系统设计

1. 引言

文件备份系统设计说明书是在指导老师和小组成员以及相关人员的帮助下完成的。在撰写本文档前，笔者全面了解了设计文档的要求，通过小组研讨确认了软件系统的设计架构。
通过阅读本说明书，可以全面了解本软件实现需求功能的方式，软件开发的详细流程以及系统整体设计架构。

1.1 编写目的

本软件系统概要设计报告是基于数据备份软件需求规格说明书编写的。系统设计说明书在满足功能需求的前提下，对被开发软件的系统进行了合理设计，对其具体功能实现方法、软件开发流程进行了适当规定，为软件的开发提供了基本模板和具体范式。读者通过阅读本书可以对开发流程、 软件架构有所了解。

1.2 项目风险

首要风险承担者及其各自在本阶段所需要承担的主要风险包括：

- 任务提出者：系统设计与预期设计不一致。
- 软件开发者：功能实现与系统设计不一致。
- 产品使用者：使用需求与软件设计不一致。

1.3 预期读者和阅读建议

可能的读者及其阅读建议包括：

- 用户：适用于具体用户界面部分。
- 开发人员：适用于全文档。
- 项目经理：适用于全文档。
- 测试人员：适用于全文档。
- 文档编写入员：适用于全文档。

1.4 参考资料

软件产品需求分析报告时所用到的参考文献及资料包括：

- 本项目的需求分析说明书。
- 本项目参考的课程 PDF。

2. 设计概述

本节描述了现有开发条件和需要实现的目标，说明了进行概要设计时应该遵循的设计原则和必须采用的设计方法。

2.1 限制和约束

该项目主要受到如下约束

- 技术条件：本科开发水平
- 人数：3
- 开发环境(包括：工具和平台)：Ubuntu 22.04.1
- 开发语言限制：C++
- 其他限制：不使用外部依赖库

在以上条件下，项目应完成一个具有文件类型支持，元数据支持，路径自定义备份，压缩解压，打包解包，加密备份，友好图形界面功能的文件备份软件。

2.2 设计原则和设计要求

本软件系统进行概要设计的原则，包括以下几方面的内容：

- 命名规则；文件命名具有可读性，如 compress、file_info 等
- 模块独立性原则；模块可独立运行、独立测试功能
- 系统易操作性要求；用户操作界面简洁

