

**APpointment Organizer (APO)**

*A Novel Scheduling Assistant for Your Daily Appointments*

**COMP2432 Operating System Group Project - Stage 1 Report**

Yuan Yuncheng, Zhang Wenxuan, Chen Ru Bing, DU Haoxun

Contents

[1. Project design 2](#_Toc128749818)

[1.1 Overall Design 2](#_Toc128749819)

[1.2 Program Structure 3](#_Toc128749820)

[1.2.1 Overview 3](#_Toc128749821)

[1.2.2 Main function 3](#_Toc128749822)

[1.2.3 APO module 3](#_Toc128749823)

[1.2.4 User process 3](#_Toc128749824)

[1.2.5 Classes 4](#_Toc128749825)

[1.2.6 Protocol 4](#_Toc128749826)

[2. Work Distribution 5](#_Toc128749827)

[3. Project Timeline 6](#_Toc128749828)

[3.1 Stage 1 6](#_Toc128749829)

[3.2 Stage 2 6](#_Toc128749830)

[4. Prototype Demonstration 7](#_Toc128749831)

# Project design

## 1.1 Overall Design

In today’s fast-paced world, individuals have numerous commitments such as appointments, meetings and social events. To manage their schedules effectively and prevent scheduling conflicts, people often rely on digital assistant tools like PDA or phone organizers to keep track of their engagements. A scheduling assistant for setting a meeting time or gathering date is in burgeoning request. APO is a C based scheduling program for making the process of multi-people scheduling appointments more efficient and easier to interact.

For the whole structure, we divide the project into two parts, which respectfully are the prototype and relatedly detailed modules. In the first stage, we discussed the construction of the project together while accomplishing the prototype. The diagram version of the structure of the whole project is shown as **Figure 1**.

The diagram indicates the program flow, including a process of completing the program from input to the final output. We receive the users’ input command line and pass through it into several modules depending on the particular requirement, which implement various functions and maximize the user satisfaction.

