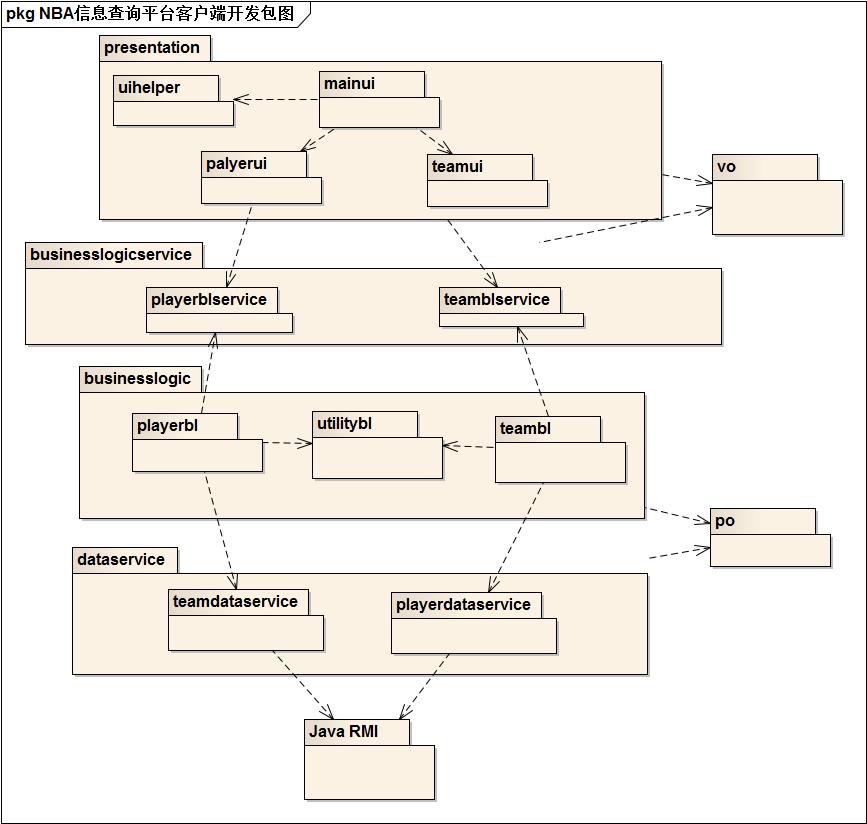
**3.1 开发包图**

NBA信息查询平台的最终开发包设计如表1所示。

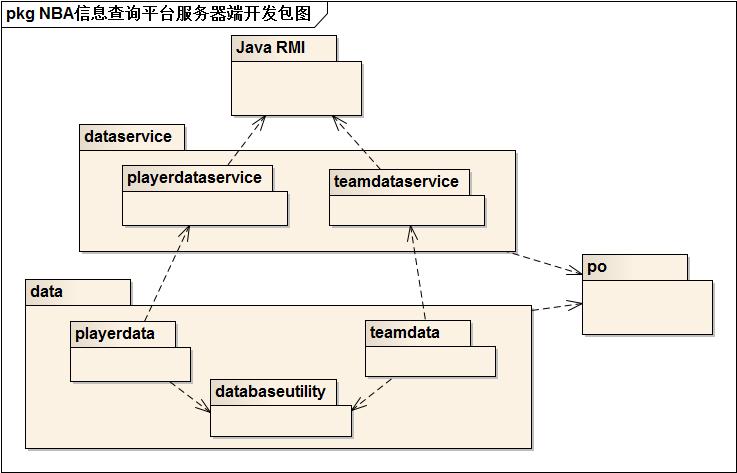
**表1 NBA信息查询平台的最终开发包设计**

|  |  |
| --- | --- |
| **开发（物理）包** | **依赖的其他开发包** |
| **mainui** | **playerui,teamui, vo,uihelper** |
| **playerui** | **playerblservice，界面类库包,vo** |
| **playerblservice** |  |
| **playersbl** | **palyerslservice,playerdataservice,po** |
| **playerdataservice** | **java rmi,po** |
| **salesdata** | **databaseutility,po,playerdataservice** |
| **teamui** | **teamblservice,界面类库包** |
| **teamblservice** |  |
| **teambl** | **teamblservice, teamdataservice,po** |
| **teamdataservice** | **java rmi,po** |
| **teamdata** | **java rmi,po,databaseutility** |
| **vo** |  |
| **po** |  |
| **utilitybl** |  |
| **uihelper** |  |
| **界面类库包** |  |
| **java rmi** |  |
| **databaseutility** | **JDBC,mysql** |

