**股票分析系统**

**详细设计描述文档**

学 院：南京大学软件学院

成 员：常德隆 曹鸿荣 陈进 武秀峰

完成日期：2017年4月20日

目录

[更新历史 4](#_Toc480474799)

[1. 引言 5](#_Toc480474800)

[1.1编制目的 5](#_Toc480474801)

[1.2词汇表 5](#_Toc480474802)

[1.3参考资料 5](#_Toc480474803)

[2. 中层设计 6](#_Toc480474804)

[2.1 KAndEMABl模块的静态结构和动态行为 6](#_Toc480474805)

[2.1.1 KAndEMABl模块局部模块的职责 6](#_Toc480474806)

[2.1.2 KAndEMABl模块局部模块的接口规范 7](#_Toc480474807)

[2.1.3 KAndEMABl模块的行为 12](#_Toc480474808)

[2.2 MarketSituationBl模块的静态结构和动态行为 12](#_Toc480474809)

[2.2.1 MarketSituationBl模块局部模块的职责 12](#_Toc480474810)

[2.2.2 getMarketSituatioBl模块局部模块的接口规范 14](#_Toc480474811)

[2.2.3 MarketSituationBl模块的行为 17](#_Toc480474812)

[2.3 quantourCompareBl模块的静态结构和动态行为 17](#_Toc480474813)

[2.3.1 quantourCompareBl模块局部模块的职责 17](#_Toc480474814)

[2.3.2 quantourCompareBl模块局部模块的接口规范 18](#_Toc480474815)

[2.3.3 quantourCompareBl模块的行为 22](#_Toc480474816)

[2.4 searchBl模块的静态结构和动态行为 22](#_Toc480474817)

[2.4.1 searchBl模块局部模块的职责 22](#_Toc480474818)

[2.4.2 searchBl模块局部模块的接口规范 23](#_Toc480474819)

[2.4.3 searchBl模块的行为 24](#_Toc480474820)

[2.5 plateBl模块的静态结构和动态行为 25](#_Toc480474821)

[2.5.1 plateBl模块局部模块的职责 25](#_Toc480474822)

[2.5.2 plateBl模块局部模块的接口规范 26](#_Toc480474823)

[2.6 userBl模块的静态结构和动态行为 29](#_Toc480474824)

[2.6.1 userBl模块局部模块的职责 29](#_Toc480474825)

[2.6.2 userBl模块局部模块的接口规范 30](#_Toc480474826)

[2.7 strategyBl模块的静态结构和动态行为 37](#_Toc480474827)

[2.7.1 strategyBl模块局部模块的职责 37](#_Toc480474828)

[2.7.2 strategyBl模块局部模块的接口规范 38](#_Toc480474829)

[2.8 依赖视角 49](#_Toc480474830)

[2.8.1 包之间的依赖关系 49](#_Toc480474831)

[2.9 界面 50](#_Toc480474832)

[2.9.1 LoginUI模块 50](#_Toc480474833)

[2.9.2 MainUI模块 51](#_Toc480474834)

[2.9.3 MessageUI模块 53](#_Toc480474835)

[2.9.4 UserUI模块 54](#_Toc480474836)

[2.9.5 Server-MainUI模块 55](#_Toc480474837)

[2.10 数据层的分解 56](#_Toc480474838)

[2.10.1 data模块 56](#_Toc480474839)

# 更新历史

|  |  |  |  |
| --- | --- | --- | --- |
| **修改人员** | **修改日期** | **修改原因** | **版本号** |
| 全体组员 | 2017-3-6 | 初步文档 | V1.0 |
| 武秀峰 | 2017-3-18 | 结构修改 | V1.1 |
| 全体成员 | 2017-4-20 | 迭代二结构修改 | V2.0 |

# 引言

## 1.1编制目的

本报告详细完成对股票分析系统的详细设计，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 |
| Frame | 表示某展示层 |  |
| Bl | 表示某逻辑层 |  |
| Data | 表示某数据层 |  |

## 1.3参考资料

1. IEEE std 1471-2000
2. 丁二玉，刘钦.计算与软件工程（卷二）[M]机械工业出版2012：134—182

# 中层设计

## 2.1 KAndEMABl模块的静态结构和动态行为

### 2.1.1 KAndEMABl模块局部模块的职责

KAndEMABl模块承担的需求参见需求规格说明文档功能需求及相关非功能需求。KAndEMABl模块的职责及接口参见软件体系结构描述文档。根据体系结构的设计，采用分层风格，将系统分为展示层，业务逻辑层，数据层。每一层之间为了灵活性，添加了接口，以实现针对接口编程，隔离数据传输的职责，降低层与层之间耦合，添加了KAndEMABlser, findStockInfoDataser 两个接口。

KAndEMABl模块的设计如图2.1.1-1所示.

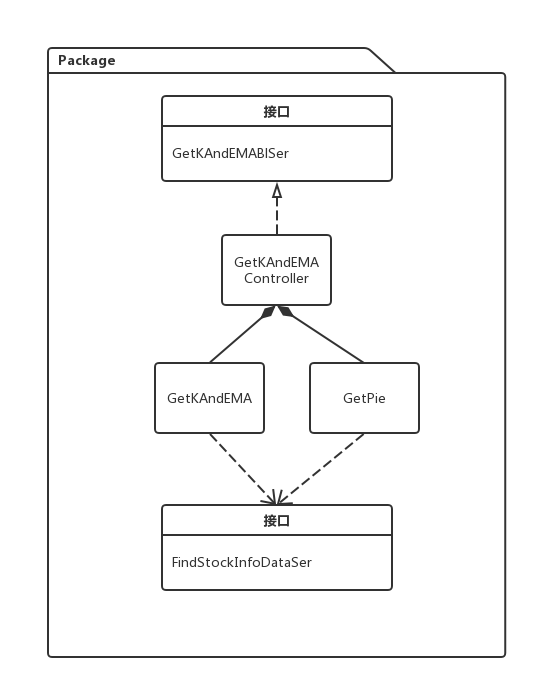


图2.1.1-1 KAndEMABl模块类的设计

|  |  |
| --- | --- |
| 类 | 职责 |
| KAndEMAController | 调用 |
| KAndEMA | 负责获取K线图和均线图的信息 |
| Pie | 负责获取饼状图的信息 |

