# 需求分析

需求分析是产品研发前期的铺垫工作，也是重要的基础工作之一。它是软件工程的一个重要阶段，没有良好的需求分析就无法编写出满足用户需要并且优秀的软件。

## 任务概述

### 目标

本文的目标是开发一个敏捷开发平台，具体要求如下：

* 支持文档模版的查询、新增以及提取使用
* 支持代码的生成，包括复用常规功能代码、通过数据库表生成相关业务代码通过UML类图生成相关代码。

## UML面向对象需求分析

UML(Unified Modeling Language)是一种建模语言，为软件开发的各个阶段提供统一标准的可视化模型，它涵盖了软件开发的各个阶段，包括需求分析以及系统详细设计等等，并且，它也融入了软件工程的新思想，即面向对象的分析与设计。本文使用UML用例图来进行需求分析，通过建立UML用例模型来确定本系统所应该具备的功能。

### 系统参与者

参与者是系统外部的一个实体，它以某种方式参与用例的执行过程。

对于敏捷开发平台系统来说，系统的参与者应为：开发者。系统中参与者解释见表3.1 参与者描述表，该表描述了该系统与者以及其相对应的详细描述。

表3.1 参与者描述表

|  |  |
| --- | --- |
| 参与者 | 描述 |
| 开发者 | 使用敏捷开发平台开发软件的开发人员 |

### UML用例规约描述与用例图

用例图是指由系统参与者、用例，边界以及它们之间的关系构成的用于描述系统功能的视图；用例规约描述则是用例图的详细文字说明，它包含用例相关的前置条件、后置条件等等相关规约。通过用例图以及用例规约描述便可以详尽的描述系统相关的功能需求。

开发者通过敏捷开发平台生成文档的用例图见图3- 1 生成文档用例图，该用例图包含四个用例分别为文档模版查询、文档模版上传、模板下载以及模版删除，其中，模板下载以及模版删除是文档模版查询所扩展的用例。



图3- 1 生成文档用例图

文档模版查询用例规约描述见表3- 1 文档模版查询用例规约描述表。

表3- 1 文档模版查询用例规约描述表

|  |  |
| --- | --- |
| 用例名称 | 文档模版查询 |
| 参与者 | 开发者 |
| 简要说明 | 开发者查询相关文档模版 |
| 前置条件 | 无 |
| 后置条件 | 成功显示文档模版列表 |
| 被包含的用例 | 无 |
| 被扩展的用例 | 模版下载、模版删除 |
| 主要事件流 | 1. 开发者点击敏捷开发平台左侧菜单栏中的文档模版查询; 2. 开发者在文档模版列表中输入文档模版名称后点击查询按钮 |
| 其他事件流 | 无 |
| 异常事件流 | 查询时发生系统异常，提示服务器繁忙。 |

文档模版下载用例规约描述见表3- 2 文档模版下载用例规约描述表

表3- 2 文档模版下载用例规约描述表

|  |  |
| --- | --- |
| 用例名称 | 模版下载 |
| 参与者 | 开发者 |
| 简要说明 | 开发者下载相关文档模版 |
| 前置条件 | 开发者查询文档模版列表 |
| 后置条件 | 成功下载文档模版 |
| 被包含的用例 | 无 |
| 被扩展的用例 | 无 |
| 主要事件流 | 1. 开发者在文档模版列表项中点击下载链接。 |
| 其他事件流 | 无 |
| 异常事件流 | 下载时发生系统异常，提示服务器繁忙。 |

文档模版删除用例规约描述表见表3- 3 文档模版删除用例规约描述表

表3- 3 文档模版删除用例规约描述表

|  |  |
| --- | --- |
| 用例名称 | 模版下载 |
| 参与者 | 开发者 |
| 简要说明 | 开发者删除相关文档模版 |
| 前置条件 | 开发者查询文档模版列表 |
| 后置条件 | 成功删除文档模版 |
| 被包含的用例 | 无 |
| 被扩展的用例 | 无 |
| 主要事件流 | 1. 开发者在文档模版列表项中点击删除链接。 |
| 其他事件流 | 无 |
| 异常事件流 | 删除时发生系统异常，提示服务器繁忙。 |

