DUAL-ALARM AM/FM CLOCK RADIO

Team W.O.R.K

Matthew Kay, Omer Omer, Ricky Ramos, Colton Williams

TABLE OF CONTENTS

Vision

The goal of this project is to create the software that controls a Dual Alarm AM/FM Clock Radio by following the steps outlined in the Unified Process. The software will be written in Java using the NetBeans IDE. Various diagrams like domain models and sequence diagrams will be designed in order to guide the implementation. The software will utilize a 12-hour time scheme and will show the time in the following format: HH:MM:SS AM/PM. The software must support the use of two alarms that can be snoozed and set to a desired time. The clock radio must also be able to switch between AM and FM and connect to specific radio stations. The FM radio setting will support stations from 88.1 - 107.3. The AM radio setting will support stations from 550 - 1480. A user-friendly, intuitive graphical user interface will also be implemented.

Use Cases

Use Case UC1: Setting the Alarm

Primary Actor	Clock Owner
Goal in Context	Setting the alarm so it goes off at the desired time
Scope	Dual Alarm Clock Radio Software
Level	User Goal
Stakeholders and Interests	Clock owner: wants an alarm to go off at the specified time
Precondition	Clock is on and is able to accept hours, minutes, and AM/PM settings for the alarms. Also, the software needs to distinguish between setting the alarm and setting the current time.The software need to know if the alarm is on or off. The software needs to make sure it goes off and makes a noise at the desired time.
Success Guarantees	The clock takes in the desired hours, minutes, and AM/PM settings and saves them in memory for alarm 1 or 2. The alarm will sound at the correct time as long as the user has not disabled the alarm.
Main Success Scenario	The clock owner enters the alarm time into the alarm, and the software saves the time in memory. The software saves the desired time of 8:40 AM to alarm 1. The alarm goes off at 8:40 AM, and the clock owner wakes up at the correct time and is not late for work.
Extensions	Extension 1: The clock owner wants to cancel an alarm that is currently in the process of being set. The user will just disable the alarm.
Special Requirements	The software must recognize HH:MM AM/PM format for displaying the time.
Open Issues	

Use Case UC2: Switch Between AM/FM Radio

Primary Actor	Clock Owner
Goal in Context	User is able to switch between AM or FM radio stations
Scope	Dual Alarm Clock Radio Software
Level	User Goal
Stakeholders and Interests	Clock Owner: Wants the ability to listen to various stations on either AM or FM radio Guests: Want the ability to change the radio AM/FM setting
Precondition	Clock is on and the software is able to recognize whether it is set to AM or FM.
Success Guarantees	User is able to change the radio to either AM or FM radio. The user is able to switch back and forth between AM or FM radio at any given time.
Main Success Scenario	The user turns on the radio. The radio is currently set to AM and is on a frequency with a static sound. The user switches the radio to FM radio.
Extensions	Extension 1: The user turns on the radio, and it is currently set to AM radio. The user attempts to switch to FM radio, but fails to do so.
Special Requirements	The software must be able to recognize and receive input from stations.
Open Issues	Does the software support foreign radio stations?

Use Case UC3: Setting the Time

Primary Actor	Clock Owner
Goal in Context	Setting the time on screen
Scope	Dual Alarm Clock Radio Software
Level	User Goal
Stakeholders and Interests	Clock Owner: wants to set the clock to the proper time
Precondition	Clock is connected to the power source and has functioning buttons. Software must recognize HH:MM:SS AM/PM format.
Success Guarantees	The clock takes in the desired hours, minutes, seconds, and AM/PM settings and saves them in memory.
Main Success Scenario	The clock owner connects the clock to a power source, and it turns on. The display time is flashing at 12:00 AM. The user looks at the time on a phone and adjusts the clock to the appropriate time by operating the respective buttons.
Extensions	Extension 1: The clock owner attempts to change the time after moving to a city with a different time zone. The owner enters the appropriate settings in HH:MM:SS AM/PM format. The user presses the “Set” button, and the correct time is now displayed.
Special Requirements	The software must recognize HH:MM:SS AM/PM format for displaying the time.
Open Issues	

Use Case UC4: Snoozing the Alarm

Primary Actor	Clock Owner
Goal in Context	Snoozing the alarm for 9 minutes
Scope	Dual Alarm Clock Radio Software
Level	User Goal
Stakeholders and Interests	Clock Owner: wants to set to snooze the alarm for 9 minutes before going off again
Precondition	Clock is connected to the power source and at least one of the alarms has been set.
Success Guarantees	The alarm will stop sounding and then go off again after 9 minutes if the snooze button is pressed.
Main Success Scenario	The clock owner connects the clock to a power source and sets an alarm for the morning. The alarm goes off at the correct time, and the user presses the snooze button. The alarm successfully goes off again after 9 minutes.
Extensions	Extension 1: The clock owner connects the clock to a power source and sets an alarm for the morning. The snooze button is pressed and the alarm stops sounding, but the alarm fails to go off again 9 minutes later. Extension 2: The clock owner connects the clock to a power source and sets an alarm for the morning. The snooze button is pressed, but the alarm fails to stop sounding.
Special Requirements	
Open Issues	

Use Case UC5: Adjust the Volume

Primary Actor	Clock Owner
Goal in Context	Control the volume of the radio
Scope	Dual Alarm Clock Radio Software
Level	User Goal
Stakeholders and Interests	Clock Owner: wants to adjust the radio to a comfortable volume Guests: want to adjust the radio to a comfortable volume
Precondition	Clock is connected to the power source and software supports volume control.
Success Guarantees	The volume of the radio will increase or decrease as adjusted by the user.
Main Success Scenario	The clock owner connects the clock to a power source and turns on the radio. The radio is initially too loud, so the user operates the volume knob to get to a comfortable volume
Extensions	Extension 1: The clock owner connects the clock to a power source and turns on the radio. The volume is too high, so the user attempts to adjust the volume knob. The radio stays at the same volume.
Special Requirements	
Open Issues	