NBA信息查询平台客户端开发包图如图2所示，服务器端开发包图如图3所示。



**图3客户器端开发包图**

****

**图4服务器端开发包图**

**3.2 运行时进程**

在NBA信息查询平台中，会有多个客户端进程和一个服务器端进程，其进程图如图4所示。结合部署图，客户端进程是在客户端机器上运行，服务器端进程在服务器端机器上运行。

****

**图4 进程图**

**3.3 物理部署**

NBA信息查询平台中客户端构件是放在客户端机器上，服务器端构件是放在服务器端机器上。在客户端节点上，还要部署RMIStub构件。由于Java RMI构件属于JDK6.0的一部分。所以，在系统JDK环境已经设置好的情况下，不需要再独立部署。部署图如图5所示。



**图5 部署图**

# 4、接口视角

**4.1 模块的职责**

客户端模块和服务器模块视图分别如图6和图7所示。客户端各层和服务器端各层的职责分别如表2和表3所示。



**图6 客户端模块视图**



**图7 服务器端模块视图**

**表2 客户端各层的职责**

|  |  |
| --- | --- |
| **层** | **职责** |
| 启动模块 | 负责初始化网络通信机制，启动用户界面 |
| 用户界面层 | 基于窗口的NBA信息查询平台客户端用户界面 |
| 业务逻辑层 | 对于用户界面的输入进行响应并进行业务处理逻辑 |
| 客户端网络模块 | 利用Java RMI机制查找RMI服务 |

**表3 服务器端各层的职责**

|  |  |
| --- | --- |
| **层** | **职责** |
| 启动模块 | 负责初始化网络通信机制，启动用户界面 |
| 数据层 | 负责数据库的维护，数据的持久化及数据访问接口 |
| 服务器端网络模块 | 利用Java RMI机制开启RMI服务，注册RMI服务 |

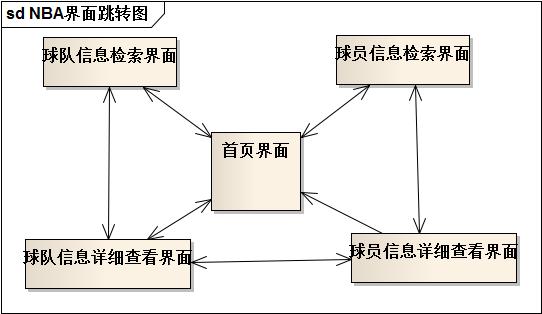
每一层只是使用下方直接接触的层。层与层之间仅仅是通过接口的调用来完成的。层之间调用的接口如表4所示。

**表4 层之间调用的接口**

|  |  |  |
| --- | --- | --- |
| **接口** | **服务调用方** | **服务提供方** |
| TeamBLService  PlayerBLService | 客户端展示层 | 客户端业务逻辑层 |
| TeamDataService  PlayerDataService | 客户端业务逻辑层 | 服务器端数据层 |

**4.2 用户界面层的分解**

根据迭代一需求，系统存在5个用户界面：首页界面、球员个人信息详细查看界面、球队信息详细查看界面、球员信息搜索界面(卡片模式,列表模式，对比模式)、球队信息搜索界面(卡片模式，列表模式，对比模式)。界面跳转如图8所示。



**图8 用户界面跳转**

服务器端和客户端的用户界面设计接口是一致的，只是具体的页面不一样。用户界面类如图9所示。



**图9 用户界面类**

**4.2.1用户界面模块的职责**

如表5所示为用户界面层模块的职责

**表5 用户界面层模块的职责**

|  |  |
| --- | --- |
| **模块** | **职责** |
| Mainui | 主界面框架，负责界面的显示和界面跳转 |
| UIHelper | 界面辅助模块，负责窗体样式控制 |

