Dual-Alarm Clock Radio

AJ, William, and Luke (DaLAW)

Our Vision (*other than get an A)*

- Create software to model the logic of a dual-alarm FM/AM Radio Clock.

- Learn the process of Software Development while completing this task.

The Process: Use Cases

UC 1:   Using Alarms

Pre-Conditions: *The clock time is correctly set*

Success Guarantee: *The alarm goes off at the specified time*

UC 2: Listening to the radio

Pre-Conditions: *Radio can recieve transmissions and respond to software*

Success Guarantee: *The station is playing and user controlled*
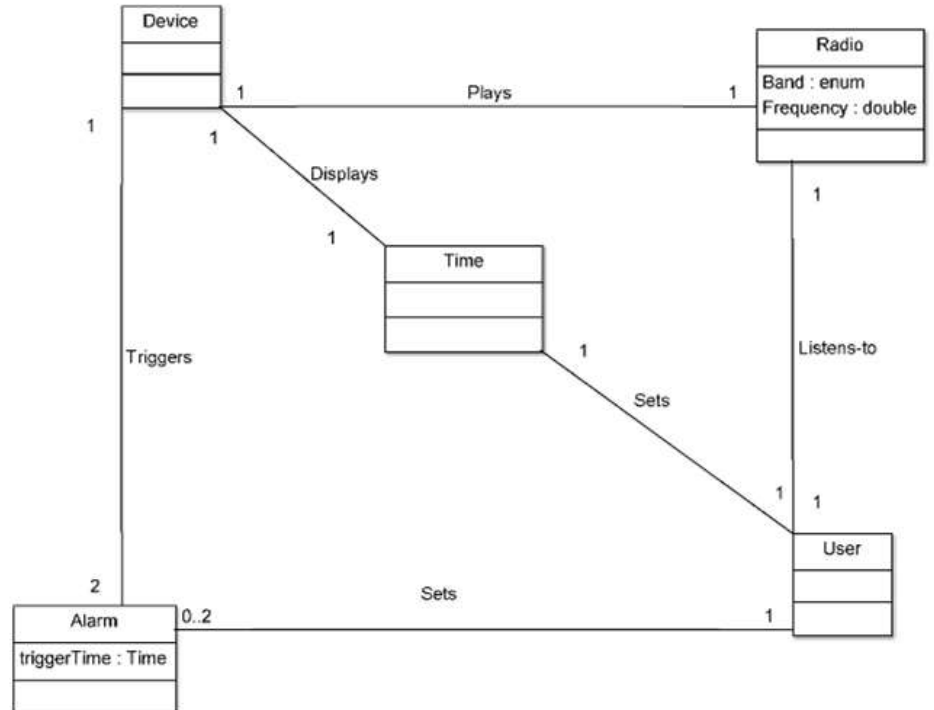
UC 3 : Setting Clock Time