Use Case UC6: Tune the Radio

Primary Actor	Clock Owner
Goal in Context	User is able to tune the radio on either AM or FM radio stations
Scope	Dual Alarm Clock Radio Software
Level	User Goal
Stakeholders and Interests	Clock Owner: Wants the ability to listen to various stations on either AM or FM radio Radio Stations: Want to know how many people listen to their stations Guests: Want the ability to tune the radio
Precondition	Clock is on and the software is able to recognize whether it is set to AM or FM and change the station.
Success Guarantees	User is able to tune the radio to any station within the frequency boundaries on AM/FM .
Main Success Scenario	The user turns on the radio. The radio is currently set to AM and is on a frequency with no music playing. The user switches the radio to FM radio and tunes to a station playing music.
Extensions	Extension 1: The user turns on the radio, and it is currently set to AM radio. The user attempts to switch to FM radio. The user begins to tune the radio, but it stays on the same station.
Special Requirements	The software must be able to recognize and receive input from antenna. The software must be able to recognize when the tuning knob is being adjusted
Open Issues	

Use Case UC7: Turn the Radio ON/OFF

Primary Actor	Clock Owner
Goal in Context	User is able to turn the radio on or off
Scope	Dual Alarm Clock Radio Software
Level	User Goal
Stakeholders and Interests	Clock Owner: Wants the ability to listen to turn the radio on or off
Precondition	The clock's software must recognize when the "Power" button is pressed by the user
Success Guarantees	The user is able to turn the radio on or off at any given moment.
Main Success Scenario	The user turns on the radio. The radio is currently set to AM and is only outputting static sound. The user decides to turn the radio off by pressing the "Power" button.
Extensions	Extension 1: The user turns on the radio, and it is currently set to AM radio. The user attempts to switch to FM radio. After listening for an hour, the user decides to turn the radio off.

Supplementary Specification

- **Functionality:**
    - **User will have the ability to:**
        - Set either of the two alarms
        - Switch between AM/FM radio stations
        - Set the time
        - Snooze the alarm(s)
        - Adjust the volume
        - Tune the radio
        - Turn the Radio On/Off

- **Usability:**
    - **Human Factors:**
        - The user should have the basic ability to read the buttons.
        - Buttons:
            - Alarm 1 On/Off
            - Alarm 2 On/Off
            - Alarm 1 Time Set
            - Alarm 2 Time Set
            - Radio Station Set
            - Radio AM Set
            - Radio PM Set
            - Clock Set

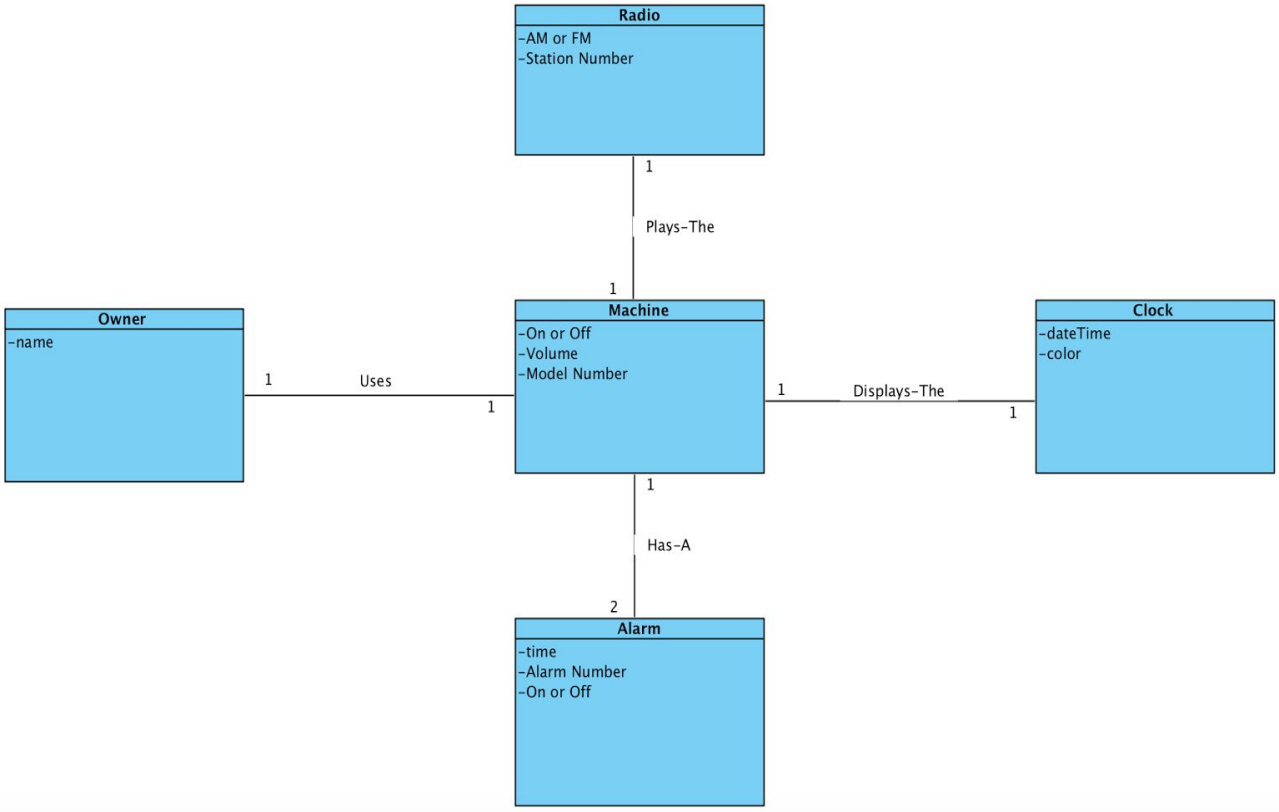
      The user will be able to see the the time on a large display. Therefore:
        - The time should be easily read from more than a few meters away.
        - The contrast between the different backgrounds and text should be sufficient enough to clearly make out the text.

- **Reliability:**
    - The alarm(s) should not fail to go off at the correct time.