表2.1.1-2 KAndEMABl模块的各个类的职责

### 2.1.2 KAndEMABl模块局部模块的接口规范

表2.1.2-1 KAndEMABl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 | |

### 2.1.3 KAndEMABl模块的行为

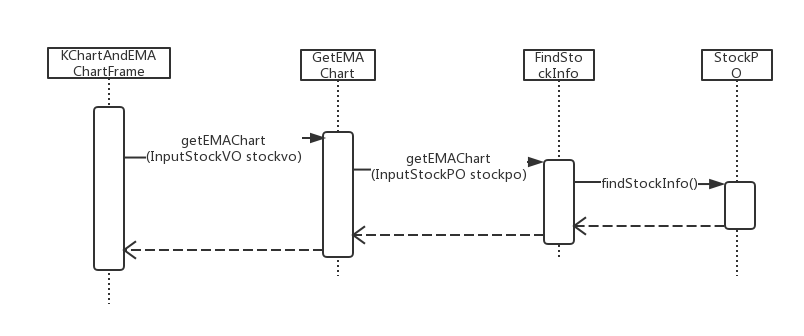


图2.1.3 获取均线图的顺序图

## 2.2 MarketSituationBl模块的静态结构和动态行为

### 2.2.1 MarketSituationBl模块局部模块的职责

MarketSituationBl模块承担的需求参见需求规格说明文档功能需求及相关非功能需求。

MarketSituationBl模块的职责及接口参见软件体系结构描述文档。根据体系结构的设计，采用分层风格，将系统分为展示层，业务逻辑层，数据层。每一层之间为了灵活性，添加了接口，以实现针对接口编程，隔离数据传输的职责，降低层与层之间耦合，添加了MarketSituationBlSer, FindStockInfoDataSer 两个接口。

MarketSituationBl模块的设计如图2.3.1-1所示.

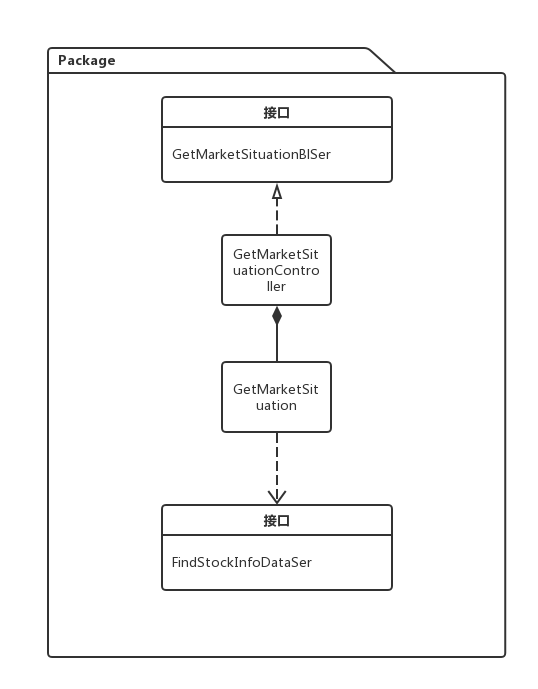


图2.3.1-1 MarketSituationBl模块的各个类设计

MarketSituationBl模块的各个类的职责如表2.3.1-1所示。

|  |  |
| --- | --- |
| 类 | 职责 |
| GetMarketSituationController | 调用GetMarketSituation |
| GetMarketSituation | 负责获取市场信息 |

### 2.2.2 getMarketSituatioBl模块局部模块的接口规范

表2.3.2-1 GetMarketSituation类的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 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 |

### 2.2.3 MarketSituationBl模块的行为

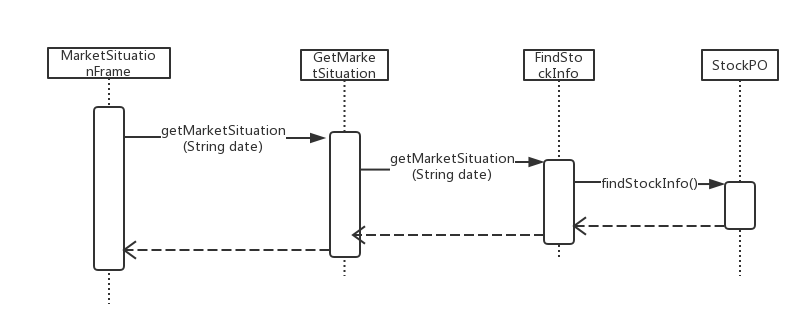


图2.3.3-1 获取市场情况的顺序图

## 2.3 quantourCompareBl模块的静态结构和动态行为

### 2.3.1 quantourCompareBl模块局部模块的职责

quantourCompareBl模块承担的需求参见需求规格说明文档功能需求及相关非功能需求。

quantourCompareBl模块的职责及接口参见软件体系结构描述文档。根据体系结构的设计，采用分层风格，将系统分为展示层，业务逻辑层，数据层。每一层之间为了灵活性，添加了接口，以实现针对接口编程，隔离数据传输的职责，降低层与层之间耦合，添加了QuantourCompareBlSer, FindStockInfoDataSer 两个接口。StockPO是作为库存信息的持久化对象被添加到设计模型中的。

quantourCompareBl模块的设计如图2.4.1-1所示。

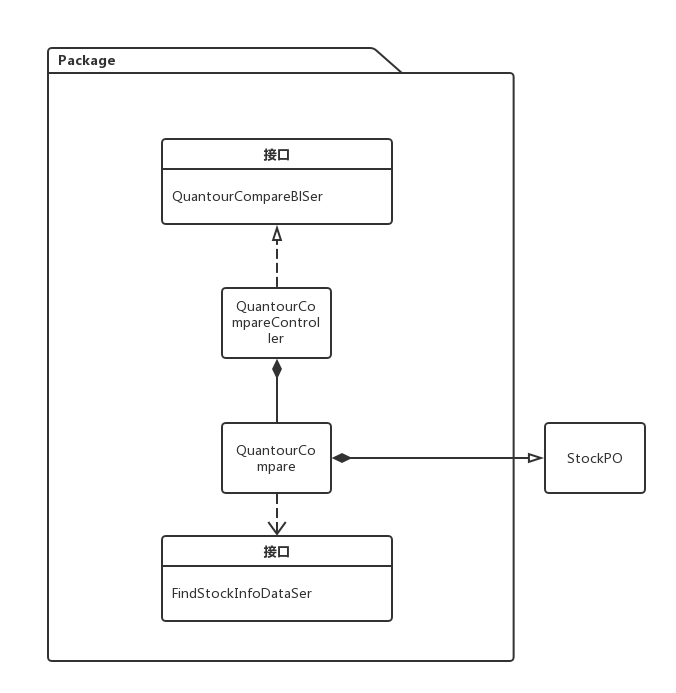


图2.3.1-1 quantourCompareBl模块的各个类设计

quantourCompareBl模块的各个类的职责如表2.3.1-1所示。

|  |  |
| --- | --- |
| 类 | 职责 |
| QuantourCompareController | 调用 |
| QuantourCompare | 负责比较两只或多只股票之间的信息并展示 |

### 2.3.2 quantourCompareBl模块局部模块的接口规范

表2.3.2-1 quantourCompareBl模块的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 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 |

