Group 9 Mountain Lion Detection System

Team Members: Alberto Ames, Carlos Paredes, and Bobby Tomlinson

System Description

The Group 9 Mountain Lion Detection System will help to detect mountain lions within San Diego County Parks in order to detect nearby mountain lions that may pose an immediate threat to park goers. This system will help to ensure the safety of park goers, ensure the safety of mountain lions, and allow for safe evacuation of humans and removal of mountain lions. The Group 9 Mountain Lion Detection System, monitored by park rangers and capable of sounding alarms, will significantly advance efforts to preserve and respect natural wildlife, and prevent danger to citizens of San Diego county.

Noise detection sensors, with power and connection to a central computer, will constantly analyze audio being detected to search for matches to user-programmed audio of target animal noises. Upon finding a match to initially programmed audio, the sensor will collect data regarding the strength of the noise (in dB), the predicted type of noise, the location of the noise (accurate within 3 meters), the time of the noise, and the audio of the noise itself. This data, along with the name of the sensor where the audio was detected, will be transmitted to the central computer in a park ranger station. Upon being received at the controlling computer, it will become an alert, requiring a park ranger to respond and assign the detection one of the 3 following classifications based on the sound: definite, suspected, and false. Any noise that constitutes an alert will have this previously mentioned data saved on a central computer for up to 30 days. After 30 days, the data for an alert will become included and saved within a summary of alert information on the central computer for up to a year, upon which it is removed from the saved summary.

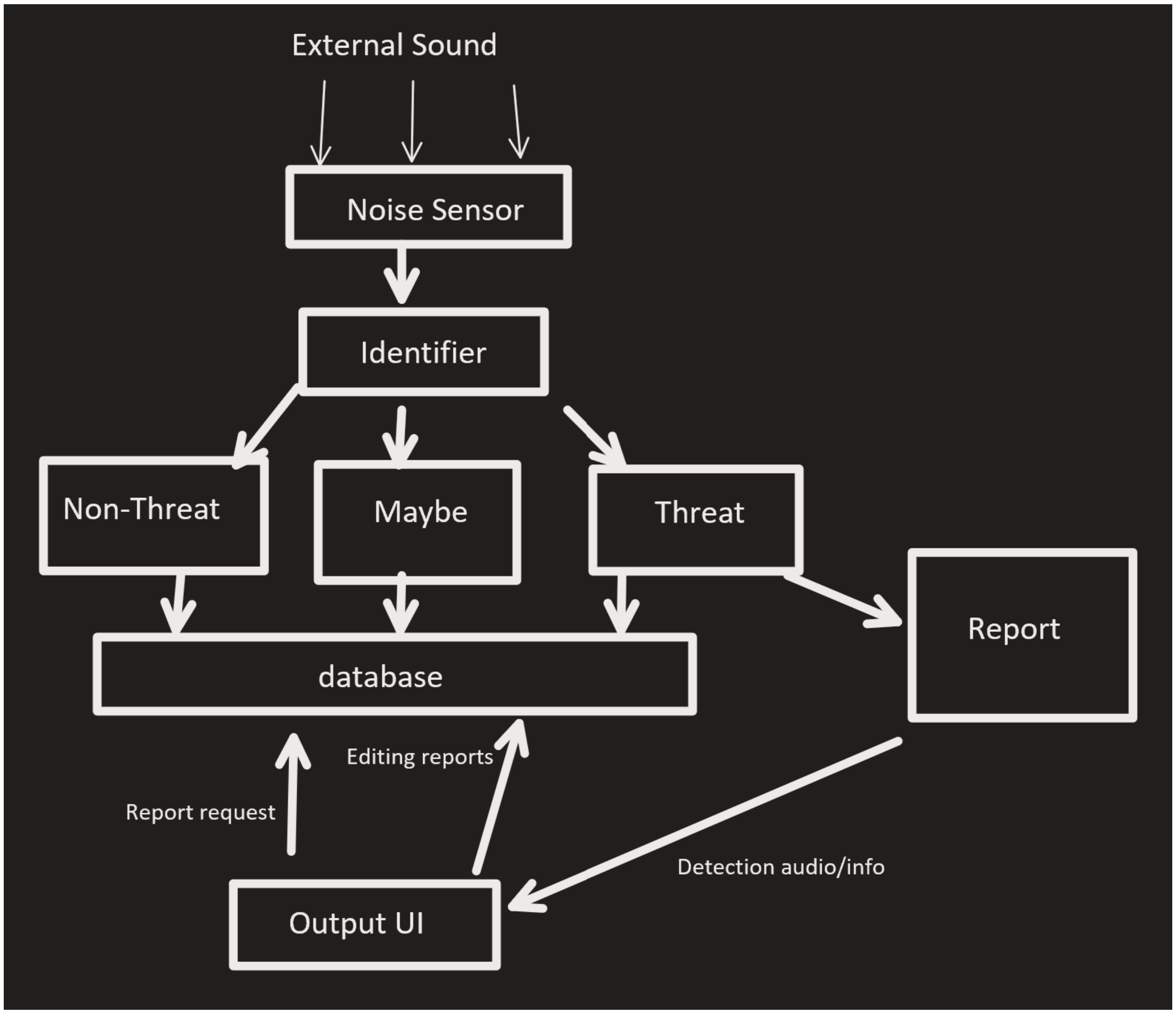
The central computer will be able to access saved data, and compile data into various possible reports upon request from the user. Possible reports include:

1. One that provides all mountain lion detections by date and by ranger classification
2. One that provides all mountain lion detections from a particular sensor
3. One that graphically displays mountain lion detections on a map
4. One that shows detection classifications from specific rangers

The Tomlinson Mountain Lion Detection System will be durable and able to withstand a range of duress within the park, including physical impacts, moisture, and precipitation. It will also be able to function in any temperature, and its hardware will be resistant to vandalism and theft. All aspects of the system save for the central computer must be installed inconspicuously. The detection sensors “in the field” must be installed so as to not attract attention or disturb the natural environment that they are in.

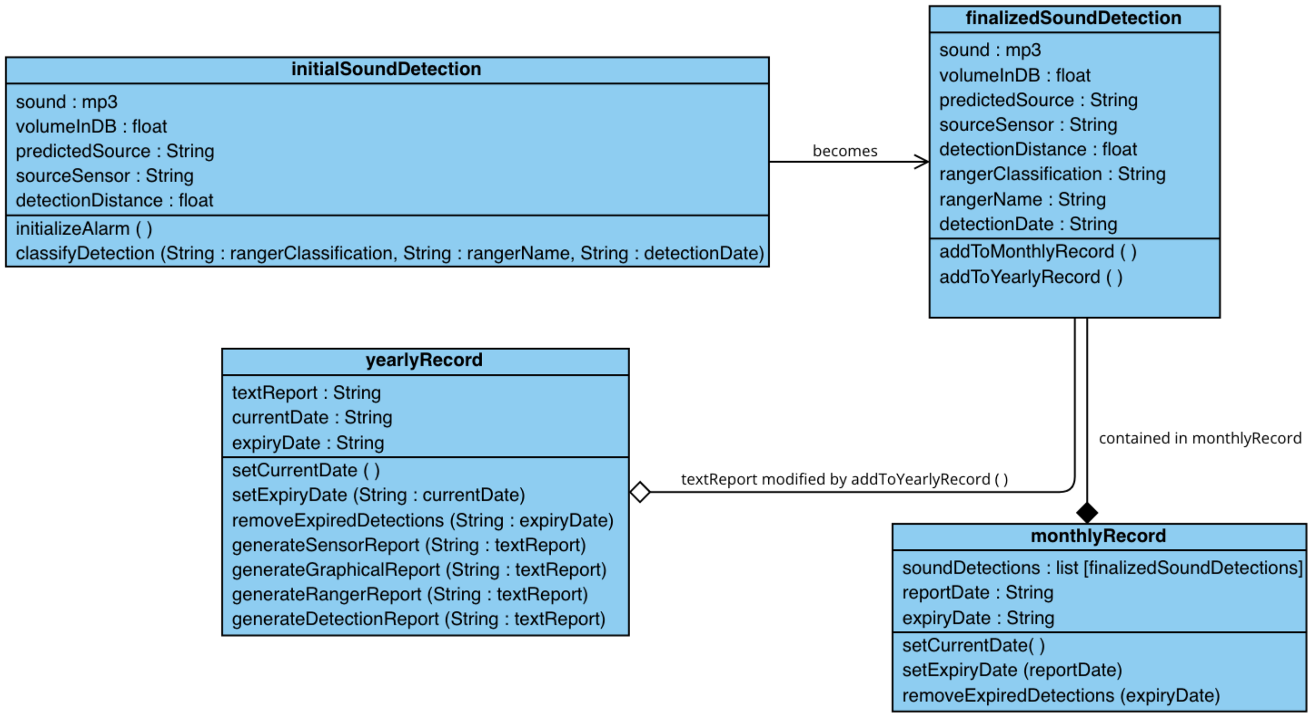
Software Architecture Overview

Architectural Diagram of All Major Components



The main components of the Mountain Lion Detection System consist of a noise sensor which takes in audio data, an identifier system which categorizes the sounds as threats, non-threats and maybe-threats, the database which holds reports of all audio detections, a report-generating system which sends reports to the user depending on the category of the sound, and a user interface which allows users to view, edit, and request reports. The user is able to hear audio if a report is generated, and edit the category it is classified under if they determine that the sound doesn't constitute a threat. A report is generated if the audio input is determined by the identifier to be a threat, otherwise the input is simply logged in the database. The user can request to view previous audio input data, and they will be able to edit it.

Group 9 MLDS UML Class Diagram


initialSoundDetection
sound : mp3
volumeInDB : float
predictedSource : String
sourceSensor : String
detectionDistance : float
initializeAlarm ( )
classifyDetection (String : rangerClassification, String : rangerName, String : detectionDate)
becomes
finalizedSoundDetection
sound : mp3
volumeInDB : float
predictedSource : String
sourceSensor : String
detectionDistance : float
rangerClassification : String
rangerName : String
detectionDate : String
addToMonthlyRecord ( )
addToYearlyRecord ( )
yearlyRecord
textReport : String
