SEAD Backend Testing Document

Table of Contents

Database Division

Interaction with the API

User Story: *As a GUI developer, I want to query the database for signals within a certain range.*

Actor	System
1. Initiate an HTTP GET request to the API, providing appropriate filters at GET parameters.	2. Parse GET request and build a query according the the specified filters. This query is executed and the results are formatted and rendered. Results are sent to the user.
3. The client receives the raw data in a two dimensional Python array, with the first row being the header of the columns.	

Interaction with the Landing Zone

Actor	System
1. SEAD plug requests configuration from server by sending its header containing its serial number and time.	2. Landing zone reads header, sends an ACK, constructs a configuration and returns it to the client. Landing zone sends an OKAY after it is finished sending the configuration.
3. SEAD plug receives packet(s) and begins to collect data. It sends it to the server when unknown criteria has been met.	4. Landing zone receives data and bulk inserts the data to the database. Landing zone sends ACK.
5. Client receives acknowledgement from the server.	

Test Coverage Summary

The landing zone and API have not been fully unit tested. Due to the short development period and ambitious scope of the project, we prioritized features over through testing.
In order to reduce the number of possible bugs, we used a simple compiled strictly and statically typed language for the landing zone and made all of our code as simple as possible to avoid hidden bugs. We also avoided objects and inheritance which can be sources of complexity and bugs. The single set of unit tests is present only to ensure accuracy of some conversion functions needed by the module which decodes packets from the SEAD plug.

Known issues

The subset API option does not always return the specified number of points even when doing so is possible. This is because we are selecting every nth point where n is an integer. We may need to tweak our rounding code.

Analysis and Classification Division

Features to be Tested

We plan to test the software on SEAD plug data files in .csv format only from only the following device types:

* Microwaves
* Laptop computers
* Incandescent lamps that draw 40 or more watts

The data files used to perform the testing should be gather from devices satisfyingly similar to the devices measured while building the classifier running under typical conditions. (i.e., no circa-1980s machines with CRT monitors, etc). Additionally, test data files must meet the following additional requirements:

* At least one minute of data
* Data given in one contiguous chunk
* Data files recorded from exactly one device

Features not to be Tested

We reserve the right to add elements to this section as necessary that may not have been thought of when the document was prepared. Features not to be tested include, at minimum:

* Detection and/or classification of unknown device types
* Detection and/or classification of device types other than those listed above
* Live functioning, e.g., working in step with the frontend.

Testing Data Handling

Data used for testing must been separated from the data used to train the classifier, prior to training the classifier.

Evaluating Classification Methods

Cross-validation was used to select an effective classification method while working with minimal data. One appliance from each category was not entered into the training set data and later used to test the classifier for accuracy.

Expected Outputs and Pass / Fail Decision Procedure

The classifier is expected to properly classify devices correctly with better-than chance accuracy. These baselines are calculated given that the classifier is trained with an equal amount of data from each device type.

Fail condition:

Classifier recognizes device types with an accuracy of chance or below.
Proportion correct <= 1/(number of device types)
Proportion correct <= 33%

Pass condition:

Classifier recognizes device types with an accuracy better than chance.
Proportion correct > 1/(number of device types)
Proportion correct > 33%

System Test

Scenario:

A user has a laptop computer plugged into a SEAD plug from time X to time Y. The user access the database classification API with the appropriate URL, specify the proper SEAD plug serial number, and correct times X and Y. The system returns ‘computer’.

Unit Test

Components:

Classifier

Input classes:

1. Expected device type plugged into SEAD plug {laptop computer, lamp, microwave}
   To test: Plug device into SEAD plug, follow classification procedure. Success is marked by a correct response. A failure is defined as incorrect response.

2. Unknown device type plugged into SEAD plug
   To test: Plug blow drier into SEAD plug, follow classification procedure. Success is marked by any response. A failure is if no response.

3. No device plugged into SEAD plug
   To test: Make sure nothing is plugged into the SEAD plug, follow classification procedure. Success is marked by any response. A failure is if no response.