## 2.3.3 quantourCompareBl模块的行为

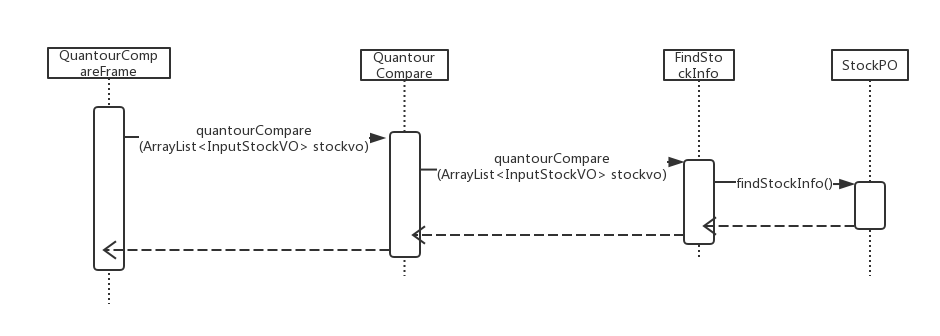


图2.3.3-1 股票信息比较的顺序图

## 2.4 searchBl模块的静态结构和动态行为

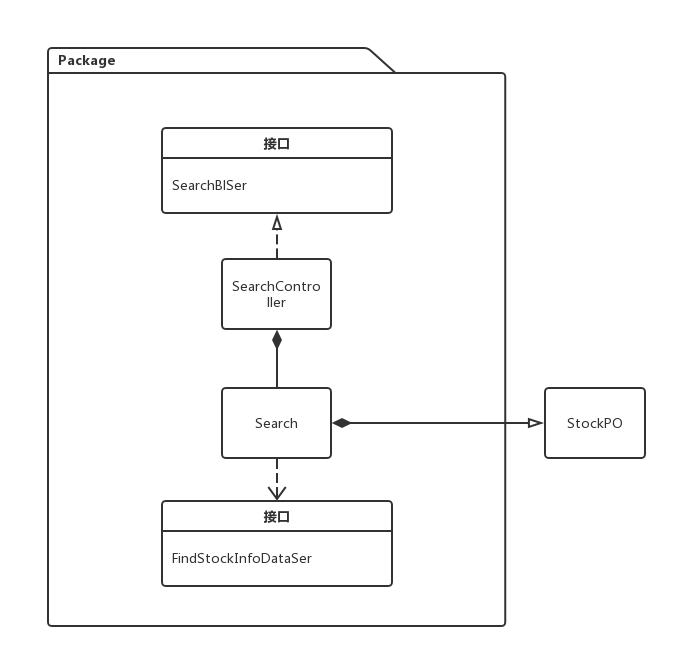
### 2.4.1 searchBl模块局部模块的职责

searchBl模块承担的需求参见需求规格说明文档功能需求及相关非功能需求。

searchBl模块的职责及接口参见软件体系结构描述文档。根据体系结构的设计，采用分层风格，将系统分为展示层，业务逻辑层，数据层。每一层之间为了灵活性，添加了接口，以实现针对接口编程，隔离数据传输的职责，降低层与层之间耦合，添加了SearchBlSer, FindStockInfoDataSer 两个接口。StockPO是作为库存信息的持久化对象被添加到设计模型中的。

searchBl模块的设计如图2.4.1-1所示。

图2.4.1-1 searchBl模块的各个类设计



### searchBl模块局部模块的接口规范

表2.4.2-1 searchBl模块的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 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 |

### searchBl模块的行为

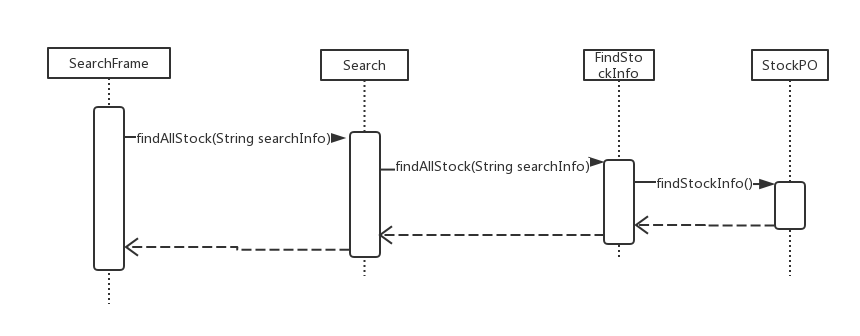


图2.4.3-1 查找信息的顺序图

## 2.5 plateBl模块的静态结构和动态行为

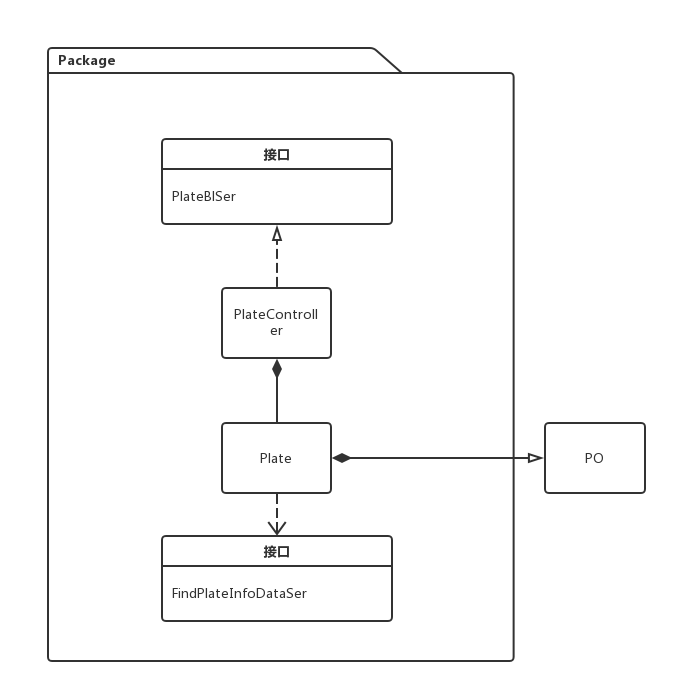
### 2.5.1 plateBl模块局部模块的职责

plateBl模块承担的需求参见需求规格说明文档功能需求及相关非功能需求。

plateBl模块的职责及接口参见软件体系结构描述文档。根据体系结构的设计，采用分层风格，将系统分为展示层，业务逻辑层，数据层。每一层之间为了灵活性，添加了接口，以实现针对接口编程，隔离数据传输的职责，降低层与层之间耦合，添加了PlateBlSer, FindStockInfoDataSer 两个接口。StockPO是作为库存信息的持久化对象被添加到设计模型中的。

plateBl模块的设计如图2.5.1-1所示。

图2.5.1-1 plateBl模块的各个类设计



### plateBl模块局部模块的接口规范

表2.5.2-1 plateBl模块的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 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 |