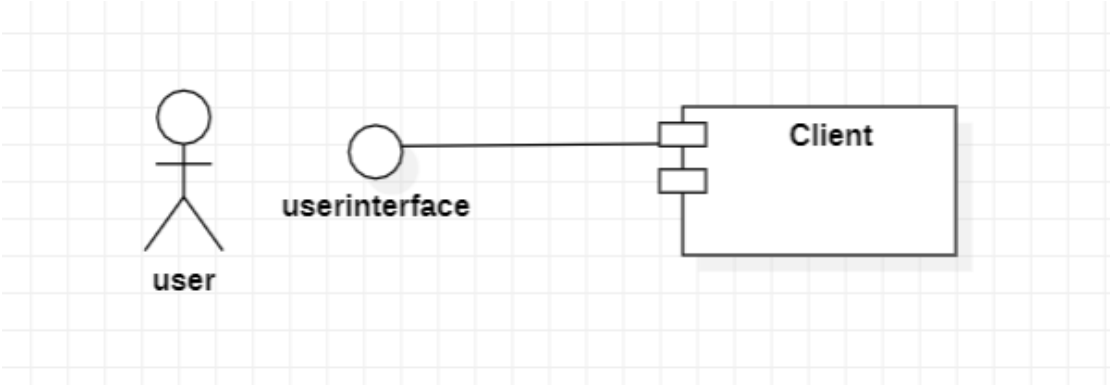
3. 总体设计

3.1 系统结构设计

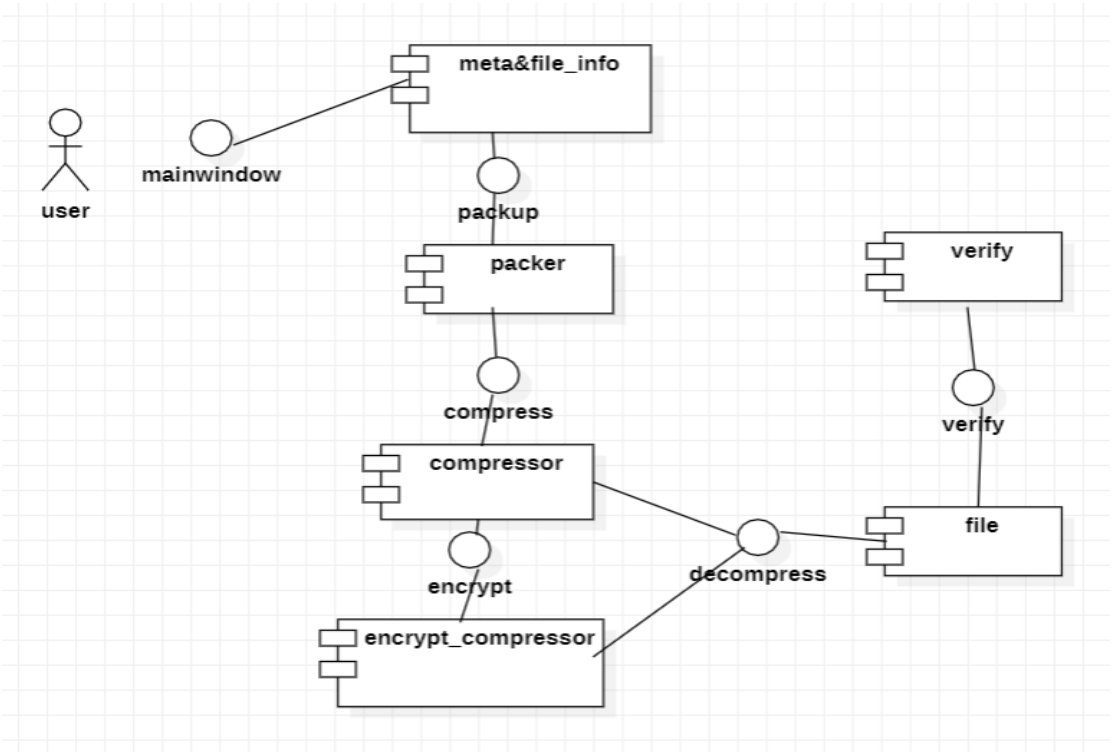
此处提供了高层系统结构的描述，使用构件图显示了本系统主要的组件及组件间的交互。

3.1.1 顶层系统结构

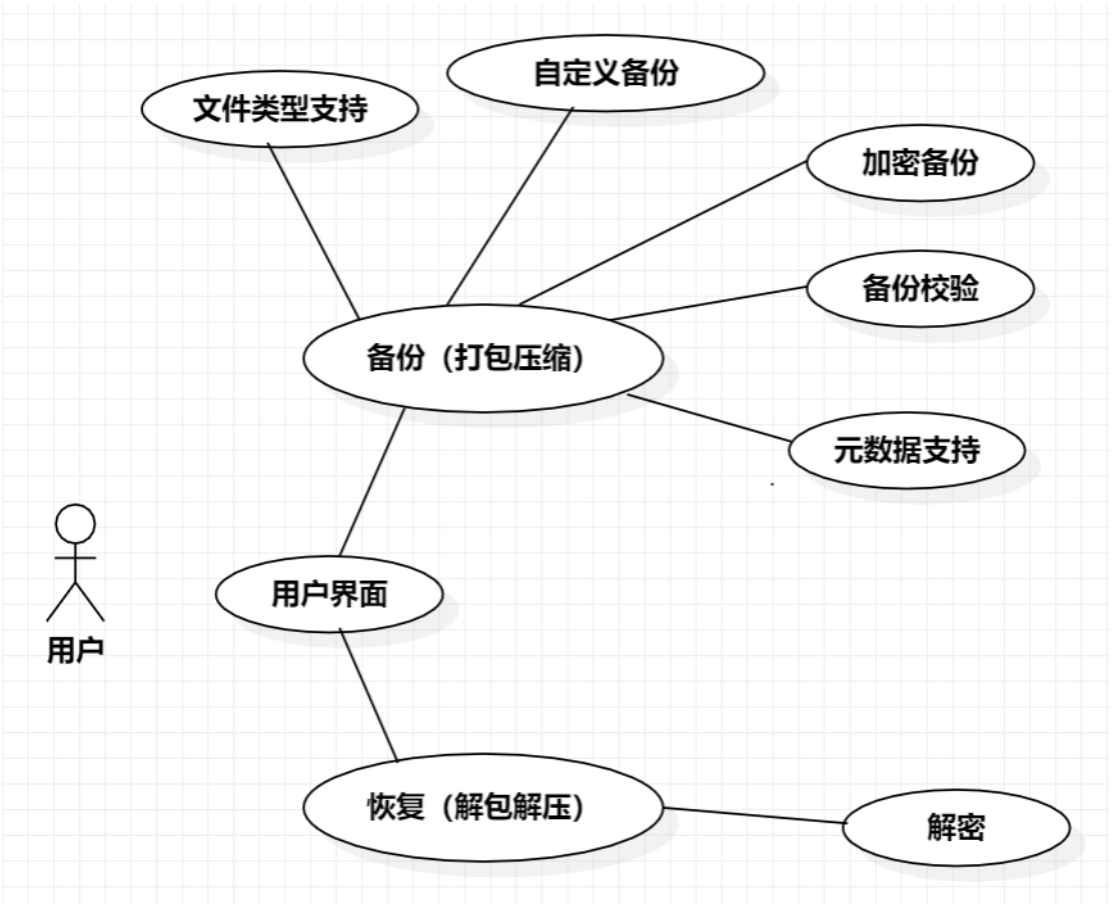
本软件未提供网络和数据库功能，因而系统的顶层系统结构仅由客户端构成。



3.1.2 客户端结构



