知识点列表

编号	名称	描述	级别
1	开发流程	从需求设计再到编码和测试的瀑布式项目开发流程	*
2	三层体系架构	由用户界面层、业务逻辑层和数据访问层组成的三层体系架构	*
3	三层逻辑模型	由接口层、实现层和逻辑对象层组成的三层逻辑模型	*
4	接口设计实现	基于继承与多态的抽象接口设计与实现	*
5	用户界面设计	面向控制台应用的字符界面设计	**
6	业务逻辑设计	连接界面与数据的业务逻辑流程	**
7	数据存储设计	基于文件系统的数据存储与访问	**
8	多文件构建	基于Makefile的多源文件构建技术	**

> *理解级别 **掌握级别 ***应用级别

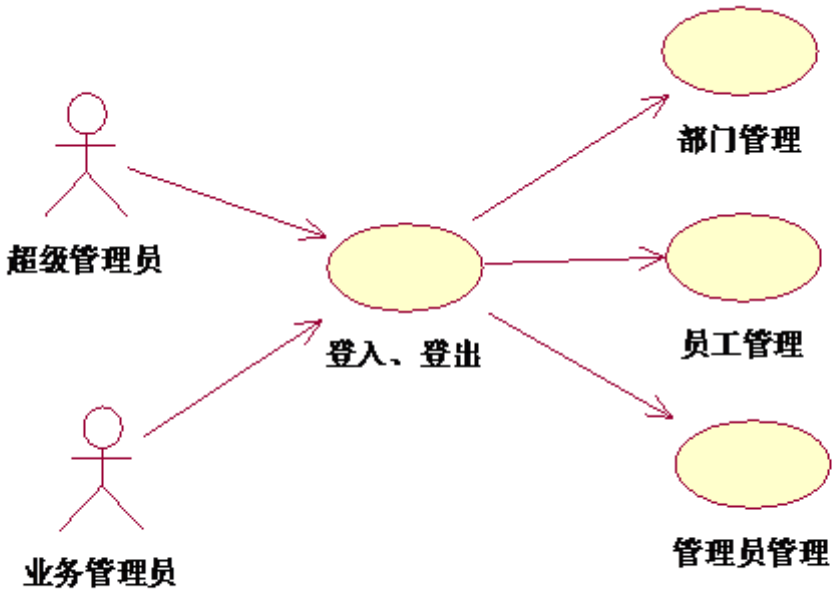
1. 概述

- 本项目旨在通过一个简化的企业管理信息系统（Enterprise Management Information System，EMIS）项目，使学生在完成对 C/C++程序设计语言和基本数据结构与算法课程的学习后，综合运用所学到的语法和算法知识，构建一个接近实际应用场景的软件系统，以达到复习和巩固前期课程内容并为后续课程奠定基础的目的。
- 通过本项目的实施，学生可以初步了解包括需求分析、概要设计、详细设计、开发计划、编码测试等环节在内的软件项目开发流程，以及相关技术文档的撰写规范，为以后从事软件项目研发工作增加实践经验。
- 本案在系统设计方面有意识地采用多层体系架构的设计理念，旨在帮助学生逐步树立产品观念，从更高的角度，以更广的视野，综合考虑用户需求、技术路线和研发成本间的矛盾，深刻理解软件系统的可维护性、可扩展性对企业可持续性发展的重要意义。

