量化交易系统QuantGee

项目设计文档（迭代三）

学 院：南京大学软件学院

成 员：王友运 王雪 赵德宇卞纯源

完成日期：2017年6月2日

目录

[引言： 3](#_Toc476475515)

[系统的分层架构 4](#_Toc476475516)

[系统的架构设计 6](#_Toc476475517)

[logicservice接口规范： 7](#_Toc476475518)

[dataDao接口规范 8](#_Toc476475519)

[VO、PO数据规范 8](#_Toc476475520)

**更新历史：**

|  |  |  |  |
| --- | --- | --- | --- |
| **修改人员** | **日期** | **变更原因** | **版本号** |
| 全体成员 | 2017.4.29 | 最初草稿 | V1.0草稿 |
| 全体成员 | 2017.6.2 | 根据代码进行了适量改动 | V2.0 |

# 引言：

|  |  |
| --- | --- |
| 内容和说明 |  |
| **编写目的** | 本文档提供QuantGee股票数据分析系统的软件架构概览，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 |
| **对象与范围** | 本文档的读者是软工小队团队内部的开发和管理人员，参考了《软件工程与计算卷三》，用于指导下一循环的代码开发和测试工作 |
| **参考文档** | 《QuantGee需求规格说明书》 |
| **名词与术语** |  |

# 系统框架：

SSH框架

SSH是 struts+spring+hibernate的一个集成[框架](http://baike.baidu.com/item/%E6%A1%86%E6%9E%B6" \t "_blank)，是目前比较流行的一种Web应用程序开源框架。

集成SSH框架的系统从职责上分为四层：[表示层](http://baike.baidu.com/item/%E8%A1%A8%E7%A4%BA%E5%B1%82" \t "_blank)、[业务逻辑层](http://baike.baidu.com/item/%E4%B8%9A%E5%8A%A1%E9%80%BB%E8%BE%91%E5%B1%82" \t "_blank)、[数据持久层](http://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E6%8C%81%E4%B9%85%E5%B1%82" \t "_blank)和域模块层，以帮助开发人员在短期内搭建结构清晰、可复用性好、[维护](http://baike.baidu.com/item/%E7%BB%B4%E6%8A%A4" \t "_blank)方便的[Web应用程序](http://baike.baidu.com/item/Web%E5%BA%94%E7%94%A8%E7%A8%8B%E5%BA%8F" \t "_blank)。其中使用Struts作为系统的整体基础架构，负责MVC的分离，在Struts框架的[模型](http://baike.baidu.com/item/%E6%A8%A1%E5%9E%8B" \t "_blank)部分，控制业务跳转，利用Hibernate框架对[持久层](http://baike.baidu.com/item/%E6%8C%81%E4%B9%85%E5%B1%82" \t "_blank)提供支持，Spring做管理，管理struts和hibernate。具体做法是：用[面向对象](http://baike.baidu.com/item/%E9%9D%A2%E5%90%91%E5%AF%B9%E8%B1%A1" \t "_blank)的分析方法根据需求提出一些模型，将这些模型实现为基本的Java对象，然后编写基本的DAO(Data Access Objects)接口，并给出Hibernate的DAO实现，采用Hibernate架构实现的DAO类来实现[Java](http://baike.baidu.com/item/Java/85979" \t "_blank)类与数据库之间的转换和访问，最后由Spring做[管理](http://baike.baidu.com/item/%E7%AE%A1%E7%90%86" \t "_blank)，管理struts和hibernate。

系统分层架构：

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。
2. 业务逻辑层：包含业务控制和逻辑的层次。
3. 数据持久层：定义和存储系统中相关数据的层次。

