SmartQuant系统设计文档

（迭代二）

1.前言 2

1.1发布的日期和状态 2

1.2发布的组织机构 2

1.3作者 2

1.4变更历史 2

2.总体介绍 2

2.1目的 2

2.2范围 2

2.3参考 3

2.4导读 3

3.词汇表 3

4.系统设计描述主题 4

4.1逻辑视角和设计视图 4

4.2组合视角和设计视图 5

4.2.1开发包图 5

4.2.2物理部署 6

4.3接口视角和设计视图 7

4.3.1模块的职责 7

4.3.2用户界面层的分解 8

4.3.3业务逻辑层的分解 10

4.3.4数据层的分解 12

4.4 信息视角 14

4.4.1 数据来源 14

4.4.2 数据对象 14

# 1.前言

## 1.1发布的日期和状态

发布日期：2016/3/9

状态：V1.0正式版

## 1.2发布的组织机构

团队名称：141250047\_cseiii\_AnyQuant

## 1.3作者

何永俊141250047软件学院

## 1.4变更历史

|  |  |  |  |
| --- | --- | --- | --- |
| 修改人员 | 日期 | 变更原因 | 版本号 |
| 何永俊 | 2016/3/8 | 最初草稿，提供给成员阅读并评审 | V0.0草稿 |
| 何永俊 | 2016/3/9 | 团队架构评审会议后的第一次更新 | V0.1正式版 |
| 黄迪璇 | 2016/3/25 | 迭代二设计完成 | V1.0正式版 |

# 2.总体介绍

## 2.1目的

本文档提供SmartQuant的软件架构概览，采用若干架构视图描述系统的不同方面，以便表示构造系统所需要的重要架构决策。

## 2.2范围

本文档的读者是141250047\_cseiii\_AnyQuant团队内部的开发和管理人员，详细完成对该系统的概要设计，使系统分析人员在此基础上进一步提出详细设计说明和完成后续设计和开发工作。

## 2.3参考

《SmartQuant系统软件需求规格文档v0.1》 141250047\_cseiii\_AnyQuant

《SmartQuant系统用例文档v0.1》 141250047\_cseiii\_AnyQuant

《SmartQuant接口规范与服务v0.1》 141250047\_cseiii\_AnyQuant

《系统设计文档模板》 IEEE 1016-2009

## 2.4导读

本文档分为四个部分，第一部分是前言，主要是关于文档发布的基本信息；第二部分是项目总体介绍，是SmartQuant系统设计文档的基本信息；第三部分是文档的词汇表；第四部分是逻辑视角、组合视角、接口视角、信息视角，从多个视角分别进行软件结构描述。

# 3.词汇表

|  |  |  |
| --- | --- | --- |
| 词汇名称 | 词汇含义 | 备注 |
| SmartQuant | 股票量化分析软件 | 为本次工程的开发产品 |
| presentation | 展示层 | 为用户界面 |
| systemlogic | 系统逻辑层 | 处理业务逻辑 |
| data | 数据层 | 持久化数据 |
| data.message | 值对象 | 用于业务层之间的数据传递 |

