# 小组成员

|  |  |
| --- | --- |
| 姓名 | 学号 |
| 肖瑞 | 131250002 |
| 张亚飞 | 131250023 |
| 赖铭津 | 131250032 |
| 郭玥 | 131250033 |
| 张怡 | 131250036 |
| 曾春晖（PM） | 131250049 |
| 许露佳 | 131250104 |
| 雷莳芳 | 131250107 |

# 模块分工

|  |  |
| --- | --- |
| 模块 | 设计者 |
| GoodsSearch | **许露佳** |
| GoodsSort | **许露佳** |
| Shield | **雷莳芳** |
| PriceParity | **雷莳芳** |
| SafeGuard | **肖瑞** |
| Crawler | **张怡** |
| CrawlerDataUpdate | **张怡** |
| CrawlAdapter | **张亚飞** |
| BrowerAdapter | **曾春晖** |
| CommentCheck | **曾春晖** |
| Pay | **赖铭津** |
| Advertisement | **赖铭津** |
| DataBase | **郭玥** |
| 数据库表设计 | **郭玥** |

# GoodsSearch模块设计

## 概述

**承担的需求（主要秘密）**

根据用户输入的关键字与数据库中的商品信息进行快速匹配，返回搜索的结果

本模块需要对输入的关键字进行预处理，将一些无用的词汇删除，如英语的a、an、the，中文的了，啊，呢，吧，哦等等

本模块需要将输入的关键词在同义词表中找出相对应的同义词，并根据同义词进行搜索，得到检索列表

**可能会修改的实现（次要秘密）**

同义词匹配算法

检索匹配算法

输入词汇预处理算法

**角色**

**对外接口**

## 类图

## 类描述

（类方法（重要方法要给出前置与后置条件）与重要数据结构描述）

## 重要协作

（顺序图与协作描述）

## 使用的设计模式

（使用场景；所用的设计模式；要达到的效果）

# GoodsSort模块设计

## 概述

**承担的需求（主要秘密）**

用户输入关键字并向系统提交查询请求后，系统在查询结果页面根据如下的排序规则对结果进行排序并展示给用户：

根据用户输入的关键词的相关度进行排序，关键字越吻合，相关度越大，排名越靠前

根据来源的商城的可靠度进行排序，商城知名度越大，关键字相关度一致的条件下排名越靠前

允许使用竞价排序规则调整排序

**可能会修改的实现（次要秘密）**

排序算法

竞价规则排序策略

**角色**

**对外接口**

## 类图

## 类描述

## 重要协作

## 使用的设计模式

# Shield模块

## 概述

### 承担的需求（主要秘密）

商品屏蔽模块的职责是屏蔽与搜索关键字匹配的商品列表中的负面商品，因为负面商品可以灵活的添加和修改，且屏蔽规则必须符合法律法规，故要求屏蔽规则具有高度的可修改性，可扩展性和灵活性。

**质量属性要求**：商品搜索要求具有很高的可修改性、可扩展性，而变更的部分主要是待屏蔽商品信息的变化。因此该模块的设计就要着重考虑词库的可扩展性和可修改性，可以灵活变更又不影响其他模块的调用。同时，由于搜索返回结果往往比较庞大，待屏蔽商品库也较庞大，因此商品过滤效率也直接影响到用户体验，需要尽量采用高效的过滤算法。用户总共进行商品查询的时间不能多于2s。

**商品屏蔽用例图**



**用例描述**

1. 负面商品信息增删改

系统管理员可以在后台管理界面灵活地增加、修改、删除负面商品。

1. 进行链接过滤

当一个商品被加入负面商品信息表中后，网页上指向该商品的连接必须失效，可以通过url内容比对结合访问控制来实现。

1. 进行内容过滤

当用户输入搜索关键字后，系统将检测搜索到的商品列表，并屏蔽其中的负面商品。

可能会修改的实现（次要秘密）

负面商品信息表

商品过滤规则

**角色**

**向外提供接口**

|  |  |
| --- | --- |
| **接口名** | **职责** |
| Public GoodsList search(Keywords keywords) | 根据输入的关键词，先调用底层模块找到符合关键词的商品列表，再进行过滤，返回符合规则的商品列表。 |
| Public URL filter(URL goodsURL) | 根据输入的URL判断是否指向被屏蔽的商品，若被屏蔽，则重定向到另一个URL，否则返回原URL。 |
| Public void addShieldGoods(Goods goods) | 增加一条负面商品信息 |
| Public void updateShieldGoods(Goods goods) | 修改一条负面商品信息 |
| Public void deleteShieldGoods(Goods goods) | 删除一条负面商品信息 |