业务逻辑相关组件

业务层

界面部分，包括普通UI

出错和配置界面

表示层

数据层

持久化数据；数据基本

读/写逻辑

系统可以部署在以下3个物理层次。

1. 访问层：用于用户访问系统的层次。
2. 业务层：部署业务控制和逻辑的层次。
3. 数据层：部署和存储系统中相关数据的层次。

数据库服务器

业务服务器

客户端

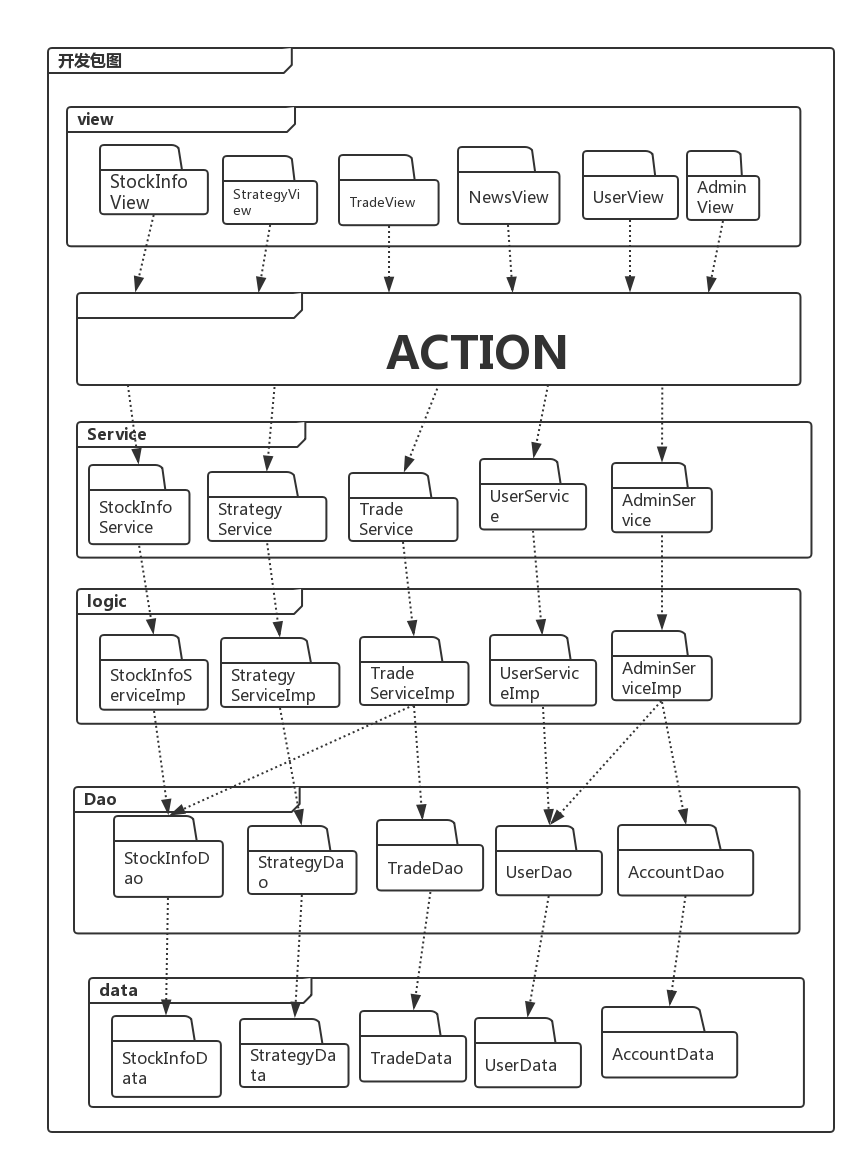
用户

访问层

业务层

数据层

# 系统的架构设计



## logicservice接口规范：

|  |  |  |
| --- | --- | --- |
| DataCalculationService接口信息 | | |
| DataCalculation  Service.  getStockInfoByCode | 语法 | StockVO getStockInfoByCode(String stockCode, Date startDate, Date endDate) |
|  | 根据股票编号获取股票信息 |
| DataCalculation  Service.  getMarketInfo | 语法 | MarketInfoVO getMarketInfo(Date data) |
|  | 根据时间获取股票市场行情数据 |
| DataCalculation  Service.  getStockInfoByName | 语法 | StockVO getStockInfoByCode(String stockName, Date startDate, Date endDate) |
|  | 根据股票名称获取股票信息 |

|  |  |  |
| --- | --- | --- |
| GraphCalculationService接口信息 | | |
| GraphCalculation Service.  getKLineInfoByCode | 语法 | ArrayList<KLineVO> getKLineInfoByCode(Date startDate, Date endDate, String StockCode) |
|  | 根据开始日期，结束日期和股票编号绘画k线图 |
| GraphCalculation Service.  getKLineInfoByName | 语法 | ArrayList<KLineVO> getKLineInfoByCode(Date startDate, Date endDate, String StockName) |
|  | 根据开始日期，结束日期和股票名称绘制K线图 |
| GraphCalculation Service.  getAverageLineInfoByCode | 语法 | ArrayList<AverageLineVO> getAverageLineInfoByCode(Date startDate, Date endDate, AverageLineType averageLine, Type, String stockCode) |
|  | 根据开始日期、结束日期、时间间隔和股票编号绘画均线图图 |
| GraphCalculation Service.  getAverageLineInfoByName | 语法 | ArrayList<AverageLineVO> getAverageLineInfoByName(Date startDate, Date endDate, AverageLineType averageLine, Type, String stockName) |
|  |  |
| StrategyCalculationService接口信息 | | |
| StratefyCalculation  Service.  getStrategyBack  TestingGraphInfo | 语法 | BackTestingResultVO getStrategyBackTestingGraphInfo  (StrategyType strategyType, StrategyInputVO strategyInputVO) |
|  | 根据策略类型和策略信息显示回测结果 |
| StratefyCalculation  Service.  getAbnormal  ReturnGraphInfo | 语法 | AbnormalReturnGraphVO getAbnormalReturnGraphInfo  (StrategyType strategyType,StrategyInputVO strategyInputVO,booleanisHoldingPeriod) |
|  | 根据策略类型、策略信息和持有形成期显示超额收益率 |