Pre-Conditions: *There is already some time set*

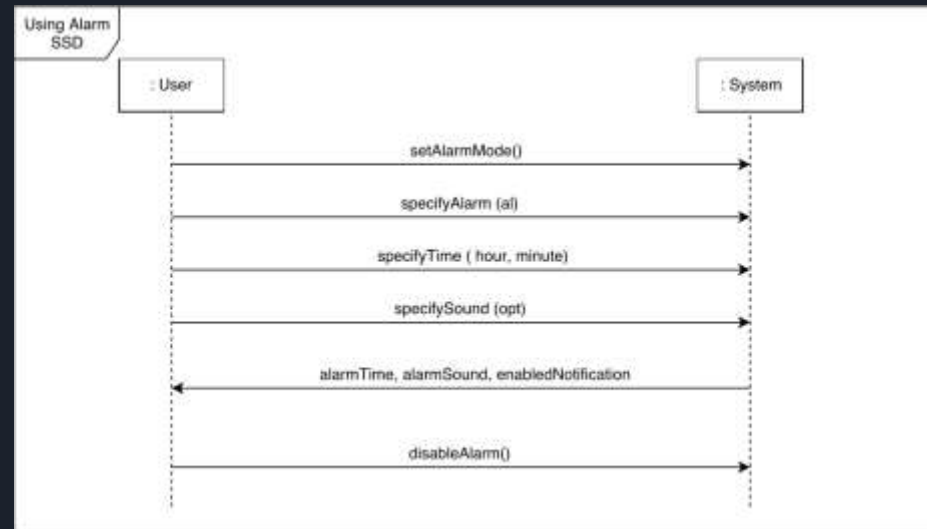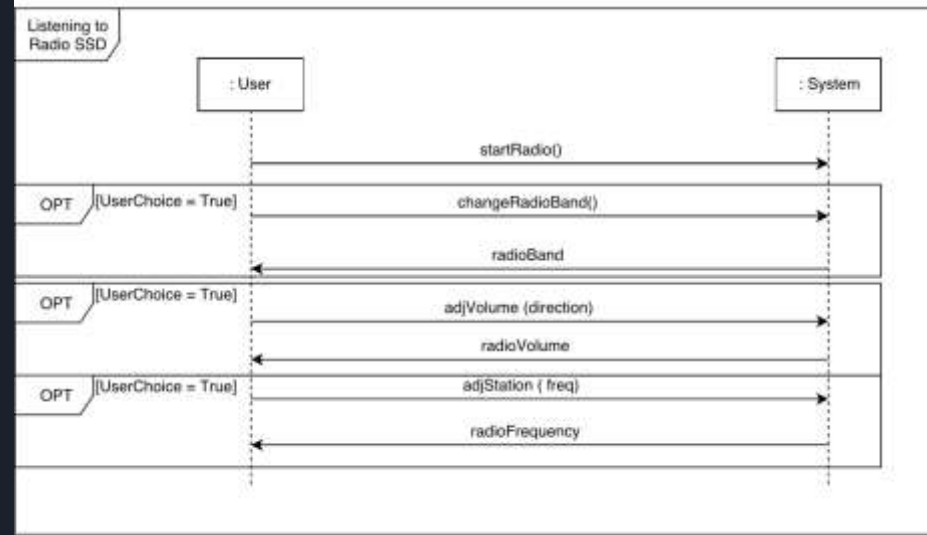