3.1.3 系统设计



3.2 系统界面

系统界面为 QT 提供的素材设计而成。
进入界面后会有以下提示。

由于系统界面较简单且清晰，因此不做更详细的介绍，根据提示做操作即可。

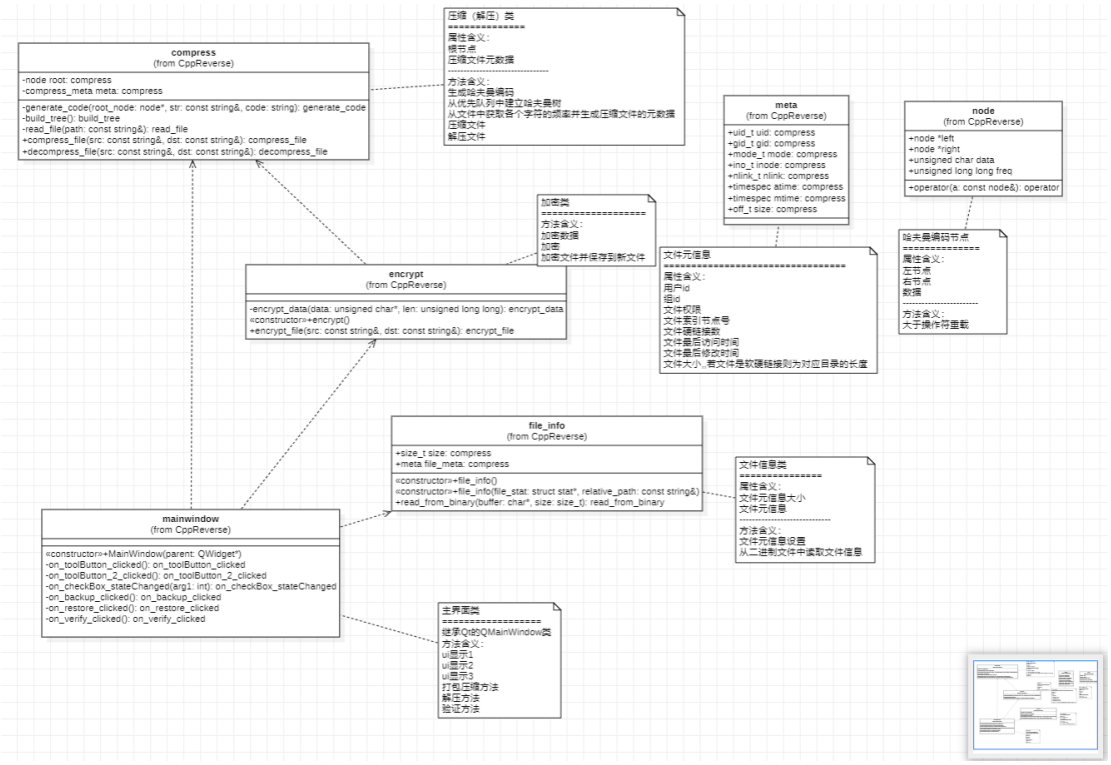
3.3 约束和假定

4.3.1 客户约束：
依赖库约束：项目不能使用第三方依赖库。
4.3.2 其他约束语言约束：
统一使用 C++进行软件开发。由于 QT 支持对 C++的编译，为了使项目代码实现、管理较为方便，均采用 C++作为开发语言。

