# 股票分析系统体系结构设计文档

学 院：南京大学软件学院

成 员：常德隆 武秀峰 曹鸿荣 陈进

完成日期：2017 年4月20日

[股票分析系统体系结构设计文档 1](#_Toc8484)

[1、引言 3](#_Toc25646)

[1.1 编制目的 3](#_Toc28955)

[1.2 词汇表 3](#_Toc7334)

[1.3 参考资料 3](#_Toc26050)

[2、产品描述 3](#_Toc7973)

[3、逻辑视角 3](#_Toc3331)

[4、组合视角 5](#_Toc11461)

[4.1 开发包图 5](#_Toc11038)

[4.2 运行时进程 9](#_Toc2597)

[4.3 物理部署 10](#_Toc17935)

[5、接口视角 10](#_Toc4544)

[5.1 模块的职责 10](#_Toc20748)

[5.2 用户界面层的分解 13](#_Toc2558)

[5.3 业务逻辑层的分解 16](#_Toc8412)

[5.4 数据层的分解 22](#_Toc25749)

[6、信息视角 29](#_Toc22563)

[6.1 数据持久化对象 29](#_Toc31291)

[6.2 Csv持久化格式 29](#_Toc13016)

[6.3数据库表 30](#_Toc11459)

## 1、引言

### 1.1 编制目的

在已经提供的 05 年 2 月到 14 年 4 月 791 只股票 1048575 条 A 股信息数据以及未来可能添加新的数据的基础上，通过对数据的分析处理，实现使用量化交易算法的股票回溯和演算系统，用以深入分析股票行情，模拟和比较量化模型。

### 1.2 词汇表

无

### 1.3 参考资料

无

## 2、产品描述

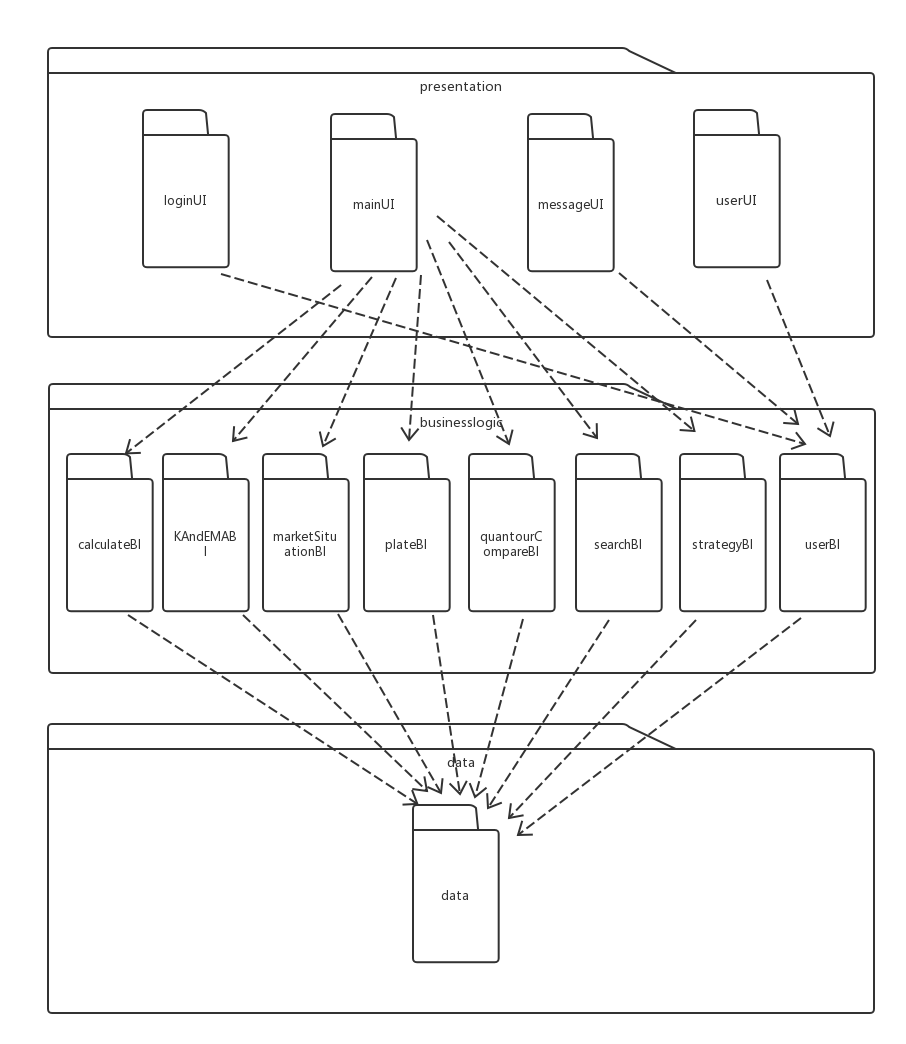
参考股票分析系统用例文档和股票分析系统软件需求规格说明中对产品的概括描述。

## 3、逻辑视角

股票分析系统中，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)能够很好地示意整个高层抽象。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和逻辑设计方案如图1和图2所示。



图一 参照体系结构风格的包图表达逻辑视角



图二 软件体系结构逻辑设计方案

## 4、组合视角

### 4.1 开发包图

表4.1-1 股票分析系统的最终开发包设计

|  |  |
| --- | --- |
| 开发（物理）包 | 依赖的其他开发包 |
| mainFrame | kChartAndEMAChartFrame,QuantourCompareFrame,marketSituationFrame,VO |
| kChartAndEMAChartFrame | kChartAndEMAChartFrameBlSer,界面类库包,VO |
| quantourCompareFrame | QuantourCompareFrameblSer,界面类库包,VO |
| marketSituationFrame | marketSituationFrameBlSer,界面类库包,VO |
| KAndEMABlSer |  |
| KAndEMABl | KAndEMABlSer ,FindStockInfoDataSer, calculateBl ,PO |
| QuantourCompareBlSer |  |
| QuantourCompareBl | QuantourCompareBlSer ,FindStockInfoDataSer ,calculateBl,PO |
| MarketSituationBlSer |  |
| marketSituationBl | MarketSituationBlSer,FindStockInfoDataSer, calculateBl ,PO |
| SearchBlSer |  |
| searchBl | SearchBlSer ,FindStockInfoDataSer,PO |
| StrategyBlSer |  |
| strategyBl | StrategyBlSer ,StrategyDataSer, calculateBl ,PO |
| UserBlSer |  |
| userBl | UserBlSer ,UserDataSer,PO |
| PlateBlSer |  |
| plateBl | PlateBlSer,FindPlateInfoDataSer |
| calculateBl |  |
| FindStockInfoDataSer | Java RMI,po, databaseutility |
| FindPlateInfoDataSer | Java RMI,po, databaseutility |
| StrategyDataSer | Java RMI,po, |
| UserDataSer | Java RMI,po,databaseutility |
| VO |  |
| PO |  |
| utilitybl |  |
| databaseutility | java.JDBC |
| 界面类库包 |  |
| JAVA RMI |  |