**===============以下部分copy自进销存 ，尚未修改===========**

**4.2.2用户界面模块的接口规范**

用户界面层模块的接口规范如表6所示。

**表6 用户界面层模块的接口规范**

|  |  |  |
| --- | --- | --- |
| Mainui | 语法 | main (args:String[]) |
| 前置条件 | 无 |
| 后置条件 | 显示startPanel和MainFramel |

用户界面层需要的服务接口如表7所示。

**表7用户界面层模块需要的服务接口**

|  |  |
| --- | --- |
| **服务名** | **服务** |
| businesslogicservice.UserBLService | 登录界面的业务逻辑（身份验证）接口  用户的增删改查的业务逻辑接口 |
| businesslogicservice.ReceiptBLService | 查看界面和单据审批的业务逻辑接口 |
| businesslogicservice.SalesBLService.viewInfoService | 查看销售各类单据的业务逻辑接口(主为其他模块调用) |

**5.2.3用户界面模块设计原理**

用户界面利用Java的Swing和AWT库来实现。

**5.3业务逻辑层的分解**

业务逻辑层包括多个针对界面的业务逻辑处理对象。例如，userbl模块负责处理登录界面的业务逻辑，promotionbl负责处理选择与促销策略有关的界面的业务逻辑。业务逻

辑层的设计如图11所示。

****

**图11 业务逻辑层的设计**

**5.3.1业务逻辑层模块的职责**

业务逻辑层模块的职责如表8所示

**表8 业务逻辑层模块的职责**

|  |  |
| --- | --- |
| **模块** | **职责** |
| **memberbl** | 负责实现对应于会员增删改查所需要的服务 |
| **salesbl** | 负责实现所有关于进货，进货退货，销售，销售退货，查看销售明细界面所需要的服务 |
| **userbl** | 负责实现对应于用户登录，管理员增删改查用户界面所需要的服务 |
| **stockbl** | 负责实现所有关于商品及其分类管理,库存报溢,报损，报警及库存赠送界面所需要的服务 |
| **financebl** | 负责实现所有对应于创建收付款单，现金费用单，账户管理，期初建账，查看经营情况表所需要的服务 |
| **Receiptbl** | 负责实现所有单据的审批，修改，红冲，查看经营历程的各项单据业务所需的服务 |
| **promotionbl** | 负责实现对应于促销策略制定，修改，发布，查看界面所需的服务 |

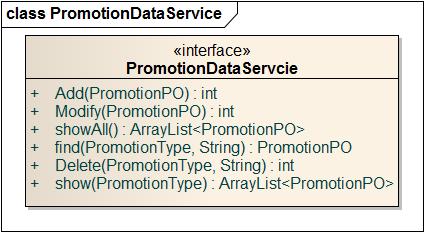
**5.3.2业务逻辑层模块的接口规范**

**表9 userbl模块的接口规范**

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| User.login | 语法 | public int login(String ID,String password) | |
| 前置条件 | password符合输入规则 | |
| 后置条件 | 查找是否存在相同的User，根据输入的password返回登录验证的结果 | |
| User.addUser | 语法 | public int addUser(UserVO vo) | |
| 前置条件 | User经过验证有添加新用户的权利，输入了有效的新用户信息 | |
| 后置条件 | 向系统中输入新增的用户账户信息 | |
| User.deleteUser | 语法 | public int deleteUser(UserVO vo) | |
| 前置条件 | User经过验证有删除用户的权利，输入了有效的待删除用户信息 | |
| 后置条件 | 系统查找对应的用户账户信息，并删除该用户对象 | |
| User.modifyUser | 语法 | public int modifyUser(UserVO vo) | |
| 前置条件 | User经过验证有修改用户的权利，输入了有效的用户修改信息 | |
| 后置条件 | 系统根据输入的用户信息和用户编号更新用户信息 | |
| User.showUser | 语法 | public UserPO showUser(String UserID) | |
| 前置条件 | User经过验证有查看所有用户的权利，输入了有效的用户ID信息 | |
| 后置条件 | 根据输入的ID查找用户，并返回用户信息 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| FileFactory.getUserData | | | 得到User文件的引用 |
| UserDataService.add(UserPO po) | | | 在文件记录中插入UserPO对象 |
| UserDataService.check(UserPO po) | | | 得到根据输入的ID和password验证的结果 |
| UserDataService.delete(UserPO po) | | | 在文件中删除一个po对象 |
| userDataService.modify(UserPO po) | | | 在文件中更新一个po对象 |
| Info.showUserInfo(long ID) | | | 得到根据输入的ID返回对应的po对象 |