2. 需求分析

2.1. 总体需求

企业管理信息系统主要用于实现对企业基本信息的管理。具体包括对企业部门的管理、对企业员工的管理，以及对管理信息系统本身的管理。



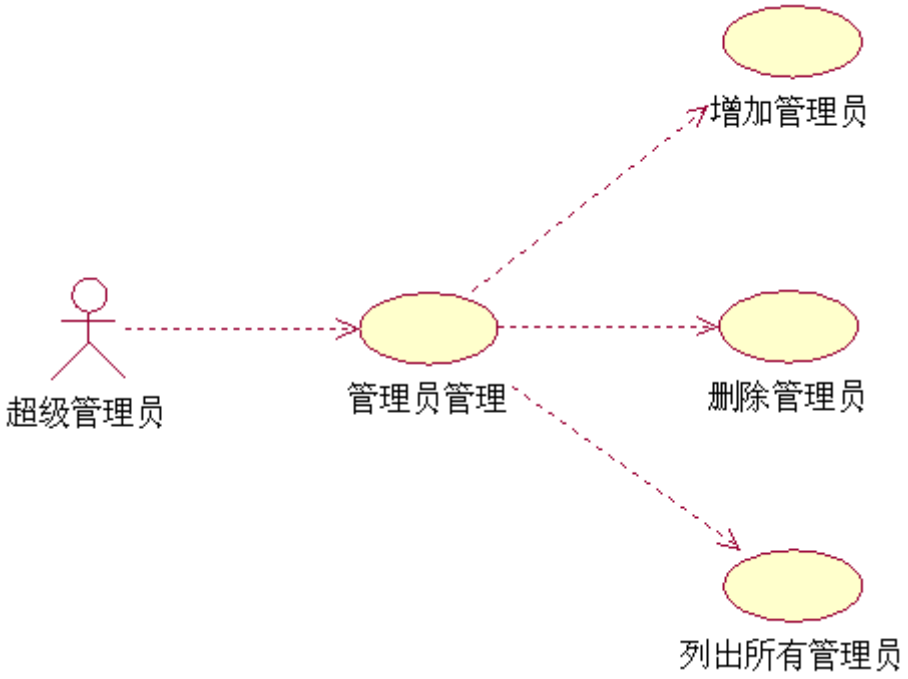
其中，对管理信息系统本身的管理主要是指对管理员的管理，这方面的需求可被归纳为管理需求，而对企业部门和员工的管理则被归纳为业务管理。

2.2. 管理需求

管理需求主要包括：

- 增加管理员：根据屏幕提示依次输入管理员的用户名和密码，系统自动为其分配ID号，并向用户提供反馈信息。
- 删除管理员：根据屏幕提示输入欲删除管理员的ID号，系统将该管理员删除，并向用户提供反馈信息。