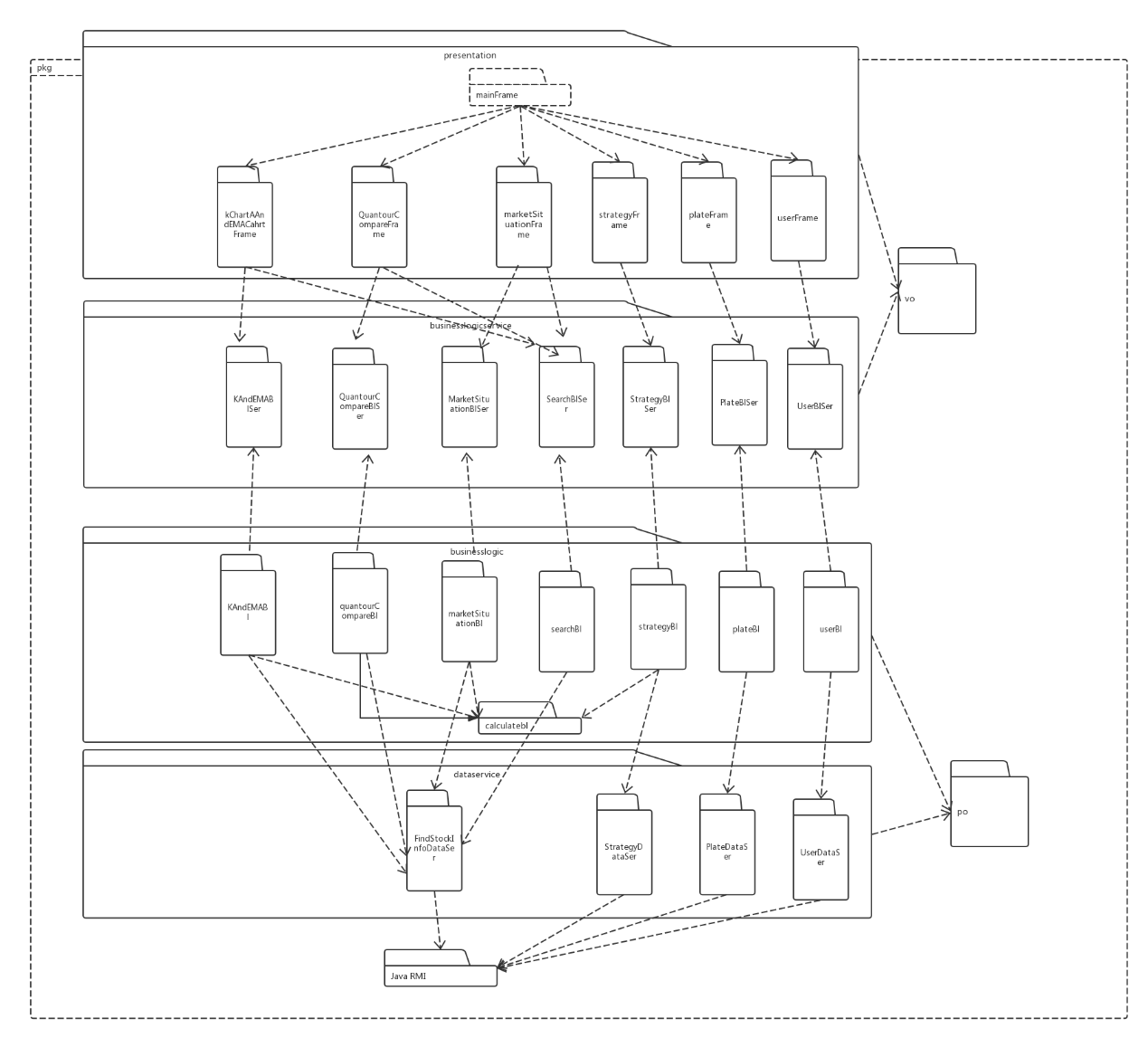


图4.1-1 股票分析系统客户端开发包图

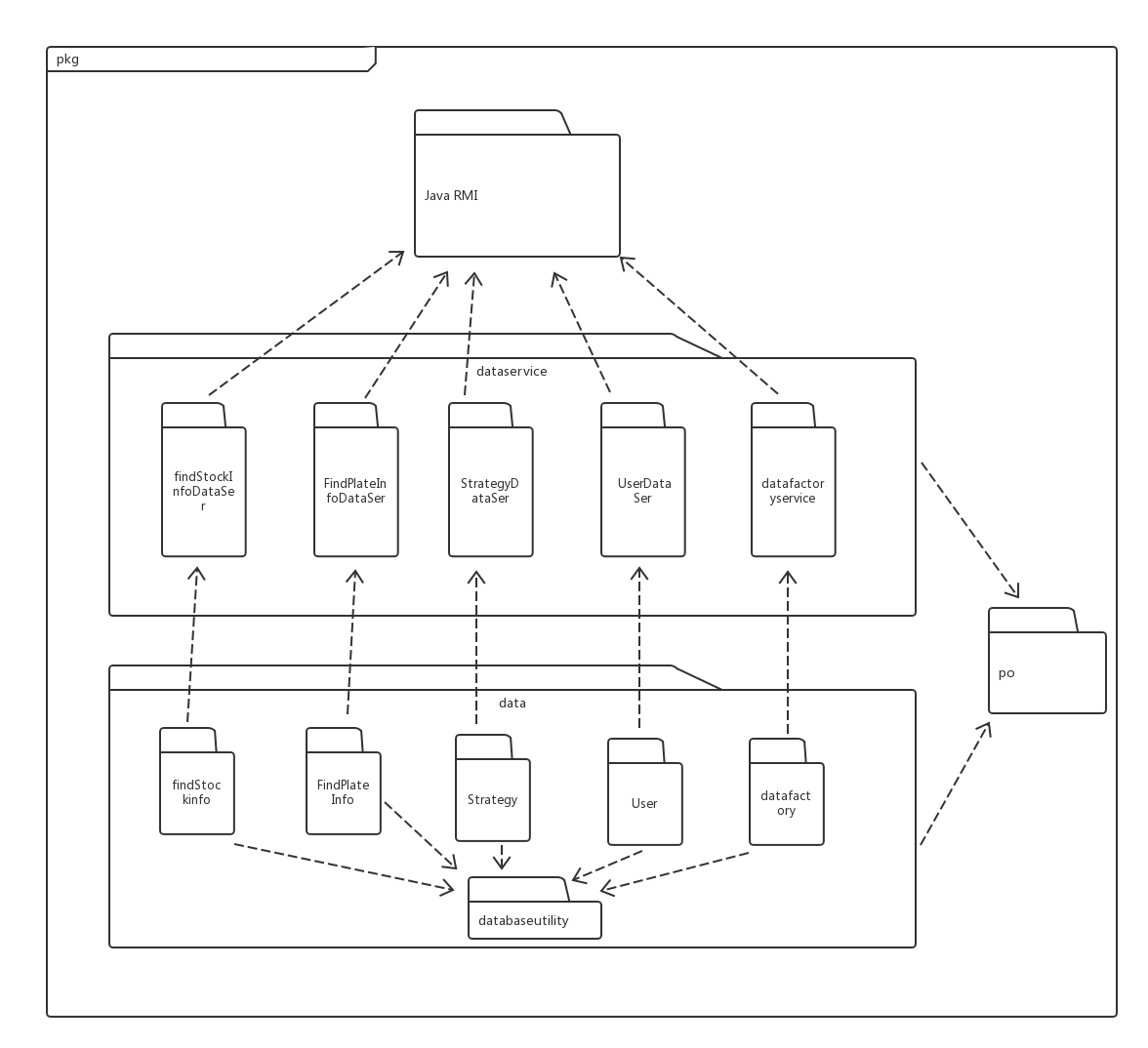


图4.1-2 股票分析系统服务器端开发包图

### 4.2 运行时进程

在股票分析系统中，会有多个客户端进程和一个服务器端进程，其进程图如图4.2所示。结合部署图，客户端进程实在客户端机器上运行，服务器端进程是在服务器端机器上运行。

图4.2 进程图

### 4.3 物理部署

股票分析系统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JavaRMI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4.3所示。



4.3 部署图

## 5、接口视角

### 5.1 模块的职责

客户端模块和服务器端模块视图分别如图5.1-1和图5.1-2所示。客户端各层和服务端各层的职责分别如表5.1-1和表5.1-2所示。



图5.1-1 客户端模块视图



图5.1-2 服务器端模块视图

表5.1-1 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 | 基于窗口的股票分析系统客户端用户界面。 |
| 业务逻辑层 | 对于用户界面的输入响应和业务处理逻辑。 |
| 客户端网络模块 | 利用Java RMI机制查找RMI服务,检测网络连接状态，进行断线重连 |

表5.1-2 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5.1-3所示。

|  |  |  |
| --- | --- | --- |
| 接口 | 服务调用方 | 服务提供方 |
| KAndEMABlSer  QuantourCompareBlSer  MarketSituationBlSer  SearchBlSer  StrategyBlSer  UserBlSer | 客户端展示层 | 客户端业务逻辑层 |
| FindStockInfoDataSer  FindPlateInfoDataSer  StrategyDataSer  UserDataSer | 客户端业务逻辑层 | 服务器端数据层 |

