**STOCK STATISTIC  
SYSTEM  
详细设计文档**

2016-4-15

朱鼎

徐亚帆

吴秦月

汪文藻

**文档更新记录表：**

|  |  |  |  |
| --- | --- | --- | --- |
| 修改人员 | 日期 | 变更原因 | 版本号 |
| 朱鼎 | 2016/3/19 | 初稿 | V1.0 |
| 吴秦月 | 2016/4/15 | 迭代二修改 | V2.0 |
| 朱鼎 | 2016/4/16 | 迭代二修改 | V2.1 |

目录

[1. 引言 3](#_Toc446180057)

[1.1. 编制目的 3](#_Toc446180058)

[2. 产品描述 3](#_Toc446180059)

[3.逻辑视角 4](#_Toc446180060)

[4组合视角 5](#_Toc446180061)

[4.1开发包图 5](#_Toc446180062)

[5接口视角 7](#_Toc446180063)

[5.1.业务逻辑层的分解 7](#_Toc446180064)

[5.1.1.业务逻辑层模块的职责 7](#_Toc446180065)

[5.1.2业务逻辑模块的接口模范 7](#_Toc446180066)

[5.2数据层的分解 13](#_Toc446180067)

[5.2.1数据层模块的职责 13](#_Toc446180068)

[5.2.2数据层模块的接口规范 14](#_Toc446180069)

# 引言

## 编制目的

本文档详细完成对Stock Statistics 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。

# 产品描述

将AnyQuant数据api应用到项目中，构建一个基于Java Applet 的股票分析展现软件，通过软件对股票数据的展现和分析，能给出让人比较感兴趣的结论和报告，或者有比较新颖的展现图表等方式。