文档模版添加用例规约描述表见表3- 4 文档模版添加用例规约描述表

表3- 4 文档模版添加用例规约描述表

|  |  |
| --- | --- |
| 用例名称 | 文档模版添加 |
| 参与者 | 开发者 |
| 简要说明 | 开发者添加相关文档模版 |
| 前置条件 | 开发者点击文档模版添加Tab页 |
| 后置条件 | 成功添加文档模版 |
| 被包含的用例 | 无 |
| 被扩展的用例 | 无 |
| 主要事件流 | 1. 点击添加按钮 2. 选择添加的模版文件 3. 填写文档模版相关信息 4. 点击保存 |
| 其他事件流 | 无 |
| 异常事件流 | 保存时发生系统异常，提示服务器繁忙。 |

开发者通过敏捷开发平台生成代码的用例图见图3- 2 生成代码用例图，该用例图包含二十一个用例，其中代码生成用例包含三个子用例分别为业务表生成、代码样例生成、UML类图生成，其中业务表生成包含子用例业务表配置，业务表配置用例包含子用例添加业务表以及查询业务表，查询业务表用例扩展了三个用例分别为修改业务表、删除业务表、生成代码；代码样例生成包括子用例代码样例管理，代码样例管理用例包含子用例添加代码样例以及查询代码样例，查询代码样例扩展了下载代码样例和删除代码样例用例；UML类图生成包含子用例UML类图管理，UML类图管理包含了两个子用例UML类图添加和UML类图查询，UML类图查询用例扩展了删除类图和下载类图用例。



图3- 2 生成代码用例图

代码生成用例规约描述表见表3- 5代码生成用例规约描述表。

表3- 5 代码生成用例规约描述表

|  |  |
| --- | --- |
| 用例名称 | 代码生成 |
| 参与者 | 开发者 |
| 简要说明 | 开发者选择代码生成菜单 |
| 前置条件 | 无 |
| 后置条件 | 展开三个子菜单，业务表生成、UML类图生成、代码样例生成 |
| 被包含的用例 | 业务表生成、UML类图生成、代码样例生成 |
| 被扩展的用例 | 无 |
| 主要事件流 | 1. 开发者点击菜单栏上的代码生成菜单 |
| 其他事件流 | 无 |
| 异常事件流 | 点击后，菜单无法展开 |

业务表生成用例规约描述表见表3- 6 业务表生成用例规约描述表

表3- 6 业务表生成用例规约描述表

|  |  |
| --- | --- |
| 用例名称 | 业务表生成 |
| 参与者 | 开发者 |
| 简要说明 | 开发者选择通过业务表生成菜单 |
| 前置条件 | 开发者点击代码生成菜单 |
| 后置条件 | 展开一个子菜单项，业务表配置 |
| 被包含的用例 | 业务表配置 |
| 被扩展的用例 | 无 |
| 主要事件流 | 1. 开发者点击业务表生成菜单 |
| 其他事件流 | 无 |
| 异常事件流 | 点击后，菜单无法展开 |

业务表配置用例规约描述表见表3- 7 业务表配置用例规约描述表

表3- 7 业务表配置用例规约描述表

|  |  |
| --- | --- |
| 用例名称 | 业务表配置 |
| 参与者 | 开发者 |
| 简要说明 | 开发者选择业务表配置菜单 |
| 前置条件 | 开发者点击通过业务表生成菜单 |
| 后置条件 | 右侧主窗口显示业务表列表 |
| 被包含的用例 | 添加业务表，查询业务表 |
| 被扩展的用例 | 无 |
| 主要事件流 | 1. 开发者点击业务表配置菜单 |
| 其他事件流 | 无 |
| 异常事件流 | 点击后，主窗口无显示，提示服务器繁忙。 |

添加业务表用例规约描述表见表3- 8 添加业务表用例规约描述表

表3- 8 添加业务表用例规约描述表

|  |  |
| --- | --- |
| 用例名称 | 添加业务表 |
| 参与者 | 开发者 |
| 简要说明 | 开发者在平台中添加所开发软件的业务表 |
| 前置条件 | 开发者点击业务表配置主窗口中的添加业务表Tab页 |
| 后置条件 | 保存成功 |
| 被包含的用例 | 无 |
| 被扩展的用例 | 无 |
| 主要事件流 | 1. 选择要添加的业务表，点击下一步。 2. 选择相关条件，点击保存 |
| 其他事件流 | 1. 点击上一步，可返回选择业务表页面 |
| 异常事件流 | 保存时系统异常，提示服务器繁忙 |