# 4.系统设计描述主题

## 4.1逻辑视角和设计视图

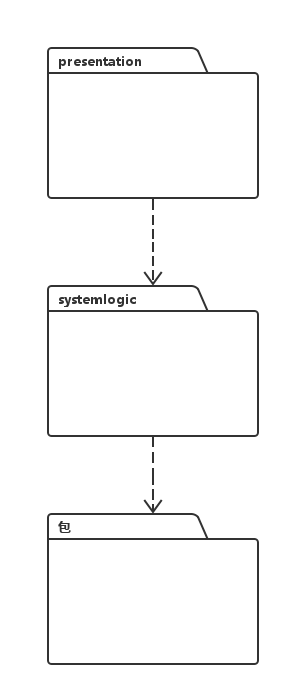


图1 参照体系结构风格的包图表达逻辑视角

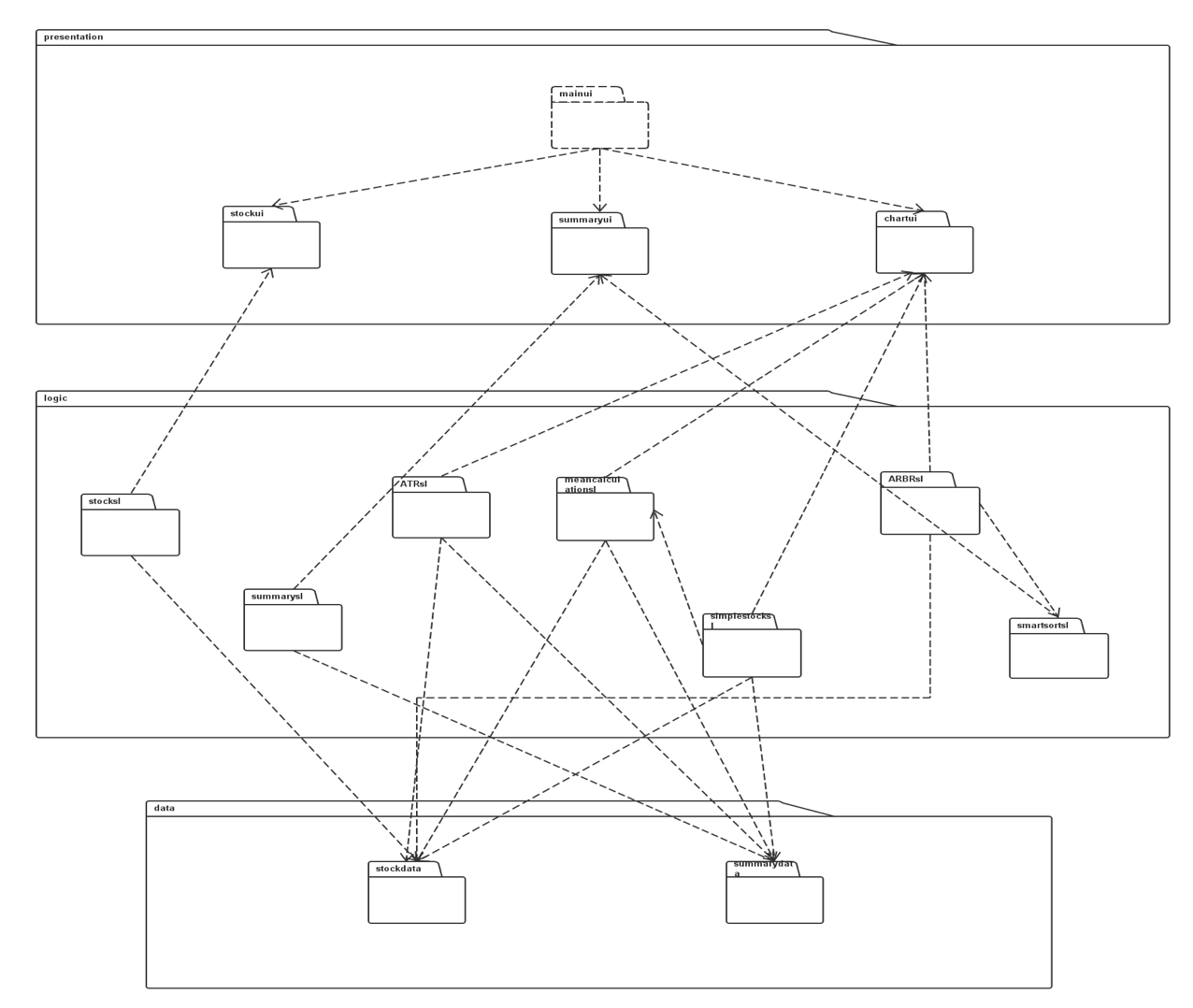


图2软件体系结构逻辑设计方案

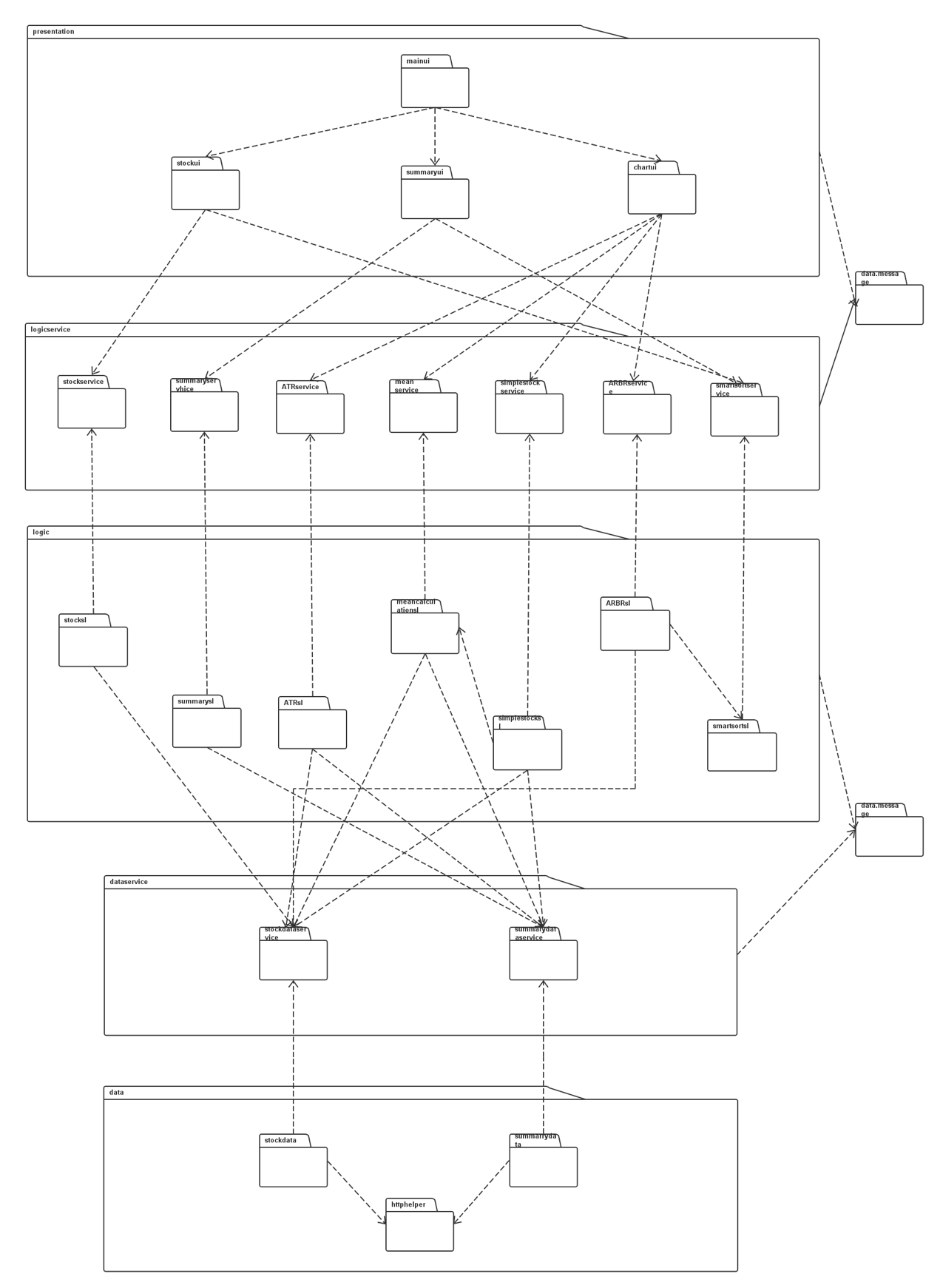
## 4.2组合视角和设计视图

## 4.2.1开发包图

表1 股票量化分析软件迭代一开发包设计

|  |  |
| --- | --- |
| 开发（物理包） | 依赖的其他开发包 |
| mainui | stockui, summaryui, chartui |
| stockui | stockservice, data.message, smartsortslservice, 界面类库包(smartui) |
| summaryui | summaryservice, smartsortservice, data.message, 界面类库包(smartui) |
| chartui | ATRslservice, meanservice, simplestockservice, ARBRservice, data.message, 界面类库包(smartui) |
| stockslservice |  |
| summaryslservice |  |
| ATRservice |  |