currentDate : String
expiryDate : String
setCurrentDate ( )
setExpiryDate (String : currentDate)
removeExpiredDetections (String : expiryDate)
generateSensorReport (String : textReport)
generateGraphicalReport (String : textReport)
generateRangerReport (String : textReport)
generateDetectionReport (String : textReport)
textReport modified by addToYearlyRecord ( )
contained in monthlyRecord
monthlyRecord
soundDetections : list [finalizedSoundDetections]
reportDate : String
expiryDate : String
setCurrentDate( )
setExpiryDate (reportDate)
removeExpiredDetections (expiryDate)

Classes

The Group 9 Mountain Lion Detection System has the following classes: initialSoundDetection, finalizedSoundDetection, yearlyRecord, and monthlyRecord. initialSoundDetection is an object that is instantiated by the sound sensor and is transmitted to the central computer as an alert. Its main purpose is to store data regarding the detection; however, within this class, the detection initializes the alert to park rangers, and the user uses the class's classifyDetection operation to determine the threat level of the detection. Once this is done, the initialSoundDetection becomes a new object, based on the class finalizedSoundDetection. This class, once again, stores data regarding the detection. Through its operations, finalizedSoundDetection also has the ability to add itself and remove itself from the computer's month-long record and its year-long record. The class monthlyRecord stores a list that contains finalizedSoundDetection objects. monthlyRecord also routinely uses the operations setCurrentDate and setExpiryDate to update its attributes reportDate and expiryDate.

monthlyRecord uses these attributes to call removeExpiredDetections and remove finalizedSoundDetections from its list. yearlyRecord, like monthlyRecord, has attributes for its expiry and current dates with corresponding routine operations to update these attributes. Additionally, it can generate a variety of reports, and contains a textReport attribute with a summary of detections that is updated as detections add themselves to the class. Thus, initialSoundDetection is an object that eventually becomes finalizedSoundDetection through user interactions. finalizedSoundDetections access the attributes of yearlyRecord, and are contained within monthlyRecord.

