Online-bug-detection-engine-for-java-programs

软件构架文档 V2.0

2009-9-7

上海交通大学软件学院

Baldur 软件开发小组

修订记录

|  |  |  |  |
| --- | --- | --- | --- |
| **时间** | **版本** | **描述** | **作者** |
| 2009-7-10 | v1.0 | 初始版本 | 胡超 |
| 2009-9-7 | V2.0 | 第二版本 | 季尚松 |
|  |  |  |  |
|  |  |  |  |

目录

[目录 2](#_Toc236641960)

[1 简介 4](#_Toc236641961)

[1.1 目的 4](#_Toc236641962)

[1.2 范围 4](#_Toc236641963)

[1.3 定义、缩写词、术语 4](#_Toc236641964)

[1.4 参考资料 4](#_Toc236641965)

[1.5 概述 4](#_Toc236641966)

[2 整体说明 5](#_Toc236641967)

[2.1 产品整体效果 5](#_Toc236641968)

[2.2 功能摘要 5](#_Toc236641969)

[用户支持系统 5](#_Toc236641970)

[2.3 用户特征 5](#_Toc236641971)

[2.4 约束 5](#_Toc236641972)

[2.5 假设与依赖关系 6](#_Toc236641973)

[3 具体需求 6](#_Toc236641974)

[3.1 主要功能 6](#_Toc236641975)

[3.1.1 用户注册 7](#_Toc236641976)

[3.1.2 提交文件的单一文件纠错 8](#_Toc236641977)

[3.1.3 粘贴代码的单一文件纠错 10](#_Toc236641978)

[3.1.4 查看缺陷信息 12](#_Toc236641979)

[3.1.5 统计缺陷 12](#_Toc236641980)

[3.1.6 项目纠错 13](#_Toc236641981)

[3.2 可用性 15](#_Toc236641982)

[3.2.1 可靠性 15](#_Toc236641983)

[3.2.2 性能 15](#_Toc236641984)

[3.2.3 可支持性 16](#_Toc236641985)

[3.3 设计约束 16](#_Toc236641986)

[3.4 联机用户文档和帮助系统需求 16](#_Toc236641987)

[3.5 接口 16](#_Toc236641988)

[3.6 适用标准 16](#_Toc236641989)

[4 支持信息 16](#_Toc236641990)

# 简介

## 目的

软件构架文档是我们在设计阶段的重要产品，它将会对我们的Java程序在线纠错引擎（Online-bug-detection-engine-for-Java-program）的整体构架做一个概述。该文档提供开发人员一个日后实现的参考以及其他涉众一个了解本项目内部构架的一个参考。

## 范围

此文档将会影响Baldur开发小组、的应用程序、特性、与其相关的用例模型以及日后开发过程中的所有文档。

## 定义、缩写词、术语

|  |  |
| --- | --- |
| Baldur | 本项目的开发团队，成员包括徐昊，季尚松，胡超。 |
| FindBugs | 一个对JAVA字节码进行BUG分析的工具。 |
| PMD | 一个对JAVA源文件进行BUG分析的工具 |

## 参考资料

|  |  |  |
| --- | --- | --- |
| **名称** | **来源** | **日期** |
| 软件构架文档模板 | 上海交通大学软件学院 | 2008-3-12 |
| FindBugs User Manual | FindBugs 官方网站 | 2009-6-4 |
| PMD User Manual | PMD 官方网站 | 2009-2-8 |

## 概述

本文档的以下部分的内容依次为：

**构架表示方式：**说明当前系统所使用的软件构架及其表示方式。

**构架设计目标和约束：**说明对构架具有某种重要影响的软件需求和目标。

**用例视图：**列出用例模型中的一些用例或场景。

**逻辑视图：**明设计模型在构架方面具有重要意义的部分。

**进程视图：**说明将系统分解为轻量级进程（单个控制线程）和重量级进程（成组的轻量级进程）的情况。

**部署视图：**说明用来部署和运行该软件的一种或多种物理网络（硬件）配置。

**实施视图：**说明实施模型的整体结构以及所有在构架方面具有重要意义的构件。

**数据视图：**说明永久性数据的存储情况。

**大小和性能：**说明软件中会对构架产生影响的主要尺寸特征，以及目标性能约束。

**质量：**说明软件构架如何促成诸如可扩展性、可靠性、可移植性等所有系统能力（而非功能）的实现。

# 构架表示方式

本文档以一系列的视图（View）来表示系统的软件构架，主要包括用例视图、逻辑视图、进程视图、部署视图、实施视图（即RUP推荐的4+1视图）等；每个视图拥有一个或多个模型（Model）（例如逻辑视图包含分析模型、设计模型和数据模型等）；并围绕相关视图来描述系统的基本结构、组成机制与工作原理等。