# 3.逻辑视角

Stock statistics system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

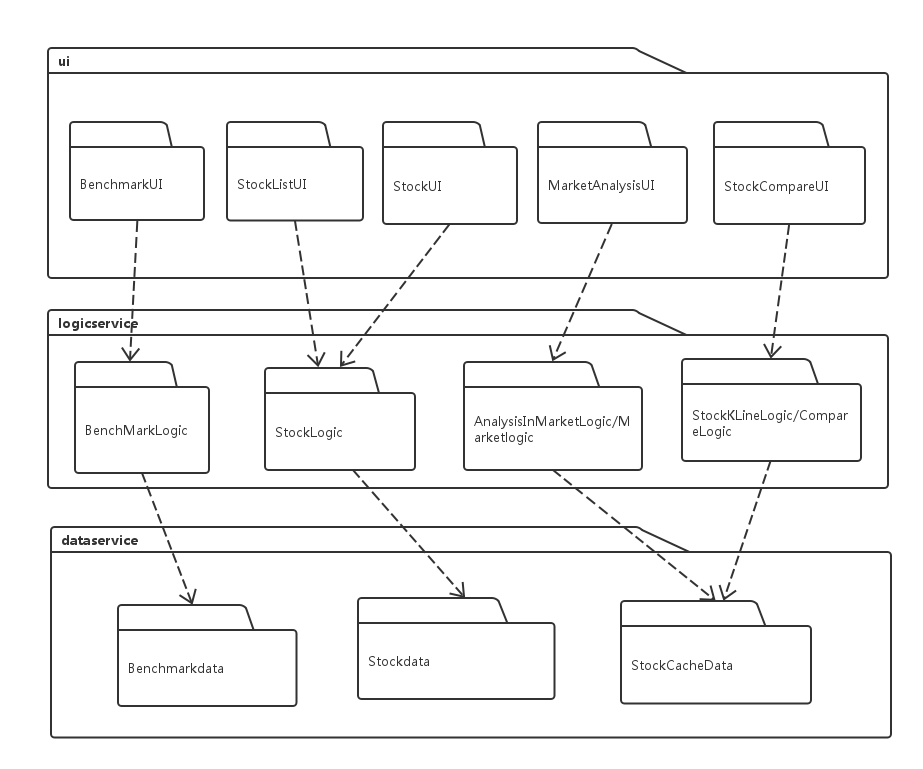


图 2 软件体系结构逻辑设计方案

# 4组合视角

## 4.1开发包图

Stock分析系统的最终开发包图设计如图3和表1所示

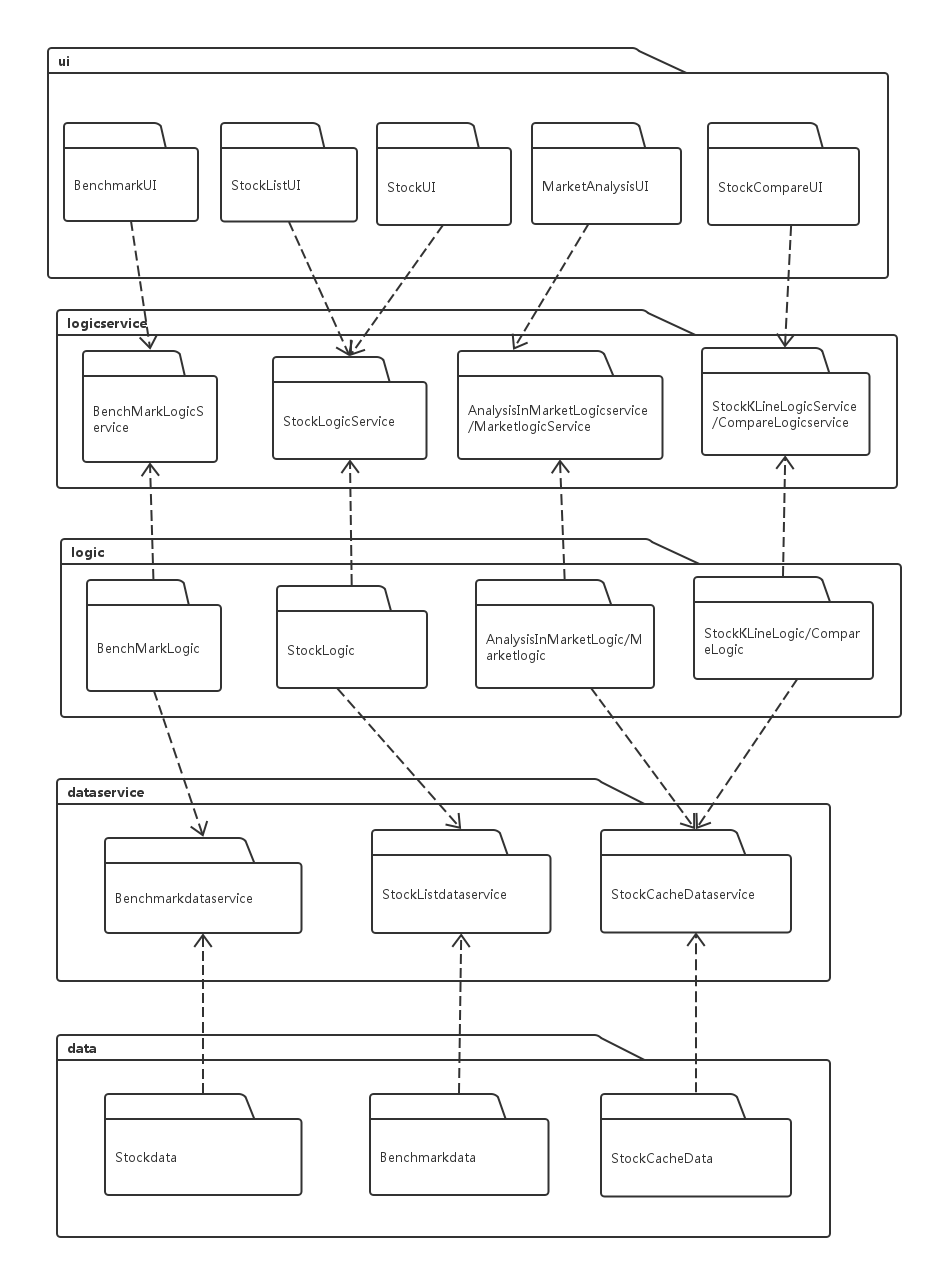


图 3开发包图

**1 层之间调用的接口**

|  |  |  |
| --- | --- | --- |
| 接口 | 服务调用方 | 服务提供方 |

|  |  |  |
| --- | --- | --- |
| AnalysisInMarketLogicservice  BenchmarkLogicservice  CompareLogicservice  initialLogicservice  MacdcalculateLogicservice  StockListLogicservice  StockKLineLogicservice  MarketLogicservice | 展示层 | 业务逻辑层 |
| BenchmarkDataservice  StockListDataservice  StockCacheDataService | 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** |
| BenchmarkLogic | 负责实现大盘指数信息界面所需要的服务 |
| StockListLogic | 负责实现所有股票列表信息界面所需要的服务 |
| MarketLogic | 负责实现银行股市行情分析展示界面所需要的服务 |
| AnalysisInMarketLogic | 行情分析数据所需要的服务 |
| CompareLogic | 负责统计分析，主要是股票间的对比（包括大盘）所需要的服务 |
| StockKLineLogic | 负责实现股票k线图所需要的服务 |
| MacdcalculateLogic | 负责实现股票macd指标所需要的服务 |
| initialLogic | 负责实现初始化16只股票所需要的服务 |

### 5.1.2业务逻辑模块的接口模范

**BenchmarkLogic接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| BenchmarkLogic. getByName | 语法 | public ArrayList<NBenchMarkVO> getByName(String name); |
| 前置条件 | 无 |
| 后置条件 | 根据大盘名返回该大盘所有可获取的数据 |
| BenchmarkLogic. getResultList | 语法 | Public getResultList(ArrayList<NBenchMarkVO> vo,Filter filter) |
| 前置条件 | 无 |
| 后置条件 | 根据筛选项属性和需要筛选的大盘信息返回经筛选过的大盘数据 |
| BenchmarkLogic. sort | 语法 | Public ArrayList<NBenchMarkVO> sort(ArrayList<NBenchMarkVO> vo,SortType sortType); |
| 前置条件 | 无 |
| 后置条件 | 根据排序方法和需要排序的数据返回经排序过的大盘数据 |
| BenchmarkLogic. getKLineVOs | 语法 | Public ArrayList<KLineVO> getKLineVOs(KLineType kLineType,Date start, Date end); |
| 前置条件 | 无 |
| 后置条件 | 依据所给时间条件和k线种类，返回k线所需数据 |
| BenchmarkLogic. getMAVOs | 语法 | Public ArrayList<MAVO> getMAVOs(Date start, Date end); |