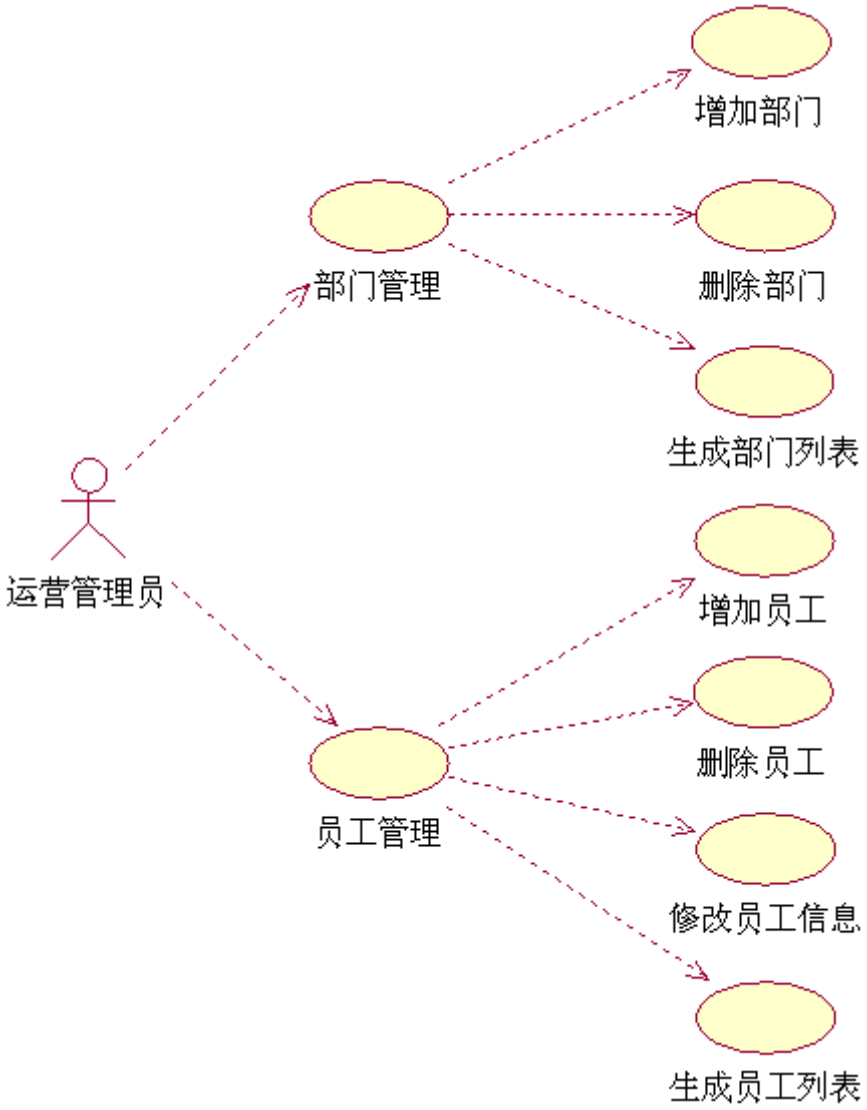
- 列出所有管理员：系统以列表形式显示所有管理员的ID号、用户名和密码。



2.3. 业务需求

业务需求主要包括：

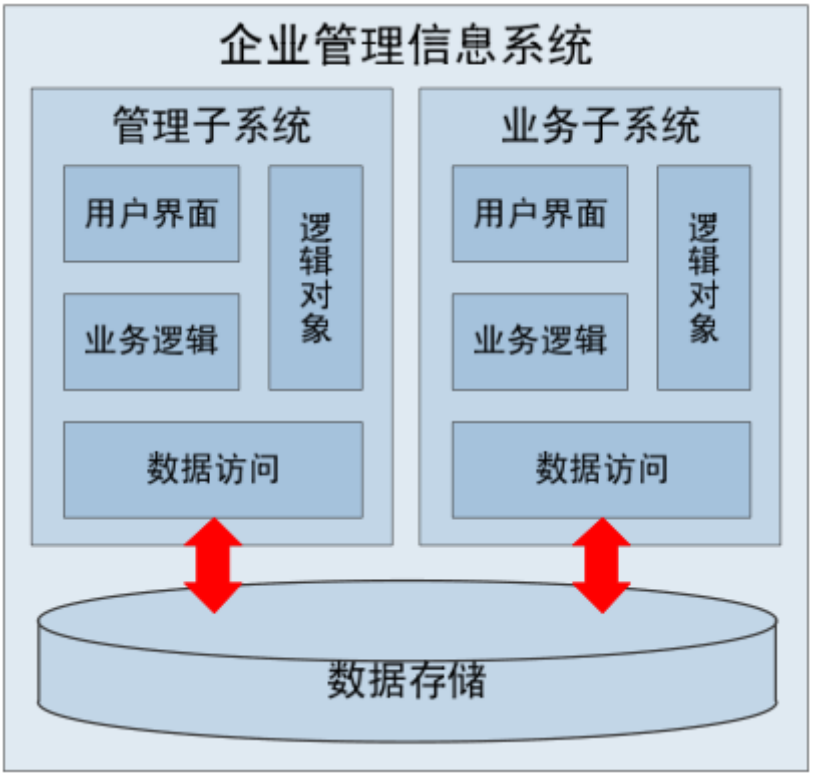
- 部门管理包括：
  - 增加部门：根据屏幕提示输入部门名称，系统自动为其分配ID号，并向用户提供反馈信息。
  - 删除部门：根据屏幕提示输入欲删除部门的ID号，系统将该部门删除，并向用户提供反馈信息。
  - 列出所有部门：系统以列表形式显示所有部门的ID号、部门名称和员工人数。
- 员工管理则包括：
  - 增加员工：根据屏幕提示依次输入员工的姓名、性别、年龄，以及所属部门的ID号等信息，系统自动为其分配ID号，并向用户提供反馈信息。
  - 删除员工：根据屏幕提示输入欲删除员工的ID号，系统将该员工删除，并向用户提供反馈信息。
  - 修改员工信息：根据屏幕提示选择要修改的员工信息然后并输入，系统更新与该员工有关的信息数据，并向用户提供反馈信息。
  - 生成员工列表：根据屏幕提示输入部门的ID号，系统以列表形式显示该部门所有员工的ID号、姓名、性别和年龄。
  - 列出所有员工：系统以列表形式显示所有员工的部门、ID号、姓名、性别和年龄。



