logo

**Visual-Vers——可视化版本控制工具**

模块设计报告

|  |  |  |
| --- | --- | --- |
| 文件状态：  [ ] 草稿  [√] 正式发布  [ ] 正在修改 | 文件标识： | ZJGSU |
| 当前版本： | 1.0 |
| 作 者： | 汤英杰 顾启隽 叶扬 |
| 完成日期： | 2022-5-31 |

浙江工商大学计算机与信息工程学院

目 录

[1．文档介绍 2](#_Toc20181)

[1.1文档目的 2](#_Toc31716)

[1.2文档范围 2](#_Toc19843)

[1.3读者对象 3](#_Toc2869)

[1.4参考文献 3](#_Toc17309)

[1.5术语与缩写解释 3](#_Toc27227)

[2．模块命名规则 3](#_Toc22771)

[3．模块汇总 3](#_Toc11430)

[3.1模块汇总表 3](#_Toc15175)

[3.1.1Visual-Vers子系统模块汇总 3](#_Toc31141)

[3.1.2Visual-Vers子系统模块汇总表 4](#_Toc10623)

[3.2模块关系图 4](#_Toc7172)

[4．用户子系统的模块设计 4](#_Toc13674)

[4.1表示层模块设计 4](#_Toc28876)

[4.1.1模块－ 4](#_Toc22680)

[4.1.2模块－ 5](#_Toc330)

[4.1.3模块－ 5](#_Toc7454)

[4.1.4模块－ 5](#_Toc9562)

[4.1.5模块－ 5](#_Toc24882)

[4.1.6模块－ 5](#_Toc14577)

[4.2数据处理层 6](#_Toc13582)

[4.2.1模块－ 6](#_Toc1820)

[4.2.2模块－ 6](#_Toc3189)

[4.2.3模块－ 6](#_Toc24012)

[4.2.4模块－ 7](#_Toc2510)

# 1．文档介绍

## 1.1文档目的

本文档的目的是明确系统的各模块的命名规则，明确各模块的功能，同时对每个模块进行详细设计，包括接口与属性设计、数据结构与算法设计，明确各模块的具体实现策略，使系统开发人员和产品管理人员明确产品功能，可以有针对性的进行系统开发、测试、验收等各方面的工作。