## 2.6 userBl模块的静态结构和动态行为

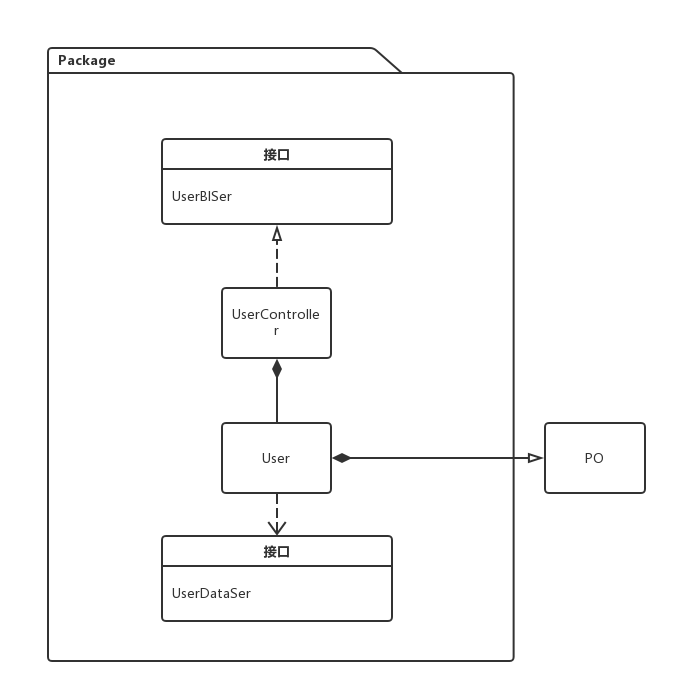
### 2.6.1 userBl模块局部模块的职责

userBl模块承担的需求参见需求规格说明文档功能需求及相关非功能需求。

userBl模块的职责及接口参见软件体系结构描述文档。根据体系结构的设计，采用分层风格，将系统分为展示层，业务逻辑层，数据层。每一层之间为了灵活性，添加了接口，以实现针对接口编程，隔离数据传输的职责，降低层与层之间耦合，添加了UserBlSer, FindStockInfoDataSer 两个接口。StockPO是作为库存信息的持久化对象被添加到设计模型中的。

userBl模块的设计如图2.4.1-1所示。

图2.6.1-1 userBl模块的各个类设计



### userBl模块局部模块的接口规范

表2.6.2-1 userBl模块的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 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. deleteOneSelfSelectStock | 语法 | public boolean deleteOneSelfSelectStock(String account, String stockCodeOrName) | |
| 前置条件 | 已知用户要删除的一支股票 | |
| 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. deleteOneSelfSelectStock | 语法 | public boolean deleteOneSelfSelectStock(String account, String stockCodeOrName) | |
| 前置条件 | 已知用户要删除的一支股票 | |
| 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 |

## 2.7 strategyBl模块的静态结构和动态行为

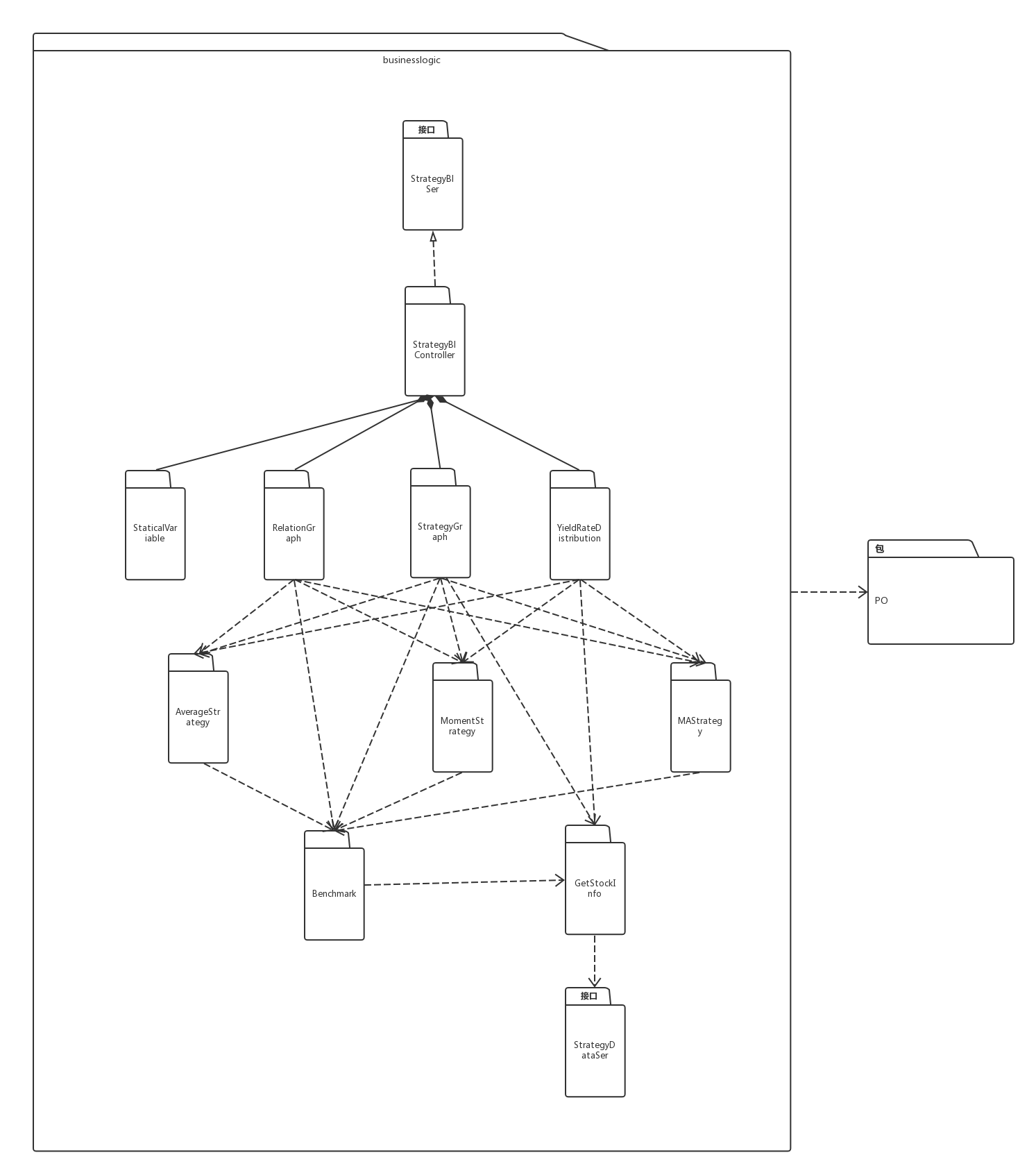
### 2.7.1 strategyBl模块局部模块的职责

strategyBl模块承担的需求参见需求规格说明文档功能需求及相关非功能需求。

strategyBl模块的职责及接口参见软件体系结构描述文档。根据体系结构的设计，采用分层风格，将系统分为展示层，业务逻辑层，数据层。每一层之间为了灵活性，添加了接口，以实现针对接口编程，隔离数据传输的职责，降低层与层之间耦合，添加了StrategyBlSer, FindStockInfoDataSer 两个接口。StockPO是作为库存信息的持久化对象被添加到设计模型中的。

strategyBl模块的设计如图2.7.1-1所示。

图2.7.1-1 strategyBl模块的各个类设计



### strategyBl模块局部模块的接口规范

表2.7.2-1 strategyBl模块的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 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 |