3. 概要设计

3.1. 总体架构

根据前述需求分析，本案在逻辑上可被划分为管理子系统和业务子系统两大模块，分别用于实现对管理员的管理和对部门及员工的管理功能。此外还需提供必要的数据存储策略，以实现对所有数据的持久化。系统总体架构如图所示：



□ 管理子系统：实现对管理员的管理功能。具体包括增加管理员、删除管理员、列出所有管理员。

□ 用户界面：显示主菜单、接受用户输入、向用户显示提示信息、处理结果和必要的反馈。

□ 业务逻辑：具体实现主菜单的各个功能项，以逻辑对象为载体，在用户界面和数据访问之间传递有关管理员的信息数据。

□ 数据访问：实现逻辑对象与数据存储之间的序列化与反序列化。

□ 逻辑对象：实现管理员对象的逻辑模型。

□ 业务子系统：实现对部门及员工的管理功能。具体包括增加部门、删除部门、列出部门、增加员工、删除员工、修改员工信息、列出部门员工、列出所有员工。

□ 用户界面：显示运营管理子菜单、接受用户输入、向用户显示提示信息、处理结果和必要的反馈。

□ 业务逻辑：具体实现运营管理子菜单的各个功能项，以逻辑对象为载体，在用户界面和数据访问之间传递有关部门及员工的信息数据。

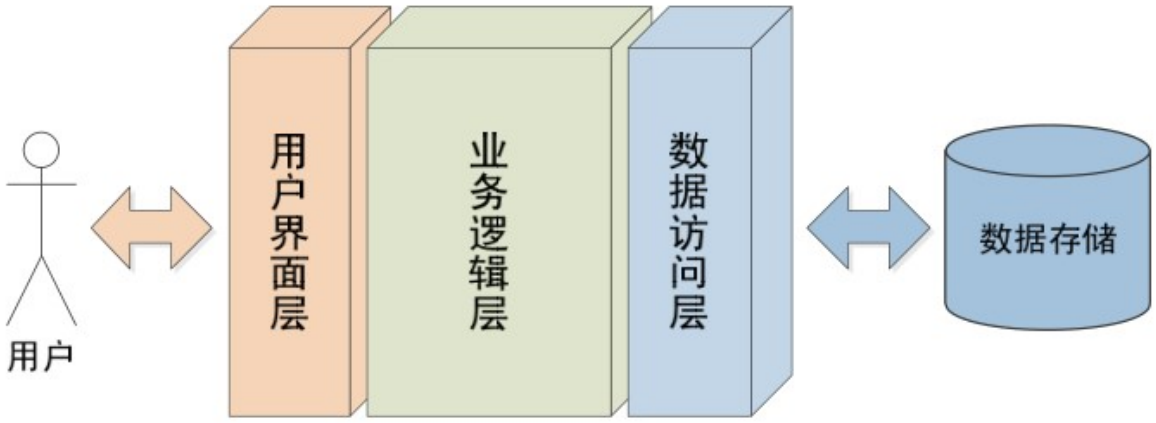
□ 数据访问：实现逻辑对象与数据存储之间的序列化与反序列化。

□ 逻辑对象：实现部门及员工对象的逻辑模型。

□ 数据存储：实现整个管理信息系统的数据持久化。

3.2. 体系架构

本案在水平方向上采用三层体系架构。如图所示：



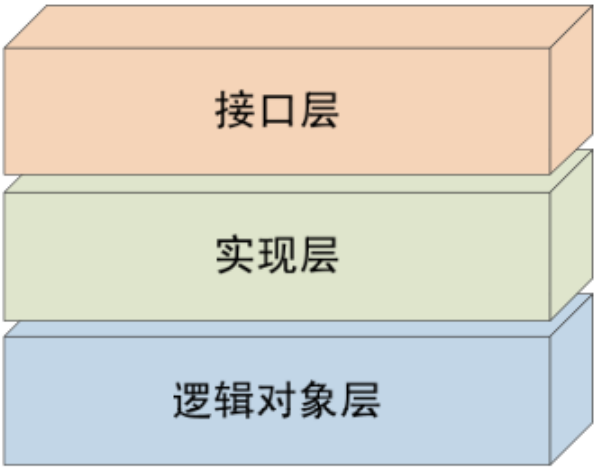
□ 用户界面层：处理与最终用户的交互，既负责从用户处收集信息，也负责向用户展现结果、给出提示或反馈。