## 1.2文档范围

本文档包括各子系统的各模块命名规则以及各模块的详细设计，包括接口与属性设计、数据结构与算法设计。

## 1.3读者对象

本文档主要面向客户、数据库设计人员、架构设计人员、界面设计人员、编码人员及测试人员。

## 1.4参考文献

陈明：《软件工程》，中央广播电视大学出版社，2002年6月版

《软件工程》第五版——清华大学出版社 张国藩编

冯玉才，数据库系统基础，199-237，武汉，华中科技大学出版社，1993

李廉等译,Web服务器技术指南,北京，机械工业出版社，1995

## 1.5术语与缩写解释

|  |  |
| --- | --- |
| **缩写、术语** | **解 释** |
|  |  |
|  |  |
|  |  |

# 2．模块命名规则

类和接口均采用名词，按照其功能含义用小写英文字母来命名。

# 3．模块汇总

## 3.1模块汇总表

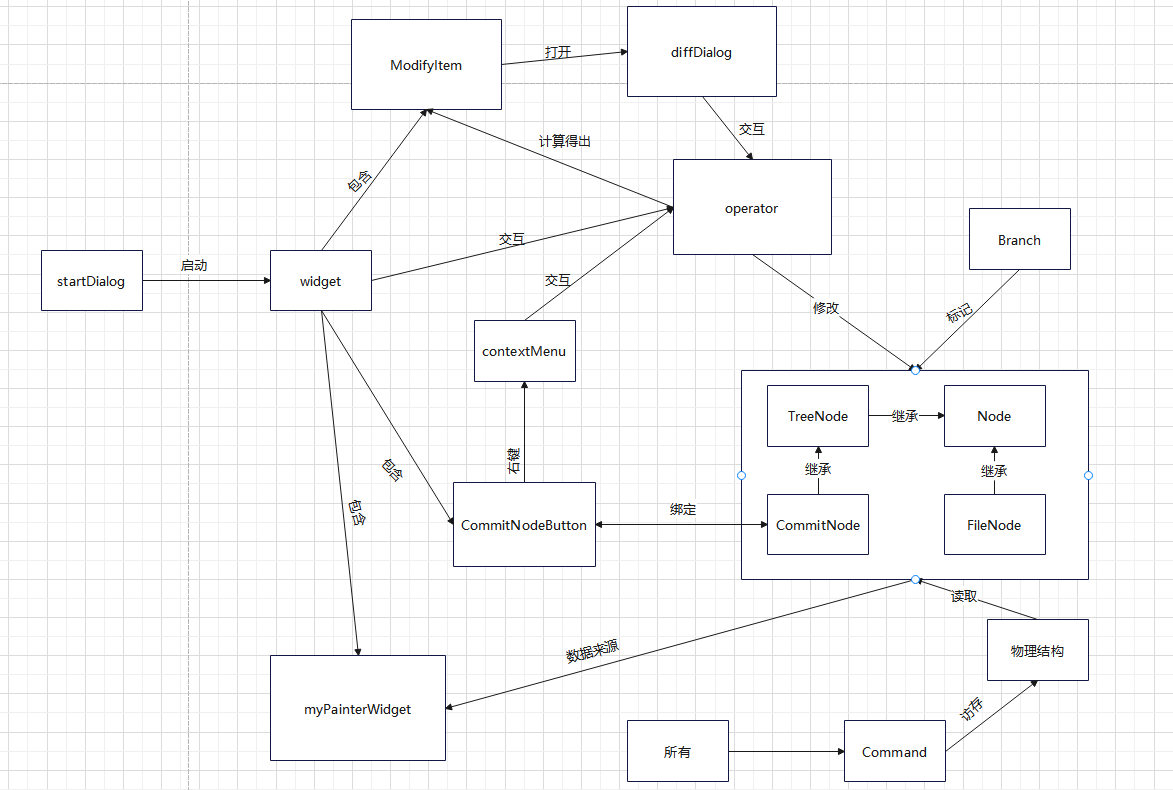
### 3.1.1Visual-Vers子系统模块汇总

1. **Command命令模块**
2. **Operator系统操作模块**
3. **Operator用户操作模块**
4. **Node节点结构模块**
5. **Branch分支模块**
6. **ModifyItem差异记录模块**
7. **CommitNodeButton数据界面衔接模块**
8. **myPainterWidget绘图界面**
9. **startDialog初始界面**
10. **Widget用户操作界面**
11. **DiffDialog差异对比界面**
12. **contextMenu弹出式交互界面**