借用用户管理用例来说明层之间的调用，如图5.1-3所示。每一层之间都是由上层依赖了一个接口（需接口），而下层实现这个接口（供接口）。GetKChartBlSer提供了KChartAndEMAChartFrame界面所需要的所有业务逻辑功能FindStockInfoDataSer提供了对数据库的增、删、改、查等操作。这样的实现就大大降低了层与层之间的耦合。

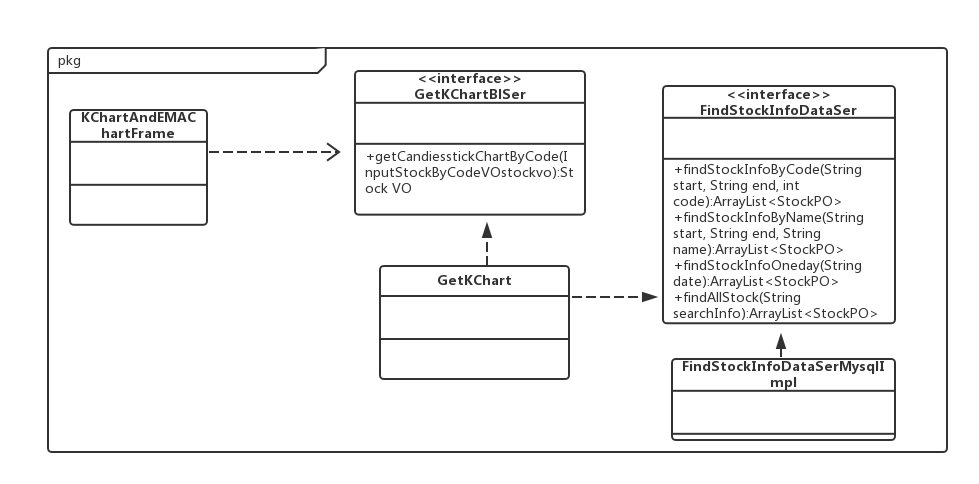


图5.1-3 画k线用例层之间调用的接口

### 5.2 用户界面层的分解

根据需求，系统存在8个用户界面：主界面包含个股界面、股票比较界面、市场情况界面、板块信息界面、投资策略界面、自选股界面另外还有登录界面及个人信息界面。

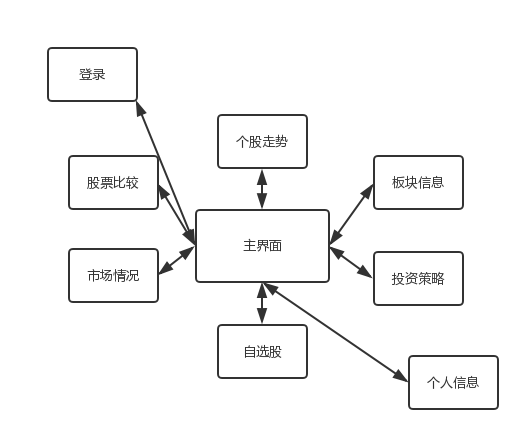


图5.2 用户界面跳转

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5.2-2所示。



#### 5.2.1 用户界面层模块的职责

如表5.2.1-1所示为用户界面层模块的职责。

|  |  |
| --- | --- |
| 模块 | 职责 |
| MainFrame | 界面Frame，负责界面的显示和界面的跳转。 |

#### 5.2.2用户界面层模块的接口规范

用户界面层模块的接口规范如表5.2.2-1所示。

表5.2.2-1 用户界面层模块的接口规范

|  |  |  |
| --- | --- | --- |
| MainFrame | 语法 | Init(args:String[]) |
| 前置条件 | 无 |
| 后置条件 | 显示主Frame |
| UserFrame | 语法 | Init(args:String[]) |
| 前置条件 | 已登录的情况下点击用户按钮 |
| 后置条件 | 显示个人信息界面 |
| LoginFrame | 语法 | Init(args:String[]) |
| 前置条件 | 未登录情况下点击用户按钮 |
| 后置条件 | 显示登录界面 |

用户界面层需要的服务接口如表5.2.2-2所示。

表5.2.2-2 用户界面层模块需要的服务接口

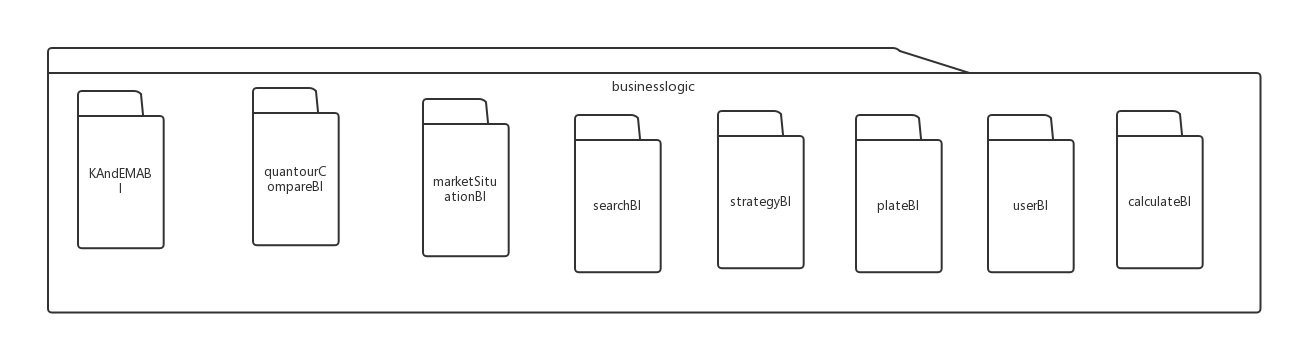
|  |  |
| --- | --- |
| 服务名 | 服务 |
| KAndEMABlser. getDataByCode | 通过股票代码获取K线图所需股票信息 |
| KAndEMABlser. getDataByName | 通过股票名称获取K线图所需股票信息 |
| KAndEMABlser. getEMAByCode | 通过股票代码获取均线图的信息 |
| KAndEMABlser. getEMAByName | 通过股票名称获取均线图的信息 |
| KAndEMABlser. getPieDataByCode | 通过股票代码获取个股饼状图信息 |
| KAndEMABlser. getPieDataByName | 通过股票名称获取个股饼状图信息 |
| MarketSituationBlSer.getMarketSituation | 获取市场整体信息 |
| MarketSituationBlSer.getIncreaseList | 获取涨幅榜信息 |
| MarketSituationBlSer.getDecreaseList | 获取跌幅榜信息 |
| MarketSituationBlSer.getVolumeList | 获取成交量排行榜信息 |
| PlateBlSer.getTotalPlateInfo | 获取板块整体信息 |
| PlateBlSer.getOnePlateInfo | 获取某个板块的具体信息 |
| QuantourCompareBlSer.quantourTableCompareByCode | 通过股票代码获取股票比较的表格信息 |
| QuantourCompareBlSer.quantourChartCompareByCode | 通过股票代码获取股票比较图表的信息 |
| QuantourCompareBlSer..quantourTableCompareByName | 通过股票名称获取股票比较表的信息 |
| QuantourCompareBlSer.quantourChartCompareByName | 通过股票名称获取股票比较图的信息 |
| StrategyBlSer.getStrategy | 获取策略信息 |
| StrategyBlSer.getRelationGraphData | 获取策略超额收益率与不同形成期/持有期的关系图信息 |
| StrategyBlSer.getYieldRateDistribution | 获取策略收益率直方图的数据 |
| UserBlSer.register | 用户登录的服务 |
| UserBlSer.login | 用户注册的服务 |
| UserBlSer.changeUserInfo | 修改用户信息的服务 |
| UserBlSer.getUserInfo | 获取用户信息的服务 |
| UserBlSer.getSelfSelectStock | 获取自选股信息的服务 |
| UserBlSer.addSelfSelectStock | 添加自选股的服务 |
| UserBlSer.deleteSelfSelectStock | 删除自选股的服务 |
| UserBlSer.isMySelfSelectStock | 判断某支股票是否是当前用户的自选股 |