4. 静态建模（对象模型）

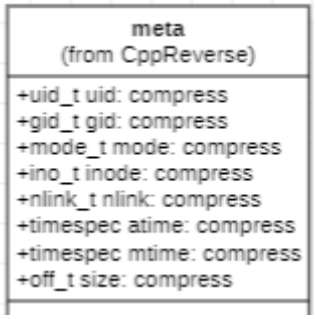
4.1 系统对象模型

5. 类（对象）描述



5.1 系统中的类（对象）与结构体

5.1.1 meta 结构体：

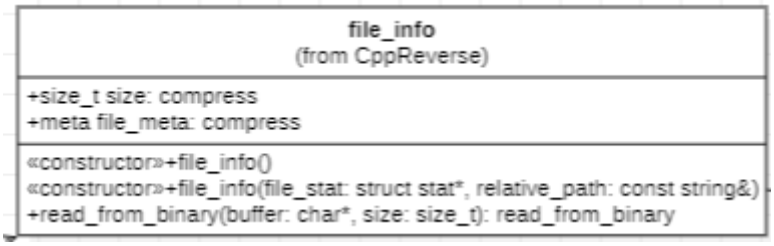


结构体说明如下：文件元信息结构体

属性含义
用户 id

组 id
文件权限
文件索引节点号
文件硬链接数
文件最后访问时间
文件最后修改时间
文件大小（软硬链接则为对应目录长度）

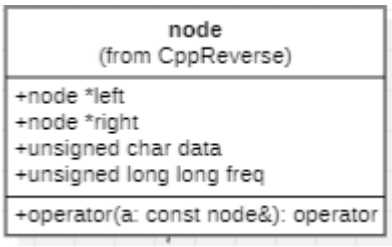
5.1.2 file_info 对象：



对象的说明如下：读取一个文件元信息

属性含义	方法含义
文件元信息大小	获取文件元信息
文件元信息	打开一个文件
	备份文件
	验证备份
	恢复文件

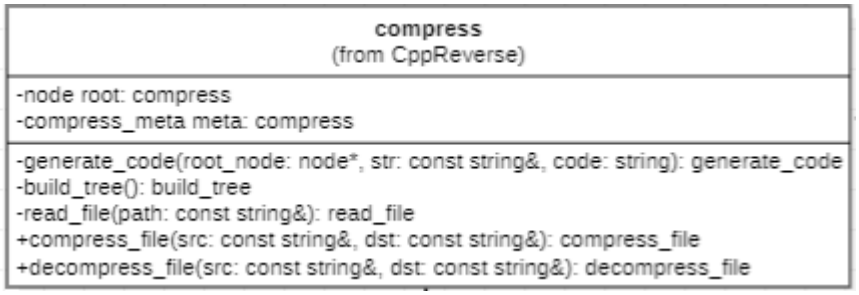
5.1.3 Node 结构体：