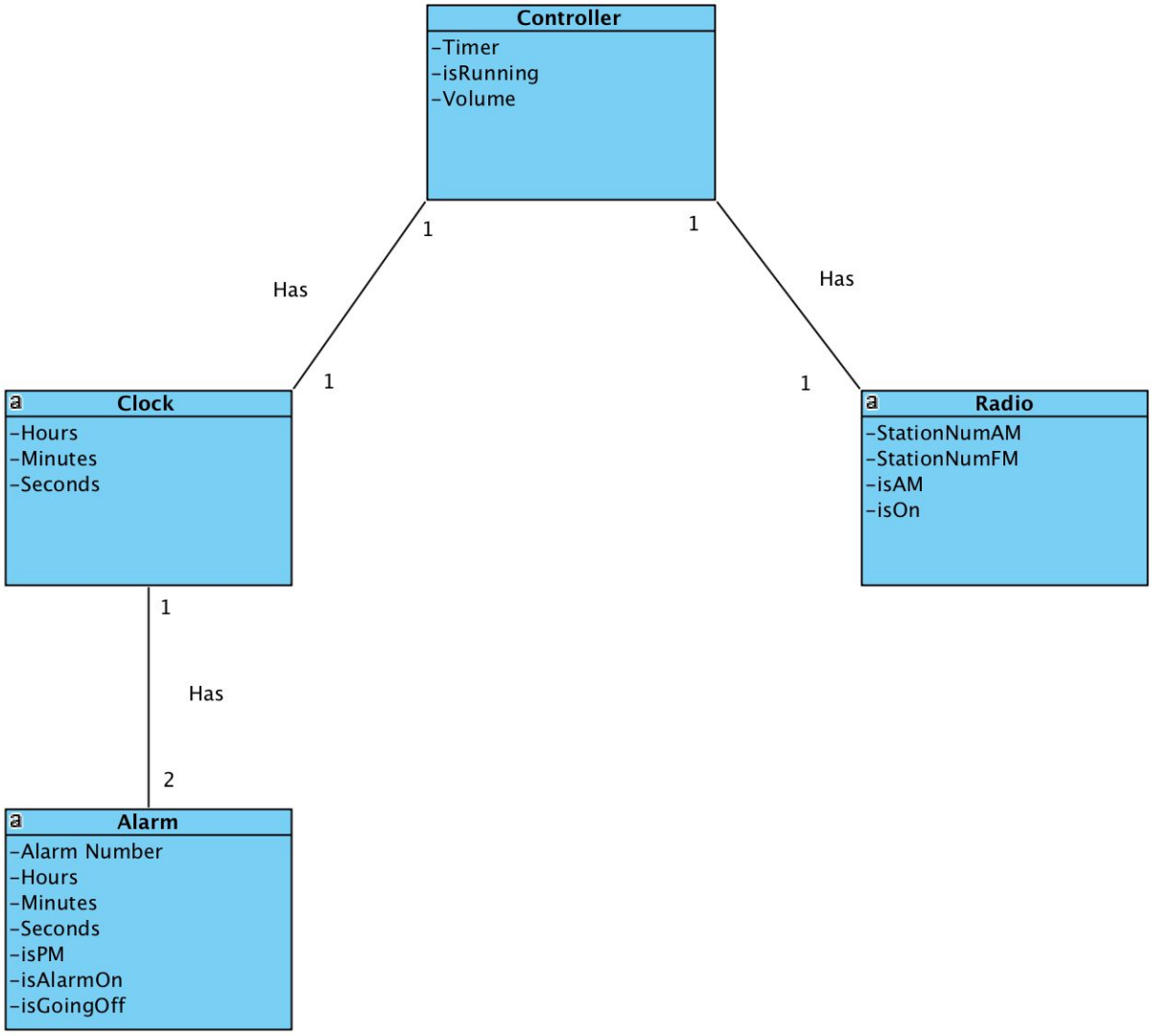
- **Performance:**
    - The radio should be able to tune to the desired station and start outputting music.
    - The clock should respond immediately to the snoozing/disabling of the alarm.

- **Supportability:**
    - The software will not need to be updated.
    - **Configurability:**
        - The alarms may be set to any time the user desires.
        - The clock only follows a 12-hour scheme.

Domain Model (Original)

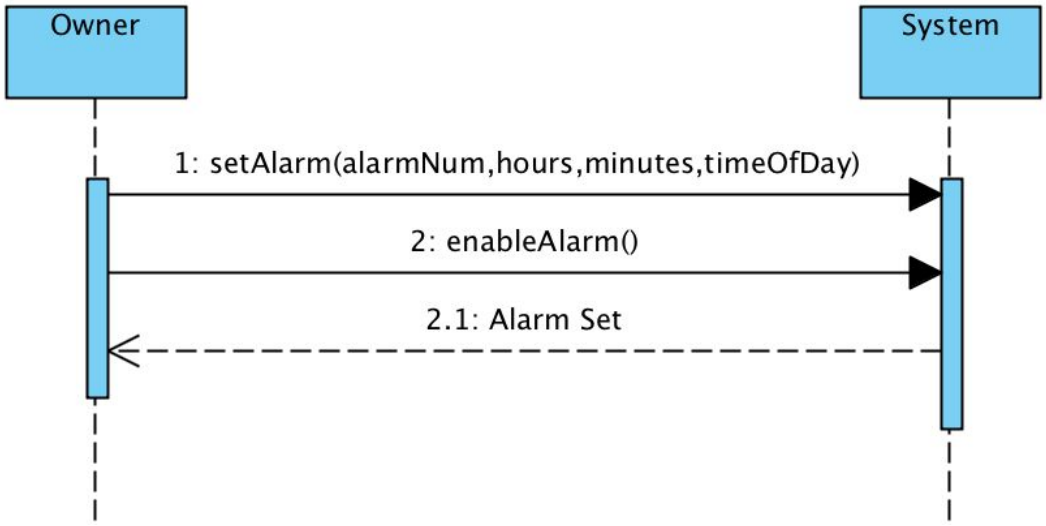


Domain Model (Updated)