| 前置条件 | 无 |
| 后置条件 | 依据所给时间期限，返回绘制ma x线所需的数据，成交量，收盘价 |
| BenchmarkLogic. getByTime | 语法 | ArrayList<NBenchMarkVO> getByTime(String name, Date startTime, Date endTime) |
| 前置条件 | 无 |
| 后置条件 | 获得指定时间段某大盘的数据信息 |
| **需要的服务（需接口）** | | |
| **服务名** | | 服务 |
| BenchmarkDataService. getByName | | 根据名称得到的具体大盘指数信息 |
| BenchmarkDataservice. getByTime | | 根据时间得到的具体大盘指数信息 |

**StockListLogic接口规范**

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| StockListLogic. getResultListAll | 语法 | | ArrayList<NStockVO> getResultListAll(String year,String exchange); |
| 前置条件 | | 无 |
| 后置条件 | | 根据年份,交易所返回股票数据 |
| **StockListLogic.search** | 语法 | | public ArrayList<NStockVO> search(String searchstr) |
| 前置条件 | | 无 |
| 后置条件 | | 根据搜索字段返回符合条件的数据 |
| StockListLogic. getResultList | 语法 | | public ArrayList<NStockVO> getResultList(ArrayList<NStockVO> vo,Filter filter); |
| 前置条件 | | 无 |
| 后置条件 | | 根据筛选项属性和需要筛选的股票信息返回经筛选过的股票数据 |
| StockListLogic. getResultList | 语法 | | public ArrayList<NStockVO> getResultList(ArrayList<NStockVO> vo,Filter filter) |
| 前置条件 | | 无 |
| 后置条件 | | 根据筛选项属性和需要筛选的股票信息返回经筛选过的股票数据 |
| StockListLogic. sortStockList | 语法 | | public ArrayList<NStockVO> sortStockList(ArrayList<NStockVO> vo,SortType sortType) |
| 前置条件 | | 无 |
| 后置条件 | | 根据排序方法和需要排序的数据返回经排序过的大盘数据 |
| StockListLogic. Click | 语法 | | public ArrayList<NStockVO> click(String name); |
| 前置条件 | | 无 |
| 后置条件 | | 根据名称返回具体一只股票的历史信息 |
| StockListLogic. getKLineVOs | 语法 | | Public ArrayList<KLineVO> getKLineVOs(KLineType kLineType,Date start, Date end); |
| 前置条件 | | 无 |
| 后置条件 | | 依据所给时间条件和k线种类，返回k线所需数据，开盘价，收盘价，最高价，最低价 |
| 需要的服务（需接口） | | | |  |
| **服务名** | | 服务 | |
| **StockListDataService.** **getByName** | | 根据名称起止条件返回股票列表 | |
| **StockListDataService. getResultListAll** | | 根据交易所返回股票列表信息 | |

