**1版本控制**

**1.1代码版本**

前后台代码均托管github，以不同分支管理大的版本，遵循如下要求：

* master：最新可运行版本，从develop合并；
* develop: 最新开发版本，可能不能运行，主要作用是代码备份，防止本地最新文件丢失；
* realese-X.X: X.X为数字泛指，如1.0、1.1、2.0，作为稳定大功能的发布版本，由master拉出；
* tags：tag版本作为废弃不用的版本存档。

**1.2文档版本**

项目周期内各种文档，尽量以两种方式呈现，分别是博客文章和docx文档，类似接口文档、数据库这种不太好博客展示的，就仅以附件文档的格式输出。

其中docx文档存放于项目doc目录下，和代码一起托管于github，同时docx文档每次修改变更均要变更版本号并标注具体变更说明。

**2系统出错处理设计**

**2.1出错信息**

为方便后续处理，系统中的异常均需要人工捕获，避免直接出现系统类异常。需要有自定义的友好提示，自定义异常尽可能详细，使得前台提示更加清晰。我们提前定义几种可能出现的异常错误。并给出对应的处理方法。

|  |  |  |  |
| --- | --- | --- | --- |
| **输出形式** | **输出值** | **含义** | **处理方法** |
| int | 1011 | 连接字符串错误 | 正确定义连接字符串 |
| int | 1012 | 连接串为空 | 正确定义连接字符串 |
| int | 1013 | 数据库连接错误 | 正确定义连接字符串，并确认数据库正常 |
| int | 1021 | 查询SQL语句错误 | 正确定义SQL语句 |
| int | 1022 | SQL语句为空 | 正确定义SQL语句 |
| int | 1031 | 操作SQL语句错误 | 正确定义SQL语句 |
| int | 1032 | SQL语句为空，数据库未打开 | 正确定义SQL语句，打开数据库 |
| int | 1041 | 数据库关闭错误 | 正常关闭数据库 |
| int | 5501 | 指定路径的客户端文件不存在 | 重新指定文件路径 |
| Int | 1000 | 操作错误 | 弹出错误提示窗口 |

**2.2补救措施**

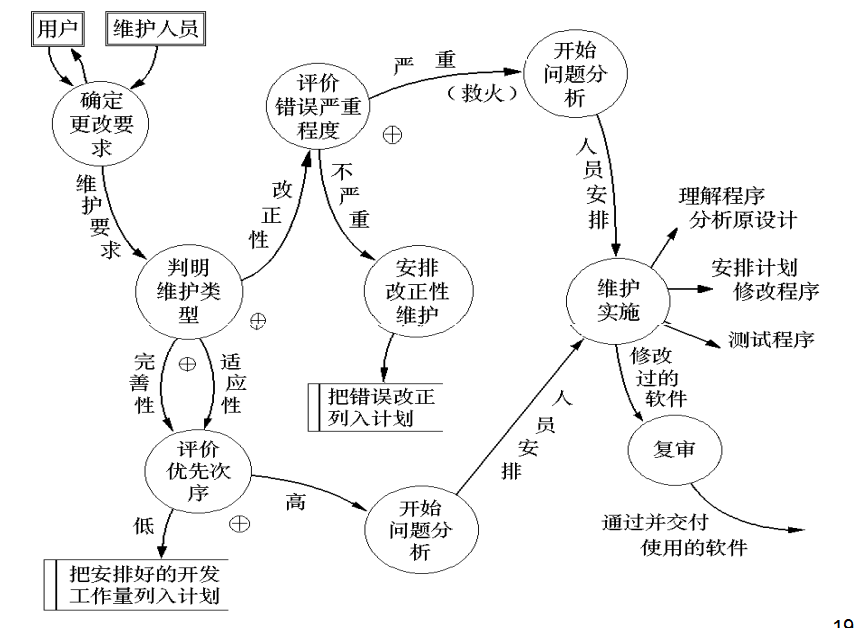
当系统运行过程中出现错误，自动捕获错误返回错误代号，并将错误信息反映到页面上。管理员根据错误提示查找对应的错误位置和处理方法。

**2.3系统维护设计**

为了有效地进行软件维护，应事先就开始做组织工作：

1. 首先建立维护的机构
2. 申明提出维护申请报告的过程及评价的过程
3. 为每一个维护申请规定标准的处理步骤
4. 建立维护活动的记录保管，并规定复审的标准

制定相应的事件流：



在出现维护需求后按照其类型和重要程度确定其优先级进行分析维护。并将维护结果存为另一版本程序上传至github。