System Sequence Diagrams and Operation Contracts

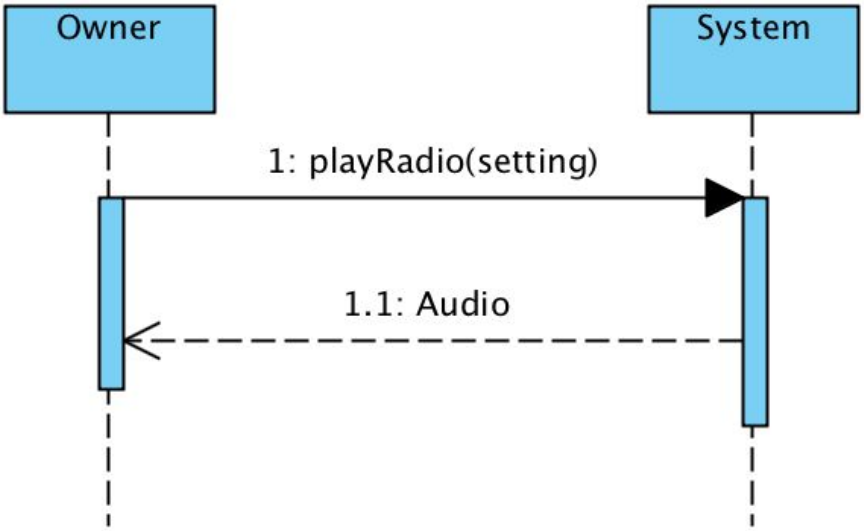
Enable Alarm:



Contract CO1: enableAlarm

Operation	enableAlarm()
Cross References	Use Cases: Set Alarm
Preconditions	Machine must be on, and the hours, minutes, and time of day have been inputted.
Postconditions	The machine was on and an alarm time was previously selected. The alarm went off at the correct time.

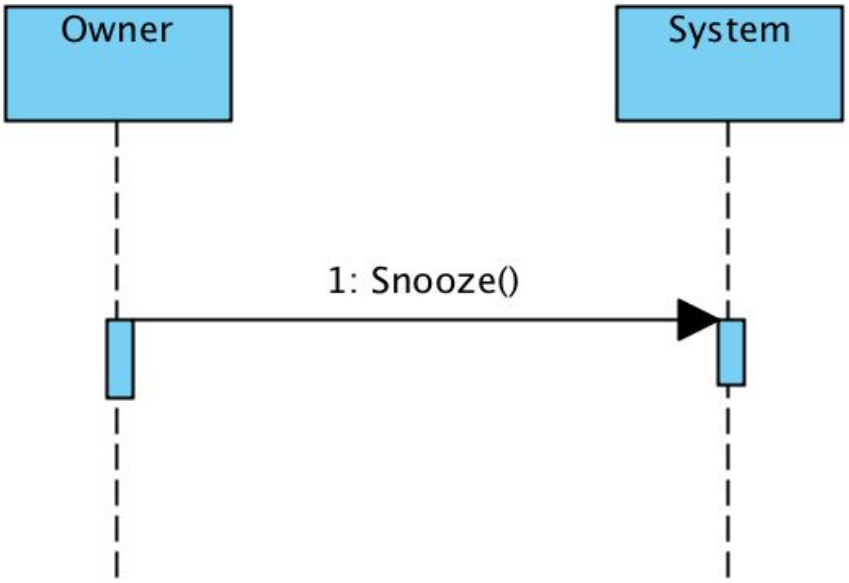
Switch Between AM/FM Radio:



Contract CO2: playRadio

Operation	playRadio(setting: int)
Cross References	Use Cases: Switch Between AM/FM Radio
Preconditions	Machine is on, and the user has decided which setting the radio will be on.
Postconditions	The machine was turned on and started playing music on AM radio. The radio was then set to FM radio.

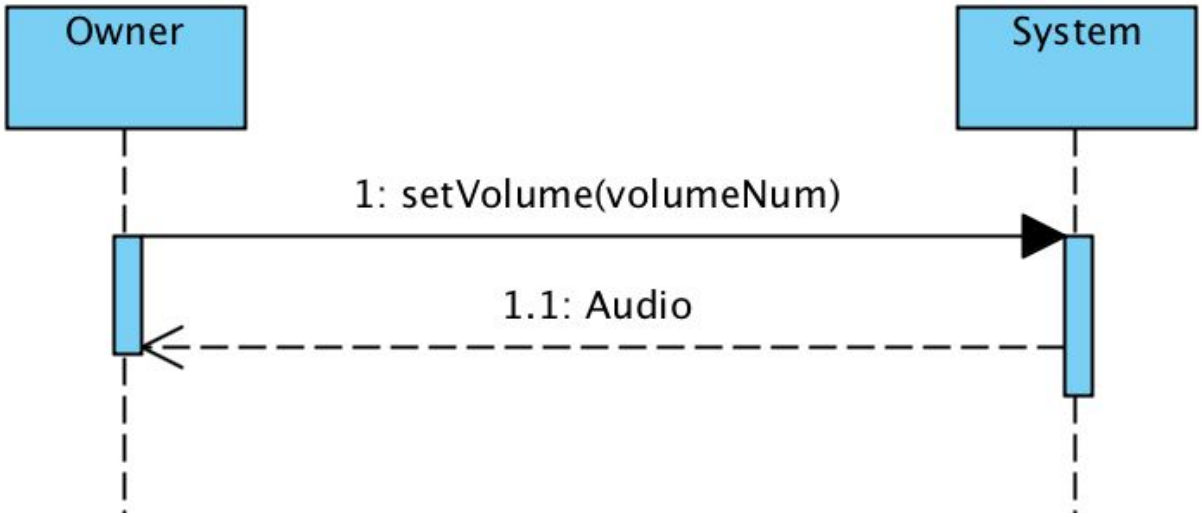
Snooze Alarm:



Contract CO3: snooze

Operation	snooze()
Cross References	Use Cases: Snooze Alarm
Preconditions	Machine is on, and the user has already set an appropriate alarm time.
Postconditions	Machine was on, and an alarm was previously set. The alarm went off at the correct time but was snoozed.