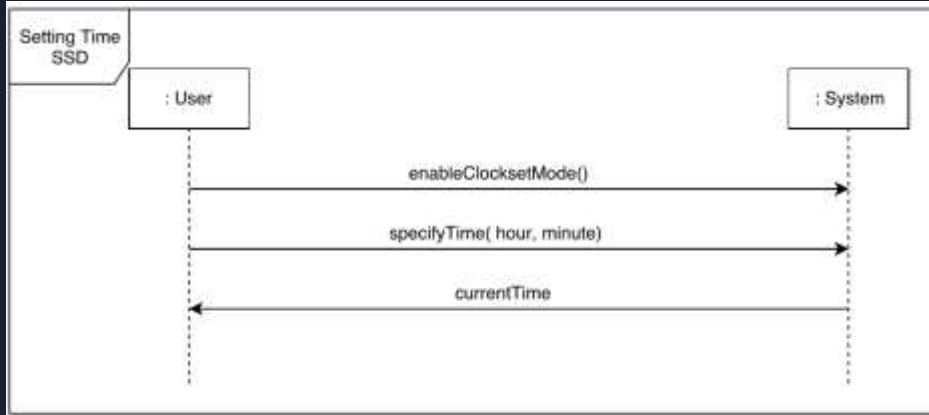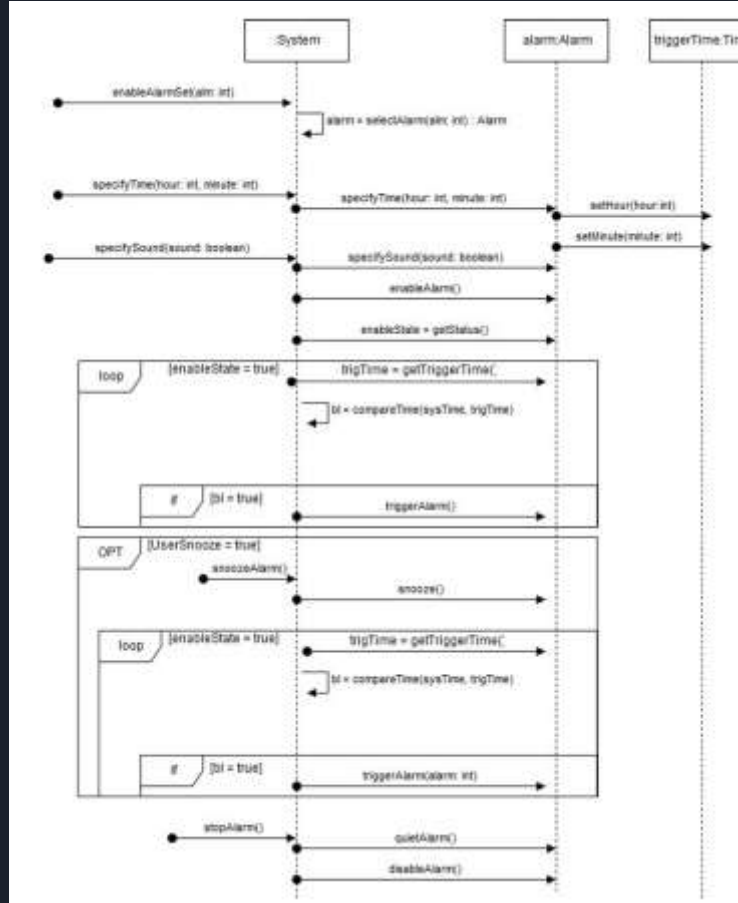Success Guarantee *The clock time is changed*