#### 5.2.3 用户界面模块设计原理

用户界面利用JavaFX来实现。

### 5.3 业务逻辑层的分解

务逻辑层包括多个针对界面的业务逻辑处理对象。例如，QuantourCompare对象负责处理股票比较界面的业务逻辑。业务逻辑层的设计如图5.3-1所示。



#### 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5.3.1-1所示。

表5.3.1-1 业务逻辑层模块的职责

|  |  |
| --- | --- |
| 模块 | 职责 |
| KAndEMABl | 负责K线图、均线图、饼状图相关的服务 |
| marketSituationBl | 负责股票比较相关的服务 |
| quantourCompareBl | 负责股票市场相关的服务 |
| searchBl | 负责搜索相关的服务 |
| strategyBl | 负责策略相关的服务 |
| plateBl | 负责板块相关的服务 |
| userBl | 负责用户相关的服务 |
| calculateBl | 负责计算相关的服务 |

#### 5.3.2 业务逻辑层模块的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| KAndEMABlController. getDataByCode | 语法 | | public ArrayList<StockVO> getDataByCode(InputStockByCodeVO stockvo); |
| 前置条件 | | 已知股票编码 |
| 后置条件 | | 返回股票K线图数据 |
| KAndEMABlController. getDataByName | 语法 | | public ArrayList<StockVO> getDataByName(InputStockByNameVO stockvo)； |
| 前置条件 | | 已知股票名字 |
| 后置条件 | | 返回股票K线图数据 |
| KAndEMABlController. getEMAByCode | 语法 | | public EMAVO getEMAByCode(InputStockByCodeVO stockvo, String numOfEMA); |
| 前置条件 | | 已知股票编码和几日均线 |
| 后置条件 | | 返回股票均线 |
| KAndEMABlController. getEMAByName | 语法 | | public EMAVO getEMAByName(InputStockByNameVO stockvo, String numOfEMA); |
| 前置条件 | | 已知股票名字和几日均线 |
| 后置条件 | | 返回股票均线 |
| KAndEMABlController. getPieDataByCode | 语法 | | public StockVO public PieVO getPieDataByCode(InputStockByCodeVO stockvo); |
| 前置条件 | | 已知股票编码 |
| 后置条件 | | 返回股票饼状图数据 |
| KAndEMABlController. getPieDataByName | 语法 | | public PieVO getPieDataByName(InputStockByNameVO stockvo); |
| 前置条件 | | 已知股票名字 |
| 后置条件 | | 返回股票饼状图数据 |
| 需要的服务（需接口） | | | |
| 服务名 | | 服务 | |
| KAndEMA. getDataByCode | | 输入开始日期，结束日期和股票编号,返回符合相关要求的K线图需要的数据 | |
| KAndEMA. getDataByName | | 输入开始日期，结束日期和股票名称，返回符合相关要求的K线图需要的数据 | |
| KAndEMA. getEMAByCode | | 输入开始日期，结束日期和股票代码，返回符合相关要求的均线图需要的数据 | |
| KAndEMA. getEMAByName | | 输入开始日期，结束日期和股票名称，返回符合相关要求的均线图需要的数据 | |
| Pie. getPieDataByCode | | 用户输入开始日期，结束日期和股票编号,返回饼状图需要的数据 | |
| Pie. getPieDataByName | | 用户输入开始日期，结束日期和股票名称,返回饼状图需要的数据 | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 提供的服务（供接口） | | | | |
| KAndEMA. getDataByCode | 语法 | | | public ArrayList<StockVO> getDataByCode(InputStockByCodeVO stockvo); |
| 前置条件 | | | 已知股票编码 |
| 后置条件 | | | 返回股票K线图数据 |
| KAndEMA. getDataByName | 语法 | | | public ArrayList<StockVO> getDataByName(InputStockByNameVO stockvo)； |
| 前置条件 | | | 已知股票名字 |
| 后置条件 | | | 返回股票K线图数据 |
| KAndEMA. getEMAByCode | 语法 | | | public EMAVO getEMAByCode(InputStockByCodeVO stockvo, String numOfEMA); |
| 前置条件 | | | 已知股票编码和几日均线 |
| 后置条件 | | | 返回股票均线 |
| KAndEMA. getEMAByName | 语法 | | | public EMAVO getEMAByName(InputStockByNameVO stockvo, String numOfEMA); |
| 前置条件 | | | 已知股票名字和几日均线 |
| 后置条件 | | | 返回股票均线 |
| Pie. getPieDataByCode | 语法 | | | public StockVO public PieVO getPieDataByCode(InputStockByCodeVO stockvo); |
| 前置条件 | | | 已知股票编码 |
| 后置条件 | | | 返回股票饼状图数据 |
| Pie. getPieDataByName | 语法 | | | public PieVO getPieDataByName(InputStockByNameVO stockvo); |
| 前置条件 | | | 已知股票名字 |
| 后置条件 | | | 返回股票饼状图数据 |
| 需要的服务（需接口） | | | | |
| 服务名 | | 服务 | | |
| FindStockInfo. findStockInfoByCode | | 查找股票 | | |
| FindStockInfo. findStockInfoByName | | 查找股票 | | |
| FindStockInfo. findStockInfoOneday | | 查找股票 | | |
| FindStockInfo. findAllStock | | 查找股票 | | |
| 提供的服务（供接口） | | | | |
| MarketSituationController. getMarketSituation | 语法 | | | public MarketSituationVO getMarketSituation(String date) |
| 前置条件 | | | 已知日期 |
| 后置条件 | | | 返回股票市场情况 |
| MarketSituationController. getIncreaseList | 语法 | | | public ArrayList<StockVO> getIncreaseList(String date)； |
| 前置条件 | | | 已知日期 |
| 后置条件 | | | 返回股票市场涨停数 |
| MarketSituationController. getDecreaseList | 语法 | | | public ArrayList<StockVO> getDecreaseList(String date) |
| 前置条件 | | | 已知日期 |
| 后置条件 | | | 返回股票市场跌停数 |
| MarketSituationController. getVolumeList | 语法 | | | public ArrayList<StockVO> getVolumeList(String date) |
| 前置条件 | | | 已知日期 |
| 后置条件 | | | 返回股票市场交易量 |
| 需要的服务（需接口） | | | | |
| 服务名 | | | 服务 | |
| MarketSituation. getMarketSituation | | | 已知日期，返回股票市场情况 | |
| MarketSituation. getIncreaseList | | | 已知日期，返回股票市场涨幅榜 | |
| MarketSituation. getDecreaseList | | | 已知日期，返回股票市场跌幅榜 | |
| MarketSituation. getVolumeList | | | 已知日期，返回股票市场交易量排行榜 | |

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| MarketSituation. getMarketSituation | 语法 | public MarketSituationVO getMarketSituation(String date) | |
| 前置条件 | 已知日期 | |
| 后置条件 | 返回股票市场情况 | |
| MarketSituation. getIncreaseList | 语法 | public ArrayList<StockVO> getIncreaseList(String date)； | |
| 前置条件 | 已知日期 | |
| 后置条件 | 返回股票市场涨幅榜 | |
| MarketSituation. getDecreaseList | 语法 | public ArrayList<StockVO> getDecreaseList(String date) | |
| 前置条件 | 已知日期 | |
| 后置条件 | 返回股票市场跌幅榜 | |
| MarketSituation. getVolumeList | 语法 | public ArrayList<StockVO> getVolumeList(String date) | |
| 前置条件 | 已知日期 | |
| 后置条件 | 返回股票市场交易量排行榜 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| FindStockInfo. findStockInfoByCode | | | 查找股票 |
| FindStockInfo. findStockInfoByName | | | 查找股票 |
| FindStockInfo. findStockInfoOneday | | | 查找股票 |
| FindStockInfo. findAllStock | | | 查找股票 |