## 类图

## 类描述

### XX类

|  |  |
| --- | --- |
| 方法名 |  |
| 前置条件 |  |
| 后置条件 |  |
| 作用 |  |

## 重要协作

## 设计模式

# PriceParity模块

## 概述

### 承担的需求（主要秘密）

商品比价模块的职责是对同一个商品，比较其在不同购物网站上的价格，并可以按照价格进行升序和降序排序。要求可以灵活地定义参与比价的网站，在未来可能会增加不同的价格计算规则，因此比价模块应该具有高度的可修改性，可扩展性和灵活性。

**商品比价用例图**



**用例描述**

1. 展示不同购物网站上的商品价格

用户查看某一商品详细信息时，系统应显示该商品在不同网站上的价格，且用户可以自定义参与比价的购物网站（从本系统提供的购物网站中选择）。

1. 根据价格排序

用户在查看某一商品时，可以点击“按价格排序”，系统将不同购物网站按商品价格升序或降序排序。

可能会修改的实现（次要秘密）

参与比价的购物网站。

价格计算规则，如是否包括运费，是否可以使用优惠券等等。

**角色**

**向外提供接口**

|  |  |
| --- | --- |
| **接口名** | **职责** |
| Public GoodsPriceList findNamesake(Goods goods) | 根据该商品查找不同购物网站上的同名商品，返回同名商品列表 |
| Public GoodsPriceList sort(GoodsPriceList goodsPriceList,Order order) | 根据Order为升序还是降序，将参数中的goodsList按价格排序 |
| Public void addShoppingSite (ShoppingSite site) | 对当前用户，增加一个参与比价的比价网站 |
| Public void deleteShoppingSite (ShoppingSite site) | 对当前用户，删除一个参与比价的比价网站 |

## 类图



## 类描述

### PriceParity接口

|  |  |
| --- | --- |
| 方法名 | Public GoodsPriceList findNamesake(Goods goods) |
| 前置条件 |  |
| 后置条件 |  |
| 作用 |  |

### PriceParityImpl类

|  |  |
| --- | --- |
| 方法名 |  |
| 前置条件 |  |
| 后置条件 |  |
| 作用 |  |

## 重要协作

## 设计模式

# SafeGuard模块详细设计

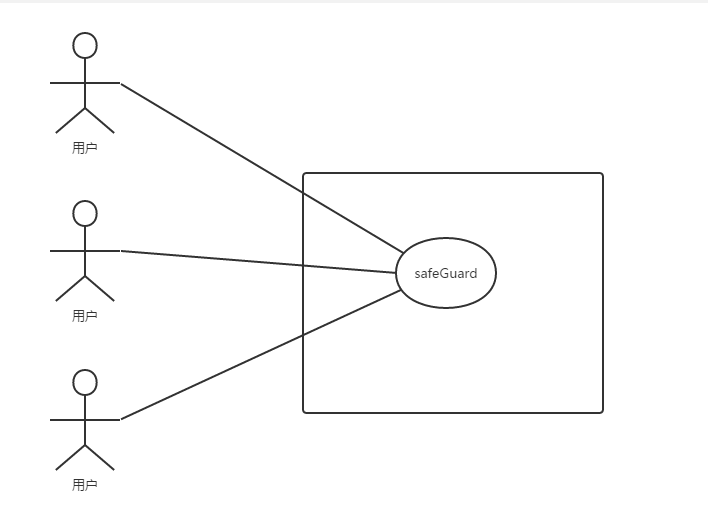
## 概述

模块需求：该模块负责保证服务器安全。用户访问服务器时需要先通过服务器检测模块，屏蔽特定IP以及访问过于频繁的IP（插件扫描攻击）。并且该模块支持可能的入侵检测规则扩展。

涉及需求：

R3 系统安全性 屏蔽99%以上的插件扫描攻击和特定的IP 促进C4

R4 入侵检测可拓展性 增加新的检测规则所需时间低于0.25人月



说明：用户的请求到达SearchProcessor模块前需要先通过SafeGuard模块进行处理

**可能会发生的修改实现**

所屏蔽的特定IP的列表

系统安全检测方法

**分配职责**

1. 该模块将用户访问中可能出现的不安全行为屏蔽，确保服务器的安全和效率

**对外接口：**

接口：Int doSafeCuard(HttpRequest request） ()