| meanservice |  |
| simplestockservice |  |
| ARBRservice |  |
| smartsortservice |  |
| stocksl | data.message, stockdataservice |
| summarysl | summarydataservice, data.message |
| ATRsl | stockdataservice, summarydataserivce, data.message |
| meancalculationsl | stockdataservice, summarydataserivce, data.message |
| simplestocksl | meancalculationsl, stockdataservice, summarydataserivce, data.message |
| ARBRsl | smartsortsl, stockdataservice, data.message |
| smartsortsl |  |
| stockdataservice | data.message |
| summarydataservice | data.message |
| stockdata | httphelper, data.message |
| summarydata | httphelper, data.message |
| api(AnyQuant) |  |

图3 股票量化分析软件开发包图



### 4.2.2物理部署

股票量化分析软件部署在以下物理层次。

访问层：用于用户访问系统的层次

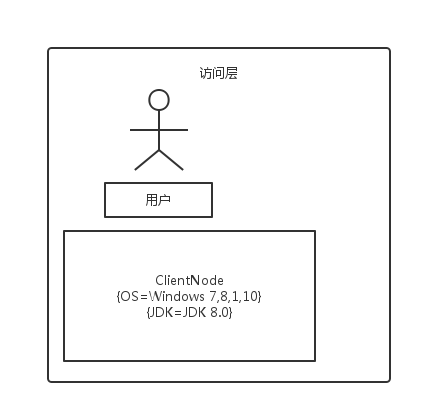


图4 部署图

## 4.3接口视角和设计视图

### 4.3.1模块的职责

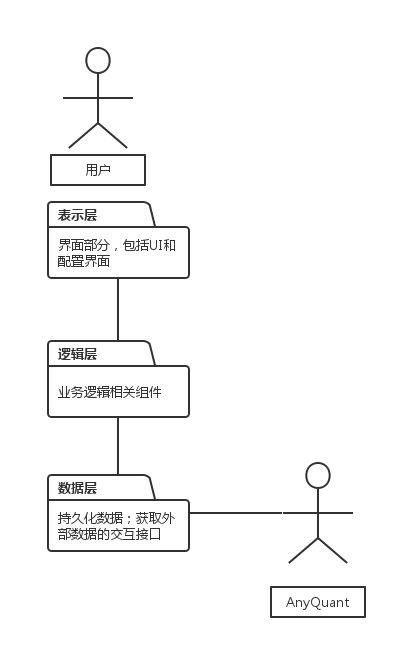


图5系统的分层架构

系统划分为以下3个逻辑层次。

1. 表示层：用于前台界面展示和配置的层次
2. 业务层：包含业务控制和逻辑的层次
3. 数据层：定义和存储系统中相关数据的层次

每一层只能调用和接触下方的接口，层与层之间调用只能通过接口来进行，从而降低了系统的耦合度。层与层之间的接口调用如表2所示。

表2 层之间调用的接口

|  |  |  |
| --- | --- | --- |
| 接口 | 服务调用方 | 服务提供方 |
| StockCheckService  SummaryCheckService  ARBRService  ATRService  MeanService  SimpleStockService  SmartSortService | 客户端展示层 | 客户端业务逻辑层 |
| StockDisposeService  SummaryDisposeService | 客户端业务逻辑层 | 客户端数据层 |