结构体的说明如下：哈夫曼树节点

属性含义	方法含义
左子树节点	重载大于操作符
右子树节点	
节点数据	
节点数据频率	

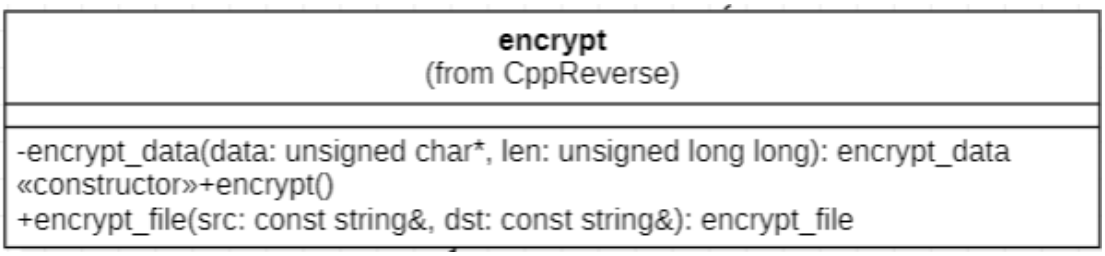
5.1.4 compress 对象：



对象的说明如下：用于文件的加解压

属性含义	方法含义
字典树根节点	生成哈夫曼编码
压缩文件元数据	构建哈夫曼树
	从文件中获取各个字符的频率并生成压缩文件的元数据
	压缩文件
	解压文件

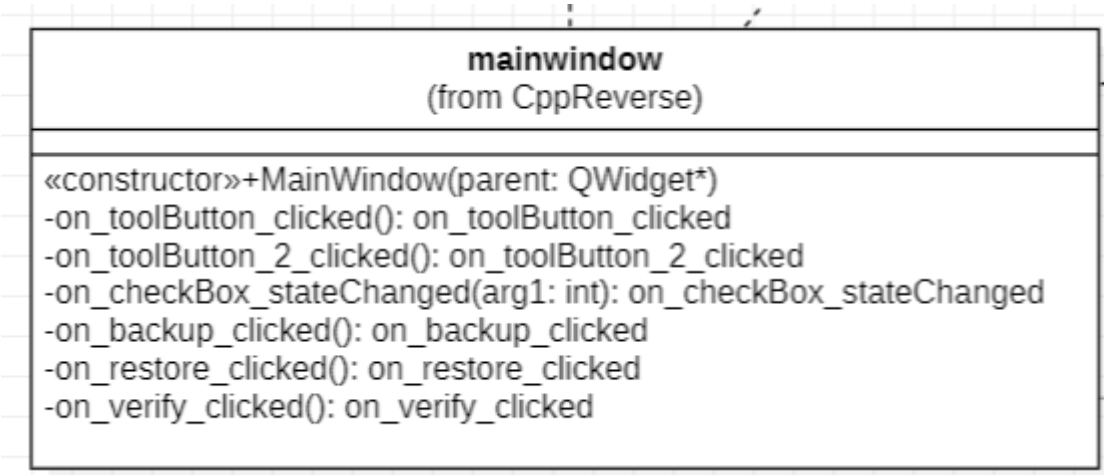
5.1.5 Encrypt 对象：



对象的说明如下：加密文件。

方法含义
加密数据
加密
加密文件并保存到新文件

5.1.6 Mainwindow 对象：



对象的说明如下：用户主界面 ui。

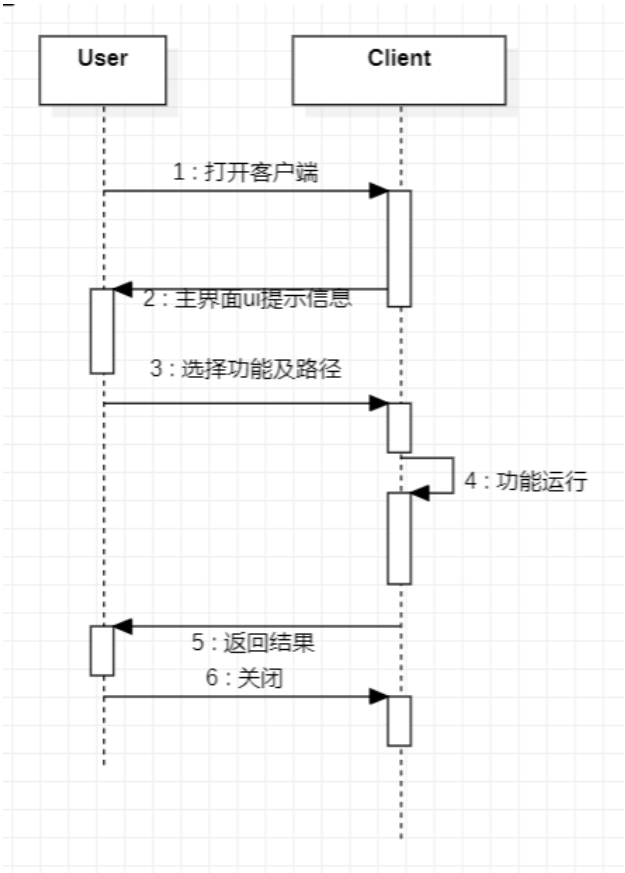
方法含义
Mainwindow 启动方法
选择备份目录 ui
选择保存路径 ui
打包压缩方法
解压方法
验证备份方法