功能：当用户的请求到达系统时，首先调用该接口，该接口将返回对该IP访问的检查结果，如果检查通过，则调用模块继续向下调用，否则屏蔽该次访问

**参数说明：**

HttpRequest request：用户请求信息，含有用户的IP、访问时间、mac地址、访问目标等

**返回值说明：**从0至5，分别表示危险评级，屏蔽3级及以上危险评级的访问

**使用方法：**

每当一个用户向服务器请求数据的时候，调用该模块检查用户是否符合访问安全要求，

**前置条件：**

传入的request中包含用户IP、mac地址、访问时间、访问目标

**后置条件**

1、返回对该次访问的安全评级

2、对储存的数据进行修改，更改某些IP地址的访问记录以及安全评价

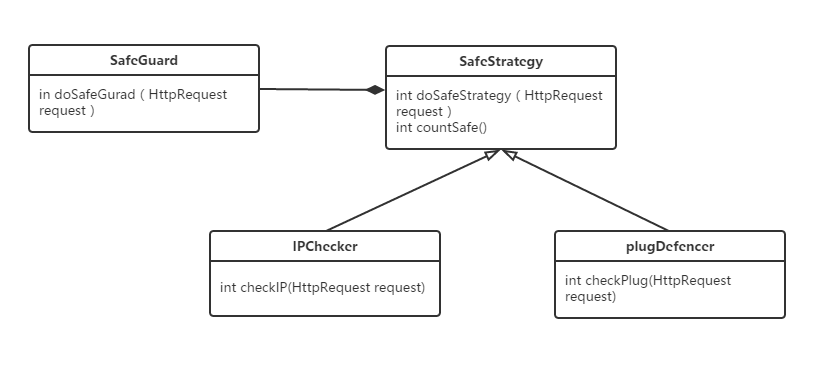
**设计方案分析**

用户的请求到达的时候，SafeGurad模块调用SafeStrategy模块进行访问分析，SafeStrategy模块则依此调用继承自SafeStrategy模块的安全策略模块，目前拥有的是IPCheck模块和PlugDefence模块

## 2类图

**静态结构**

**详细设计类图和类图说明：**



设计说明：该模块主要由两个部分组成：SafeGuard 和SafeStrategy

HttpRequest通过SafeGuard类传入SafeStrategy类，再通过SafeStrategy类调用其子类，最后通过子类返回值计算出该次request的安全评级并返回

## 3类描述

**类方法（重要方法要给出前置与后置条件）与重要数据结构描述**

|  |  |
| --- | --- |
| 类 | 说明 |
| SafeGuard | 传入HttpRequest对象，向下传递并获取返回结果 |
| SafeStrategy | 安全策略类，为所有策略的父类，调用所有策略，计算结果并返回 |
| IPChecker | 策略类的子类，检查IP是否在屏蔽列表内 |
| PlugDefencer | 策略类的子类，检查该次访问是否是插件扫描攻击 |

|  |  |
| --- | --- |
| SafeGuard | |
| DoSafeGuard | 获取HttpRequest向下传递并获取返回值 |

|  |  |
| --- | --- |
| SafeStrategy | |
| doSafeStrategy | 调用所有的检查策略，并返回该次访问的危险值 |
| CountSafe | 计算该访问的危险并返回 |