借大盘点用例来说明层之间的调用，如图6所示

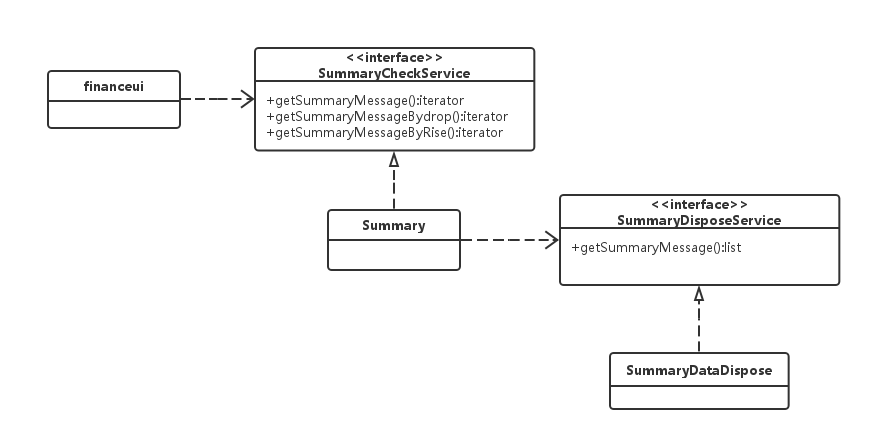


图6 大盘点用例层之间调用的接口

### 4.3.2用户界面层的分解

根据需求，系统存在4个界面：大盘点界面、股票界面。界面跳转如图7所示

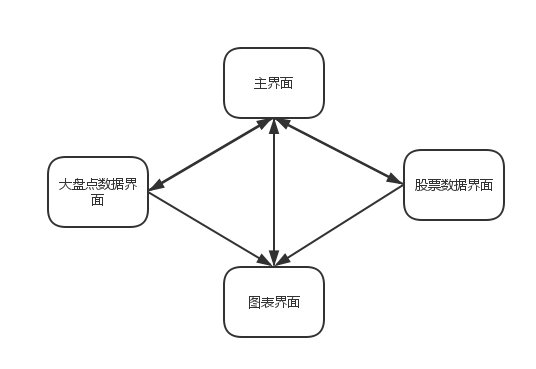


图7 用户界面跳转

#### 4.3.2.1用户界面层模块的职责

如表3所示为用户界面层模块的职责

表3 用户界面层模块的职责

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

4.3.2.2用户界面层模块的接口规范

用户界面层模块的接口规范表4所示

表4 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Frame以及MarketPanel |

用户界面层所需要的服务接口如表5所示。

表5 用户界面层模块所需要的服务接口

|  |  |
| --- | --- |
| 服务名 | 服务 |
| logicservice.\*service | 每个界面都有一个相应的业务逻辑接口 |

#### 4.3.2.3用户界面模块设计原理

基于需求中要求的图形界面设计，将利用Java的Swing和AWT库来实现用户界面。

### 4.3.3业务逻辑层的分解

业务逻辑层包括多个针对界面业务逻辑处理对象，实现logicserivce中的接口方法。

#### 4.3.3.1业务逻辑层模块的职责

业务逻辑层模块的职责如表6所示

表6 业务逻辑层模块的职责

|  |  |
| --- | --- |
| 模块 | 职责 |
| stocksl | 股票数据界面所需要的服务 |
| summarysl | 大盘点数据界面所需要的服务 |
| ATRsl | 均衡指标图表界面所需要的服务 |
| meancalculationsl | K线图界面所需要的服务 |
| simplestocksl | K线图界面所需要的服务 |
| ARBRsl | 人气意愿图表界面所需要的服务 |
| smartsortsl | 股票数据界面、大盘点数据界面所需要的服务 |

#### 4.3.3.2业务逻辑层模块的接口规范

stocksl模块的接口规范如表7所示

表7 stocksl模块的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务 | | |
| Stock.getStockMessage | 语法 | public Iterator getStockMessage(String idNumber,String beginDate,String endDate); |
| 前置条件 | 在查看股票列表的过程中，该股票被鼠标点击 |
| 后置条件 | 返回该股票的详细数据 |
| Stock.getStockList | 语法 | public Iterator getStockList(MessageType type,String Message); |
| 前置条件 | 在查看股票中，输入关键字并查询 |
| 后置条件 | 返回符合条件的股票列表 |
| Stock.getSelectedStock | 语法 | public Iterator getSelectedStock(); |
| 前置条件 | 点击刷新股票列表界面的按钮 |
| 后置条件 | 返回随机股票列表 |
| Stock.getAllStock | 语法 | public Iterator getAllStock(); |
| 前置条件 | 获取所有股票信息的按钮被点击 |
| 后置条件 | 返回所有股票信息 |