6. 动态建模

这部分的作用是描述系统如何响应各种事件，确定了不同的场景（Scenario）。

6.1 场景（Scenarios）

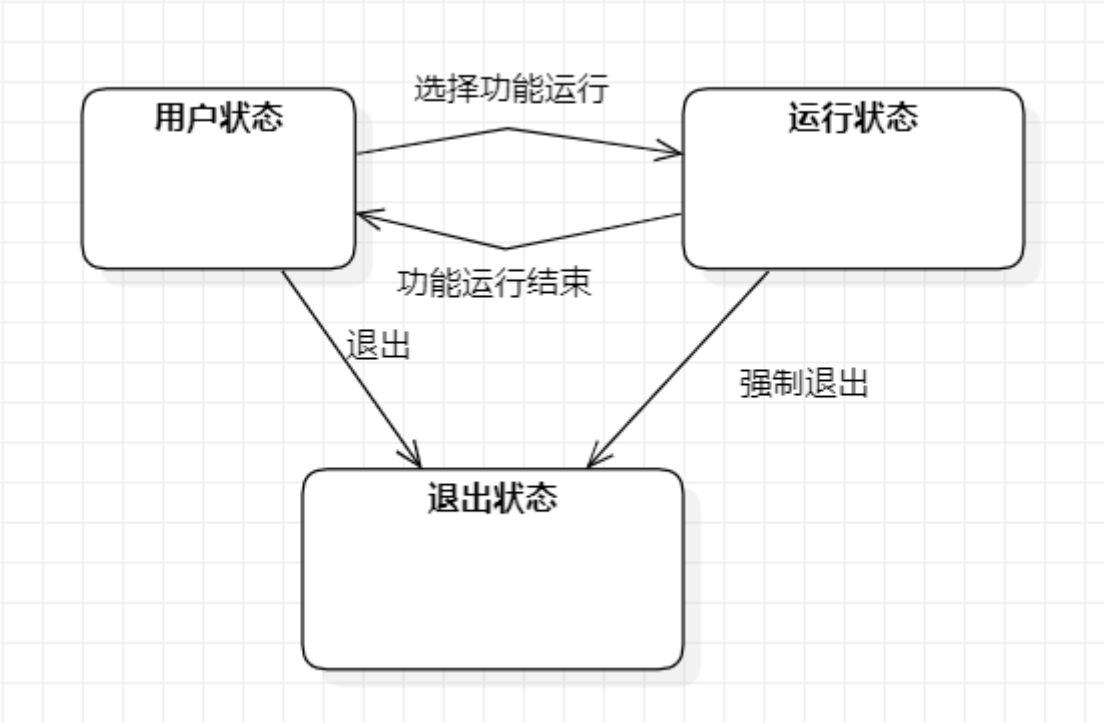
6.1.1 场景：备份文件场景
描述：文件备份
顺序图：



详细说明：
1、用户打开客户端

2、客户端 UI 提示
3、用户选择软件功能，备份、还原、校验 以及备份路径和还原路径
4、后端进行功能运行
5、返回运行结果
6、用户关闭软件

6.2 状态图



7. 支撑环境

7.1 数据备份系统

本次项目采用 QT 内置 UI 素材制作主界面 UI
数据库管理系统、以及安装配置情况如下：
编译环境 ：
产品名称：gcc
发型厂商：GNU
版本号：gcc 11.2.0
语言：C++
虚拟机运行平台：
产品名称：VMware Workstation Pro
发行厂商：vmware

版本号：Workstation 16 Pro

虚拟机镜像：

产品名称：Ubuntu

发行厂商：Cannoical

版本号：Ubuntu 22.04.1 LTS

7.2 硬件环境

- 机型；虚拟机 Ubuntu 20.04.3 LTS
- 主频；2.2Ghz
- 内存容量；2GB
- 磁盘容量；50GB
- 特殊部件；VMware Workstation
- 操作系统；Linux