**StockKLineLogic接口规范**

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| **StockKLineLogicService.** **getMAVOs** | 语法 | | ArrayList<MAVO> getMAVOs(String name,Date start, Date end) |
| 前置条件 | | 无 |
| 后置条件 | | 根据搜索字段返回符合条件的数据 |
| StockListLogic. getKLineVOs | 语法 | | Public ArrayList<KLineVO> getKLineVOs(KLineType kLineType,Date start, Date end); |
| 前置条件 | | 无 |
| 后置条件 | | 依据所给时间条件和k线种类，返回k线所需数据，开盘价，收盘价，最高价，最低价 |
| StockListLogic. getBollVOs | 语法 | | ArrayList<BollVO> getBoll(String name,Date start, Date end); |
| 前置条件 | | 无 |
| 后置条件 | | 得到boll图需要的数据 |
| StockListLogic. getStockATR | 语法 | | ArrayList<KDJVO> getKDJ(String name,Date start, Date end); |
| 前置条件 | | 无 |
| 后置条件 | | 得到kdj图需要的数据 |
| 语法 | | ArrayList<StockStatisticVO> getStockATR(String stockCode, Date start, Date end) |
| 前置条件 | | 无 |
| 后置条件 | | 得到atr图需要的数据 |
| 需要的服务（需接口） | | | |  |
| **服务名** | | 服务 | |
| **StockListDataService. getByName** | | 根据名称起止条件返回股票列表 | |
| **StockListDataService. getResultListAll** | | 根据交易所返回股票列表信息 | |

**MarketLogic接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| **MarketLogic**. getResultListAll | 语法 | Public ArrayList<NStockVO> getResultListAll(); |
| 前置条件 | 无 |
| 后置条件 | 获得16只银行股票的所有历史信息 |
| **MarketLogic**. getMarketVOs | 语法 | Public ArrayList<MarketVO> getMarketVOs(Date start,Date end); |
| 前置条件 | 无 |
| 后置条件 | 依据所给时间计算返回涨跌幅 |
|
| 需要的服务（需接口） | | |  |
| **服务名** | | 服务 |
| **StockListDataService.** **getAllByTime** | | 返回一天16支股票的信息 |
| **StockListDataservice.** **getAllByTime** | | 返回一天16支股票的信息从txt中获取 |

**AnalysisInMarketLogic接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| **AnalysisInMarketLogicservice**. ArrayList<AnalysisVO> | 语法 | ArrayList<AnalysisVO> averageMarketPrice (Date start ,Date end); |
| 前置条件 | 无 |
| 后置条件 | 获得行业均价折线图 |
| **AnalysisInMarketLogicservice**. shiYingLuBar | 语法 | ArrayList<AnalysisVO> shiYingLuBar（） |
| 前置条件 | 无 |
| 后置条件 | 获得行业内市盈率柱状图 |
| **AnalysisInMarketLogicservice**. shiJingLuBar | 语法 | ArrayList<AnalysisVO> shiJingLuBar (); |
| 前置条件 | 无 |
| 后置条件 | 获得行业内市净率柱状图 |
| **AnalysisInMarketLogicservice**. changeRateBar | 语法 | ArrayList<AnalysisVO> changeRateBar (); |
| 前置条件 | 无 |
| 后置条件 | 获得行业内涨幅柱状图 |
| **AnalysisInMarketLogicservice**. dealpriceBarandPie | 语法 | ArrayList<AnalysisVO> dealpriceBarandPie (); |
| 前置条件 | 无 |
| 后置条件 | 获得行业内成交金额柱状图 |
|  |  |
|  |  |
|
| 需要的服务（需接口） | | |  |
| **服务名** | | 服务 |
| **StockListDataService.** **getAllByTime** | | 返回一天16支股票的信息 |
| **StockListDataservice.** **getAllByTime** | | 返回一天16支股票的信息从txt中获取 |

