# AnyQuant Analyse System

# 详细设计文档



**南京大学软件学院czzZ小组**

**2016/3/1**

### 文档更新记录：

|  |  |  |  |
| --- | --- | --- | --- |
| 版本 | 作者 | 描述 | 日期 |
| V1.0 | 郑韵芝 | 发布到gitlab服务器，提供给成员阅读并评审 | 2016/3/1 |
| V1.1 | 崔忠诚 | 部分内容修改更正 | 2016/3/3 |

### 目录：

目录

[AnyQuant Analyse System 1](#_Toc444806706)

[详细设计文档 1](#_Toc444806707)

[文档更新记录： 2](#_Toc444806708)

[目录： 3](#_Toc444806709)

[0.引言： 4](#_Toc444806710)

[1.词汇表： 5](#_Toc444806711)

[2.产品概述 5](#_Toc444806712)

[3.系统的分层结构 6](#_Toc444806713)

[4.系统的架构设计 7](#_Toc444806714)

[5.用户界面层的分解 9](#_Toc444806715)

[6. 业务逻辑层的分解 11](#_Toc444806716)

[7. 数据层的分解 13](#_Toc444806717)

[8. 系统的类图设计 15](#_Toc444806718)

[9.系统顺序图 16](#_Toc444806719)

[10. 接口规范 17](#_Toc444806720)

### 0.引言：

|  |  |
| --- | --- |
| 内容和说明 | 项目 |
| 1.编写目的 | 本文档提供AnyQuant Analyse System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 |
| 2.对象与范围 | 本文档的读者是czzZ团队内部的开发和管理人员，参考了RUP的《软件架构文档模板》，用于指导下一循环的代码开发和测试工作 |
| 3.参考文献 | 1.《软件需求规格说明书》，czzZ;  2.《软件架构文档模板》，Rational Software Corporation,2002;  3.The Object Management Group(OMG),The Unified Modeling Language Specification v1.4,2003；  4.《软件工程与计算（卷二）》，刘钦，丁二玉； |
| 4.名词与术语 | AAS表示本项目的股票统计分析系统简称； |

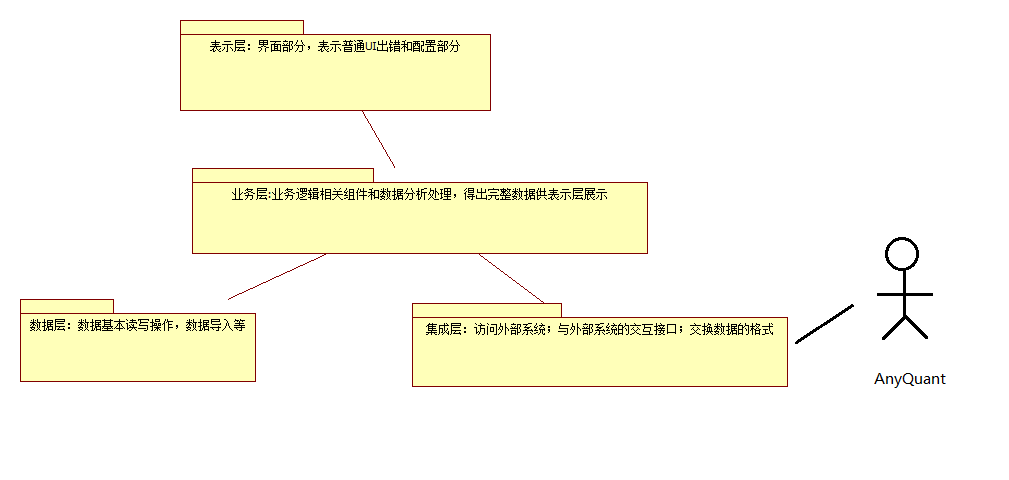
### 1.词汇表：

|  |  |  |
| --- | --- | --- |
| 词汇名称 | 词汇含义 | 备注 |
| Ui | 表示展示层 |  |
| Bl | 表示逻辑层 |  |
| Data | 表示数据层 |  |
| AAS | 表示AnyQuant Analyse System系统 |  |