本文档还将系统的构架机制描述也放在了逻辑视图之下。

本文档主要使用统一建模语言（UML）来充当相关模型的表达语言。

# 构架设计目标和约束

描述构架设计必须满足的关键系统功能需求和质量约束，这些功能需求和质量要求对软件构架有重大的影响，并决定了构架的设计。

本节同时还列明影响构架的其他相关因素，如软件的复用策略、使用商业构件、设计与实施的策略等。

## 关键功能需求

* 用户能够通过Internet访问本在线纠错引擎。
* 用户能够通过本引擎进行对Java程序的在线纠错。
* 用户能够在线查看自己和全局的缺陷统计。

## 关键质量要求

* + 1. **有效性**
* 系统平均可用时间大于99.99%。
  + 1. **性能**
* 系统并发用户在线数大于50。
* 普通数据录入、查找等操作，每单步操作最大延迟时间应小于5秒。
* 一般查询统计，结果集在100条记录以内情况下，最大延迟时间不超过30秒。
* 所有统计，其最大延迟时间不超过2分钟。
  + 1. **性能可扩展**
* 支持调用本地纠错程序方式的更改。
  + 1. **功能可扩展**
* 支持对其他语言纠错的扩展。

## 开发策略

* + 1. 软件复用策略
* 系统中重要基础构件应当具备较高的设计与构建质量，可以在产品中复用。
  + 1. 使用开源构件
* 使用时下比较流行的Java程序静态分析工具。
* 单元测试使用junit框架。
  + 1. 使用商业构件
* 不适用。

## 其它设计约束

* 目标构架总体上应采用分层结构，并全面应用面向对象设计、编程技术使系统具有较好的扩展性与重用性。
* 本系统支持与其他系统进行集成，所以要提取出良好的集成接口。

# 用例视图



## 用户注册

1. 简述

该用例允许未注册用户进行注册.

1. 流程
   1. 基本流

用户点击’ Bug Detect’ 链接, 系统跳转至’ Debug ’ 页面’ , 随后点击’ register’ 链接, 系统跳转至’ register’ 页面, 根据输入框上方的提示输入系统所需要的信息, 系统根据用户所提交的信息在数据库中生成一条新的数据, 系统跳转至注册用户页面, 注册成功.

* 1. 备选流
     1. 用户名已存在

点击’submit’按钮后,由于用户名已经存在, 系统将会显示错误信息,用户可以选择继续注册或是结束注册, 如果不继续注册,则用例失败.

* + 1. 两次输入的密码错误

点击’submit’按钮后,由于两次输入的密码不一致, 系统将会显示错误信息,用户可以选择继续注册或是结束注册, 如果不继续注册,则用例失败.

* + 1. 用户信息填写不完整

点击’submit’按钮后,由于所填写的信息, 系统将会显示错误信息, 提示用户哪些信息没有填写. 用户可以选择继续注册或是结束注册, 如果不继续注册,则用例失败.

1. 特殊需求

无

1. 前置条件

无

1. 后置条件

无

1. 扩展点

无



## 提交文件的单一文件纠错

1. 简述

该用例允许用户提交单个文件的源文件进行纠错

1. 流程
   1. 基本流

用户点击’ Bug Detect’ 链接, 系统跳转至’ Debug ’ 页面’ , 随后点击’ find bugs in a single file’ 链接, 系统跳转至’ singlereport’ 页面, 点击’浏览’ 按钮, 选择想要提交的文件后, 点击’upload’按钮, 系统调用服务器中的PMD程序对该文件进行纠错,生成报告后跳转至createSingleReport页面并将报告输出在该页面,用例成功.

* 1. 备选流
     1. 该文件通不过编译

如果该文件通不过编译，则将编译错误输出到createSingleReport页面中。

* + - 1. 用户选择不做下一步动作

用例失败.