业务表查询用例规约描述表见表3- 9 业务表查询用例规约描述表

表3- 9 业务表查询用例规约描述表

|  |  |
| --- | --- |
| 用例名称 | 业务表查询 |
| 参与者 | 开发者 |
| 简要说明 | 开发者查询系统中保存的相关业务表 |
| 前置条件 | 开发者点击业务表配置菜单项 |
| 后置条件 | 无 |
| 被包含的用例 | 无 |
| 被扩展的用例 | 修改业务表、删除业务表、生成代码 |
| 主要事件流 | 1、填写相关查询条件，点击查询按钮 |
| 其他事件流 | 无 |
| 异常事件流 | 查询时系统异常，提示服务器繁忙 |

修改业务表用例规约描述表见表3- 10 修改业务表用例规约描述表

表3- 10 修改业务表用例规约描述表

|  |  |
| --- | --- |
| 用例名称 | 修改业务表 |
| 参与者 | 开发者 |
| 简要说明 | 开发者修改保存业务表的相关信息 |
| 前置条件 | 开发者查询业务表 |
| 后置条件 | 无 |
| 被包含的用例 | 无 |
| 被扩展的用例 | 无 |
| 主要事件流 | 1. 在业务表列表上点击修改链接； 2. 填写相关修改信息，点击保存 |
| 其他事件流 | 无 |
| 异常事件流 | 保存时系统异常，提示服务器繁忙 |

删除业务表用例规约描述表见表3- 11 删除业务表用例规约描述表

表3- 11 删除业务表用例规约描述表

|  |  |
| --- | --- |
| 用例名称 | 删除业务表 |
| 参与者 | 开发者 |
| 简要说明 | 开发者删除系统中保存的相关业务表 |
| 前置条件 | 开发者查询业务表 |
| 后置条件 | 无 |
| 被包含的用例 | 无 |
| 被扩展的用例 | 无 |
| 主要事件流 | 1. 开发者点击业务表列表中的删除链接 |
| 其他事件流 | 无 |
| 异常事件流 | 删除时系统异常，提示服务器繁忙 |

生成代码用例规约描述表见表3- 12 生成代码用例规约描述表

表3- 12 生成代码用例规约描述表

|  |  |
| --- | --- |
| 用例名称 | 生成代码 |
| 参与者 | 开发者 |
| 简要说明 | 开发者根据配置的业务表生成代码 |
| 前置条件 | 开发者查询业务表 |
| 后置条件 | 无 |
| 被包含的用例 | 无 |
| 被扩展的用例 | 无 |
| 主要事件流 | 1. 开发者点击业务表列表中的生成代码链接； 2. 填写相关生成代码选项，点击保存并生成代码按钮 |
| 其他事件流 | 无 |
| 异常事件流 | 生成代码时系统异常，提示服务器繁忙 |

代码样例生成用例规约描述表见表3- 13 代码样例生成用例规约描述表

表3- 13 代码样例生成用例规约描述表

|  |  |
| --- | --- |
| 用例名称 | 代码样例生成 |
| 参与者 | 开发者 |
| 简要说明 | 开发者选择代码样例生成菜单 |
| 前置条件 | 开发者选择代码生成菜单 |
| 后置条件 | 展开一项子菜单代码样例列表 |
| 被包含的用例 | 代码样例管理 |
| 被扩展的用例 | 无 |
| 主要事件流 | 1. 开发者点击代码样例生成菜单 |
| 其他事件流 | 无 |
| 异常事件流 | 点击时无法展开子菜单 |

代码样例管理用例规约描述表见表3- 14 代码样例管理用例规约描述表

表3- 14 代码样例管理用例规约描述表

|  |  |
| --- | --- |
| 用例名称 | 代码样例管理 |
| 参与者 | 开发者 |
| 简要说明 | 开发者进行代码样例管理 |
| 前置条件 | 开发者点击代码样例生成菜单项 |
| 后置条件 | 主窗口显示代码样例列表 |
| 被包含的用例 | 添加代码样例、查询代码样例 |
| 被扩展的用例 | 无 |
| 主要事件流 | 1. 开发者点击代码样例管理菜单 |
| 其他事件流 | 无 |
| 异常事件流 | 点击时系统异常，提示服务器繁忙 |