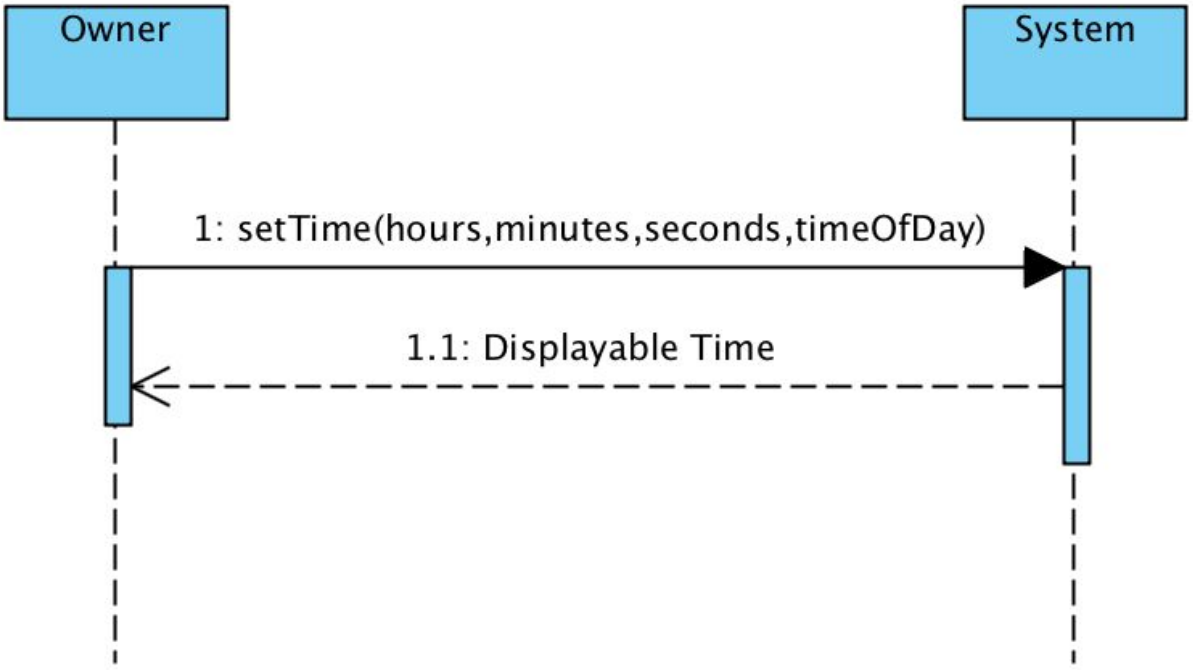
Set the Volume:



Contract CO4: setVolume

Operation	setVolume(volumeNum: int)
Cross References	Use Cases: Changing the Volume
Preconditions	The machine is on, and audio is playing.
Postconditions	The machine was on and playing music. The volume was too loud, so it was adjusted to a comfortable setting.

Set the Time:

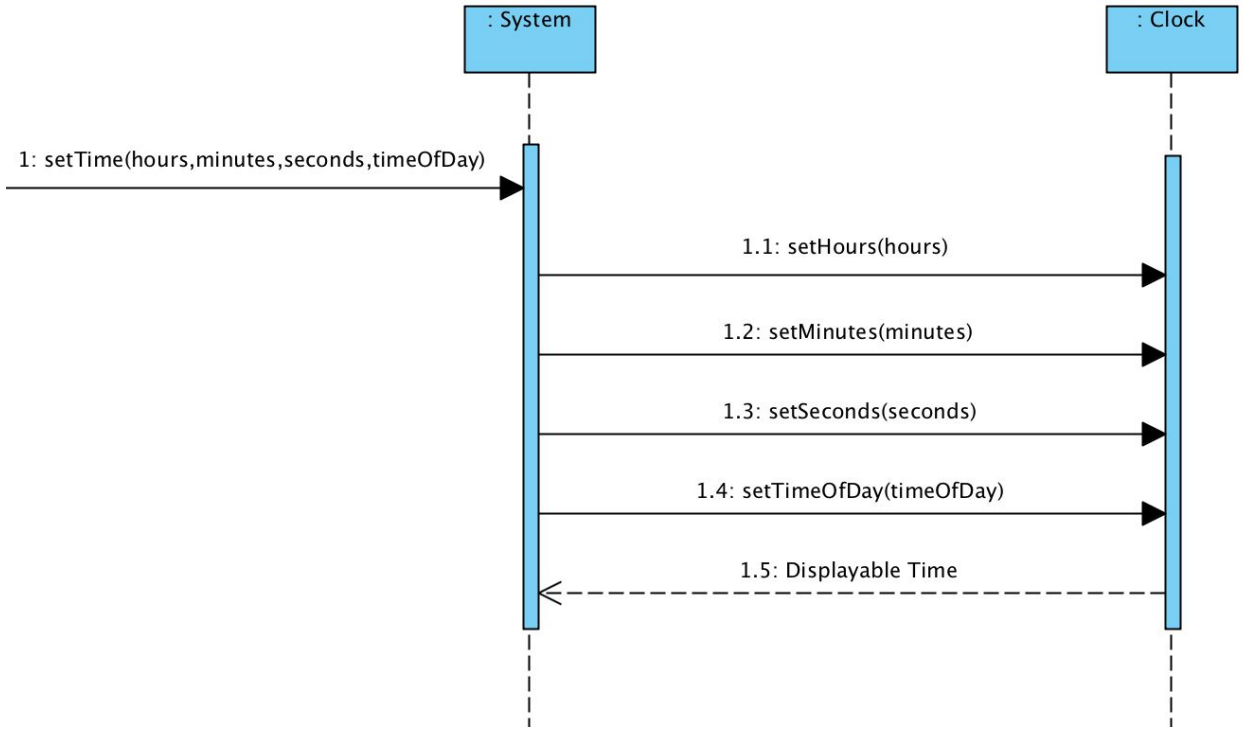


Contract CO5: setTime

Operation	setTime(hours: int, minutes: int, seconds: int timeOfDay: String)
Cross References	Use Cases: Setting the Time
Preconditions	The machine is on, and the user has decided to adjust the time.
Postconditions	The machine was on, and the time was incorrect. The clock was adjusted, and the correct time was displayed.