**5.4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。包括对于持久化数据的增、删、改、查。这些服务由DataService接口提供。在本系统中我们采用序列化文件的形式保存持久化数据。具体如图12—17所示。



**图12 数据层模块（PromotionDataService）的描述**

**5.4.1数据层模块的职责**

数据层模块的职责如表16－22所示。

**表16 数据层模块（UserDataService）的职责**

|  |  |
| --- | --- |
| **模块** | **职责** |
| UserDataService | 持久化外部文件的接口，提供集体载入、集体保存、增、删、改、查服务 |

**5.4.2数据层模块的接口规范**

数据库模块的接口规范如表23－29所示

**表23 数据层模块（UserDataService）的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| UserDataService.add | 语法 | public int add(UserPO po) |
| 前置条件 | 同样ID的po在文件中不存在 |
| 后置条件 | 在外部文件中增加一个po记录 |
| UserDataService.check | 语法 | public boolean check(UserPO po) |
| 前置条件 | 无 |
| 后置条件 | 返回用户名和密码是否匹配 |
| UserDataService.delete | 语法 | public int delete(UserPO po) |
| 前置条件 | 在外部文件中存在某一个ID 的po需要删除 |
| 后置条件 | 删除一个po对象 |
| UserDataService.modify | 语法 | public int modify(UserPO po) |
| 前置条件 | 在外部文件中存在某一个ID 的po需要更新 |
| 后置条件 | 更新一个po对象 |
| UserDataService.showUserInfo | 语法 | public UserPO showUserInfo (String ID) |
| 前置条件 | 无 |
| 后置条件 | 按ID进行查找返回相应的UserPO的结果 |

**5.5客户端启动模块的分解**

客户端启动模块逻辑上是一个横跨客户端各层的模块，为客户端各层提供初始化启动服务。客户端启动模块主要包含主程序类ClientStarter。主程序类是整个程序的入口，负责整个程序的展示层初始化、业务逻辑层初始化和网络初始化。其结构如图18所示。



**图18客户端启动模块结构图**

**5.5.1 客户端启动模块的职责**

如表30所示为客户端启动模块的职责。

**表30 客户端启动模块的职责**

|  |  |
| --- | --- |
| **模块** | **职责** |
| ClientStarter | 程序入口。负责展示层初始化、业务逻辑层初始化和网络初始化。 |

**5.5.2客户端启动模块的接口规范**

客户端启动模块的接口规范如表31所示。

**表31 客户端启动模块的接口规范**

|  |  |  |
| --- | --- | --- |
| ClientStarter | 语法 | main（args:String[]） |
| 前置条件 | 无 |
| 后置条件 | 根据配置初始化好相应的网络连接并进行展示层和业务逻辑层初始化。 |

客户端启动模块需要的服务接口如表32所示。

**表32 客户端启动模块的服务接口**

|  |  |
| --- | --- |
| **服务名** | **服务** |
| java.rmi.LocateRegistry.createRegistry() | 启动RMI注册服务 |
| java.rmi.Naming.rebind() | 注册RMI服务 |

**5.5.3客户端启动模块设计原理**

启动模块是一个独立于分层体系之外的模块，可以直接访问各层的接口，完成系统启动的需求。

**5.6服务器端启动模块的分解**

服务器端启动模块逻辑上是一个横跨服务器端各层的模块，为服务器端各层提供初始化启动服务。服务器端启动模块主要包含主程序类SeverStarter。主程序类是整个程序的入口，负责整个程序的网络初始化和外部文件初始化。其结构如图19所示。