□ 业务逻辑层：针对用户界面层所体现的功能项，以数据访问层为基础，实现与业务逻辑相关的算法和流程。

□ 数据访问层：实现对数据存储介质的访问，为业务逻辑层提供数据源，并接受其处理结果。

3.3. 逻辑模型

本案在垂直方向上采用三层逻辑模型。如图所示：



□ 接口层：定义各功能模块的抽象接口，降低模块间的耦合性，提高代码复用率，降低维护成本。

□ 实现层：对抽象接口的具体实现。本案用户界面层的接口实现拟采用控制台方式，而数据访问层的接口实现则采用文件系统方式。

□ 逻辑对象层：以逻辑模型的方式对系统中的相关数据加以组织，并构成从用户界面到业务逻辑再到数据访问各层之间的信息载体。逻辑对象包括：管理员、部门和员工。

3.4. 平台约束

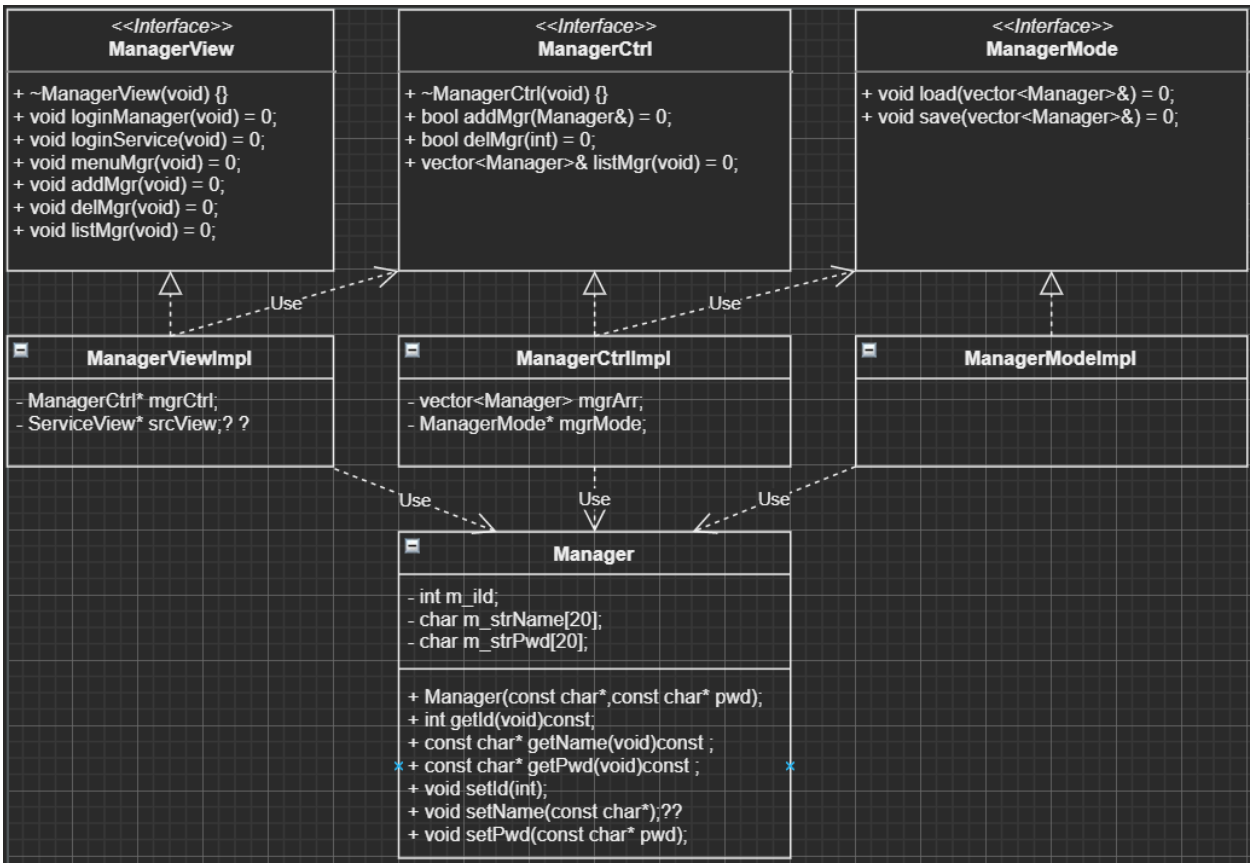
平台约束	说明
硬件环境	32 位 Intel x86 及其兼容处理器的个人计算机
操作系统	Ubuntu 12.04 LTS
开发工具	GCC 4.6.3，C/C++标准库
应用类型	命令行应用程序
用户界面	非全屏模式的控制台字符界面
数据存储	二进制及纯文本文件
平台中立	不要求
交叉编译	不要求