代码样例添加用例规约描述表见表3- 15 代码样例添加用例规约描述表

表3- 15 代码样例添加用例规约描述表

|  |  |
| --- | --- |
| 用例名称 | 代码样例添加 |
| 参与者 | 开发者 |
| 简要说明 | 开发者添加相关代码样例 |
| 前置条件 | 开发者点击代码样例管理并选择主窗口的添加代码样例Tab页 |
| 后置条件 | 添加成功 |
| 被包含的用例 | 无 |
| 被扩展的用例 | 无 |
| 主要事件流 | 1. 开发者点击添加按钮，选择相应代码样例上传； 2. 添加相关信息后点击保存按钮。 |
| 其他事件流 | 无 |
| 异常事件流 | 保存或上传时系统异常，提示服务器繁忙 |

查询代码样例用例规约描述表见

表3- 16 查询代码样例用例规约描述表

|  |  |
| --- | --- |
| 用例名称 | 查询代码样例 |
| 参与者 | 开发者 |
| 简要说明 | 开发者查询系统中保存的代码样例 |
| 前置条件 | 开发者点击代码样例管理 |
| 后置条件 | 无 |
| 被包含的用例 | 无 |
| 被扩展的用例 | 下载代码样例、删除代码样例 |
| 主要事件流 | 1. 填写相关查询条件，点击查询按钮 |
| 其他事件流 | 无 |
| 异常事件流 | 查询时系统异常，提示服务器繁忙 |

下载代码样例用例规约描述表见表3- 17 下载代码样例用例规约描述表

表3- 17 下载代码样例用例规约描述表

|  |  |
| --- | --- |
| 用例名称 | 下载代码样例 |
| 参与者 | 开发者 |
| 简要说明 | 开发者下载系统中保存的代码样例 |
| 前置条件 | 开发者查询代码样例 |
| 后置条件 | 成功下载 |
| 被包含的用例 | 无 |
| 被扩展的用例 | 无 |
| 主要事件流 | 1. 开发者点击代码样例列表中的下载链接 |
| 其他事件流 | 无 |
| 异常事件流 | 下载时系统异常，下载失败。 |

删除代码样例用例规约描述表见表3- 18 删除代码样例用例规约描述表

表3- 18 删除代码样例用例规约描述表

|  |  |
| --- | --- |
| 用例名称 | 删除代码样例 |
| 参与者 | 开发者 |
| 简要说明 | 开发者删除系统中保存的代码样例 |
| 前置条件 | 开发者查询代码样例 |
| 后置条件 | 无 |
| 被包含的用例 | 无 |
| 被扩展的用例 | 无 |
| 主要事件流 | 1. 开发者点击代码样例列表中的删除链接 |
| 其他事件流 | 无 |
| 异常事件流 | 删除时系统异常，提示服务器繁忙 |

UML类图生成用例规约描述见表表3- 19 UML类图生成用例规约表

表3- 19 UML类图生成用例规约表

|  |  |
| --- | --- |
| 用例名称 | UML类图生成 |
| 参与者 | 开发者 |
| 简要说明 | 开发者点击UML类图生成菜单 |
| 前置条件 | 开发者点击代码生成菜单 |
| 后置条件 | 菜单展开项为UML类图列表 |
| 被包含的用例 | UML类图管理 |
| 被扩展的用例 | 无 |
| 主要事件流 | 1. 开发者点击UML类图生成菜单 |
| 其他事件流 | 无 |
| 异常事件流 | 点击时菜单无法展开 |

UML类图管理用例规约描述表见表3- 20 UML类图管理用例规约表

表3- 20 UML类图管理用例规约表

|  |  |
| --- | --- |
| 用例名称 | UML类图管理 |
| 参与者 | 开发者 |
| 简要说明 | 开发者点击UML列表菜单 |
| 前置条件 | 开发者点击UML生成菜单 |
| 后置条件 | 右侧主窗口显示系统中保存的UML类图列表 |
| 被包含的用例 | UML类图添加、UML类图查询 |
| 被扩展的用例 | 无 |
| 主要事件流 | 1. 开发者点击UML类图列表菜单 |
| 其他事件流 | 无 |
| 异常事件流 | 点击时右侧主窗口无显示，提示服务器繁忙 |

UML类图添加用例规约描述表见表3- 21 UML类图添加用例规约描述表

表3- 21 UML类图添加用例规约描述表

|  |  |
| --- | --- |
| 用例名称 | UML类图添加 |
| 参与者 | 开发者 |
| 简要说明 | 开发者向系统中保存UML类图 |
| 前置条件 | 开发者选择主窗口上的添加UML类图Tab页面 |
| 后置条件 | 添加成功 |
| 被包含的用例 | 无 |
| 被扩展的用例 | 无 |
| 主要事件流 | 1. 开发者点击添加按钮，选择上传的UML文件 2. 添加相关信息后点击保存。 |
| 其他事件流 | 无 |
| 异常事件流 | 保存时系统异常，提示服务器繁忙 |