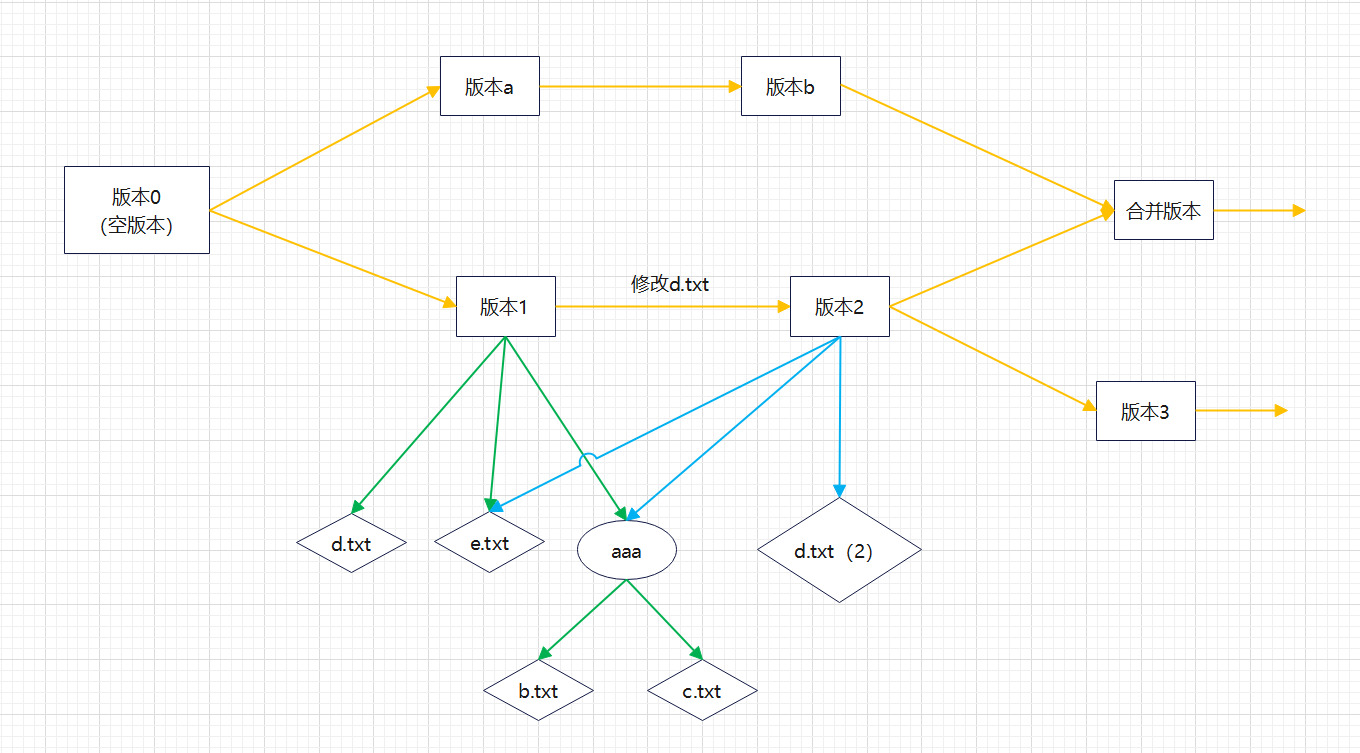
### 3.1.2Visual-Vers子系统模块汇总表

|  |  |  |
| --- | --- | --- |
| **表示层** | | |
| 页面名称 | 模块名称 | 功能简述 |
| startDialog | 开始界面 | 初始界面，打开/新建仓库，最近仓库 |
| widget | 用户操作界面 | 处理大多交互内容 |
| diffdialog | 差异对比界面 | 显示差异内容 |
| contextMenu | 弹出式交互页面 | 对具体节点的交互操作 |
| myPainterWidget | 绘图界面 | 绘制版本图 |
| **数据处理层** | | |
| 模块名称 | 功能简述 | |
| Command | 常用命令，文件操作，格式转换 | |
| Operator | 封装大多数操作，实现用户可直接调用的增删查改操作 | |
| Node | 项目核心数据结构 | |
| Branch | 实现多分支，管理当前分支 | |
| ModifyItem | 项目对比，文件对比时记录差异，打开diffdialog | |
| CommitNodeButton | 与版本节点一一对应，将用户点击操作转换为对节点的具体操作 | |

