COMP 3004 Term Project Design Document

Design Patterns

Model-View-Controller Design Pattern

After several discussions regarding the design of various classes and their interactions, the group decided to implement the MVC design pattern. In this application, each aspect of the MVC design is delineated as follows:

- View: The program includes two separate GUIs, mainwindow.ui is the GUI which the user interacts with to use the device's main functionality, whereas testui.ui is the development GUI that the user can use to view the plots of waveforms. testui.ui window is only displayed when the user starts a new session.
- Controller: There are two controller classes corresponding to the two previously introduced views.
    - MainWindow is responsible for controlling the behavior of the main GUI. This includes starting a new treatment session, pausing and stopping an ongoing session, editing the date and time of the device, and viewing the session log history. The MainWindow class receives the user actions, processes the action, sends a signal to the device (or calls a device function), and change the state of the mainwindow.ui if necessary. In addition, it creates a thread for a device object to enable sending and receiving data to the Device class while remaining responsive to user interaction.
    - TestUI is a secondary window used to plot the waveforms sent from the Device class to MainWindow. The user can select from the wave band type, EEG number, and round of treatment to view the waves produced by the hypothetical brain. TestUI uses an external library called QCustomPlot.

When the model's state is changed, it sends a signal to the MainWindow or TestUI to enable them to notify the user of the change. For example, when a treatment session is initiated by the user, a series of consecutive changes in the model's state happen that the controller need to be informed of. These changes happen when the session phase (e.g. EEG connection, analysis, treatment) changes, new wave data is obtained, a possible EEG connection loss occurs, and a session ends.

- Model: The model notifies the view when its state has changed. The classes used in the model include
    - Device simulates a Neureset device where connection to EEG headset is made. When the user starts a new treatment session, the Device class sends the request to SessionManager, while waiting for the SessionManager to perform the treatment procedure. During the treatment, EEG signals are sent to the Device to be displayed on the TestGUI, a member field of the MainWindow. Once the treatment is completed, Device receives the initial and final dominant frequencies of all EEG sites and store them in a data structure in case the user asks to upload them to the PC.
    - SessionManager is the class where the management of session activities happen. This includes functions regarding starting, resuming, and stopping a session, receiving wave data obtained by the EEG sites, and that data to the Device class for storage or presentation purposes.
    - EEG plays the role of an EEG site attached to a headset. EEG can receive the wave data generated by the WaveGenerator class and send it to the SessionManager.
    - WaveGenerator returns four wave bands when its generateWave() function is called. These wave bands include alpha, beta, delta, and theta. Each wave band is created by combining three sine functions who have received a small random value to make the waveforms of each EEG and each round of treatment different from others.

- SignalProcessor is responsible for calculating the initial and final dominant frequencies. For this purpose, it uses the following formula
$$f_d = (f_1 \times A_1^2 + f_2 \times A_2^2 + f_3 \times A_3^2)/(A_1^2 + A_2^2 + A_3^2)$$
where f_i and A_i are the frequency and amplitude of sine waves.
- PC simulates the behavior of a PC when the user chooses to upload the session data to a hypothetical pc. The data to be sent to PC is stored in the Device class and include date and time of a treatment session and the initial and final dominant frequencies of waves produced at each EEG site during that session. To simulate the PC's monitor, the main window is divided into two sections where the left half belongs to the device screen and the right half replicates the PC screen. After uploading a session data to the PC, the user can save the data on the hypothetical PC in the json format.
- Battery is the eponymous class designed to hold and retrieve the battery level of the device. During a session, the battery is discharged. If the battery level drops below 20%, a warning message is shown to the user. If the battery level falls below 5%, the device is turned off.
- defs.h is not a class but a header file to store the constants, enums, and structs used in the program.

As MVC is a composite design pattern, the program also implements other design patterns including Strategy and Observer.

Strategy Design Pattern

The view and controller implement the classic Strategy Pattern. Here, the MainWindow and TestUI classes provide the strategy while the client, i.e. the view, is only concerned with the visual representation of the program. In this pattern, the controller is responsible for the decision making process required when the user interacts with the GUI. With this setup, the Strategy pattern allows the Neureset application to offer different operating modes to user through its GUIs. Users can utilize GUI components to control how the device operates, while the underlying logic remains decoupled from the user interface and can vary independently. This promotes flexibility, extensibility, and maintainability of the design.

Observer Design Pattern

In the Neureset application, the subject is the model which undergoes changes during a treatment session. During a treatment session, a considerable amount of data is produced by the EEG sites. When the waveforms are received by the device during each round of treatment, the device notifies the TestUI so it can provide the user with the option to plot the newly received waveform. In addition, the device notifies the other observer, i.e. MainWindow, of changes in the EEG sites connection status and the current phase of the treatment session. By employing this pattern, the model remains entirely decoupled from the views and controllers. This flexibility enables us to employ various views with the same model. For example, we can switch between views with and without the PC functionality by employing different UIs.

Design Decisions

Multi-threading

One of the extensions of use case 1 of the design document requires the application to pause the treatment session when the pause button is pressed by the user. Doing so involves running the device in a separate thread. Otherwise, when a treatment session starts, the GUI becomes irresponsive until the end of session making the pause and stop button irrelevant.
In addition to the device, the instance of SessionManager needs to run in its own thread to make the GUIs accessible when a treatment session is in progress. Since various activities, including retrieval and analysis of waveforms should take a few seconds to perform, QThread::msleep() is used to add a delay wherever necessary.

User Interface Design

To meet the requirements specified in the project documents and also to simulate a real handheld device more closely, the following design decisions regarding the user interface were implemented.

- The main UI is divided into two sections, where the left section acts as the device display and buttons and the right section as a PC where the session log is being uploaded to.
- As per professor instructions TestUI will be used only in the development phase of the application, therefore, it was decided to add it as a separate window.
- An ON/OFF button is added to simulate the process of turning the device off and on. After the application is launched, the device should be turned on to perform any task.
- Low battery warnings have been added to give the device a more authentic experience. If the battery level drops below 20%, a warning message is shown to the user. If the battery level falls below 5%, the device is turned off.