UML类图查询用例规约描述表见表3- 22 UML类图查询用例规约描述表

表3- 22 UML类图查询用例规约描述表

|  |  |
| --- | --- |
| 用例名称 | UML类图查询 |
| 参与者 | 开发者 |
| 简要说明 | 开发者查询系统中所保存的UML类图 |
| 前置条件 | 开发者点击UML类图列表菜单 |
| 后置条件 | 无 |
| 被包含的用例 | 无 |
| 被扩展的用例 | 删除类图、下载类图、生成代码 |
| 主要事件流 | 1. 开发者输入查询条件、点击查询按钮 |
| 其他事件流 | 无 |
| 异常事件流 | 查询时系统异常，提示服务器繁忙 |

下载类图用例规约描述表见表3- 23 下载类图用例规约描述表

表3- 23 下载类图用例规约描述表

|  |  |
| --- | --- |
| 用例名称 | 下载类图 |
| 参与者 | 开发者 |
| 简要说明 | 开发者下载系统中保存的UML类图 |
| 前置条件 | 开发者查询UML类图 |
| 后置条件 | 下载成功 |
| 被包含的用例 | 无 |
| 被扩展的用例 | 无 |
| 主要事件流 | 1. 开发者点击UML类图列表中的下载链接 |
| 其他事件流 | 无 |
| 异常事件流 | 下载时系统异常，无法下载 |

删除类图用例规约描述表见表3- 24 删除类图用例规约描述表

表3- 24 删除类图用例规约描述表

|  |  |
| --- | --- |
| 用例名称 | 删除类图 |
| 参与者 | 开发者 |
| 简要说明 | 开发者删除系统中保存的UML类图 |
| 前置条件 | 开发者查询UML类图 |
| 后置条件 | 删除成功 |
| 被包含的用例 | 无 |
| 被扩展的用例 | 无 |
| 主要事件流 | 1. 开发者点击UML类图列表中的删除链接 |
| 其他事件流 | 无 |
| 异常事件流 | 删除时系统异常，提示服务器繁忙 |

生成代码用例规约描述表见表3- 25 生成代码用例规约描述表

表3- 25 生成代码用例规约描述表

|  |  |
| --- | --- |
| 用例名称 | 生成代码 |
| 参与者 | 开发者 |
| 简要说明 | 开发者根据UML类图生成代码 |
| 前置条件 | 开发者查询UML类图 |
| 后置条件 | 生成成功 |
| 被包含的用例 | 无 |
| 被扩展的用例 | 无 |
| 主要事件流 | 1. 开发者点击UML列表中的生成代码链接 2. 填写相关配置，点击保存并生成代码按钮 |
| 其他事件流 | 无 |
| 异常事件流 | 生成代码时系统异常，提示服务器繁忙。 |

## 模块功能划分

敏捷开发平台共分为以下三大功能模块：

* UI模块：主要包括前端页面，负责用户与系统交互与反馈，详细分为首页UI模块、文档生成UI模块、代码生成UI模块。
* 生成器模块：主要负责系统对于文档以及代码的生成处理，详细分为代码生成器、文档生成器，其中代码生成器又划分出三个子模块，分别为：业务表代码生成器、UML类图代码生成器、代码样例生成器。
* 通用组件模块：主要负责系统中通用的工具以及封装系统前后台交互的消息体。详细分为通用工具模块、消息体模块。

模块划分图见图3- 3 敏捷开发平台模块功能划分图



图3- 3 敏捷开发平台模块功能划分图

# 敏捷开发平台详细设计与实现

## 系统总体设计

### 系统架构设计

本文的敏捷开发平台采用分层设计架构，分为业务实体层、数据访问层、业务

逻辑层、控制层以及视图层。分层结构见图图4- 1 敏捷开发平台分层架构，其中开发者通过视图层的HTML页面与系统进行交互，之后通过 Fetch技术调用控制层（由SpringMVC承担），接着由控制层调用业务逻辑层执行相关业务处理，再由业务逻辑层调用数据访问层（由Mybaits承担）获取数据并封装成相应实体类返回到控制层后，控制层在对数据进行消息体封装使用JSON数据格式（由SpringBoot集成FastJSON承担）返回给视图层。