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| QuantourCompareController. quantourTableCompareByCode | 语法 | public ArrayList<StockCompareTotalVO> quantourTableCompareByCode(ArrayList<InputStockByCodeVO> stockvo)； | |
| 前置条件 | 已知两个股票编码 | |
| 后置条件 | 返回两股票比较情况 | |
| QuantourCompareController. quantourChartCompareByCode | 语法 | publicArrayList<ArrayList<StockCompareEverydayVO>>quantourChartCompareByCode (ArrayList<InputStockByCodeVO> stockvo)； | |
| 前置条件 | 已知两个股票编码 | |
| 后置条件 | 返回两股票比较情况 | |
| QuantourCompareController. quantourTableCompareByName | 语法 | public ArrayList<StockCompareTotalVO> quantourTableCompareByName(ArrayList<InputStockByNameVO> stockvo)； | |
| 前置条件 | 已知两个股票名字 | |
| 后置条件 | 返回两股票比较情况 | |
| QuantourCompareController. quantourChartCompareByName | 语法 | public ArrayList<StockCompareTotalVO> quantourTableCompareByName(ArrayList<InputStockByNameVO> stockvo) | |
| 前置条件 | 已知两个股票名字 | |
| 后置条件 | 返回两股票比较情况 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| QuantourCompare. quantourTableCompareByCode | | | 已知两个股票编码,返回两股票比较情况表 |
| QuantourCompare. quantourTableCompareByName | | | 已知两个股票名称,返回两股票比较情况表 |
| QuantourCompare. quantourChartCompareByCode | | | 已知两个股票编码,返回两股票比较情况图 |
| QuantourCompare. quantourChartCompareByName | | | 已知两个股票名称,返回两股票比较情况图 |

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| QuantourCompare. quantourTableCompareByCode | 语法 | public ArrayList<StockCompareTotalVO> quantourTableCompareByCode(ArrayList<InputStockByCodeVO> stockvo)； | |
| 前置条件 | 已知两个股票编码 | |
| 后置条件 | 返回两股票比较情况 | |
| QuantourCompare. quantourChartCompareByCode | 语法 | publicArrayList<ArrayList<StockCompareEverydayVO>>quantourChartCompareByCode (ArrayList<InputStockByCodeVO> stockvo)； | |
| 前置条件 | 已知两个股票编码 | |
| 后置条件 | 返回两股票比较情况 | |
| QuantourCompare. quantourTableCompareByName | 语法 | public ArrayList<StockCompareTotalVO> quantourTableCompareByName(ArrayList<InputStockByNameVO> stockvo)； | |
| 前置条件 | 已知两个股票名字 | |
| 后置条件 | 返回两股票比较情况 | |
| QuantourCompare. quantourChartCompareByName | 语法 | public ArrayList<StockCompareTotalVO> quantourTableCompareByName(ArrayList<InputStockByNameVO> stockvo) | |
| 前置条件 | 已知两个股票名字 | |
| 后置条件 | 返回两股票比较情况 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| FindStockInfo. findStockInfoByCode | | | 查找股票 |
| FindStockInfo. findStockInfoByName | | | 查找股票 |
| FindStockInfo. findStockInfoOneday | | | 查找股票 |
| FindStockInfo. findAllStock | | | 查找股票 |

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| SearchController. findAllStock | 语法 | | public ArrayList<StockVO> findAllStock(String searchInfo); |
| 前置条件 | | 已知股票部分信息 |
| 后置条件 | | 返回股票全部信息 |
| 需要的服务（需接口） | | | |
| 服务名 | | 服务 | |
| Search. findAllStock | | 已知股票部分信息，返回股票全部信息 | |

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| Search. findAllStock | 语法 | public ArrayList<StockVO> findAllStock(String searchInfo); | |
| 前置条件 | 已知股票部分信息 | |
| 后置条件 | 返回股票全部信息 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| FindStockInfo. findAllStock | | | 查找股票 |

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| PlateController. getTotalPlateInfo | 语法 | public TotalPlateVO getTotalPlateInfo()； | |
| 前置条件 | 有看板块信息的需要 | |
| 后置条件 | 返回板块整体信息 | |
| PlateController. getTotalPlateInfo | 语法 | public TotalPlateVO getTotalPlateInfo(String date)； | |
| 前置条件 | 已知日期 | |
| 后置条件 | 返回当日板块整体信息 | |
| PlateController.getOnePlateInfo | 语法 | public OnePlateVO getOnePlateInfo(StockPoolBl stockPoolBl) | |
| 前置条件 | 已知板块 | |
| 后置条件 | 返回该板块详细信息 | |
| PlateController.getOnePlateInfo | 语法 | public OnePlateVO getOnePlateInfo(StockPoolBl stockPoolBl, String date) | |
| 前置条件 | 已知板块和日期 | |
| 后置条件 | 返回当日该板块详细信息 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| Plate. getTotalPlateInfo | | | 返回板块整体信息 |
| Plate.getOnePlateInfo | | | 返回某板块内详细信息 |

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| Plate. getTotalPlateInfo | 语法 | public TotalPlateVO getTotalPlateInfo()； | |
| 前置条件 | 有看板块信息的需要 | |
| 后置条件 | 返回板块整体信息 | |
| Plate. getTotalPlateInfo | 语法 | public TotalPlateVO getTotalPlateInfo(String date)； | |
| 前置条件 | 已知日期 | |
| 后置条件 | 返回当日板块整体信息 | |
| Plate.getOnePlateInfo | 语法 | public OnePlateVO getOnePlateInfo(StockPoolBl stockPoolBl) | |
| 前置条件 | 已知板块 | |
| 后置条件 | 返回该板块详细信息 | |
| Plate.getOnePlateInfo | 语法 | public OnePlateVO getOnePlateInfo(StockPoolBl stockPoolBl, String date) | |
| 前置条件 | 已知板块和日期 | |
| 后置条件 | 返回当日该板块详细信息 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| FindPlateInfoDataSer. findTotalPlateInfo | | | 返回板块整体信息 |
| FindPlateInfoDataSer. findOnePlateInfo | | | 返回某板块内详细信息 |

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| UserController. register | 语法 | public boolean register(UserVO userVO) | |
| 前置条件 | 已知用户注册信息 | |
| 后置条件 | 返回是否注册成功 | |
| UserController. login | 语法 | public boolean login(InputLoginVO inputLoginVO)； | |
| 前置条件 | 已知用户登录信息 | |
| 后置条件 | 返回是否登录成功 | |
| UserController.changeUserInfo | 语法 | public boolean changeUserInfo(UserVO userVO) | |
| 前置条件 | 已知用户修改的信息 | |
| 后置条件 | 返回用户是否修改信息成功 | |
| UserController.getUserInfo | 语法 | public UserVO getUserInfo(String account) | |
| 前置条件 | 已知用户账号 | |
| 后置条件 | 返回用户详细信息 | |
| UserController. getSelfSelectStock | 语法 | public SelfSelectStockVO getSelfSelectStock(String account) | |
| 前置条件 | 已知用户账号 | |
| 后置条件 | 返回用户自选股信息 | |
| UserController. addSelfSelectStock | 语法 | public boolean addSelfSelectStock(SelfSelectStockVO selfSelectStockVO) | |
| 前置条件 | 已知用户添加股票 | |
| 后置条件 | 返回是否添加成功 | |
| UserController. addOneSelfSelectStock | 语法 | public boolean addOneSelfSelectStock(String account, String stockCodeOrName) | |
| 前置条件 | 已知用户添加的一支股票 | |
| 后置条件 | 返回是否添加成功 | |
| UserController. deleteSelfSelectStock | 语法 | public boolean deleteSelfSelectStock(SelfSelectStockVO selfSelectStockVO) | |
| 前置条件 | 已知用户删除自选股 | |
| 后置条件 | 返回是否删除成功 | |
| UserController.isMySelfSelectStock | 语法 | public boolean isMySelfSelectStock(String stockIdOrName,String account) | |
| 前置条件 | 已知用户和某股票 | |
| 后置条件 | 返回该股票是否是用户的自选股 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| User. register | | | 返回是否注册成功 |
| User.login | | | 返回是否登录成功 |
| User. changeUserInfo | | | 返回是否修改信息成功 |
| User.getUserInfo | | | 返回获取的用户 |
| User.getSelfSelectStock | | | 返回用户自选股信息 |
| User.addSelfSelectStock | | | 增加用户自选股 |
| User.addOneSelfSelectStock | | | 增加用户一支自选股 |
| User.deleteSelfSelectStock | | | 删除用户自选股 |
| User.isMySelfSelectStock | | | 返回股票是否是用户自选股 |