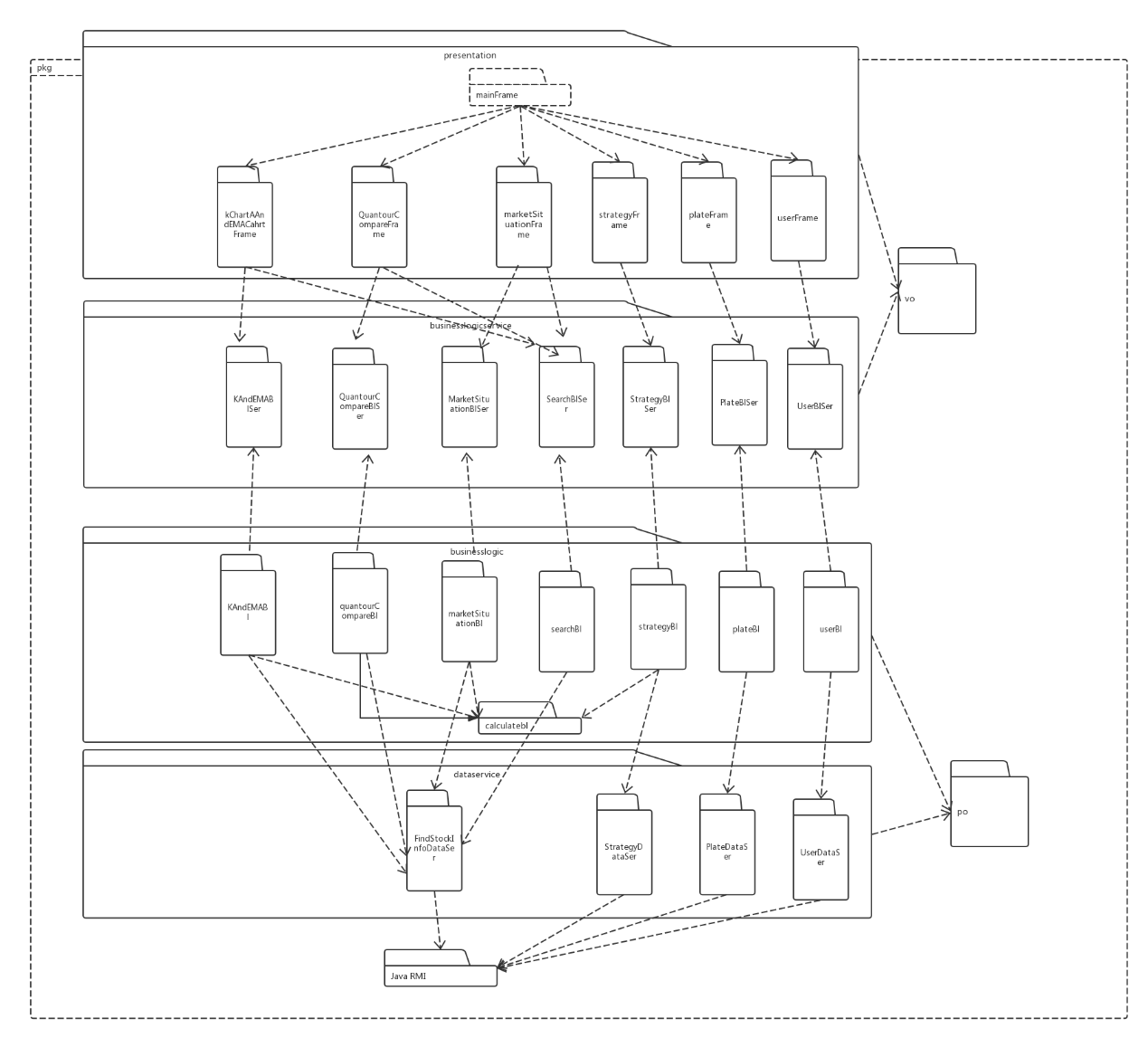
|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| AverageStrategy. getAverageData | 语法 | public ArrayList<String> getAverageData | |
| 前置条件 | 已知策略股票信息 | |
| 后置条件 | 返回均值回归策略数据 | |
| AverageStrategy. getAverageData | 语法 | public ArrayList<Double> getOneRelationData； | |
| 前置条件 | 已知策略股票信息 | |
| 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 |