图4- 1 敏捷开发平台分层架构

代码层次则使用Maven技术依据模块划分工程为三个子工程分别为web模块（负责系统的控制层及视图层）、gen模块（负责代码生成模块的业务实体层、数据访问层、业务逻辑层）、commons模块（负责提供通用组建以及消息体封装）

### 系统开发与运行环境

* 开发环境

系统所使用开发环境见表表4- 1 系统开发环境表

表4- 1 系统开发环境表

|  |  |
| --- | --- |
| 操作系统 | Windows 10 |
| 处理器 | Core i7 |
| 内存 | 16G |
| JDK | 1.8 |
| 集成开发环境 | IDEA |
| 数据库 | MySQL5.7 |

* 运行环境

系统运行环境见表表4- 2 系统运行环境表

表4- 2 系统运行环境表

|  |  |
| --- | --- |
| 操作系统 | Windows XP及以上 |
| 硬件 | 普通PC机 |
| 软件 | MySQL5.0以上、JDK1.8以上 |

## 数据库设计

数据库设计是指根据用户的需求, 在某一具体的数据库管理系统上, 设计数据库的结构和建立数据库的过程。

### 表设计

系统中所使用的数据表如下：

* gen\_table(id,table\_name,table\_comments,class\_name)

该表用于存储开发者在系统中所添加保存的业务表，表定义见表4- 3 gen\_table定义表

表4- 3 gen\_table定义表

|  |  |  |  |
| --- | --- | --- | --- |
| **字段名** | **字段类型** | **字段大小** | **字段含义** |
| id | Int | 11 | Id |
| table\_name | varchar | 100 | 业务表名称 |
| table\_comments | varchar | 500 | 业务表说明 |
| class\_name | varchar | 100 | 业务表映射类名 |

* gen\_table\_column(id,gen\_table\_id,name,comments,jdbc\_type,java\_type,java\_field,is\_pk,is\_null,is\_insert,is\_edit,is\_list,is\_query,query\_type)

该表用于存储业务表字段生成代码的相关配置选项。表定义见表4- 4 gen\_table\_column定义表

表4- 4 gen\_table\_column定义表

|  |  |  |  |
| --- | --- | --- | --- |
| **字段名** | **字段类型** | **字段大小** | **字段含义** |
| id | Int | 11 | Id |
| gen\_table\_id | Int | 11 | 关联业务表Id |
| name | varchar | 100 | 字段名称 |
| comments | varchar | 200 | 说明 |
| jdbc\_type | varchar | 50 | JDBC查询类型 |
| java\_type | varchar | 50 | JAVA类型 |
| java\_field | varchar | 100 | JAVA属性名称 |
| is\_pk | char | 1 | 是否为主键 |
| is\_null | char | 1 | 是否为空 |
| is\_insert | char | 1 | 是否可插入 |
| is\_edit | char | 1 | 是否可编辑 |
| is\_list | char | 1 | 是否为列表 |
| is\_query | char | 1 | 是否为查询字段 |
| query\_type | varchar | 100 | 查询类型 |

* gen\_uml\_class\_diagram(id,class\_diagram\_name,comments,format\_json\_str,real\_name)

该表用于存储系统保存的UML类图信息。表定义见

|  |  |  |  |
| --- | --- | --- | --- |
| **字段名** | **字段类型** | **字段大小** | **字段含义** |
| id | Int | 11 | Id |
| class\_diagram\_name | varchar | 25 | 类图名称 |
| comments | varchar | 500 | 类图说明 |
| format\_json\_str | longtext |  | 类图格式化JSON |
| real\_name | varchar | 128 | 真实名称 |

* gen\_doc(id,doc\_name,doc\_size,upload\_date,real\_name)

该表用于存储系统中保存的文档模版相关信息。表定义见表4- 5 gen\_doc表定义

表4- 5 gen\_doc表定义

|  |  |  |  |
| --- | --- | --- | --- |
| **字段名** | **字段类型** | **字段大小** | **字段含义** |
| id | Int | 11 | Id |
| doc\_name | varchar | 100 | 文档模版名称 |
| doc\_size | Int | 11 | 文档模版大小 |
| upload\_date | datetime |  | 文档创建日期 |
| real\_name | varchar | 128 | 真实名称 |

* gen\_code\_template(id,name,function,comments,real\_name)