需要的服务（需接口）

|  |  |
| --- | --- |
| 服务名 | 需要的服务 |
| StockDisposeService.getStockList(String year) | 根据年份进行查找股票列表 |
| StockDisposeService.getStockMessage(String idNumber,String beginDate,String endDate) | 根据股票代码，开始日期，结束日期进行查找股票具体信息 |
| StockDisposeService.getStockDayMessage(String idNumber) | 返回指定代码的股票最新一天的信息 |

summarysl模块的接口规范如表8所示

表8 summarysl模块的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务 | | |
| Summary.getSummaryMessage | 语法 | public Iterator getSummaryMessage(MarketType type,String beginDate,String endDate); |
| 前置条件 | 启动一个大盘点回合 |
| 后置条件 | 返回大盘点数据 |

需要的服务（需接口）

|  |  |
| --- | --- |
| 服务名 | 需要的服务 |
| SummaryDisposeService.getSummaryMessage(MarketType type,String beginDate,String endDate) | 获取API提供的大盘点数据 |

ATRsl模块的接口规范如表9所示

表9 ATRsl模块的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务 | | |
| ARTService.ATRData | 语法 | public Iterator ATRData(String name,String beginDate,String endDate); |
| 前置条件 | 查看均衡指标图表 |
| 后置条件 | 返回所需的均衡指标数据 |
| ARTService.MATRData | 语法 | public Iterator MATRData(String name,String beginDate,String endDate); |
| 前置条件 | 查看均衡指标图表 |
| 后置条件 | 返回所需的均衡指标数据 |

需要的服务（需接口）

|  |  |
| --- | --- |
| 服务名 | 需要的服务 |
| StockDisposeService.getStockMessage(String name,String beginDate,String endDate) | 获取API提供的股票数据 |

meancalculationsl模块的接口规范如表10所示

表10 meancalculationsl模块的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务 | | |
| MeanCalculationService.meanClose | 语法 | public Iterator meanClose(String name, Cyc cyc, String beginDate, String endDate) ; |
| 前置条件 | 查看K线图 |
| 后置条件 | 返回所需的收盘价结点平均数数据 |
| MeanCalculationService.meanVolume | 语法 | public Iterator meanVolume(String name, Cyc cyc, String beginDate, String endDate) ; |
| 前置条件 | 查看K线图 |
| 后置条件 | 返回所需的成交量结点平均数数据 |

需要的服务（需接口）

|  |  |
| --- | --- |
| 服务名 | 需要的服务 |
| StockDisposeService.getStockMessage(String name,String beginDate,String endDate) | 获取API提供的股票数据 |
| SummaryDisposeService.getSummaryMessage(MarketType type,String beginDate,String endDate) | 获取API提供的大盘点数据 |
| SimpleStockService.getSimpleWeekNodes(String name,String beginDate,String endDate) | 获取周结点股票或大盘点数据 |
| SimpleStockService.getSimpleMonthNodes(String name,String beginDate,String endDate) | 获取月结点股票或大盘点数据 |

simplestocksl模块的接口规范如表11所示

表11 simplestocksl模块的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务 | | |
| SimpleStockService.getSimpeWeekNodes | 语法 | public Iterator getSimpleWeekNodems(String name,String beginDate, String endDate) ; |
| 前置条件 | 查看K线图 |
| 后置条件 | 返回所需的股票或大盘点周结点数据 |
| SimpleStockService.getSimpeMonthNodes | 语法 | public Iterator getSimpleMonthNodems(String name,String beginDate, String endDate) ; |
| 前置条件 | 查看K线图 |
| 后置条件 | 返回所需的股票或大盘点月结点数据 |

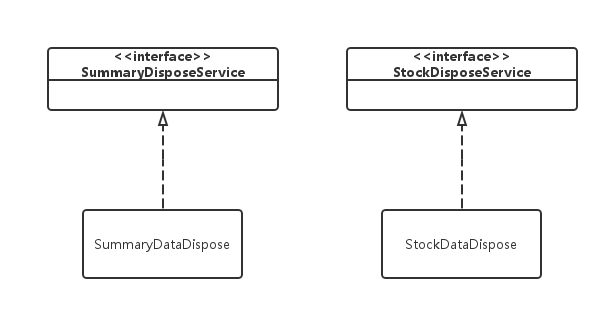
需要的服务（需接口）

|  |  |
| --- | --- |
| 服务名 | 需要的服务 |
| StockDisposeService.getStockMessage(String name,String beginDate,String endDate) | 获取API提供的股票数据 |
| SummaryDisposeService.getSummaryMessage(MarketType type,String beginDate,String endDate) | 获取API提供的大盘点数据 |
| SmartSortService.sort(DataType type, Iterator iteratior) | 对数据进行排序 |