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| User. register | 语法 | public boolean register(UserVO userVO) | |
| 前置条件 | 已知用户注册信息 | |
| 后置条件 | 返回是否注册成功 | |
| User. login | 语法 | public boolean login(InputLoginVO inputLoginVO)； | |
| 前置条件 | 已知用户登录信息 | |
| 后置条件 | 返回是否登录成功 | |
| User.changeUserInfo | 语法 | public boolean changeUserInfo(UserVO userVO) | |
| 前置条件 | 已知用户修改的信息 | |
| 后置条件 | 返回用户是否修改信息成功 | |
| User.getUserInfo | 语法 | public UserVO getUserInfo(String account) | |
| 前置条件 | 已知用户账号 | |
| 后置条件 | 返回用户详细信息 | |
| User. getSelfSelectStock | 语法 | public SelfSelectStockVO getSelfSelectStock(String account) | |
| 前置条件 | 已知用户账号 | |
| 后置条件 | 返回用户自选股信息 | |
| User. addSelfSelectStock | 语法 | public boolean addSelfSelectStock(SelfSelectStockVO selfSelectStockVO) | |
| 前置条件 | 已知用户添加股票 | |
| 后置条件 | 返回是否添加成功 | |
| User. addOneSelfSelectStock | 语法 | public boolean addOneSelfSelectStock(String account, String stockCodeOrName) | |
| 前置条件 | 已知用户添加的一支股票 | |
| 后置条件 | 返回是否添加成功 | |
| User. deleteSelfSelectStock | 语法 | public boolean deleteSelfSelectStock(SelfSelectStockVO selfSelectStockVO) | |
| 前置条件 | 已知用户删除自选股 | |
| 后置条件 | 返回是否删除成功 | |
| User.isMySelfSelectStock | 语法 | public boolean isMySelfSelectStock(String stockIdOrName,String account) | |
| 前置条件 | 已知用户和某股票 | |
| 后置条件 | 返回该股票是否是用户的自选股 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| UserDataSer. register | | | 返回是否注册成功 |
| UserDataSer.login | | | 返回是否登录成功 |
| UserDataSer. changeUserInfo | | | 返回是否修改信息成功 |
| UserDataSer.getUserInfo | | | 返回获取的用户 |
| UserDataSer.getSelfSelectStock | | | 返回用户自选股信息 |
| UserDataSer.addSelfSelectStock | | | 增加用户自选股 |
| UserDataSer.addOneSelfSelectStock | | | 增加用户一支自选股 |
| UserDataSer.deleteSelfSelectStock | | | 删除用户自选股 |
| UserDataSer.isMySelfSelectStock | | | 返回股票是否是用户自选股 |

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| StrategyController. getStrategy | 语法 | public StrategyVO getStrategy(InputStrategyVO inputStrategyVO) | |
| 前置条件 | 已知用户获取策略输入信息 | |
| 后置条件 | 返回获取的策略信息，包括策略的统计变量、策略的策略和基准的累计收益率比较图的数据 | |
| StrategyController. getRelationGraphData | 语法 | public RelationGraphVO getRelationGraphData(InputStrategyVO inputStrategyVO, int type)； | |
| 前置条件 | 已知用户获取策略输入信息 | |
| 后置条件 | 返回获取的策略超额收益率与不同形成期/持有期的关系图以及策略胜率与不同形成期/持有期的关系图的数据 | |
| StrategyController. getYieldRateDistribution | 语法 | public YieldRateDistributionVO getYieldRateDistribution(InputStrategyVO inputStrategyVO) | |
| 前置条件 | 已知用户获取策略输入信息 | |
| 后置条件 | 返回获取的策略收益率分布直方图的数据 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| StaticalVariable. getStaticalVariable | | | 返回策略的统计变量 |
| RelationGraph. getRelationGraphData | | | 返回策略超额收益率与不同形成期/持有期的关系图以及策略胜率与不同形成期/持有期的关系图的数据 |
| RelationGraph. getOneRelationData | | | 返回关系图的其中一个数据 |
| StrategyGraph. getGraphData | | | 返回策略的策略和基准的累计收益率比较图的数据 |
| StrategyGraph. getStrategyEvaluate | | | 返回策略评估雷达图的数据 |
| YieldRateDistribution. getYieldRateDistribution | | | 返回策略收益率分布直方图的数据 |

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| StaticalVariable. getStaticalVariable | 语法 | public StaticalVariableVO getStaticalVariable(StrategyGraphVO strategyGraphVO,String start, String end) |
| 前置条件 | 已知策略要求信息 |
| 后置条件 | 返回策略的统计变量 |

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| RelationGraph. getRelationGraphData | 语法 | public RelationGraphVO getRelationGraphData(InputStrategyVO inputStrategyVO, int type) | |
| 前置条件 | 已知用户策略要求信息 | |
| 后置条件 | 返回策略超额收益率与不同形成期/持有期的关系图以及策略胜率与不同形成期/持有期的关系图的数据 | |
| RelationGraph. getOneRelationData | 语法 | public ArrayList<Double> getOneRelationData(StrategyType type,ArrayList<StrategyStockPO> strategyStockPOS, int formationPeriod, int holdingPeriod, int valueless)； | |
| 前置条件 | 已知用户策略要求信息 | |
| 后置条件 | 返回关系图的其中一个数据 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| Benchmark.getBenchmarkOneRelationData | | | 返回关系图的其中一个数据 |
| MomentStrategy.getOneRelationData | | | 返回关系图的其中一个数据 |
| AverageStrategy.getOneRelationData | | | 返回关系图的其中一个数据 |
| MAStrategy.getOneRelationData | | | 返回关系图的其中一个数据 |

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| StrategyGraph. getGraphData | 语法 | public StrategyGraphVO getGraphData(InputStrategyVO inputStrategyVO) | |
| 前置条件 | 已知用户获取策略输入信息 | |
| 后置条件 | 返回策略的策略和基准的累计收益率比较图的数据 | |
| StrategyGraph. getStrategyEvaluate | 语法 | StrategyEvaluateVO getStrategyEvaluate(String strategyYearYieldStr,String maxWithdrawStr, String betaStr,String sharpeRatioStr,String alphaStr)； | |
| 前置条件 | 已知部分策略信息 | |
| 后置条件 | 返回策略评估雷达图的数据 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| GetStockInfo.getStrategyStockInfo | | | 返回股票信息 |
| Benchmark.getPlateBenchmarkData | | | 返回策略超额收益率与不同形成期/持有期的关系图以及策略胜率与不同形成期/持有期的关系图的数据 |
| Benchmark.getSelfSelectBenchmarkData | | | 返回关系图的其中一个数据 |
| MomentStrategy.getMomentData | | | 返回策略的策略和基准的累计收益率比较图的数据 |
| AverageStrategy.getAverageData | | | 返回策略的策略和基准的累计收益率比较图的数据 |
| MAStrategy.getMAData | | | 返回策略的策略和基准的累计收益率比较图的数据 |
| Benchmark.getMaxWithdraw | | | 返回策略评估雷达图的数据 |

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| YieldRateDistribution. getYieldRateDistribution | 语法 | public YieldRateDistributionVO getYieldRateDistribution(InputStrategyVO inputStrategyVO) | |
| 前置条件 | 已知用户获取策略输入信息 | |
| 后置条件 | 返回策略收益率分布直方图的数据 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| GetStockInfo.getStrategyStockInfo | | | 返回股票信息 |
| MomentStrategy.getOneRelationData | | | 返回关系图中的一个数据 |
| AverageStrategy.getOneRelationData | | | 返回关系图中的一个数据 |
| MAStrategy.getOneRelationData | | | 返回关系图中的一个数据 |