### 2.产品概述

AAS系统是用于查询显示近期股票并能对股票信息进行筛选并统计分析的应用软件系统，主要由Java语言开发，利用数据库、web信息获取等相关方式配合多种复杂数据逻辑统计计算，为用户提供一个多角度全方位的股票查询统计系统，良好的用户交互体验和准确全面的数据分析，完成该系统的服务目标。

### 3.系统的分层结构



系统划分为以下4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

在本次迭代中可以部署在以下的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）集成层：部署和获取系统外部相关数据的层次

### 4.系统的架构设计

系统架构中的对象分为7类：  
 1）UI对象，负责处理系统数据的展现和用户的交互

2）IController对象，控制器负责获取用户输入，并调用IService模块的服务

3）IService对象，负责提供服务的抽象接口，获取从数据端组装好的数据

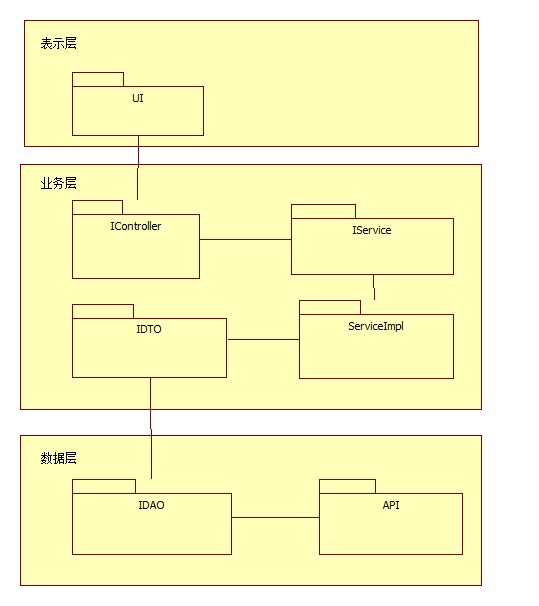
4）ServiceImpl对象，负责对于抽象接口的实现模块

5）IDTO(Data Transfer Object)对象，负责封装从IDAO获取的批量数据的接口

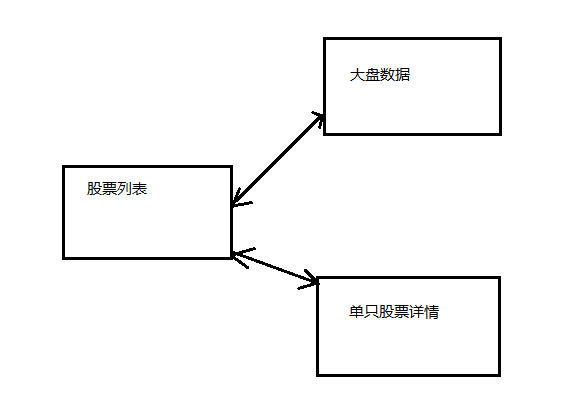
6）IDAO(Data Access Object)对象，负责与数据库实体交互，获取数据