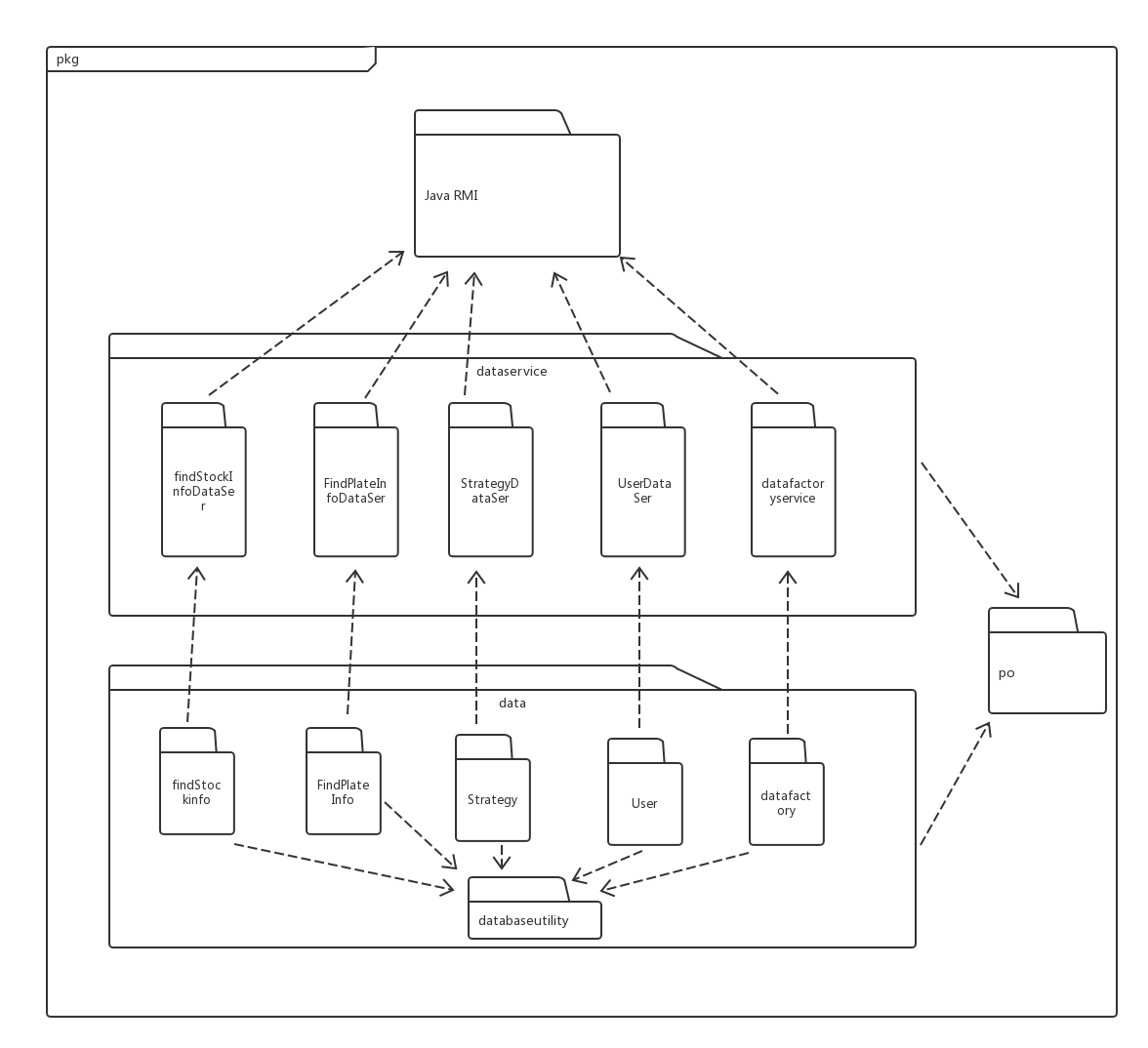
|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| MAStrategy. getMAData | 语法 | public ArrayList<String> getMAData | |
| 前置条件 | 已知策略股票信息 | |
| 后置条件 | 返回MA择时策略数据 | |
| MAStrategy. getMAData | 语法 | public ArrayList<Double> getOneRelationData； | |
| 前置条件 | 已知策略股票信息 | |
| 后置条件 | 返回关系图中的一个数据 | |
| 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 |

## 2.8 依赖视角

### 2.8.1 包之间的依赖关系

2.8.1-1 客户端包图



2.8.1-2 服务器端包图

## 2.9 界面

展示层的开发包图参见软件体系结构设计文档

### 2.9.1 LoginUI模块

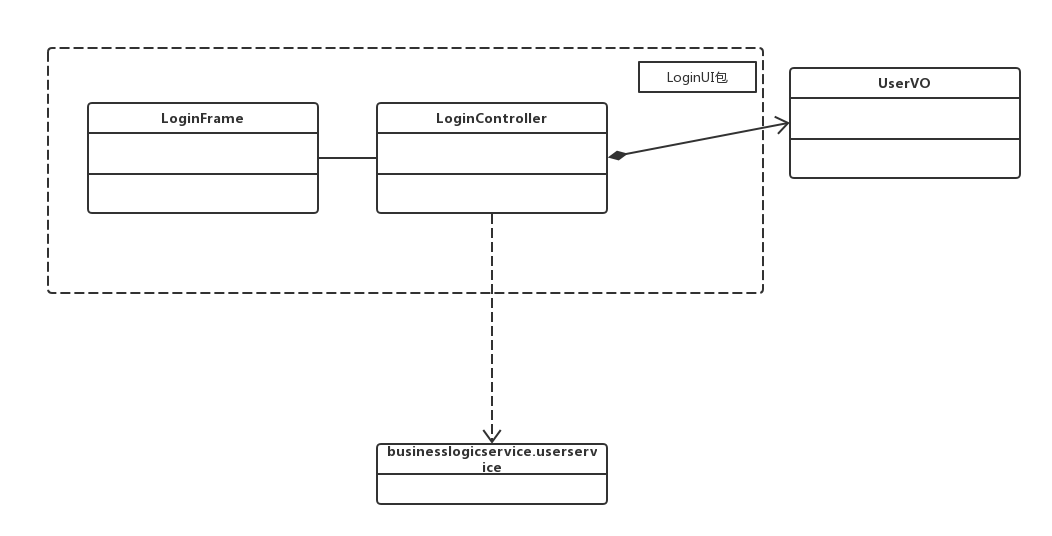
1. 模块概述

LoginUI模块承担的需求为用户登录和注册

LoginUI模块的责任是为用户提供登录和注册提供图形化界面入口

1. 整体结构

LoginUI模块的设计如图所示：



LoginUI模块各个类的设计

LoginUI模块各个类的职责如表所示：

|  |  |
| --- | --- |
| 模块 | 职责 |
| LoginFrame | 负责搭建用户登录和注册的界面 |
| LoginController | 负责给用户登录和注册的界面提供监听 |