|  |  |
| --- | --- |
| IPChecker | |
| checkIP | 检查IP是否合法 |

|  |  |
| --- | --- |
| PlugDefencer | |
| checkPlug | 检查该次访问是否是插件扫描攻击 |

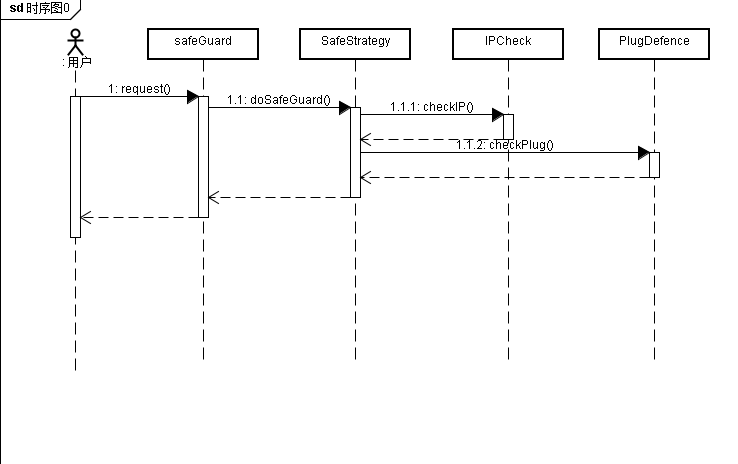
**重要的数据结构描述**

用户请求

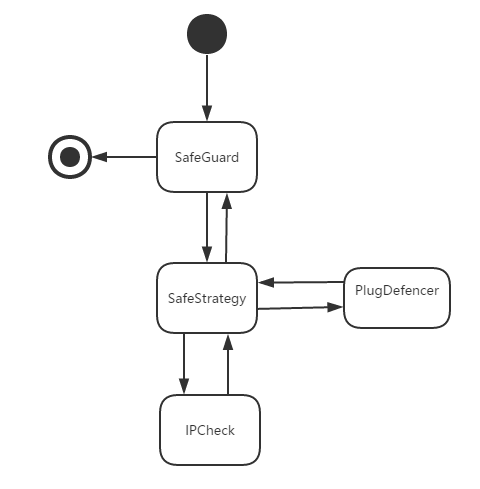
HttpRequest：其中有用户IP、mac地址、访问时间、访问目标

## 4重要协作

**顺序图**



**状态图**



## 5使用的设计模式

**策略模式：**模块含有一个策略的调用者SafeStrategy和若干个继承自SafeStrategy的安全策略类，SafeGuard调用SafeStrategy类，由safeStrategy类调用其记录的所有策略类。当安全检查策略发生修改或删除时，只需修改对应的策略类，当增加安全策略时，只需要增加安全策略类，并在SafeStrategy类中增加该类的调用

**场景：**检查用户的访问安全

**目的：**屏蔽不安全的请求

1. Crawler模块

1.1 概述

**承担的需求（主要秘密）：**

该模块主要承担的任务是从其他网站上抓（爬）取商品价格信息，抓（爬）取策略的选择和配置以及周期性抓取新数据以保证数据有效性。

模块需要实现自动抓（爬）取指定网页的功能，初步过滤掉无效信息，然后将商品相关的信息交给数据处理模块处理。对一个网页进行数据抓（爬）取时，将页面中的有效链接放入到一个URL抓（爬）取队列中，在当前网页抓（爬）取完毕后，从URL抓（爬）取队列中选取下一个URL作为抓（爬）取对象。以此操作，直到URL抓（爬）取队列为空。

模块需要实现抓（爬）取策略可变更的功能，能够根据爬虫管理员的选择来采取相应的网页抓（爬）取策略。抓（爬）取策略存储和配置存储在配置文件中。当系统初始化或者抓（爬）取策略变更时，爬虫类从配置文件中读取选定的抓（爬）取策略，然后开始网站抓（爬）取工作。

模块需要实现一个可配置的定时器来完成周期性抓（爬）取的功能。定时器的各项配置保存在配置文件中。爬虫管理员需要提前设定好定时器的周期。在开启爬虫程序后，一旦定时器达到指定周期，就会触发爬虫去抓（爬）取网站信息。在爬虫程序运行过程中，爬虫管理员可以随时改变定时器的周期，然后定时器的配置文件会做出相应的更改，然后在适当的时间（即爬虫程序空闲时）重启爬虫程序来使配置生效。