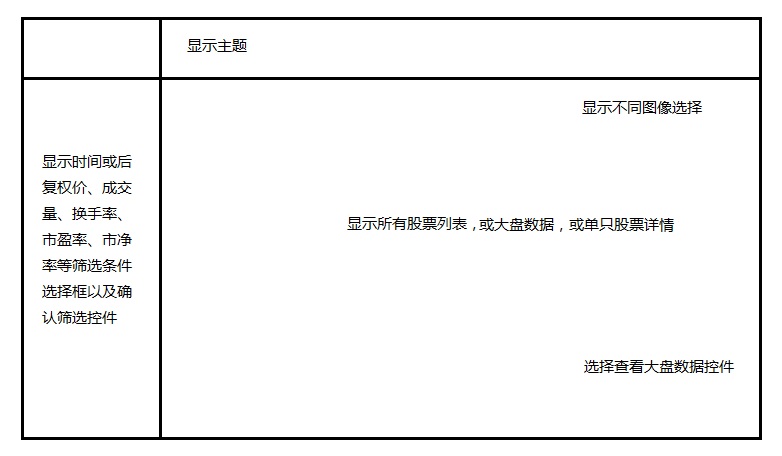
7）API对象，负责提供服务的抽象接口，从web网页中获取数据

系统中的组件和组件接口：



### 5.用户界面层的分解

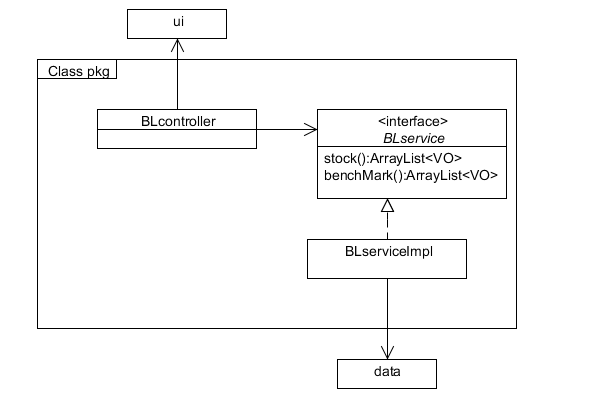






### 业务逻辑层的分解

业务逻辑层为针对具体问题的操作，也可以理解为对集成层的操作，是对数据进行业务处理，包括对界面层所需要服务的接口提供以及对集成层的调用。业务逻辑模块的具体描述如图：



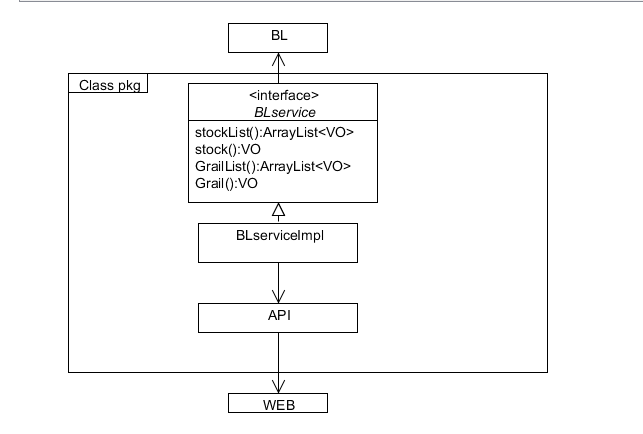
接口组件设计结构如下表：

|  |  |  |  |
| --- | --- | --- | --- |
| 接口ID | 连接组件 | 接口信息 | |
| I1 | 连接UI与BLcontroller | 语法 | Return():ArrayList<VO> |
| 前置条件 | 用户的输入正确 |
| 后置条件 | 处理控制组件处理请求并且响应 |
| 不变量 | 用户请求信息 |
| I2 | 连接BLcontroller与BLservice | 语法 | Return():ArrayList<VO> |
| 前置条件 | 无 |
| 后置条件 | 对应的BLService执行对应的业务逻辑 |
| 不变量 | 无 |
| I3 | 连接BLservice与dataservice | 语法 | Return():ArrayList<VO> |
| 前置条件 | 无 |
| 后置条件 | 对应的data模块组件调用特定类获取API数据，并返回数据集 |
| 不变量 | 无 |