该表用于存储系统中保存的代码样例相关信息。表定义见表4- 6 gen\_code\_template表定义

表4- 6 gen\_code\_template表定义

|  |  |  |  |
| --- | --- | --- | --- |
| **字段名** | **字段类型** | **字段大小** | **字段含义** |
| id | Int | 11 | Id |
| name | varchar | 50 | 代码样例名称 |
| function | varchar | 50 | 代码样例功能 |
| comments | varchar | 256 | 说明 |
| real\_name | varchar | 128 | 真实名称 |

* gen\_scheme(id,name,category,package\_name,module\_name,sub\_module\_name,function\_name,function\_name\_simple,function\_author,ref\_id,ref\_type)

该表用于存储代码生成方案相关信息，表定义见表4- 7 gen\_scheme表定义

表4- 7 gen\_scheme表定义

|  |  |  |  |
| --- | --- | --- | --- |
| **字段名** | **字段类型** | **字段大小** | **字段含义** |
| id | Int | 11 | Id |
| name | varchar | 50 | 代码生成方案名称 |
| category | varchar | 50 | 生成方案模版类别 |
| package\_name | varchar | 256 | 包名 |
| module\_name | varchar | 128 | 模块名 |
| sub\_module\_name | varchar | 128 | 子模块名 |
| function\_name | varchar | 100 | 功能名 |
| function\_name\_simple | varchar | 100 | 功能名简写 |
| function\_author | varchar | 100 | 功能作者 |
| ref\_id | int | 11 | 方案引用业务表或类图ID |
| ref\_type | int | 1 | 方案引用类型 |

### E-R图

敏捷开发平台E-R图见图4- 2 敏捷开发平台E-R图，gen\_table与gen\_table\_column通过gen\_table\_id字段进行关联，gen\_table拥有多个gen\_column，与gen\_scheme表通过ref\_id与ref\_type字段相关联，并且为一一对应关系，相应的，gen\_uml\_class\_diagram表与gen\_scheme表也通过ref\_id与ref\_type字段相关联，并且为一一对应关系。



图4- 2 敏捷开发平台E-R图

## 模块功能设计与实现

### UI模块

UI模块主要使用HTML5技术以及Semantic-UI、jQuery等前端框架作为基础实现。

* 首页设计

首页部分分为三块，第一部分为头部分，用于展示平台信息以及Logo；第二部分为左侧菜单栏，用于显示系统可操作菜单；第三部分为系统主窗口部分，用于系统主要功能界面。

* 文档生成页面设计

文档生成页面包括两个标签页，第一个标签页为文档模版列表，其中包含文档模版列表及其查询条件；第二个标签页为文档模版添加标签页，其中包含一个表单，该表单用于提供相应信息将模版保存入系统。

* 业务表配置页面设计

业务表配置页面包括两个标签页，第一个标签页为业务表列表，其中包含业务表列表及其相关查询条件；第二个标签页为业务表添加，其中为两步操作页面，第一步为下拉框选择相应业务表，第二步为业务表信息表单。

* 代码样例页面设计

代码样例页面包括两个标签页，第一个标签页为代码样例列表，其中包含代码样例列表及其相关查询条件；第二个标签页为添加代码样例，其中包含一个表单，该表单用于提供相应信息将代码样例存入系统

* UML类图管理页面设计

UML类图管理页面包括两个标签页，第一个标签页为UML类图列表，其中包含系统中保存的UML类图列表及其相关查询条件；第二个标签页为UML类图添加，其中包含一个表单，该表单用于提供相应信息保存UML类图到系统中。

### 生成器模块

* 文档生成功能设计与实现

文档生成功能类图见图4- 3 文档生成功能类图，GenDoc为文档实体类，其继承了Page类以提供分页支持；IGenDocDao是DAO层接口，提供数据访问方法；GenDocService为业务逻辑层，其组合了IGenDocDao接口，提供业务逻辑处理以及调用DAO层操作；GenDocController为控制层，其组合了GenDocService，并且依赖与GenDoc以及Message，用于处理请求调用业务逻辑层并且返回数据。



图4- 3 文档生成功能类图

1. 文档查询功能实现

前端页面通过Fetch发起文档查询请求，GET请求地址为/gen/getGenDocList，之后通过SpringMVC的参数绑定技术将页面请求参数注入到GenDoc对象中，之后调用GenDocSevice获取文档列表（分页），之后使用Message消息体封装返回值以JSON格式返回前端。