Attributes

The classes of the Group 9 Mountain Lion Detection System follow the principles of Object-Oriented Programming, and as such, are primarily intended as stores of data. Thus, it is important to note the purposes of each class's attributes. First, the initialSoundDetection class makes use of mp3 audio libraries in order to store the audio from the detection within the initialSoundDetection with a type of mp3. Next, the volume of the detection, measured in decibels, is stored within initialSoundDetection as volumeInDB of float type. Next, the predictedSource of the audio, based on user-programmed audio and a corresponding user-entered string, is stored within initialSoundDetection as predictedSource of string type. Next, the sensor that detected the sound of interest is assigned a user-entered name upon installation, and when a detection occurs, the name of the sensor is stored within initialSoundDetection as sourceSensor of string type. The sensor's predicted distance (in meters) from the sound of interest is stored within initialSoundDetection as detectionDistance of type float. When initialSoundDetection is instantiated, and its alert causes a park ranger to respond, initialSoundDetection becomes finalizedSoundDetection. finalizedSoundDetection stores all of the previously mentioned attributes of initialSoundDetection, with some additional attributes due to user interactions. Supplementally, finalizedSoundDetection stores the responding ranger's classification of the threat stored as rangerClassification, which is subjectively decided by the ranger based on the audio. This variable can be one of three strings: Definite, Suspected, or False. Additionally, finalizedSoundDetection stores the user-entered full name of the responding park ranger within the variable rangerName of type string. Furthermore, finalizedSoundDetection stores the date of the detection as the variable detectionDate. This data is carefully entered by the responding ranger according to a specific format upon which the functionalities of further operations are dependent, and this data is of string type. monthlyRecord contains the necessary libraries to allow for a dynamically-created list that stores instances of finalizedSoundDetection of object type "finalizedSoundDetection." Additionally, monthlyReport also contains a reportDate attribute of type string that is updated daily by setCurrentDate, and this string represents the most recent date that the object contains detections for. Additionally, monthlyReport contains a variable setExpiryDate of type string that is updated daily by setExpiryDate, and this string represents the least-recent date that the object contains detections for. Similarly the class yearlyReport also contains attributes for currentDate and expiryDate, both of type string, both containing the most recent and least recent dates for which yearlyReport contains detections for. yearlyReport also contains a variable textReport of type string that is relatively long and contains a natural language summary of all of the detections occurring for the past year. This string is dynamically updated as new detections are added and summarized.