Object Sequence Diagrams with Operation Contracts

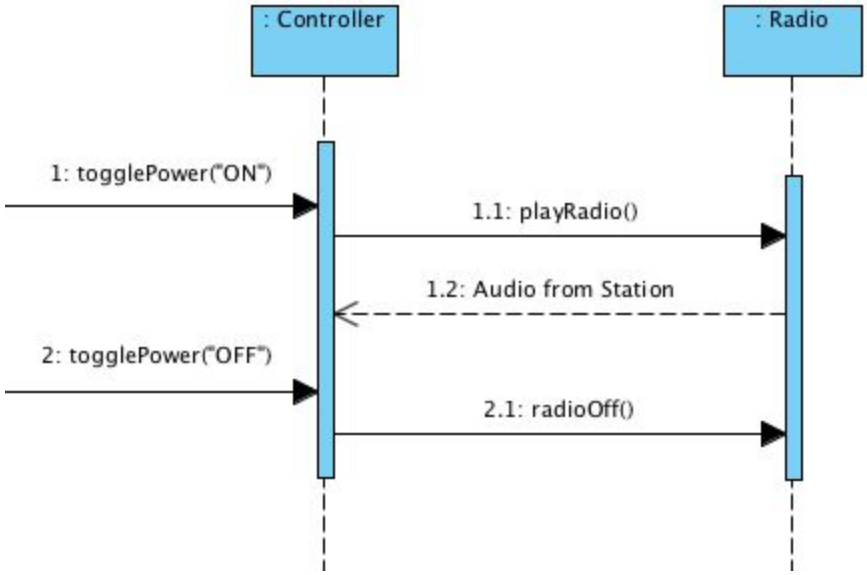
Set the Time:



Contract CO1: setTime

Operation	setTime(hours: int, minutes: int, seconds: int, timeOfDay: String)
Cross References	Use Cases: Setting the Time
Preconditions	The controller was requested to set the time.
Postconditions	-A new instance of Clock was created. -The Clock's hours attribute was set to desired setting. -The Clock's minutes attribute is set to desired setting. -The Clock's seconds attribute is set to desired setting. -The Clock's timeOfDay attribute was set to desired time of day.

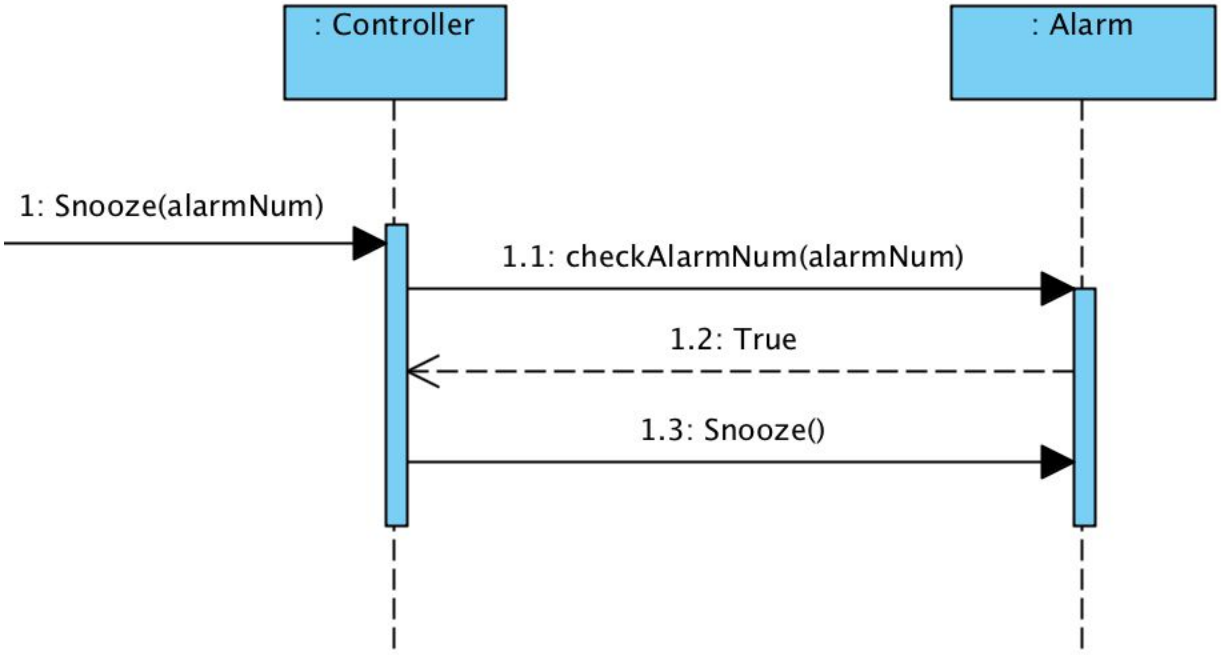
Turn the Radio ON/OFF:



Contract CO2: turnRadioON/OFF

Operation	togglePower(power: String)
Cross References	Use Cases: Turn the Radio ON/OFF
Preconditions	The controller is being requested to turn the radio on.
Postconditions	-A new instance of Radio was created. -The power attribute was changed to desired setting. -The method playRadio() was called. -Audio was sent from the Radio object to the controller.