1. 文档添加功能实现

前端页面通过Jquery-FileUpload插件选择上传文件并提交表单，PUT请求地址为/gen/uploadDoc，之后调用GenDocService生成GUID唯一文件名，存储到磁盘中并将相关信息调用IGenDocDao的saveOrUpdate方法入库，调用完成后将信息封装为Message对象以JSON形式返回前端。

1. 文档下载功能实现

前端页面通过Fetch发起GET请求，请求地址为/gen/downLoadDoc，之后GenDocController通过解析请求携带参数Id调用GenDocService查询相关实体对象，通过实体对象的realName属性获取磁盘文件，最后设置HTTP请求头并将文件以字节写入响应后完成下载。

1. 文档删除功能实现

前端页面使用Fetch发起DELETE请求，请求地址为/gen/deleteDoc，之后GenDocController解析请求参数Id并调用GenDocService删除相关文件，接着调用IGenDocDao的delete方法删除数据库信息，最后，将成功或者失败信息封装为Message对象以JSON格式返回前端。

* 业务表生成模块设计与实现

业务表生成模块类图见图4- 4 业务表生成模块类图，其中GenTable为业务实体类，其中组合了多个GenTableColumn实体类并且继承了Page类以支持分页查询，GenTableService为业务逻辑层，组合了IGenTableDao和IGenTableColumnDao接口来实现对于GenTable的增删改查功能，GenTableSchemaService同样为业务逻辑层，其继承了虚基类BaseGenSchemaService，虚基类实现了IGenCodeService接口，并且依赖GenUtils工具类，主要负责通过业务表配置规则生成代码的业务逻辑，GenTableControllerw为控制层，它组合了GenTableService以及GenTableSchemaService来实现业务逻辑的处理，并且依赖于GenTable以及Message对象来向前端返回数据。



图4- 4 业务表生成模块类图

1. 业务表查询功能

前端页面通过Fetch发起业务表查询请求，GET请求地址为/gen/ getBusinessTables，之后通过SpringMVC的参数绑定技术将请求参数注入到GenTable对象中，之后根据GenTable对象中的信息调用GenTableService获取业务表列表，之后使用Message消息体封装返回值以JSON格式返回前端显示。

1. 业务表修改功能

前端页面首先发起GET请求/gen/ modifyGenTable获取到待修改的GenTable详细信息，之后将获取到的GenTable信息展现到修改表单中，最后提交表单，GenTableController将表单信息注入到GenTable对象中，调用GenTableService执行更新操作，最后将操作信息封装入Message对象中，以JSON格式返回前端显示。

1. 业务表删除功能

前端页面使用Fetch发起DELETE请求，请求地址为/gen/deleteGenTable，GenTableController根据请求中的Id调用GenTableService执行删除操作，需同时删除关联的GenTableColumn信息以及GenTableSchema信息，最后将操作信息封装入Message对象中，以JSON格式返回前端显示。

1. 业务表添加功能

前端页面首先使用Fetch发起GET请求，请求地址为/gen/getPhysicalTables，GenTableController调用GenTableService，GenTableService调用IGenTableDao获取数据库中的物理表列表，获取后封装入Message对象以JSON返回前端，前端将列表信息显示到下拉框中，开发者选择相应业务表点击下一步时，再使用Fetch发起GET请求，请求地址为/gen/getTableInfo，GenTableController根据请求中的参数调用GenTableService，GenTableService调用IGenTableDao查询物理表的详细信息并返回，最后将详细信息封装入Message对象，以JSON格式返回前端。

前端将返回的信息填入表单，最后提交表单，GenTableController将表单信息注入GenTable对象，调用GenTableService执行保存操作并生成默认代码生成方案，并将操作结果封装入Message对象，以JSON格式返回前端。

1. 业务表生成代码功能

前端页面首先使用Fetch发起GET请求，请求地址为/gen/getSchemaByRefId，GenTableController根据请求携带参数信息调用GenTableSchemaService，GenTableSchemaService调用IGenTableSchemaDao查询到默认生成方案并返回，将返回的GenSchema封装入Message对象以JSON格式返回前端，前端依据返回信息将信息填入生成方案表单，填写信息提交表单后，GenTableController根据表单信息，调用GenTableSchemaService，GenTableSchemaService调用GenUtils根据填写的信息以及模版类型调用FreeMakers工具渲染模版，再通过FileUtils以及配置信息写入文件，最后更新GenSchema并将生成结果返回，封装Message对象以JSON格式返回前端。

* 代码样例生成模块