Operations

The Group 9 Mountain Lion Detection System contains relatively few operations, and most have to do with updating dates and removing/adding soundDetection objects to containers. initialSoundDetection has an operation that executes upon its instance of initialSoundDetection being received by the central computer, and results in flashing lights and sounds within the ranger station, controlled externally by the central computer. initialSoundDetection also contains the operation classifyDetection that takes input in the form of strings (ranger name, detection classification, detection date) from the user in order to construct a finalizedSoundDetection object from attributes of the original initialSoundDetection, along with new, user-entered attributes rangerClassification, rangerName, and detectionDate. finalizedSoundDetection has the operations addToMonthlyRecord and addToYearlyRecord. Both of these operations are executed once the finalizedSoundDetection is instantiated. addToMonthlyRecord accesses the monthlyRecord object's list attribute "soundDetections" in order to add finalizedSoundDetection to the list within this monthlyRecord. addToYearlyRecord converts the attributes of finalizedSoundDetections into a text summary (1-2 sentences) of type string that is prepended to the larger string attribute textReport of object yearlyRecord. monthlyRecord has the operation setCurrentDate that accesses the central computer information to determine and store the current date. monthlyRecord also has the operation setExpiryDate with the input parameter of currentDate that creates a string expiryDate that represents the date 30 days prior to the currentDate. monthlyRecord also has the operation removeExpiredDetections with the input parameter expiryDate that iterates through the list soundDetections and checks each the detectionDate of each finalizedSoundDetection, and if there is a match to the expiryDate, the corresponding finalizedSoundDetection is removed from the detectionDates list. yearlyReport also has the operation setCurrentDate that functions identically to the setCurrentDate operation of monthlyReport. yearlyReport has the operation setExpiryDate that sets the string expiryDate equal to the date that is 365 days prior to currentDate. yearlyReport has the operation removeExpiredDetections with the input parameter expiryDate in order to remove portions (sentences) of the string textReport that correspond to a detection (or detections) at the expiryDate. yearlyReport has the operations generateSensorReport, generateGraphicalReport, generateDetectionReport, and generateRangerReport. These operations each take in the textReport attribute as an input parameter to generate the reports detailed in bullets 1-4 of the System Description - generateDetectionReport corresponds to bullet 1, generateSensorReport corresponds to bullet 2, generateGraphicalReport corresponds to bullet 3, and generateRangerReport corresponds to bullet 4.

Development Plan and Timeline

<u>Overview and Duration of Each Phase</u>

#1 Research and Planning – 1 week

#2 System Design – 2 weeks

#3 System Development – 8 weeks

#4 Report Generation – 1 week

#5 System Deployment and Maintenance - FOREVER

<u>#1 Research and Planning</u>

Tasks:

1. Research on Animals-R-Here's animal detection system and its viability to detect mountain lions.
2. Establish hardware and software requirements.
3. Develop a plan, including timeline, budget, and team member duties.

Team member duties:

Bobby:	Conduct research on the Animal-R-Here's animal detection system, establish hardware requirements.
Charlie:	Research software requirements and develop the project plan.
Alberto:	Develop financial cost.

<u>#2 System Design</u>

Tasks:

1. Design the noise detection sensors placement plan for the park.
2. Design the animal noise detection algorithm to detect mountain lion sounds.
3. Development and design of alert messaging system, including noise type, noise strength, and location information.
4. Design and development of the control program for the controlling computer, which includes the alert sound, alarm off feature, and detection classification functionality.

Team member duties:

Bobby:	Design sensor placement for the park.
Charlie:	Design algorithm for noise detection sensor to detect mountain lions.
Alberto:	Design the alert messaging system and control program for controlling computer.

#3 System Development

Tasks:

1. Design and development of noise sensors hardware and firmware
2. Integration of animal noise detection algorithm with noise detection sensors.
3. Develop the alert messaging system, including the communication protocol and software to send alerts to the controlling computer.
4. Design and development of control program for the controlling computer with alarm sound and detection classification functionality.
5. Test the system.

Team member duties:

Bobby:	Develop the noise detection sensors and firmware.
Charlie:	Develop the animal noise detection algorithm and integrate it with the noise detection sensors.
Alberto:	Develop the alert messaging system, control program, and testing.

#4 Report

Tasks:

1. Develop the report module for the system, including reports for all mountain lion detections, detections at sensor locations, graphical reports, and detection classifications by ranger.

Team member duties:

Bobby:	Create the report module for all mountain lion detections and detections specific to sensor location.
Charlie:	Create the graphical report module.
Alberto:	Create the detection classification by ranger report module.

#5 System Deployment and Maintenance

Tasks:

1. Set up the system in the San Diego County Parks and Recreation Department and provide user training.
2. Provide maintenance and support for the system.
3. Create documentation for the system, including user manuals, troubleshooting guides, and technical specifications.

Team member duties:

Bobby:	Deploy the system and provide user training.
Charlie:	Provide maintenance and support for the system.
Alberto:	Develop documentation for the system.