## dataDao接口规范

|  |  |  |
| --- | --- | --- |
| StockDataDao接口信息 | | |
| StockDataDao.  getStockCodeByName | 语法 | String getStockCodeByName(String stockName) |
|  | 根据股票名称获取股票信息 |
| StockDataDao .  getStockPO | 语法 | StockPO getStockPO(String date, String stockCode) |
|  | 根据日期和股票编号获取信息 |
| StockDataDao .  getStockPOsByDate | 语法 | ArrayList<StockPO> getStockPOsByDate(String date) |
|  | 根据指定时间获取当日所有股票信息 |
| StockDataDao。  getStockPOs  ByTimeInterval | 语法 | ArrayList<StockPO> getStockPOsByTimeInterval(String startDate, String endDate, String stockCode) |
|  | 根据指定开始时间和结束时间和股票编号获取信息 |
| StockDataDao.  getStockPOs  ByBlockName | 语法 | ArrayList<ArrayList<StockPO>> getStockPOsByBlockName(String startDate, String endDate, String blockName) |
|  | 根据板块名获得股票数据 |
| StockDataDao。  getAllStockPO | 语法 | ArrayList<ArrayList<StockPO>> getAllStockPO(String startDate, String endDate) |
|  | 获得所有股票数据 |
| StockDataDao.  getBaseYield  ByBlockName | 语法 | ArrayList<BaseCumulativeYielPO> getBaseYieldByBlockName(String blockName, String startDate, String endDate) |
|  | 根据板块名获取基准股票信息 |

## VO、PO数据规范

|  |  |  |
| --- | --- | --- |
| KLineVO | kLineX | 横坐标 |
| upperShadow | 上影线值 |
| lowerShadow | 下影线值 |
| maxValue | 最高值 |
| minValue | 最低值 |
| AverageLineVO | averageLineX | 横坐标（时间） |
| averageLineY | 纵坐标（均值） |
| IncreaseOr  DecreaseRateVO | rateX | 涨跌幅柱状图横坐标 |
| rateY | 涨跌幅柱状图纵坐标 |
| StockVO | stockCode | 股票代码 |
| stockName | 股票名称 |
| stockMarket | 市场名称 |
| minValue | 最低值 |
| maxValue | 最高值 |
| inOrDecreaseRate | 涨幅／跌幅 |
| closePrice | 收盘价 |
| logarithmYield | 对数收益率 |
| logarithm YieldVariance | 对数收益率方差 |
| MarketInfoVO | volume | 当日总成交量 |
| limitUpNum | 涨停股票数 |
| limitDownNum | 跌停股票数 |
| greaterThan  FiveNum | 开盘‐收盘大于5%\*上一个交易日收盘价的股票个数 |
| lessThanFiveNum | 开盘‐收盘小于‐5%\*上一个交易日收盘价的股票个数 |
| AbnormalReturn  GraphVO | bestHoldingPeriod | 最佳持有期 |
| bestReturnPeriod | 最佳形成期 |
| bestAbnormalReturn | 最优的超额收益率 |
| bestStategyWinRate | 最优的策略胜率 |
| BackTestingResultVO | cumulativeYield  GraphVO | 累计收益率图的数据 |
| yieldHistorgram  GraphVO | 频率分布直方图的数据 |
| CumulativeYield  GraphVO | annualRevenue | 年化收益率 |
| baseAnnualRevenue | 基准年化收益率 |
| alpha |  |
| beta |  |
| sharpRatio | 夏普比率 |
| maxDrawdown | 最大回撤 |
|  | cumulativeYield  GraphDataVOS | 策略收益率 |
| baseCumulativeYield  GraphDataVOS | 基准收益率 |
| StrategyInputVO | holdingPeriod | 调仓周期（持有期） |
| returnPeriod | 基准收益率计算周期  （形成期） |
| holdingStockNum | 每次调仓持有的股票数量 |
| ratio | 持有比例 |
| YieldHistogramGraphVO | positiveEarningNum | 正收益周期数 |
| negativeEarningNum | 负收益周期数 |
| winRate | 赢率 |
| yieldHistogram  GraphDataVOS | 坐标信息 |
| StockPO | date | 日期 |
| openPrice | 开盘指数 |
| maxValue | 最高指数 |
| minValue | 最低指数 |
| closePrice | 收盘指数 |
| volume | 成交量 |
| ADJ | 复权后的收盘指数 |
| stockCode | 股票代码 |
| stockName | 股票名称 |
| stockMarket | 市场名称 |
| blockName | 板块名称 |
| BaseCumulativeYielPO | date | 日期 |
|  | baseRatio | 基准收益率 |