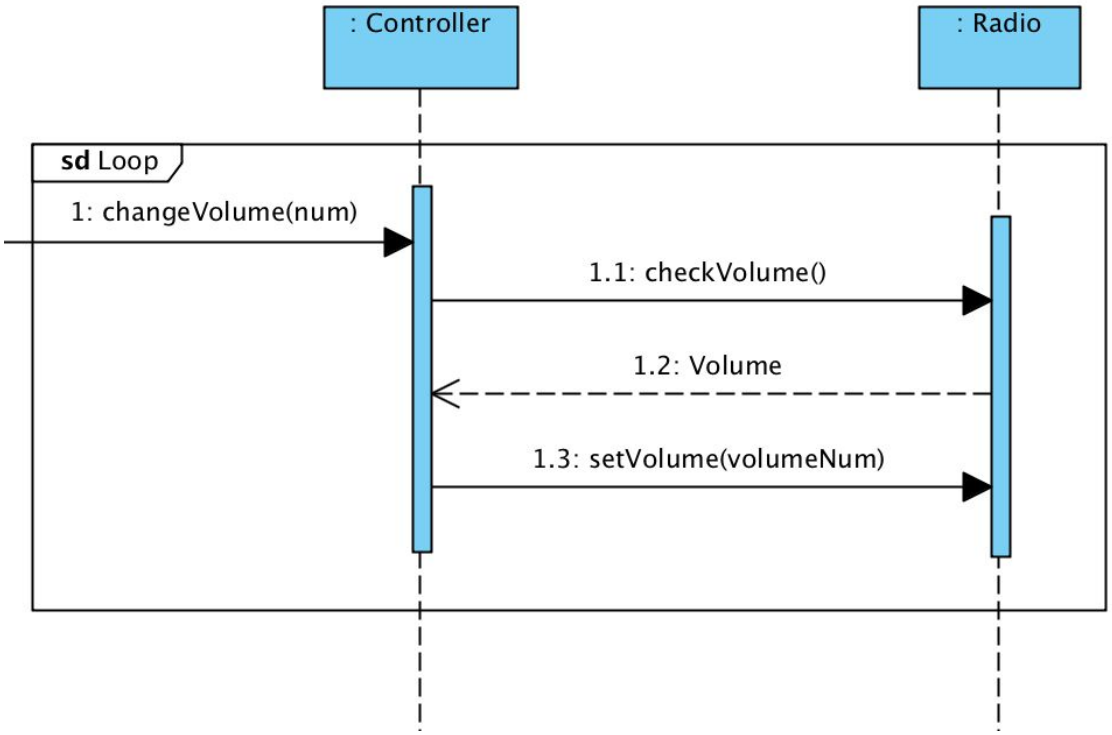
Snooze the Alarm:



Contract CO3: snoozeAlarm

Operation	snooze(alarmNum: int)
Cross References	Use Cases: Snoozing the Alarm
Preconditions	One or both alarms are going off, and the controller has been requested to snooze the alarm(s).
Postconditions	-The checkAlarm() method was called on both instances of Alarm. -The boolean value of the isGoingOff attribute in both instances was returned to the controller. -The method snoozeAlarm() in the correct Alarm was called. -The correct alarm(s) were snoozed.

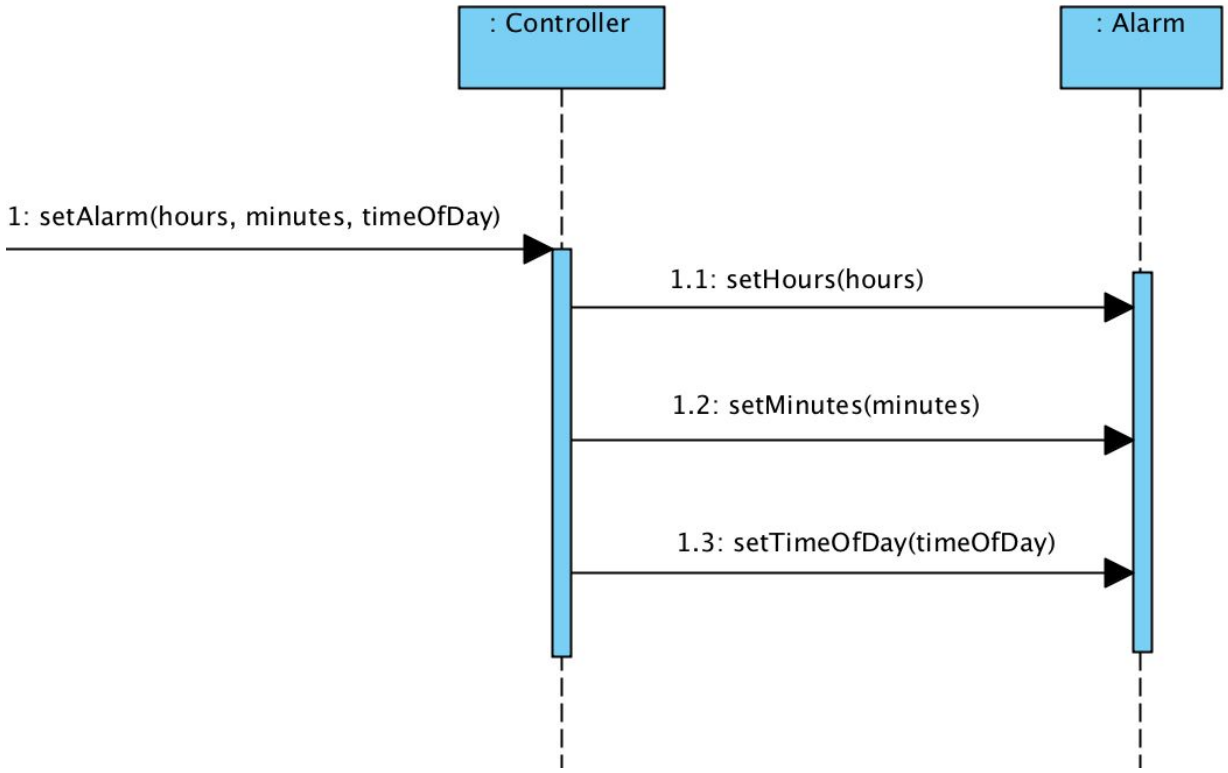
Adjust the Volume:



Contract CO4: adjustVolume

Operation	changeVolume(num: int)
Cross References	Use Cases: Adjust the Volume
Preconditions	The machine is on, and the controller has been requested to adjust the volume.
Postconditions	-A method checkVolume() was called on the controller. -An instance of Radio was previously created. -The Radio attribute volume was returned to the controller. -A method setVolume(num: int) was called with that attribute number. -The volume attribute of the Radio object was changed again to the new number.