* + - 1. 用户对代码进行修改

用户点击报告中的链接,可以对上传的文件进行修改.修改后可以选择删除文件或是保存修改. 如果保存修改结果, 则重新进行编译,如果选择删建,则将刚才提交时生成的临时文件删除.

* + 1. 该文件没有缺陷

由于该文件不存在缺陷, 故无法生成报告, 所跳转至的createSingleReport页面没有报告显示, 用例成功.

1. 特殊需求

无

1. 前置条件

无

1. 后置条件

无

1. 扩展点

无



## 粘贴代码的单一文件纠错

1. 简述

该用例允许用户粘贴的源代码进行纠错.

1. 流程
   1. 基本流

用户点击’ Bug Detect’ 链接, 系统跳转至’ Debug ’ 页面’ , 随后点击’ find bugs in a single file’ 链接, 系统跳转至’ singlereport’ 页面, 将源代码粘贴至文本框中, 点击’submit’按钮, 系统调用服务器中的PMD程序对该文件进行纠错,生成报告后跳转至createSingleReport页面并将报告输出在该页面,用例成功.

* 1. 备选流
     1. 该文件通不过编译

如果该文件通不过编译，则将编译错误输出到createSingleReport页面中。

* + - 1. 用户选择不做下一步动作

用例失败.

* + - 1. 用户对代码进行修改

用户点击报告中的链接,可以对上传的文件进行修改.修改后可以选择删除文件或是保存修改. 如果保存修改结果, 则重新进行编译,如果选择删建,则将刚才提交时生成的临时文件删除.

* + 1. 该文件没有缺陷

由于该文件不存在缺陷, 故无法生成报告, 所跳转至的createSingleReport页面没有报告显示, 用例成功.

1. 特殊需求

无

1. 前置条件

无

1. 后置条件

无

1. 扩展点

无



## 查看缺陷信息

1. 简述

该用例允许用户查看缺陷信息

1. 流程
   1. 基本流

用户点击查看信息链接,显示所有缺陷信息,用例成功.

* 1. 备选流
     1. 当前不存在缺陷信息.

如果当前不存在缺陷信息,则跳转至的BugStat页面并不包含缺陷信息, 用例成功.

1. 特殊需求

无

1. 前置条件

无

1. 后置条件

无

1. 扩展点

无



## 统计缺陷

1. 简述

该用例允许用户在纠错同时系统可以统计错误频率.

1. 流程
   1. 基本流

用户在进行纠错的同时,系统将会统计缺陷的频率, 并将其记录至数据库中.

* 1. 备选流

无

1. 特殊需求

用户对文件进行纠错.

1. 前置条件

无

1. 后置条件

无

1. 扩展点

无



## 项目纠错

1. 简述

该用例允许注册用户登录后对已上传的项目进行纠错

1. 流程
   1. 基本流

用户点击’ Bug Detect’ 链接, 系统跳转至’ Debug ’ 页面’ , 随后点击’login’ 链接, 用户进行登录, 点击’Debug’链接, 系统跳转至’ProjectDebug’ 页面, 用户选择已经上传的项目进行纠错, 生成报告,跳转至’ProjectReport’页面, 并将报告输出.

* 1. 备选流
     1. 该项目通不过编译

如果该项目通不过编译，则将编译错误输出到createProjectReport页面中。

* + - 1. 用户选择不做下一步动作

用例失败.

* + - 1. 用户对代码进行修改

用户点击报告中的链接,可以对上传的项目中的存在问题的文件进行修改.修改后可以选择删除文件或是保存修改. 如果保存修改结果, 则重新进行编译,如果选择删建,则该文件文件删除.

* + 1. 该文件没有缺陷

由于该项目不存在缺陷, 故无法生成报告, 所跳转至的createProjectReport页面没有报告显示, 用例成功.

1. 特殊需求

无

1. 前置条件

注册用户已经登录.