**CompareLogic接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| **CompareLogic**. changeRateLine | 语法 | ArrayList<AnalysisVO<Date>> changeRateLine (String name); |
| 前置条件 | 无 |
| 后置条件 | 股票涨幅折线图 |
|
| 需要的服务（需接口） | | |  |
| **服务名** | | 服务 |
| **StockListDataService.** **getAllByTime** | | 返回一天16支股票的信息 |
| **StockListDataservice.** **getAllByTime** | | 返回一天16支股票的信息从txt中获取 |

**initialLogicservice接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| **initialLogicservice**. save() | 语法 | void save(); |
| 前置条件 | 无 |
| 后置条件 | 初始化十六支股票的信息 |
|
| 需要的服务（需接口） | | |  |
| **服务名** | | 服务 |
| **StockListDataService.** **getAllByTime** | | 返回一天16支股票的信息 |
| **StockListDataservice.** **getAllByTime** | | 返回一天16支股票的信息从txt中获取 |

**MacdcalculateLogicservice接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| **MacdcalculateLogicservice.** calculateMacd | 语法 | ArrayList<MacdVO> calculateMacd(String stockName,Date start,Date end); |
| 前置条件 | 无 |
| 后置条件 | 计算从start到end时间段，macd图所需要的数据 |
|
| 需要的服务（需接口） | | |  |
| **服务名** | | 服务 |
| **StockListDataService.** **getByName** | | 根据名称起止条件返回股票列表 |
| **StockListDataService. getResultListAll** | | 根据交易所返回股票列表信息 |
|  | |  |

## 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格式持久化的股票数据。

### 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 |
| BenchmarkDataservice | 负责读取大盘信息 |
| StockListDataservice | 负责读取股票列表信息 |
| StockCacheDataservice | 负责从缓存中读取股票信息 |

### 5.2.2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**

**BenchmarkDataservice接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| BenchmarkDataservice. getByName | 语法 | ArrayList<NBenchMarkPO> getByName(String name) |
| 前置条件 | 无 |
| 后置条件 | 根据名称返回大盘信息 |
| BenchmarkDataservice. getByTime | 语法 | ArrayList<NBenchMarkPO> getByTime(String name, String startTime, String endTime) |
| 前置条件 | 无 |
| 后置条件 | 根据名称和起止时间返回大盘信息 |

StockListDataService接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| StockListDataService. getResultListAll | **语法** | **public ArrayList<NStockPO> getResultListAll(String exchange)** |
| 前置条件 | 无 |
| 后置条件 | 根据交易所返回股票列表信息 |
| StockListDataService. getByName | 语法 | ArrayList<NStockPO> getByName(String name,String startTime , String endTime) |
| **前置条件** | **无** |
| 后置条件 | 根据名称起止条件返回股票列表信息 |
| StockListDataService. getAllByTime | 语法 | ArrayList<NStockPO> getAllByTime(String date) |
| **前置条件** | **无** |
| 后置条件 | 返回一天16支股票的信息 |
|  |  |  |

StockCacheDataservice接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| StockCacheDataservice. getByName | 语法 | ArrayList<NStockPO> getByName(String name,String startTime , String endTime) |
| **前置条件** | **无** |
| 后置条件 | 根据名称起止条件返回股票列表信息 |
| StockCacheDataservice. getAllByTime | 语法 | ArrayList<NStockPO> getAllByTime(String date) |
| **前置条件** | **无** |
| 后置条件 | 返回一天16支股票的信息 |
|  |  |  |