### 数据层的分解

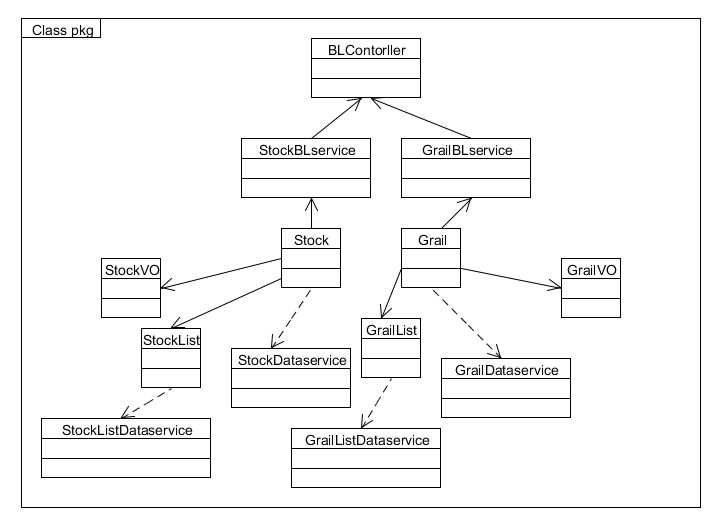
数据层访问外部系统，使用API接口与网络进行交互，返回JSON数据并进行数据格式的处理，将数据以VO或ArrayList<VO>的形式返回给业务逻辑层。

数据层的具体描述如图：

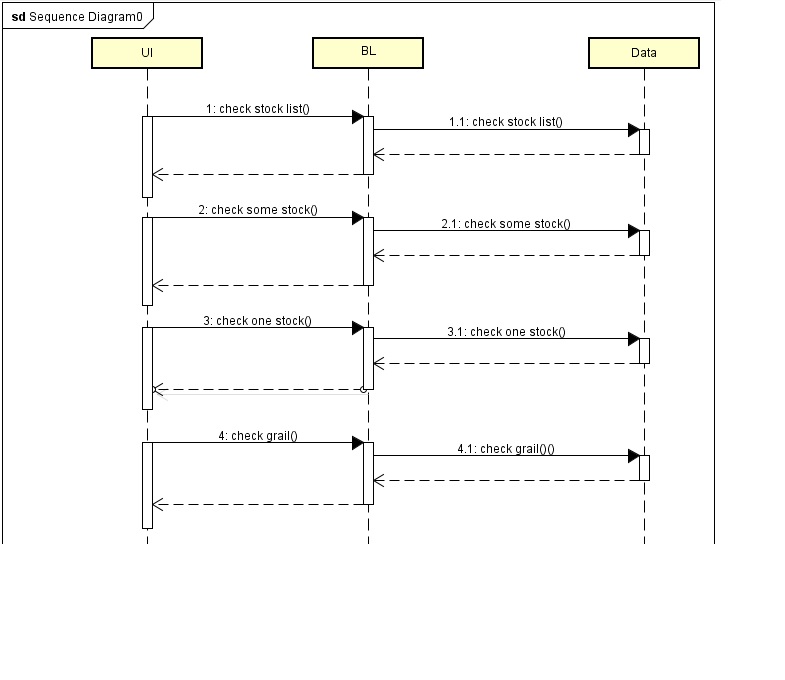


### 系统的类图设计

设计如下：



### 9.系统顺序图



### 接口规范

StockList\_service的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务 | | |
| StockList\_service.ShowAll | 语法 | public ArrayList<StockVO> ShowAll(); |
| 前置条件 | 已经创建一个StockList\_Impl领域对象 |
| 后置条件 | 显示股票列表 |
| StockList\_service.Select | 语法 | public ArrayList<StockVO> Select(String id); |
| 前置条件 | 已经创建一个StockList\_Impl领域对象 |
| 后置条件 | 显示选中股票的详细信息 |
| StockList\_service.Search | 语法 | public ArrayList<StockVO> Search(String id); |
| 前置条件 | 已经创建一个StockList\_Impl领域对象 |
| 后置条件 | 根据id搜索股票详细信息 |
| StockList\_service.Refresh | 语法 | public void Refresh(); |
| 前置条件 | 已经创建一个StockList\_Impl领域对象 |
| 后置条件 | 刷新股票列表 |
| StockList\_service.filter | 语法 | public ArrayList<StockVO> filter(String F); |
| 前置条件 | 已经创建一个StockList\_Impl领域对象 |
| 后置条件 | 根据条件筛选股票列表 |
| StockList\_service.setTime | 语法 | public ArrayList<StockVO> setTime(String time); |
| 前置条件 | 已经创建一个StockList\_Impl领域对象 |
| 后置条件 | 根据时间筛选股票列表 |
| 需要的服务 | | |
| 服务名 | | 服务 |
| StockListData\_service.StockList | | 传输相应股票列表 |