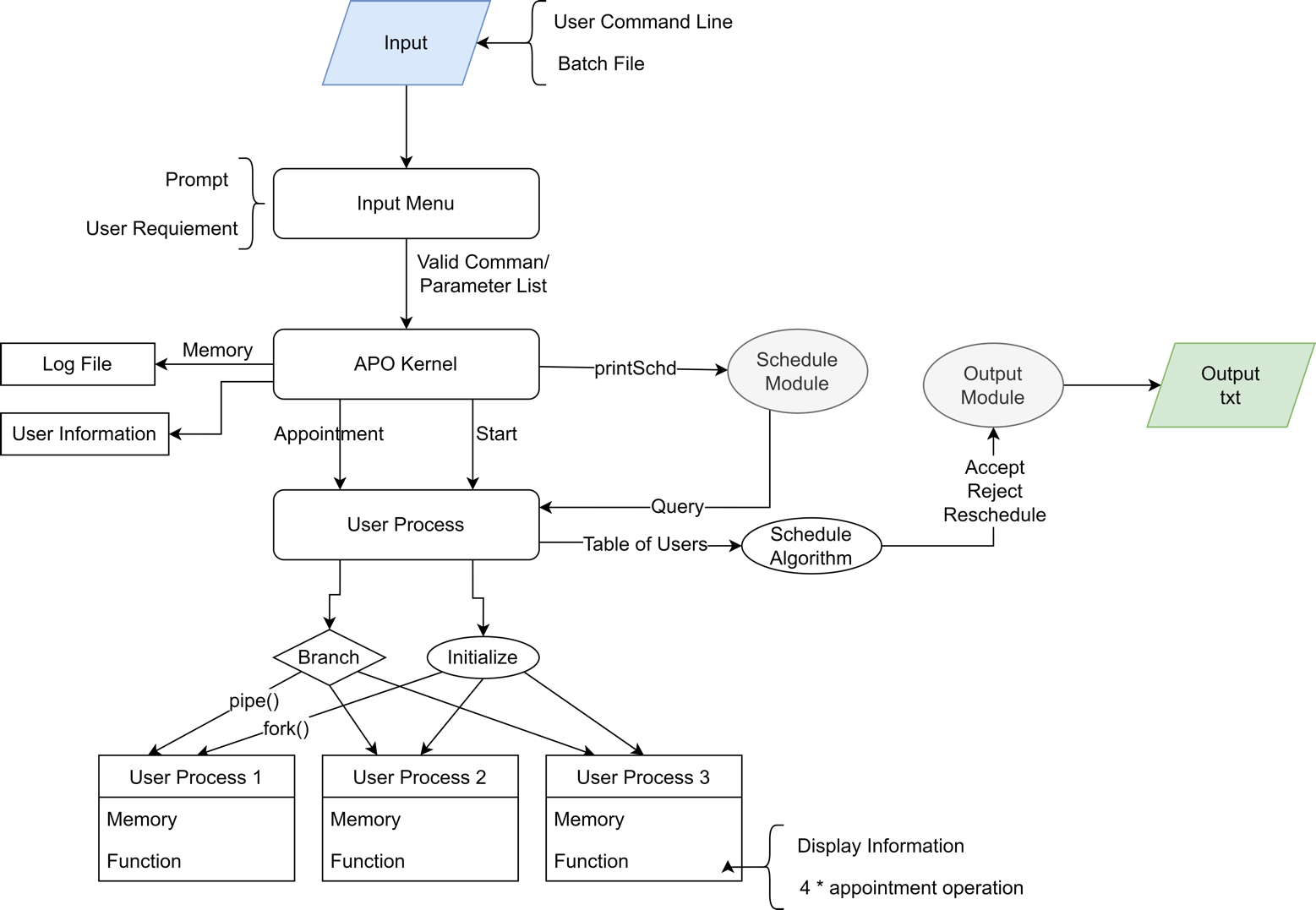


Figure 1 - Structure Design Diagram

## 1.2 Program Structure

### 1.2.1 Overview

图形用户界面, 文本

描述已自动生成This part will briefly explain the main structure of the project to show the already finished work and what has yet to be done.

The project mainly consists of 5 parts: classes, protocol, APO modules, main function, and user process function. We will talk about each of them later.

### 1.2.2 Main function

The main function is the entrance of the program. It will initialize the program, create the child processes and interact with the user. By the end of the process, the program will free up the environment before exiting the main function.

### 1.2.3 APO module

APO module is the kernel of the program. It interprets the directive from users and performs the corresponding operations. APO modules include an appointment module, schedule module, and output module.

文本

描述已自动生成The appointment module interprets the “appointment” instructions and notifies the user process by inter-process communication. The scheduling module interprets the printed instruction, collect personal schedules from user processes, then analyses the schedules. The output module translates the analysis result of APO into the standard output format.

The main parts of the APO modules are already finished, although some static functions still need to be implemented (The interfaces of them have already been defined). As for the output module, the programming work has yet to start, but its interface is defined clearly in the txt file.

### 1.2.4 User process

The user process part implements the functionality of the child process in the program. The child process saves the personal schedule of a particular user and interacts with APO. This part is already finished.

### 1.2.5 Classes

We use the concept of classes in object-oriented programming to encapsulate several frequently used data types and methods. This design greatly simplifies the process of handling the variables when developing the APO modules.

文本

描述已自动生成

The “class” is implemented by struct and Enum in C languages. The method oriented to handling these data types is also included in this part.

电脑屏幕截图

描述已自动生成

This part is about to complete except for the scheduling algorithms (The functions to implement the algorithms are regarded as methods of algorithm data type and have a unified interface. )

### 1.2.6 Protocol

The protocol part is designed to simplify the development of APO modules, letting the program achieve better encapsulation and portability. Almost all of the inter-process communication details are encapsulated in these functions. When developing the APO modules, we just need to invoke the protocol API to do the inter-process communication.

文本

描述已自动生成

There are two protocols in total. The structure imitates the application layer protocol in computer networking.

文本

描述已自动生成

This part hasn’t started for programming, but the interface of the protocols has already been defined. In addition, we develop the prototype to test the function hierarchy of IPC in the program.

# Work Distribution

We have already set up the structure of the program and implemented the main part of APO. The interfaces of the unimplemented functions are defined clearly with specific specifications about the input and output. The remaining work is to implement these interfaces.

For the second stage, we are going to fulfill the structure with further codes to finish the project. We split the rest of the work into three separate parts, which are protocol writing, output module, and tiny methods writing with inputting the scheduling algorithms.

(The division of stage 2’s mission)

Each one of the members is to take one or collaborate on one part of stage 2’s work. To avoid problems led by personal misunderstanding, we organize the weekly meeting during stage 2, offering everyone a chance to update their work and check others to guarantee everyone is on the right track.

# Project Timeline

## 3.1 Stage 1

## 3.2 Stage 2

图示, 示意图

描述已自动生成

# 4. Prototype Demonstration

The demonstration so far is taking by screenshotting the execution output of the main function.

图形用户界面, 文本

描述已自动生成