**可能会修改的实现（次要秘密）：**

增加新的网站抓（爬）取源时，需要增加新的网页内容抓（爬）取方式。

增加新的网站抓（爬）取策略时，需要增加新的抓（爬）取策略实现。

**角色：**

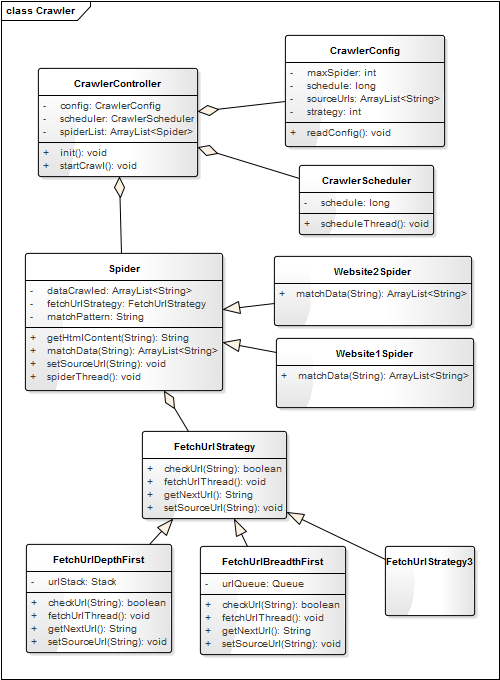
在整个系统中扮演商品数据信息获取的角色，获取的数据交给商品兼容处理模块。

**对外接口：**

1）void analyseData();（参数为从网站爬取的原始商品信息）

提供给商品数据兼容处理模块，目的是让其能够获取从网站上爬取的原始商品信息。

1.2 类图



* 1. 类描述

**CrawlerController类：**爬虫模块的集中控制类，包括爬虫配置文件读取、爬虫定时器的启动、初始化爬虫和开始爬虫程序。类中维护了一个爬虫列表。

**Init():void----**爬虫程序初始化方法，通过CrawlerConfig读取配置文件来获取一些运行时的基本参数。

**startCrawl():void----**爬虫程序的启动方法，里面根据config里的基本参数来新建爬虫对象，开启爬虫线程。

**CrawlerConfig类：**爬虫模块配置类，负责读取配置文件中的爬虫配置，从而获得爬虫程序的启动参数。具体参数包括爬取网站源、爬取周期、爬取策略、最大爬虫数量等。

**readConfig():void----**读取配置文件，获得基本的爬虫参数。

**CrawlerScheduler类：**爬虫模块的定时器类，负责周期性启动爬虫程序。

**scheduleThread():void----**开启定时器的计时线程，当一个周期到时，能够重新开启爬虫程序来爬取数据。

**Spider类：**爬虫模块的抓取数据类，负责根据给定的URL抓取网页的信息，然后从中提取需要的商品信息。它是具体网站信息抓取类Website1Spider、Website2Spider类的父类，包含了爬虫类的统一属性和方法。

**setSourceUrl(String):void----**设置待抓取网站的源URL，给URL爬取类使用。

**spiderThread():void----**启动爬虫页面爬取类的线程。

**getHtmlContent(String):String----**根据URL获取网页的html内容，作为返回值返回，供后面提取商品信息用。

**matchData(String):ArrayList<String>----**根据匹配规则筛选出抓取页面中的商品信息。

**WebSite1Spider类：**是适用于具体网站Website1的爬虫类。

**matchData(String):ArrayList<String>----**根据适用于Website1的信息匹配规则，提取出页面中的商品信息。

其他方法与父类中相同。

**Website2Spider类：**是适用于具体网站Website2的爬虫类。

**matchData(String):ArrayList<String>----**据适用于Website2的信息匹配规则，提取出页面中的商品信息。

其他方法与父类中相同。

**FetchUrlStrategy类：**爬虫模块的URL爬取类，负责爬取给定网站所包含的所有URL，并且对URL进行去重。爬取到的URL提供给爬虫类进行网页数据的抓取。

其中的方法为抽象方法，供具体策略类实现。

**FetchUrlDepthFirst类：**URL爬取类的具体实现，运用了深度优先遍历的URL爬取策略。

**setSourceUrl(String)----**设置待抓取网站的源URL，作为初始URL爬取对象。

**fetchUrlThread():void----**开始URL爬取线程，爬取的URL放入URL栈中。

**checkUrl(String):Boolean----**检查爬取的URL是否重复，不重复则加入URL栈中。

**getNextUrl():String----**爬虫类从URL栈中获取下一条可抓取信息的URL。

**FetchUrlBreadthFirst类：**URL爬取类的具体实现，运用了广度优先遍历的URL爬取策略。

**setSourceUrl(String)----** 设置待抓取网站的源URL，作为初始URL爬取对象。

**fetchUrlThread():void----**开始URL爬取线程，爬取的URL放入URL队列中。

**checkUrl(String):Boolean----**检查爬取的URL是否重复，不重复则加入URL队列中。

**getNextUrl():String----**爬虫类从URL队列中获取下一条可抓取信息的URL。

**FetchUrlStrategy3类：**URL爬取类的具体实现，运用了其他的URL爬取策略（Strategy3）。

* 1. 重要协作

**顺序图**

**协作描述**

* 1. 使用的设计模式

**使用场景**

**所用设计模式**

**要达到的效果**

# CrawlerDataUpdate模块

2.1 概述

**承担的需求（主要秘密）：**

该模块主要承担的任务是将规范化后的商品数据增量更新到数据库中。由于商品信息中不同的数据改变频率不同（比如商品描述变化少，商品价格、数量变化频繁），该模块会对不同的部分采取不同的更新策略，以此来提高数据更新的效率。

**可能会修改的实现（次要秘密）：**

可能会新增不同的数据更新方式和策略。

可能会修改数据库表结构的设计。

**角色：**

在爬虫部分中扮演数据更新的角色，负责将商品兼容模块处理后的数据更新进数据库。

**对外接口：**

1）void giveFormattedData();（参数为分析处理后的商品数据）

提供给商品数据兼容处理模块，目的是获取处理后的商品数据，从而更新到数据库中。

2.2 类图

* 1. 类描述

**类方法：**

**重要数据结构描述：**

* 1. 重要协作

**顺序图**

**协作描述**

* 1. 使用的设计模式

**使用场景**

**所用设计模式**

**要达到的效果**