SingleStock\_service的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的接口 | | |
| SingleStock\_service.Show | 语法 | public ArrayList<StockVO> Show(String id); |
| 前置条件 | 已经创建一个SingleStock\_Impl领域对象 |
| 后置条件 | 显示某支股票详细信息 |
| SingleStock\_service.close | 语法 | public void close(); |
| 前置条件 | 已经创建一个SingleStock\_Impl领域对象 |
| 后置条件 | 返回股票列表界面 |
| SingleStock\_service.setTime | 语法 | public ArrayList<StockVO> setTime(String id,String time); |
| 前置条件 | 已经创建一个SingleStock\_Impl领域对象 |
| 后置条件 | 根据时间显示单支股票详细信息 |
| 需要的服务 | | |
| 服务名 | | 服务 |
| SingleStockData\_service.StockList | | 传输相应股票详细信息 |

Grail\_service的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的接口 | | | |
| Grail\_service.ShowGrailList | 语法 | | Public ArrayList<GrailVO> ShowGrailList(); |
| 前置条件 | | 已经创建一个Grail\_Impl领域对象 |
| 后置条件 | | 显示大盘信息列表 |
| Grail\_service.setTime | 语法 | | public ArrayList<GrailVO> setTime(String time); |
| 前置条件 | | 已经创建一个Grail\_Impl领域对象 |
| 后置条件 | | 显示要求时间段内的大盘信息 |
| Grail\_service.close | 语法 | | public void close(); |
| 前置条件 | | 已经创建一个Grail\_Impl领域对象 |
| 后置条件 | | 返回股票列表 |
| Grail\_service.filter | 语法 | | public ArrayList<GrailVO> filter(String F); |
| 前置条件 | | 已经创建一个Grail\_Impl领域对象 |
| 后置条件 | | 对大盘数据进行筛选 |
| 需要的服务 | | | |
| 服务名 | | 服务 | |
| GrailData\_service.StockList | | 传输相应股票详细信息 | |

StockListData\_service的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的接口 | | | |
| StockListData\_service.StockList | 语法 | | public ArrayList<StockVO> StockList(); |
| 前置条件 | | 已经创建一个StockListData\_Impl领域对象 |
| 后置条件 | | 获得股票信息列表 |
| 需要的服务 | | | |
| 服务名 | | 服务 | |
| API.GetData | | 传输相应股票详细信息 | |

SingleStockData\_service的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的接口 | | | |
| SingleStockData\_service.Singlestock | 语法 | | public ArrayList<StockVO> SingleStock(String time); |
| 前置条件 | | 已经创建一个SingleStockData\_Impl领域对象 |
| 后置条件 | | 获得股票信息列表 |
| 需要的服务 | | | |
| 服务名 | | 服务 | |
| API.GetData | | 传输相应股票详细信息 | |

GrailListData\_service的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的接口 | | | |
| GrailListData\_service.GrailList | 语法 | | public ArrayList<GrailVO> GrailList(String time); |
| 前置条件 | | 已经创建一个GrailListData\_Impl领域对象 |
| 后置条件 | | 获得股票信息列表 |
| 需要的服务 | | | |
| 服务名 | | 服务 | |
| API.GetData | | 传输相应股票详细信息 | |