1. 后置条件

无

1. 扩展点

无



## 可用性

# 逻辑视图

## 概述

* 逻辑视图从系统内在逻辑结构的角度描述系统的基本结构与动态行为，通常包括分析模型（Analysis Model）、设计模型（Design Model）以及数据模型（Data Model）等。
* 设计模型说明了系统的组成元素、组织架构和关系，并描述了各组成元素的协作以及状态转换关系等（通过用例实现Use Case Realization予以表达）。本节将分别在系统层次结构模型中描述系统的层次组织结构；在主要的包和子系统中说明系统的具体组成；并在架构机制中详述系统中的各种构架机制；最后在关键用例实现中通过描述最重要的用例实现，来说明构架的典型协作（动态行为）。
* 分析模型对等于设计模型，是在更高的抽象层次上定义系统的结构，作为可选项，本文档将不予说明。

## 系统层次结构模型



## 在构架上具有重要意义的设计包

* Application

这个层包括了所有UseCase实现的包。比如Engine,Utils包等

* Business Services

这个层包括了University Artifacts包。

* Middleware

这个层包括了JSP包。

## 重要体系结构设计包

* JSP包

JSP包包含了系统所有的界面视图，就如这个包的名字所暗示的一样，组成这个包的所有元素均是普通的JSP页面。我们使用JSP页面进行网页页面的渲染，接受用户的指令，并将系统操作后产生的结果返回给用户。值得注意的是，我们并没有在JSP页面中镶嵌任何代码或者脚本，以此保证JSP页面没有参与任何的业务逻辑

* Servlet包

Servlet包由所有的Servlet类组成，负责处理视图层发过来的所有请求，由于我们的系统由常用的JSP+SERVLET+TOMCAT搭建，采用基本的MVC结构。

* Utils包

Utils包主要包含了在系统设计过程中要用到的一些工具类

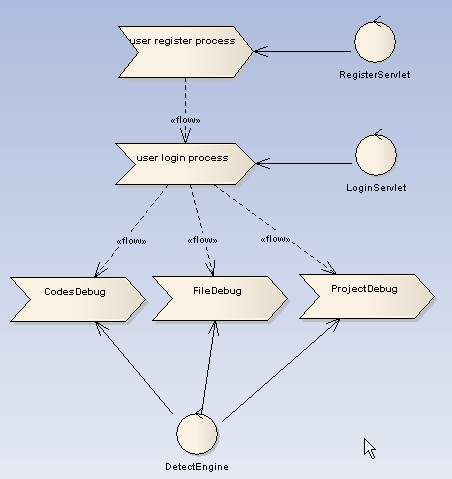
* Engine包

该包主要包括了将FINDBUGS和PMD两者集成在一起的类。

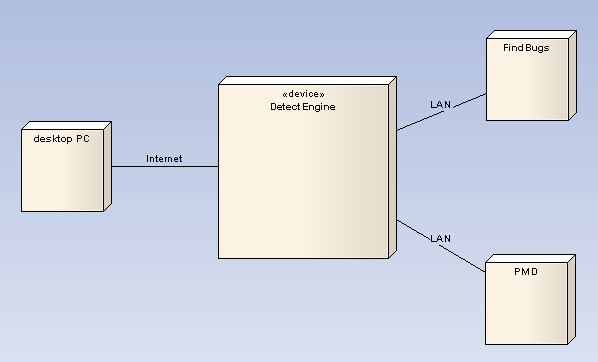
# 流程视图

## 概述

进程视图从系统运行时刻的角度，描述系统划分为进程、线程的结构，及其动态关系。模型主要说明进程、线程的分类，系统构架敏感的主要边界类、控制类对象等在进程、线程中的分布，以及它们之间的创建、交互与消息通讯关系等。



# 部署视图



# 数据视图（可选）

略

# 大小和性能

所选择的系统架构将支持所规定的关键的大小和性能：

* 该系统将支持多达200个用户在任何时刻对中央数据库的访问，并支持多大50个用户在任何时刻对服务器的访问。
* 该系统对数据库的访问将不超过5秒的延迟。
* 该系统将在10秒内处理粘贴代码和单一文件的纠错，15秒内完成对单个项目分析纠错。
* 系统客户端能在486或更高的处理器上运行，至少需要20MB磁盘空间和32MB内存空间。

# 质量

* 桌面用户的界面应高于Windows 98的要求。
* 系统的界面设计应满足易用性，同时用户在不需要培训的情况下也能轻松使用。
* 系统的每个功能应为用户提供在线帮助，并且在线帮助应具体明示操作的每一步，在线帮助应包括操作的定义，术语和缩略语。
* 系统应能在一天的24小时，一周的7天皆可用，不得超过有4%的停机时间。

系统总共的故障时间不得超过300小时