**图19 服务器端启动模块结构图**

**5.6.1 服务器端启动模块的职责**

如表33所示为服务器端启动模块的职责。

**表33 服务器端启动模块的职责**

|  |  |
| --- | --- |
| **模块** | **职责** |
| ServerStarter | 程序入口。负责网络初始化和外部文件初始化。 |

**5.6.2 服务器端启动模块的接口规范**

服务器端启动模块的接口规范如表34所示。

**表34服务器端启动模块的接口规范**

|  |  |  |
| --- | --- | --- |
| ServerStarter | 语法 | main（args:String[]） |
| 前置条件 | 无 |
| 后置条件 | 根据配置初始化好相应的网络连接和进行序列化文件初始化。 |

服务器端启动模块需要的服务接口如表35所示。

**表35服务器端启动模块的服务接口**

|  |  |
| --- | --- |
| **服务名** | **服务** |
| java.rmi.LocateRegistry.createRegistry() | 启动RMI注册服务 |
| java.rmi.Naming.rebind() | 注册RMI服务 |

**5.6.3服务器端启动模块设计原理**

启动模块是一个独立于分层体系之外的模块，可以直接访问各层的接口，完成系统启动的需求。

**5.7客户端网络模块的分解**

由于数据是存储在服务器上的，客户端访问数据必须通过网络通信层，具体是由客户端的业务逻辑层和服务器端的数据层通过网络通信层进行通信。如图19所示。

具体网络通信层的实现是通过远程方法调用RMI来实现的。



**图20客户端数据通信层模块的描述**

其实网络通信层的实际运作代码是由RMI API所完成的，在客户端我们只需要部署RMI服务的Stub文件和Java RMI类库即可。如表36所示。

**表36网络通信层模块的职责**

|  |  |
| --- | --- |
| **模块** | **职责** |
| IDataRemoteService\_Stub | 给业务逻辑层提供服务接口 |
| Java RMI 类库 | 实现RMI机制 |

**5.8服务器端网络模块的分解**

由于数据是存储在服务器上的，客户端访问数据必须通过网络通信层，具体是由客户端的业务逻辑层和服务器端的数据层通过网络通信层进行通信。

具体服务器端模块中只需要包含Java RMI类库即可。但是其数据层实现RMI的类需要符合以下条件：

****

**图21 服务器数据层实现RMI的类的描述**

DatabaseService 是被业务逻辑层调用的接口。为了实现RMI， DatabaseService需要继承Remote接口，而数据层中的实现这个接口的类SerService都需要继承UnicastRemoteObject类。

以下是服务器网络模块的RMI详细的规范：

**表37服务器网络模块的RMI接口规范**

|  |  |
| --- | --- |
| **类/接口** | **规范** |
| 远程服务接口 | extends Remote，方法 throws RemoteException |
| 远程服务的实现类 | extendsUnicastRemoteObject |

**表38服务器网络模块的RMI接口返回值的规范**

|  |  |
| --- | --- |
| **类/接口** | **规范** |
| 作为远程引用传递 | 该类需要extends Remote，方法 throws RemoteException |
| 通过序列化文件进行复制 | Implements SerFile |

**6、信息视角**

**6.1 数据持久化对象**

系统的PO类就是对应的相应的实体类，在此只做简单的介绍。

* UserPO类包含用户的用户名、密码属性。

持久化用户对象UserPO的定义如图21所示。

public class UserPO{

String id;

String name;

String password;

UserJob job;

double grade;

public UserPO(String I, String n, String p,UserRole r){

id = i;

name = n;

password = p;

role = r;

}

public String getName(){

return name;

}

public String getID(){

return id;

}

public String getPassword(){

return password;

}

Public UserRole getRole(){

Return role;

}

}

**图持久化用户对象UserPO的定义**

**图22 持久化用户对象UserPO的定义**

**6.2数据库文件列表**

数据库中包含User.ser、Stock.ser、Member.ser、sale.ser、salereturn.ser,purchase.ser、purchasereturn.ser,giftcoupon.ser,giftgoods.ser,discount.ser,pack.