### 2.9.2 MainUI模块

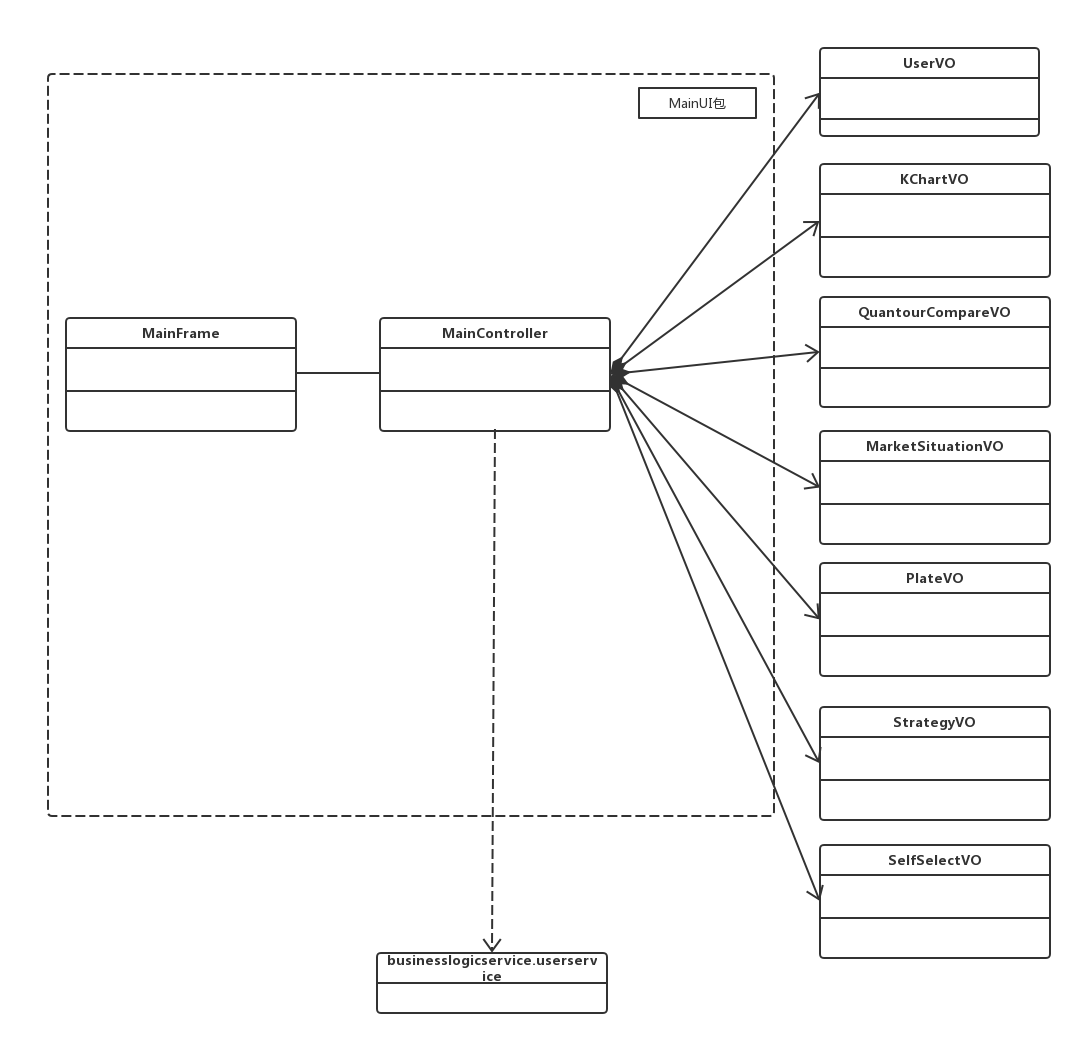
1. 模块概述

MainUI模块承担的需求为所有业务逻辑实现

MainUI模块的责任是为所有业务逻辑提供图形化界面入口

1. 整体结构

MainUI模块的设计如图所示：



MainUI模块各个类的设计

MainUI模块各个类的职责如表所示：

|  |  |
| --- | --- |
| 模块 | 职责 |
| MainFrame | 负责搭建所有业务逻辑实现的界面 |
| MainController | 负责给所有业务逻辑实现的界面提供监听 |

### 2.9.3 MessageUI模块

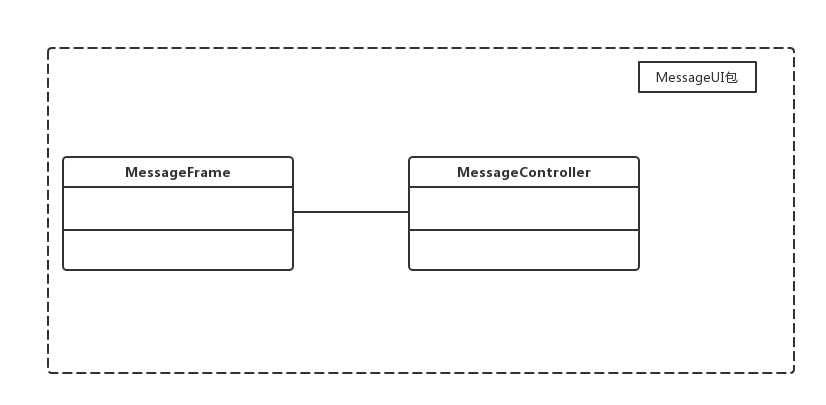
1. 模块概述

MessageUI模块承担的需求为提示用户没有登录的信息和用户登录界面入口

MessageUI模块的责任是为提示用户没有登录的信息和用户登录提供图形化界面入口

1. 整体结构

MessageUI模块的设计如图所示：



MessageUI模块各个类的设计

MessageUI模块各个类的职责如表所示：

|  |  |
| --- | --- |
| 模块 | 职责 |
| MessageFrame | 负责搭建提示用户没有登录和提供登录入口的界面 |
| MessageController | 负责给提示用户没有登录和提供登录入口的界面提供监听 |

### 2.9.4 UserUI模块

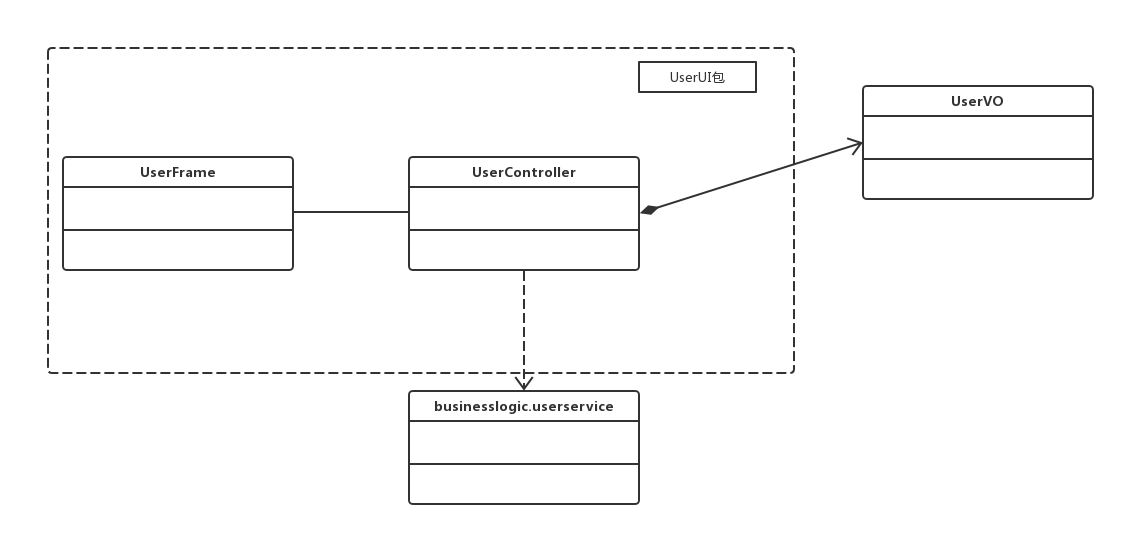
1. 模块概述

UserUI模块承担的需求为用户信息查看和修改

UserUI模块的责任是为用户信息查看和修改提供界面入口

1. 整体结构

UserUI模块的设计如图所示：



UserUI模块各个类的设计

UserUI模块各个类的职责如表所示：

|  |  |
| --- | --- |
| 模块 | 职责 |
| UserFrame | 负责搭建用户信息查看和修改的界面 |
| UserController | 负责给用户信息查看和修改的界面提供监听 |