4. 详细设计

4.1 管理子系统

本案管理子系统由用户界面、业务逻辑、数据访问和逻辑对象四部分组成。

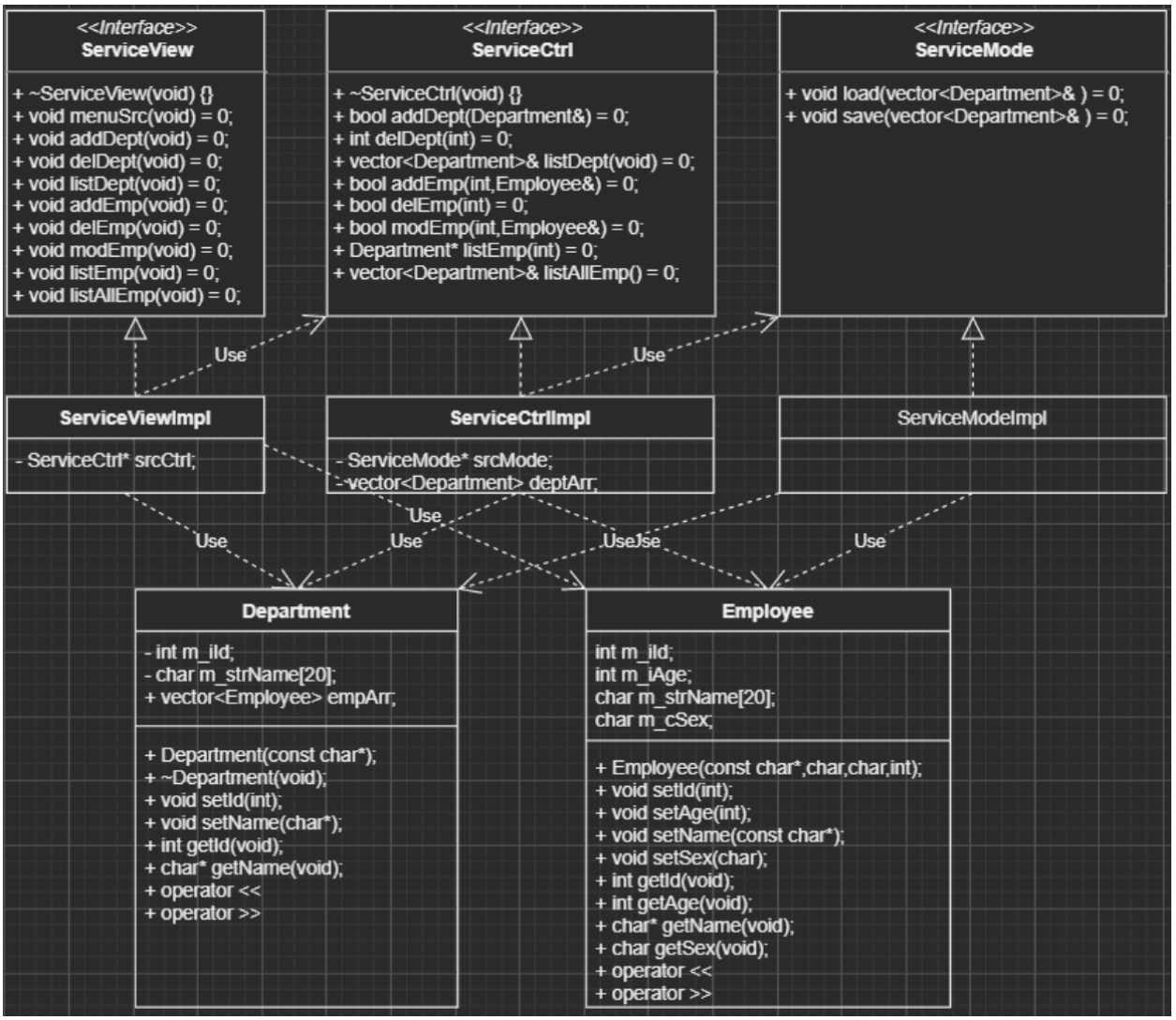
其中用户界面、业务逻辑和数据访问又分别包括接口和实现两部分。如图所示：



4.2. 业务子系统

本案业务子系统由用户界面、业务逻辑、数据访问和逻辑对象四部分组成。

其中用户界面、业务逻辑和数据访问又分别包括接口和实现两部分。如图所示：



4.3. 基础设施及辅助工具

- get_mgrid()：生成唯一的管理员ID号。
- get_deptid()：生成唯一的部门ID号。
- get_empid()：生成唯一的员工ID号。
- 注意：从id.dat配置文件中读取上次生成的ID号，将其加1后重新写入，同时返回新生成的ID号 。*

4.4. 配置文件

- id.dat：唯一ID号配置文件。保存最后一次生成的ID号，每次新生成的ID号即在此基础上加1 。

4.5. 数据存储

managers.dat：管理员信息数据库。以二进制形式保存全部Manager对象

services.dat：部门及员工信息数据库。形如：

```
103 研发部  3
1008 王曦文 m 21
1009 李政 m 22
1010 杨超 m 23
104 销售部  1
1012 林豪  w 22
101 财务部 1
1011 戚钧恒 w 23
```

1. 部门记录，包括三个字段：部门ID号、部门名称和该部门的员工人数。程序可以以部门员工人数字段的值作为后续读取员工记录的循环控制上限；
2. 每个部门记录下面紧跟着隶属于该部门的员工记录，包括四个字段：员工ID号、员工姓名、员工性别（1表示男性，0表示女性）和员工年龄；
3. 为了便于程序通过标准I/O流进行格式化访问，各字段之间以空格分隔。

5. 文件组织

5.1. 代码文件

5.1.1. 管理子系统

- manager_view.h：定义ManagerView抽象基类
- manager_view_impl.h：声明ManagerVieImpl类