## 3.2模块关系图



其中Node部分：



# 4．用户子系统的模块设计

## 4.1表示层模块设计

## 4.1.1命令模块

|  |  |
| --- | --- |
| 模块名称 | Command |
| 功能描述 | 常用命令，文件操作，格式转换 |
| 接口与属性 | //#define ROOT\_PATH ("D:\\repo")  extern string ROOT\_PATH; //各种常量定义  #define REPO\_PATH (ROOT\_PATH + "\\.vvs")  #define DATA\_PATH (REPO\_PATH + "\\data")  #define DATA\_PATH\_ (REPO\_PATH + "\\data\\")  #define COMMIT\_PATH (REPO\_PATH + "\\commits")  #define COMMIT\_PATH\_ (REPO\_PATH + "\\commits\\")  #define BRANCH\_FILE\_PATH (REPO\_PATH + "\\.branch")  #define DEFAULT\_AVATAR ":/images/img/default.png"  const string EMPTY\_HASH = "00000000000000000000000000000000";  const string ROOT\_ID = "yemmm000000000000000000000000000t";  bool doCmd(const string &cmd, string &res); //命令行Api  bool doCmd(const string &cmd);  bool CopyAFile(const string &a, const string &b); //各种文件操作  void DeleteAny(const string &a);  void CreateFolder(const string &a);  const int FOLDER\_PATH = 1; //路径判断和路径宏  const int FILE\_PATH = 2;  const int EMPTY\_PATH = 0;  bool isempty(const string &path); //判断文件是否为空  int judgePath(const string &path); //判断路径类型  //更高级的文件操作  vector<string> fileList(const string &path, bool filter = true); //列出一个目录下所有文件名，filter可选是否应用过滤  void readFile(const string &path, vector<string> &lines); //读取一个文件的内容，写入lines中  void loadFile(const string &path, vector<string> &lines); //将lines内容写入一个文件  bool isTextFile(const string &path); //判断是否为文本类型文件  void Hint(const string &info, QWidget \*parent = MainWidget); //提示Api  void Warning(const string &info, QWidget \*parent = MainWidget);  void Error(const string &info, QWidget \*parent = MainWidget);  void mergeHash(string &hs, const string &name); //将名称混入哈希值  string getHash(const string &path, const string &name, const char tag); //获取文件哈希值  int str2int(const string &str); //常用类型转换  string int2str(int x);  QString Str2Q(const string &str);  string Q2Str(const QString &qstr);  string gbk2utf(const string &gbkStr); //用于windows文件操作的utf-gbk转换  string utf2gbk(const string &utf8Str); |
| 数据结构与算法 | Windows自带的MD5哈希算法，gbk，utf转换算法 |
| 补充说明 | 由于windows系统是gbk的，本项目所有涉及文件名的操作都需要经过utf-gbk转换，故不直接读取文件，全部通过command封装的函数，输入是utf-8编码的字符串。 |

## 4.1.2操作模块

|  |  |
| --- | --- |
| 模块名称 | Operator |
| 功能描述 | 实现大多数具体操作，分两部分，一部分是处理函数，以递归调用的函数为主；另一部分用户可直接调用，参数更通俗，经过检查和处理后进入递归函数处理。 |
| 接口与属性 |  |
| 数据结构与算法 |  |
| 补充说明 |  |

## 4.1.3模块－

|  |  |
| --- | --- |
| 模块名称 |  |
| 功能描述 |  |
| 接口与属性 |  |
| 数据结构与算法 |  |
| 补充说明 |  |

## 4.1.4模块－

|  |  |
| --- | --- |
| 模块名称 |  |
| 功能描述 |  |
| 接口与属性 |  |
| 数据结构与算法 |  |
| 补充说明 |  |

## 4.1.5模块－

|  |  |
| --- | --- |
| 模块名称 |  |
| 功能描述 |  |
| 接口与属性 |  |
| 数据结构与算法 |  |
| 补充说明 |  |

## 4.1.6模块－

|  |  |
| --- | --- |
| 模块名称 |  |
| 功能描述 |  |
| 接口与属性 |  |
| 数据结构与算法 |  |
| 补充说明 |  |

## 4.2数据处理层

## 4.2.1模块－

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 模块名称 |  | | | |
| 功能描述 |  | | | |
| 接口与属性 | 函数名 | 访问级别 | 参数 | 返回值 |
|  | public | 无 |  |
| 数据结构与算法 |  | | | |
| 补充说明 |  | | | |

## 4.2.2模块－

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 模块名称 |  | | | |
| 功能描述 |  | | | |
| 接口与属性 | 函数名 | 访问级别 | 参数 | 返回值 |
|  | public | 无 | Void |
| 数据结构与算法 |  | | | |
| 补充说明 |  | | | |

## 4.2.3模块－

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 模块名称 |  | | | |
| 功能描述 | 用于处理XSS漏洞 | | | |
| 接口与属性 | 函数名 | 访问级别 | 参数 | 返回值 |
|  | public | 无 | Void |
| 数据结构与算法 | 无 | | | |
| 补充说明 |  | | | |

## 4.2.4模块－

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 模块名称 |  | | | |
| 功能描述 |  | | | |
| 接口与属性 | 函数名 | 访问级别 | 参数 | 返回值 |
|  |  |  |  |
| 数据结构与算法 | 无 | | | |
| 补充说明 |  | | | |