The Process: Modeling…

Domain Model

Sequence Diagram

Listening to Radio SSD

: User   : System

startRadio()

OPT [UserChoice = True]
changeRadioBand()
radioBand

OPT [UserChoice = True]
adjVolume (direction)
radioVolume

OPT [UserChoice = True]
adjStation (freq)
radioFrequency

Setting Time SSD

: User   : System

enableClocksetMode()

specifyTime(hour, minute)

currentTime

Using Alarm SSD

: User   : System

setAlarmMode()

specifyAlarm (al)

specifyTime (hour, minute)

specifySound (opt)

alarmTime, alarmSound, enabledNotification

disableAlarm()
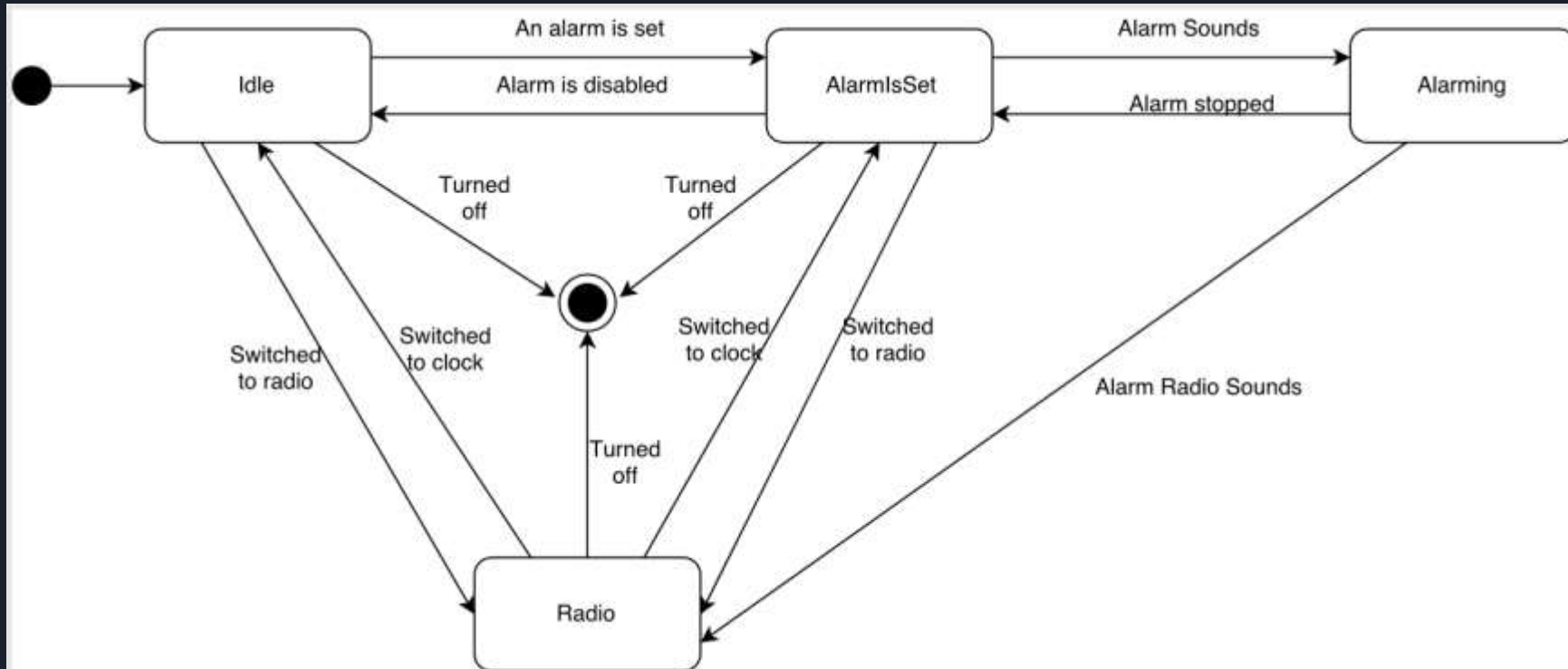
Use Case Realization: Dual-Alarm
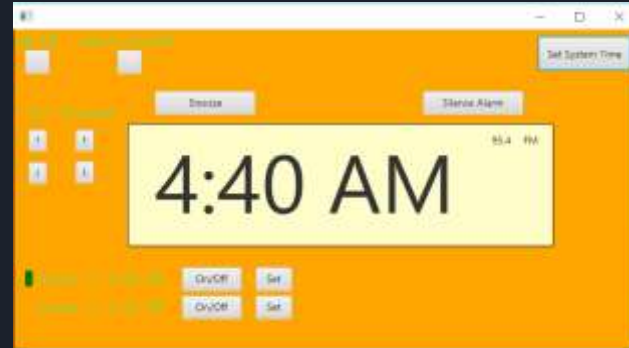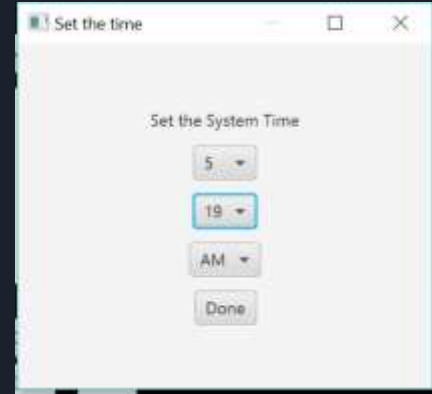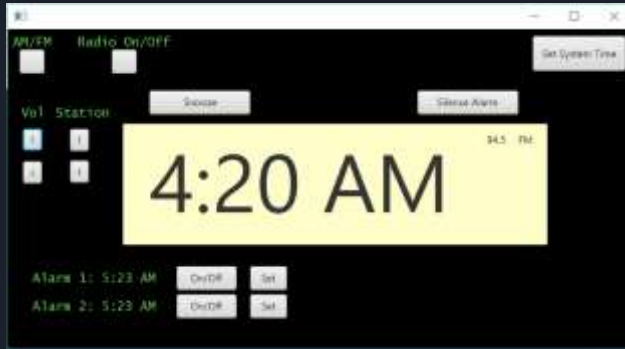
UI State Diagram

The Process: Implementation

- Coded in Java in the Netbeans IDE, using JavaFX for UI through Scenebuilder.

- GitHub for collaboration and code storage

- GroupMe for team communication

The Process: Presenting Implementation

Post-Process: Reflection on Difficulties

- Prior planning was inadequate for actual implementation

- Working with unique classes and their behaviours and limitations

Post Process: Lessons Learned

- Models are helpful when transitioning from ideas to code

- Models should not take up a large amount of time