### 2.9.5 Server-MainUI模块

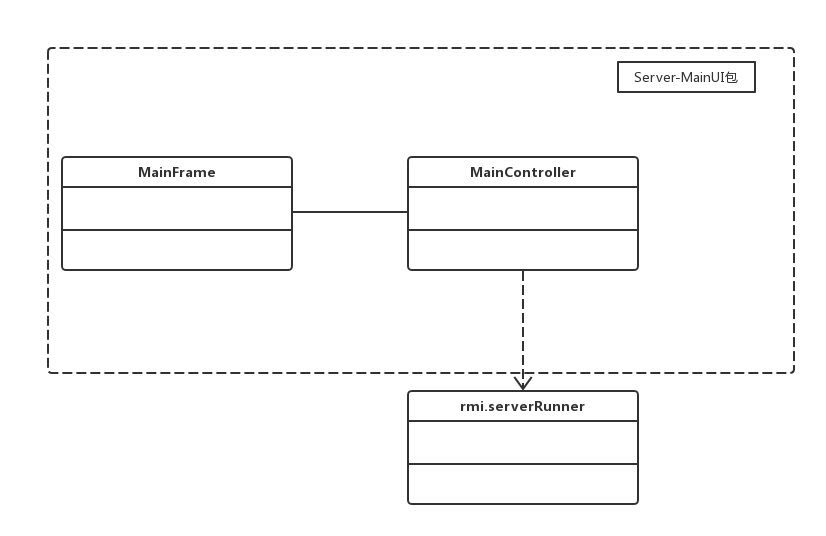
1. 模块概述

Server-MainUI模块承担的需求为服务器的启动

Server-MainUI模块的责任是为服务器启动提供图形化界面入口

1. 整体结构

Server-MainUI模块的设计如图所示：



Server-MainUI各个类的设计

Server-MainUI模块各个类的职责如表所示：

|  |  |
| --- | --- |
| 模块 | 职责 |
| MainFrame | 负责搭建服务器启动的界面 |
| MainController | 负责给用服务器启动界面提供监听 |

## 2.10 数据层的分解

数据层的开发包图参见软件体系结构文档。

### 2.10.1 data模块

（1）模块概述

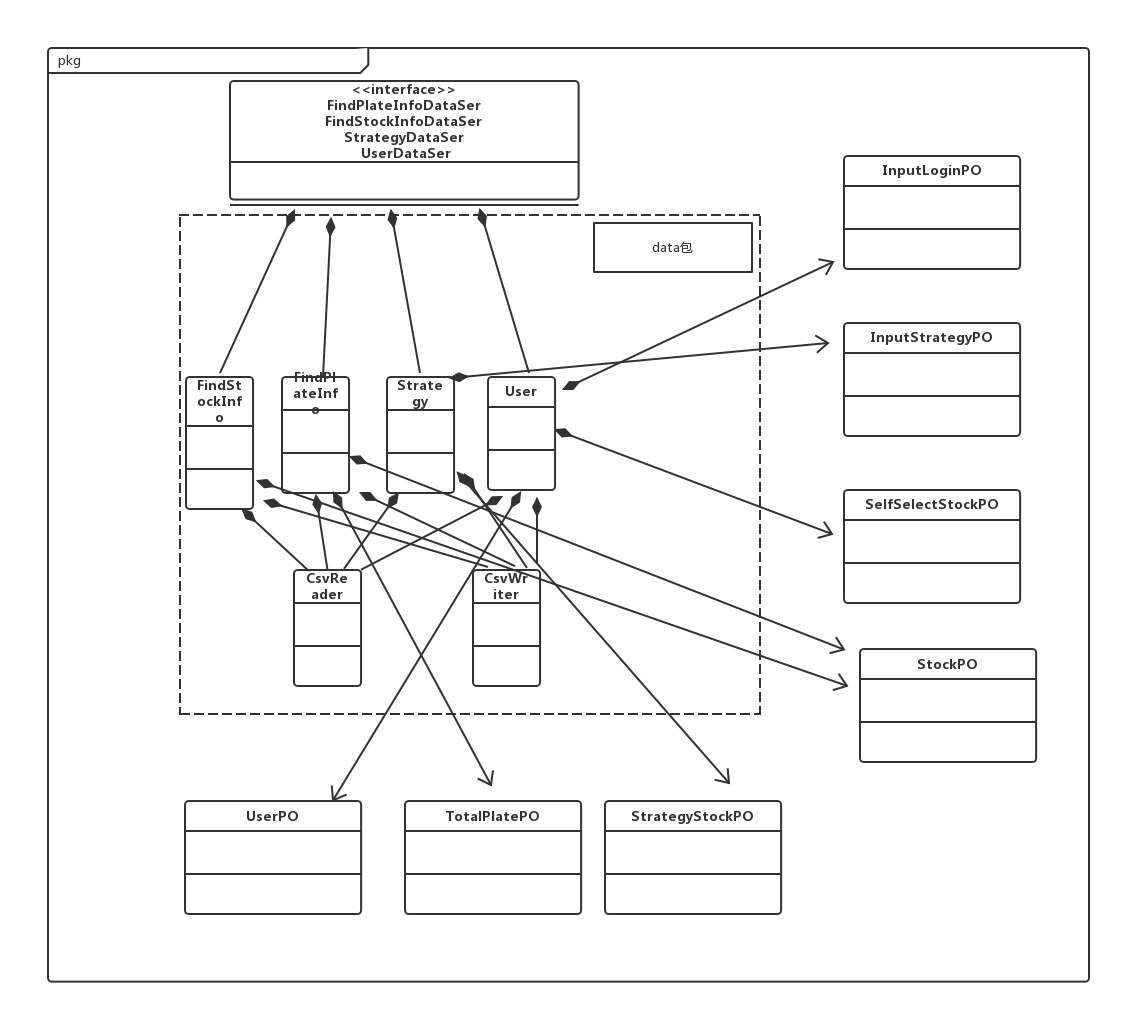
data模块承担的需求参见需求规格说明文档功能需求以及相关非功能需求。

data模块的职责以及接口参见软件系统结构描述文档。

（2）整体结构

根据体系结构的设计，我们将系统分为展示层，业务逻辑层，数据层。每一层之间为了增加灵活性，我们会添加接口。比如业务逻辑层和数据层之间添加了FindPlateInfoDataSer接口。StockPO是作为股票信息的持久化对象被添加到设计模型中去的。

data 模块的设计如图所示



Data模块各个类的职责如表所示

|  |  |
| --- | --- |
| 模块 | 职责 |
| FindStockInfo  FindPlateInfo | 负责股票信息的检索  负责板块信息的检索 |
| Strategy | 负责与策略有关信息的检索 |
| User | 负责用户个人信息，股票信息的处理 |
| CsvReader | 负责csv文件的读取 |
| CsvWriter | 负责csv文件的写入 |

#### 2.10.2 数据层模块的接口规范

表2.10.2-1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 |