### 4.3.4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。

图8数据层模块的描述



#### 4.3.4.1数据层模块的职责

数据层模块的职责如表12所示

表12 数据层模块的职责

|  |  |
| --- | --- |
| 模块 | 职责 |
| StockDisposeService | 基于AnyQuant提供的接口，提供集体载入服务 |
| SummaryDisposeService | 基于AnyQuant提供的接口，提供大盘点数据查找服务 |

#### 4.3.4.2数据层模块的接口规范

数据层模块的接口规范如表13所示

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| StockDisposeService.getStockDayMessage | 语法 | public StockDateNode getStockDayMessage(String idNumber); |
| 前置条件 | 无 |
| 后置条件 | 按ID进行查找返回相应的股票最新一天信息结果 |
| StockDisposeService.getStockList | 语法 | public Map getStockList(String year); |
| 前置条件 | 无 |
| 后置条件 | 按年份进行查找返回相应的股票列表结果 |
| StockDisposeService.getStockMessage | 语法 | public List getStockMessage(String idNumber,String beginDate,String endDate) |
| 前置条件 | 无 |
| 后置条件 | 按日期和ID进行查找返回相应的多个股票信息结果 |
| SummaryDisposeService.getSummaryMessage | 语法 | public List getSummaryMessage(MarketType markettype, String beginDate,String endDate); |
| 前置条件 | 无 |
| 后置条件 | 按指定交易所和时间进行查找返回相应时间内的大盘数据 |

## 4.4 信息视角

### 4.4.1 数据来源

数据来源为AnyQuant的股票API，通过Http方法获得JSON格式的数据封装包，由数据称对其进行解析成data.message中临时数据类

### 4.4.2 数据对象

Stock类是对股票信息进行分装的类，包含股票链接、股票代号、股票类型、股票中文名、股票日结点等属性

StockDateNode类是股票的日结点类，包含成交量、市净率、最高价、后复权价、最低价、日期、收盘价、开盘价、换手率等属性

SimpleStockNode类是简单股票／大盘点数据周／月结点类，包含起始日期、结点周期、周期内最高价、周期内最低价、开盘价、收盘价、总成交量等属性

ARBRReport类是对股票种类的人气意愿分析的封装信息类，包括平均人气指标、平均意愿指标、推荐指度、股票种类等属性

Summary类是对大盘点信息进行封装的类、包含股市名、大盘点数据日结点等属性

SummaryDateNode类是大盘点数据的日结点类，包含成交量、最高价、后复权价、最低价、日期、收盘价、开盘价等属性

Stock对象定义如下：

public class Stock {

private String link;//访问地址

private String name;

private String cName;

private StockType stockType;

private List<StockDateNode>dateMessage;

private StockDateNode todayData;

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getCName() {

return cName;

}

public void setCName(String cName) {

this.cName = cName;

}

public Iterator getAllDateMessage(){

if(!dateMessage.isEmpty()){

// Collections.sort(dateMessage);

return dateMessage.iterator();//返回value的迭代器

}

return null;

}

public List getDateMessage(){

return dateMessage;

}

public void setDateMessage(List<StockDateNode>dateMessage) {

this.dateMessage = dateMessage;

}

public String getLink() {

return link;

}

public void setLink(String link) {

this.link = link;

}

public String toString(){

return "name: " + name + " link: " + link;

}

public StockDateNode getTodayData() {

return todayData;

}

public void setTodayData(StockDateNode todayData) {

this.todayData = todayData;

}

public StockType getStockType() {

return stockType;

}

public void setStockType(StockType stockType) {

this.stockType = stockType;

}

}

其他定义请参考代码。