- manager_view_impl.cpp：实现ManagerViewImpl类
- manager_ctrl.h：定义ManagerCtrl抽象基类
- manager_ctrl_impl.h：声明ManagerCtrlImpl类
- manager_ctrl_impl.cpp：实现ManagerCtrlImpl类
- manager_mode.h：定义ManagerMode抽象基类
- manager_mode_impl.h：声明ManagerModeImpl类
- manager_mode_impl.cpp：实现ManagerModeImpl类
- manager.h：声明Manager类
- manager.cpp：实现Manager类

5.1.2. 业务子系统

- service_view.h：定义ServiceView抽象基类
- service_view_impl.h：声明ServiceViewImpl类
- service_view_impl.cpp：实现ServiceViewImpl类
- service_ctrl.h：定义ServiceCtrl抽象基类
- service_ctrl_impl.h：声明ServiceCtrlImpl类
- service_ctrl_impl.cpp：实现ServiceCtrlImpl类
- service_mode.h：定义ServiceMode抽象基类
- service_mode_impl.h：声明ServiceModeImpl类
- service_mode_impl.cpp：实现ServiceModeImpl类
- department.h：声明Department类
- department.cpp：实现Department类
- employee.h：声明Employee类
- employee.cpp：实现Employee类

5.1.3. 基础设施及辅助工具

- main.cpp：定义main()函数
- emis.h：声明全局变量
- emis.cpp：定义全局变量
- tools.h：声明工具函数
- tools.cpp：定义工具函数

5.2. 脚本文件

- Makefile：项目制作脚本