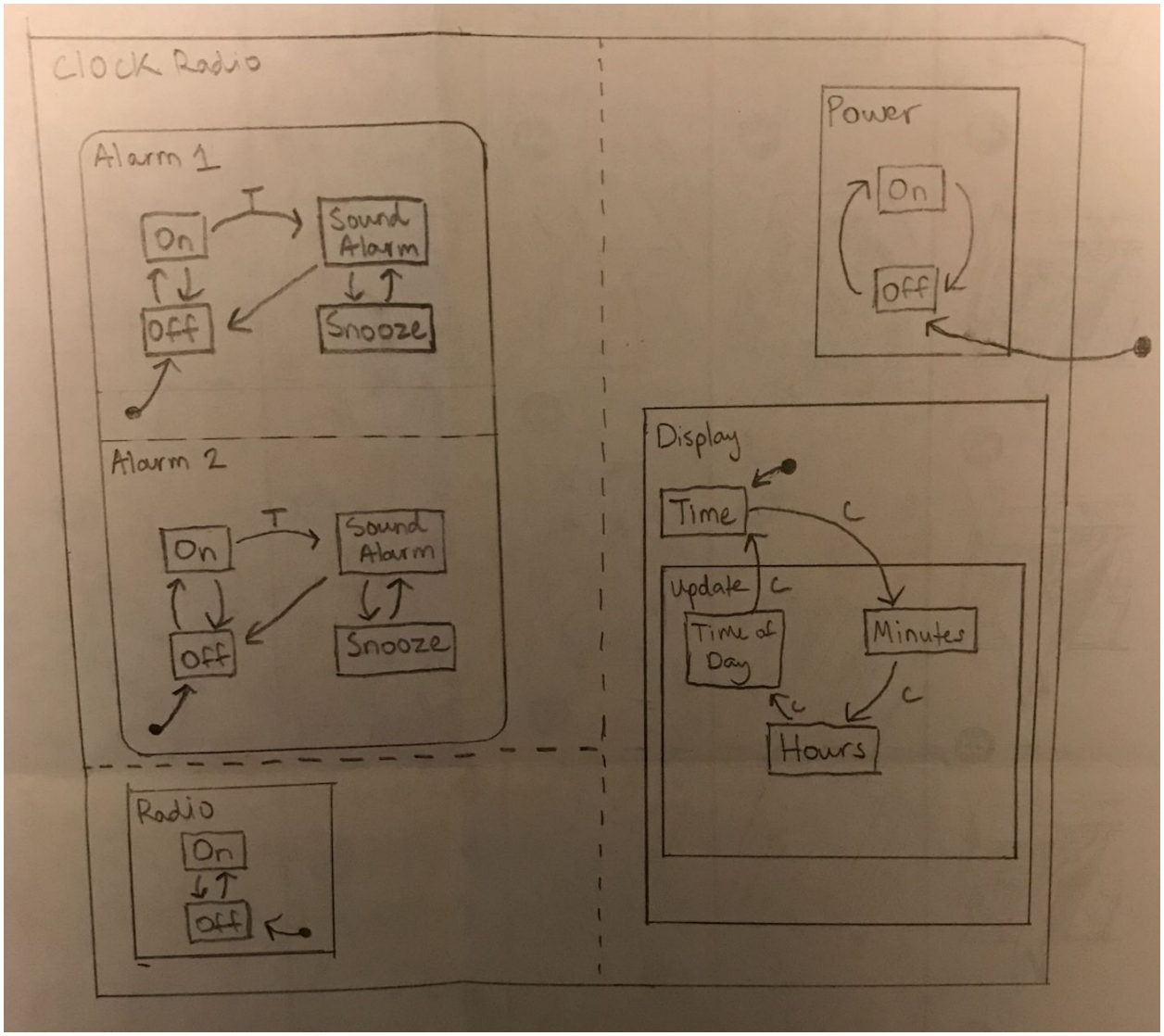
Set Alarm:



Contract CO5: setAlarm

Operation	setAlarm(hours: int, minutes: int, timeOfDay: String)
Cross References	Use Cases: Set the Alarm
Preconditions	The machine is on, and the controller has been requested to set either alarm one or alarm two.
Postconditions	-Two instances of Alarm were created. -The Alarm hours, minutes, and timeOfDay attributes of one Alarm instance were changed to the specific parameters passed in.

State Diagram



Glossary

- **AM/FM:** Amplitude Modulation and Frequency Modulation, respectively. These are forms of modulation in which either the amplitude(strength) or frequency of a carrier wave is varied. The clock radio can be used with either setting.
- **Cross References:** The use cases to which an operation contracts applies.
- **Domain:** A formal boundary that defines a particular subject or area of interest.
- **Domain model:** Illustrates noteworthy concepts in a domain.
- **Dual Alarm AM/FM Clock Radio:** A clock that supports the setting of two alarms and has the capability of receiving and interpreting radio waves to output information to the user.
- **Extensions:** Alternate scenarios in use cases.
- **Graphical User Interface (GUI):** A visual way of interacting with a computer using items such as windows, icons, and menus, used by most modern operating systems.
- **Operation Contracts:** Identifies system state changes when an operation happens.
- **Postconditions:** What must be true on successful completion of the use case.
- **Preconditions:** What must always be true before a scenario is begun in a use case.
- **Sequence Diagram:** Shows the flow of messages between software objects.
- **Snooze:** Delaying the alarm for another 9 minutes.
- **Software:** The part of the computer system that consists of encoded information or computer instructions.
- **State Diagram:** An illustration of the states an object can attain as well as the transitions between those states in the Unified Modeling Language (UML).
- **Use Case:** A list of actions or event steps, typically defining the interactions between a role and a system, to achieve a goal.
- **Vision:** A short executive overview document for quickly learning the project's big ideas.