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| GetStockInfo. getStrategyStockInfo | 语法 | public ArrayList<StrategyStockPO> getStrategyStockInfo | |
| 前置条件 | 已知用户获取策略输入信息 | |
| 后置条件 | 返回策略股票数据 | |
| GetStockInfo. getBenchmarkStockInfo | 语法 | public StrategyStockPO getBenchmarkStockInfo； | |
| 前置条件 | 已知用户获取策略输入信息 | |
| 后置条件 | 返回基准股票数据 | |
| GetStockInfo. getPreviousTradeDate | 语法 | public String getPreviousTradeDate | |
| 前置条件 | 已知当前日期 | |
| 后置条件 | 返回这个交易日的上个交易日 | |
| GetStockInfo. getBeforeSomeTradeDate | 语法 | public String getBeforeSomeTradeDate | |
| 前置条件 | 已知当前日期 | |
| 后置条件 | 返回当前交易日前somedays天的交易日 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| StrategyDataSer.findStrategyStockInfo | | | 返回策略的股票信息 |
| StrategyDataSer.findBenchmarkStockInfo | | | 返回基准的股票信息 |
| StrategyDataSer.getPreviousTradeDate | | | 返回当前交易日前somedays天的交易日 |

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| MomentStrategy. getMomentData | 语法 | public ArrayList<String> getMomentData | |
| 前置条件 | 已知策略股票信息 | |
| 后置条件 | 返回动量策略数据 | |
| MomentStrategy. getMomentData | 语法 | public ArrayList<Double> getOneRelationData； | |
| 前置条件 | 已知策略股票信息 | |
| 后置条件 | 返回关系图中的一个数据 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| Benchmark.getYield | | | 返回投资的股票每日的累计收益率 |

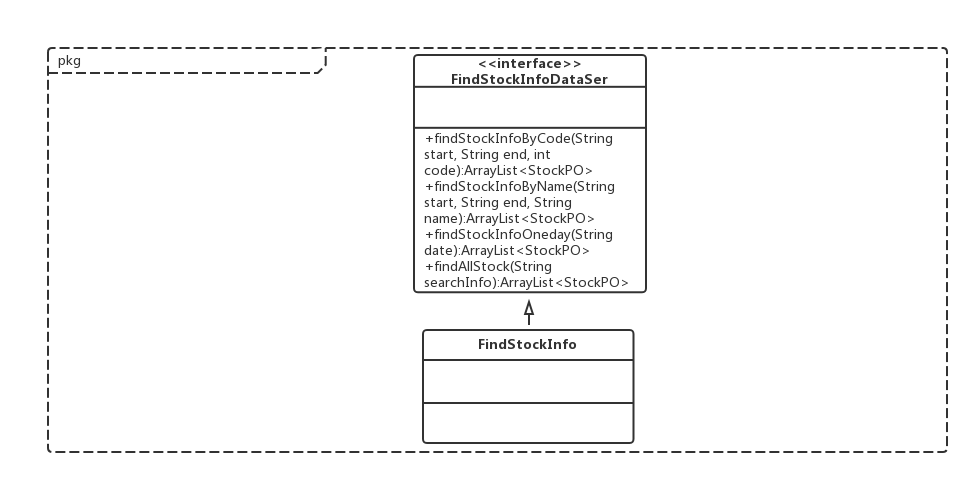
|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| AverageStrategy. getAverageData | 语法 | public ArrayList<String> getAverageData | |
| 前置条件 | 已知策略股票信息 | |
| 后置条件 | 返回均值回归策略数据 | |
| AverageStrategy. getAverageData | 语法 | public ArrayList<Double> getOneRelationData； | |
| 前置条件 | 已知策略股票信息 | |
| 后置条件 | 返回关系图中的一个数据 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| Benchmark.getYield | | | 返回投资的股票每日的累计收益率 |

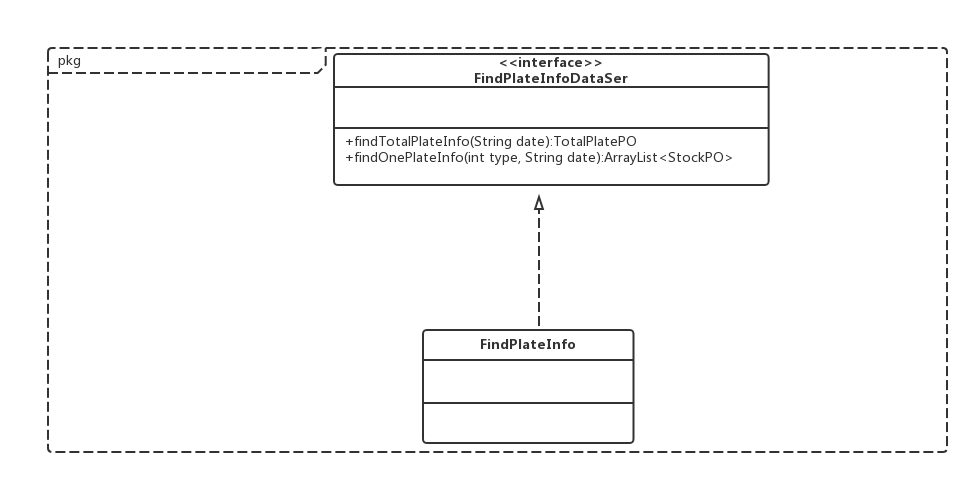
|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| MAStrategy. getMAData | 语法 | public ArrayList<String> getMAData | |
| 前置条件 | 已知策略股票信息 | |
| 后置条件 | 返回MA择时策略数据 | |
| MAStrategy. getMAData | 语法 | public ArrayList<Double> getOneRelationData； | |
| 前置条件 | 已知策略股票信息 | |
| 后置条件 | 返回关系图中的一个数据 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| Benchmark.getYield | | | 返回投资的股票每日的累计收益率 |

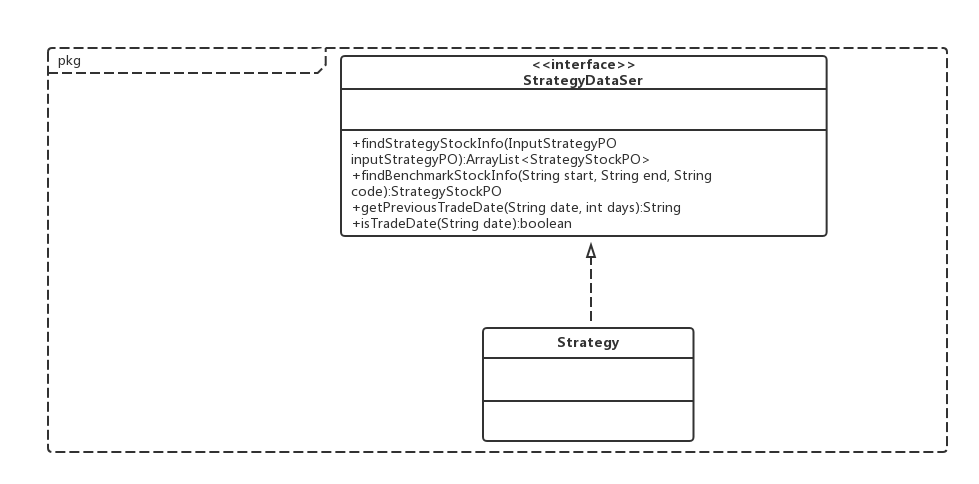
|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| Benchmark. getPlateBenchmarkData | 语法 | public ArrayList<String> getPlateBenchmarkData | |
| 前置条件 | 已知用户获取策略输入信息 | |
| 后置条件 | 返回板块的基准股票信息 | |
| Benchmark. getSelfSelectBenchmarkData | 语法 | public ArrayList<String> getSelfSelectBenchmarkData； | |
| 前置条件 | 已知策略信息 | |
| 后置条件 | 返回这些股票每日的基准累计收益率的数据 | |
| Benchmark. getBenchmarkOneRelationData | 语法 | public ArrayList<Double> getBenchmarkOneRelationData | |
| 前置条件 | 已知策略信息 | |
| 后置条件 | 返回关系图的其中一个数据 | |
| Benchmark. getYield | 语法 | public double[] getYield(ArrayList<StrategyStockPO> strategyStockPOS,int start,int end)； | |
| 前置条件 | 已知策略信息 | |
| 后置条件 | 返回strategyStockPOS中每一支股票的getAdjClose()的第start到end的每一日的累计收益率 | |
| Benchmark. getMaxWithdraw | 语法 | public ArrayList<Integer> getMaxWithdraw | |
| 前置条件 | 已知策略信息 | |
| 后置条件 | 返回最大回撤的开始日期和结束日期 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| Calculate.getBeforeSomeTradeDate | | | 返回当前日期前几天的交易日 |
| getStockInfo.getBenchmarkStockInfo | | | 返回基准股票数据 |

### 5.4 数据层的分解

数据层主要给业务逻辑层提供数据防伪服务，包括对于持久化数据的增、删、改、查。getEMA业务逻辑需要的服务由findStockInfoDataSer接口提供。由于持久化数据的保存可能存在多种形式：Txt文件、序列化文件、数据库等，所示抽象了数据服务。数据层模块的具体描述如图5.4所示。







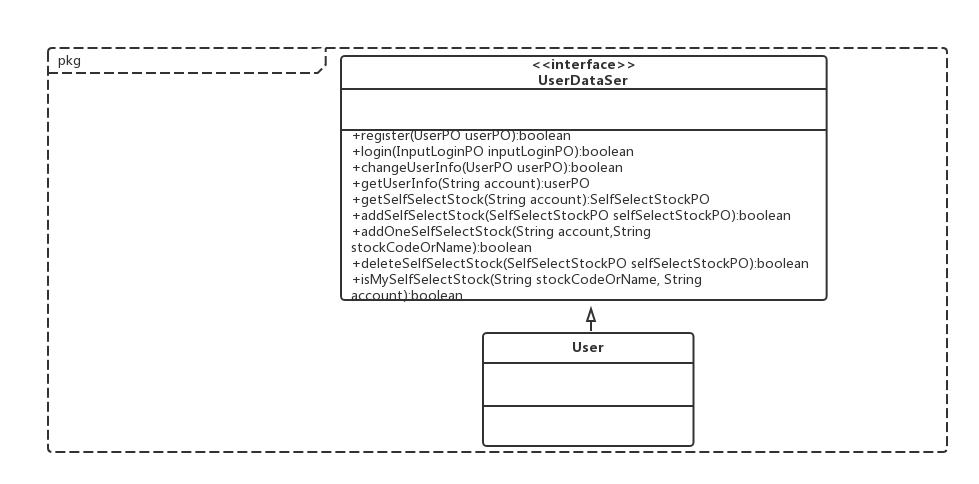


图5.4 数据层模块的描述

#### 5.4.1 数据层模块的职责

数据层模块的职责如表5.4.1所示。

表5.4.1 数据层模块的职责

|  |  |
| --- | --- |
| 模块 | 职责 |
| Data | 持久化数据库的接口，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

表5.4.2-1数据层模块的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| FindStockInfo. findStockInfoByCode | 语法 | public ArrayList<StockPO> findStockInfoByCode(String start, String end, int code); |
| 前置条件 | 输入正确的时间格式以及股票代码 |
| 后置条件 | 返回相应的StockPO |
| FindStockInfo. findStockInfoByName | 语法 | public ArrayList<StockPO> findStockInfoByName(String start, String end, String name); |
| 前置条件 | 输入正确的时间以及股票名字 |
| 后置条件 | 返回相应的StockPO |
| FindStockInfo. findStockInfoOneday | 语法 | public ArrayList<StockPO> findStockInfoOneday(String date) |
| 前置条件 | 输入正确的时间 |
| 后置条件 | 返回相应的StockPO |
| FindStockInfo. findAllStock | 语法 | public ArrayList<StockPO> findAllStock(String searchInfo) |
| 前置条件 | 已经输入搜索信息 |
| 后置条件 | 返回相应的StockPO |
| FindPlateInfo.findTotalPlateInfo | 语法 | publicTotalPlatePO findTotalPlateInfo(String date); |
| 前置条件 | 输入正确的日期 |
| 后置条件 | 返回某一天整体的板块*信息* |
| FindPlateInfo.findOnePlateInfo | 语法 | public ArrayList<StockPO> findOnePlateInfo(int type, String date); |
| 前置条件 | 输入正确的板块种类和日期 |
| 后置条件 | 返回某一个板块的股票某一天的具体信息 |
| Strategy.findStrategyStockInfo | 语法 | public ArrayList<StrategyStockPO> findStrategyStockInfo(InputStrategyPO inputStrategyPO); |
| 前置条件 | 输入正确的信息 |
| 后置条件 | 返回符合要求的信息 |
| Strategy.findBenchmarkStockInfo | 语法 | public StrategyStockPO findBenchmarkStockInfo(String start, String end, String code)； |
| 前置条件 | 输入正确的时间和股票代码 |
| 后置条件 | 返回相应时间区间内的基准股票每日的信息 |
| Strategy.getPreviousTradeDate | 语法 | public String getPreviousTradeDate(String date, int days)； |
| 前置条件 | 输入正确的时间 |
| 后置条件 | 返回某交易日days个交易日之前的交易日 |
| Strategy.isTradeDate | 语法 | public boolean isTradeDate(String date)； |
| 前置条件 | 输入正确的时间 |
| 后置条件 | 返回判断结果 |
| User.register | 语法 | public boolean register(UserPO userPO)； |
| 前置条件 | 输入注册用户信息 |
| 后置条件 | 返回是否注册成功 |
| User.login | 语法 | public boolean login(InputLoginPO inputLoginPO)； |
| 前置条件 | 输入账户，密码 |
| 后置条件 | 返回是否登录成功 |
| User.changeUserInfo | 语法 | public boolean changeUserInfo(UserPO userPO)； |
| 前置条件 | 输入新的用户信息 |
| 后置条件 | 返回是否修改成功 |
| User.getUserInfo | 语法 | public UserPO getUserInfo(String account)； |
| 前置条件 | 输入账号 |
| 后置条件 | 返回对应用户信息 |
| User.getSelfSelectStock | 语法 | public SelfSelectStockPO getSelfSelectStock(String account)； |
| 前置条件 | 输入用户账号 |
| 后置条件 | 返回用户股票信息 |
| User.addSelfSelectStock | 语法 | public boolean addSelfSelectStock(SelfSelectStockPO selfSelectStockPO)； |
| 前置条件 | 输入股票完整信息 |
| 后置条件 | 返回是否添加成功 |
| User.addOneSelfSelectStock | 语法 | public boolean addOneSelfSelectStock(String account,String stockCodeOrName)； |
| 前置条件 | 输入股票部分信息 |
| 后置条件 | 返回是否添加成功 |
| User.deleteSelfSelectStock | 语法 | public boolean deleteSelfSelectStock(SelfSelectStockPO selfSelectStockPO)； |
| 前置条件 | 输入要删除的股票信息 |
| 后置条件 | 返回是否删除成功 |
| User.isMySelfSelectStock | 语法 | public boolean isMySelfSelectStock(String stockCodeOrName, String account)； |
| 前置条件 | 输入用户账号和股票代码或名字 |
| 后置条件 | 返回判断结果 |

## 6、信息视角

### 6.1 数据持久化对象

系统的PO类就是对应的相关的实体类，在此只做简单的介绍。

StockPO类包含股票的开盘指数、最高指数、最低指数等属性。

UserPO类包括用户的id、姓名、密码。

InputLoginPO类包括用户的账号、密码。

InputStrategyPO类包括开始、结束日期，股票池种类。

SelfSelectStockPO类包括用户账号、股票代码、股票名字、股票市场。

StrategyStockPO类包括股票名称，代码，日期，收盘价。

TotalPlatePO类包括板块名称、开盘公司数等。

持久化用户对象如StockPO的定义如图: 更多定义见原型代码



### 6.2 Csv持久化格式

Csv数据格式以用户个人信息.csv为例。，每行分别对应账号，帐户名，密码，真实密码，邮箱，出生日期，手机号。中间用Tab隔开。如下所示：

b123 李四 EA73456AEBBD87E5C620896323E614CC 陈进 572008964@qq.com 19950401 13951723565

a123 张三 EA73456AEBBD87E5C620896323E614CC 陈进 572008964@qq.com 19950401 13951723565

### 6.3数据库表

无