UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

Neo Tandem Technologies

Eye Tracking
Functional Specification

Author:	*Student number:*
Duran Cole	u13329414
Michael Nunes	u12104592
Molefe Molefe	u12260429
Tebogo Seshibe	u13181442
Timothy Snayers	u13397134

October 28, 2015

Functional requirements

Eye Tracking

Github Link: https://github.com/MichaelNunes/Neo-Tandem-Tech-Eye-Tracking

Version: 0.2 Basic requirements
Version: 0.5 Added content
Version: 0.9 Added in-depth content
Version: 1.0 Stable release (version 1) requirements

Contents

List of Figures

1 Introduction

1.1 Project Background

The Geo-Information Science (GIS) department of the University of Pretoria (UP) wanted a system that could track the attention of a user on a 3D model. The system needs to provide a way to measure the amount time a user has looked at a specific point. As an added, they also wanted to be able to track the attention of a user on a video, also providing the same manner of measurement used for 3D models.

The GIS department has provided the team with The Eye Tribe eye tracking camera. The Eye Tribe also provides a Software Development Kit (SDK) for development when purchasing their camera. The camera comes as a much cheaper solution than most other eye tracking solutions such as Tobii. The camera we were provided with is currently the cheapest on the market and sells for only $99. The camera has an accuracy rating of, on average, about 75% but can achieve accuracy ratings of much higher if all perfect conditions are met.
For more information please see: http://theeyetribe.com

1.2 Project Vision

The aim of this project is to provide a Windows form application that incorporates The Eye-Tribe (SDK). The application would use the Eye-Tribe camera to track the focus of a user on various types of models. Using the SDK, it would generate a heat-map of the users focus and eye movements that can be applied to the various 2D, 3D and video models.

2 Use Case/Service Contracts

2.1 Heat Maps

This section will cover all the aspects of the heat maps use case (Figure 1). This use case is chosen as it is a vital part of the entire system. The data that is recorded will be used to create the heat maps that can then later be used to analyse the data and be used for future uses.

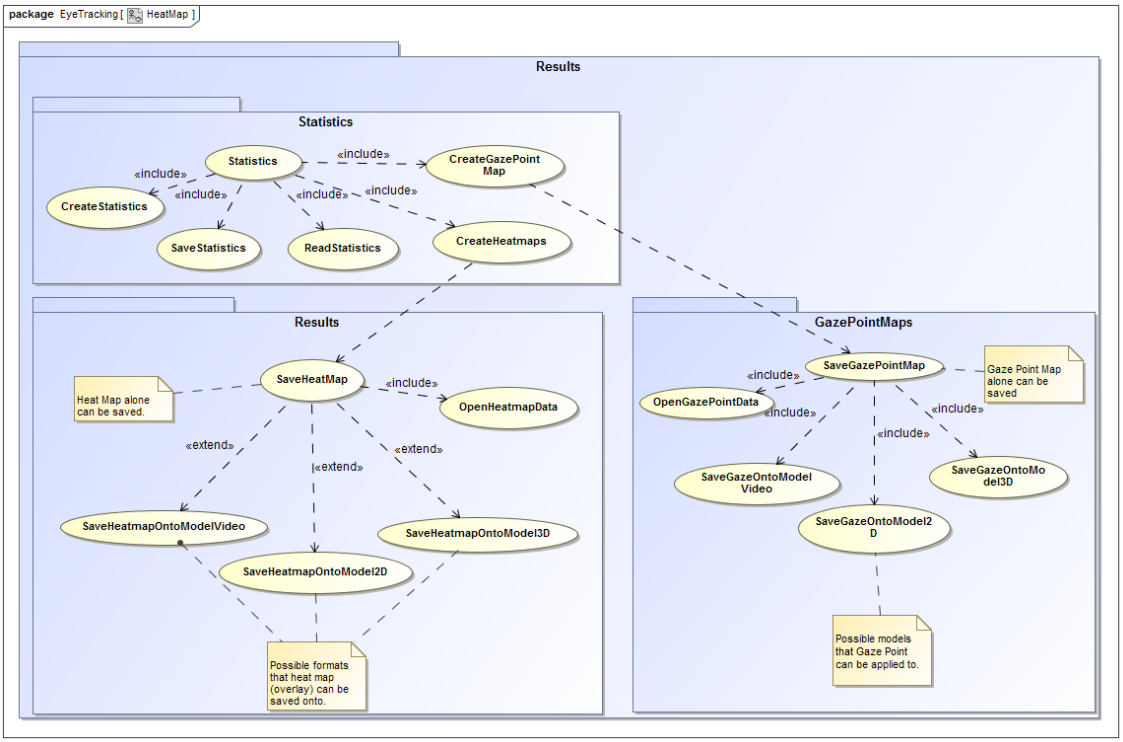

Figure 1: Heat Map Use Case

The following sub use cases are included in this use case:

2.1.1 CreateHeatMap

The create heat map use case deals with the creation of the heat map from the recorded data. The heat map shows where on the media item has been viewed the most by the user(s).

- Pre-condition: Eye tracking data must be already recorded.
- Post-condition: Heat-map of the data is created.
- Request Data Structure: Raw information recorded by eye tracking camera.
- Return data Structure: Heat-map is created

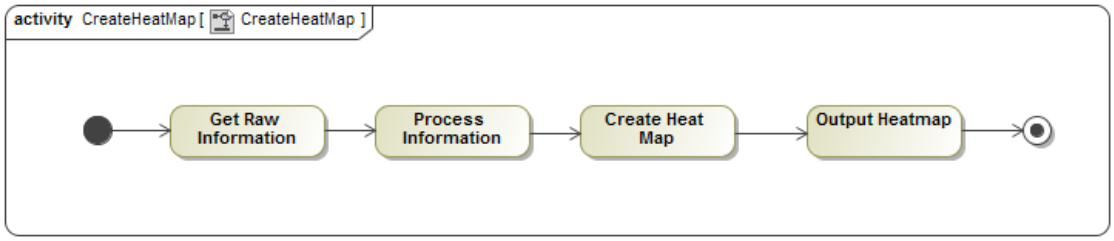

Figure 2: Create Heat Map Activity

2.1.2 ReadHeatMap

This use case deals with the viewing of heat-maps from their recorded data. Once the heat-map is created the users' will be able to create a heat-map and view the data.

- Pre-condition: Heat-map must have been created.
- Post-condition: Heat-map of the data is viewable and can be further analysed.
- Request Data Structure: The created heat-map.
- Return Data Structure: Heat-map is viewable to the user(s).

2.1.3 SaveHeatMap

This use case deals with the saving of heat-maps that are created from their respective recorded data. The heat-map will be saved on the users' computer to be viewed at a later date. This will save only the heat-map which can later be saved onto a model at a later date.

- Pre-condition: Raw information must have already been recorded so that new heat-map can be created.
- Post-condition: Heat-map of the data is saved.
- Request Data Structure: Raw eye tracking information
- Return Data Structure: Heat-map is saved.

2.2 Save Heat Maps

The saving of the heat-maps provides important functionality to the system as it will allow the users' to save the data which can later be used to either continue research or to compare data.The heat-map will serve as an overlay for the media type and be placed over the media model. The saving of heat maps can be divided into three sub-sections, namely SaveHeatMapOntoModel3D, SaveHeatMapOntoModelVideo and SaveHeatMapOntoModel2D.

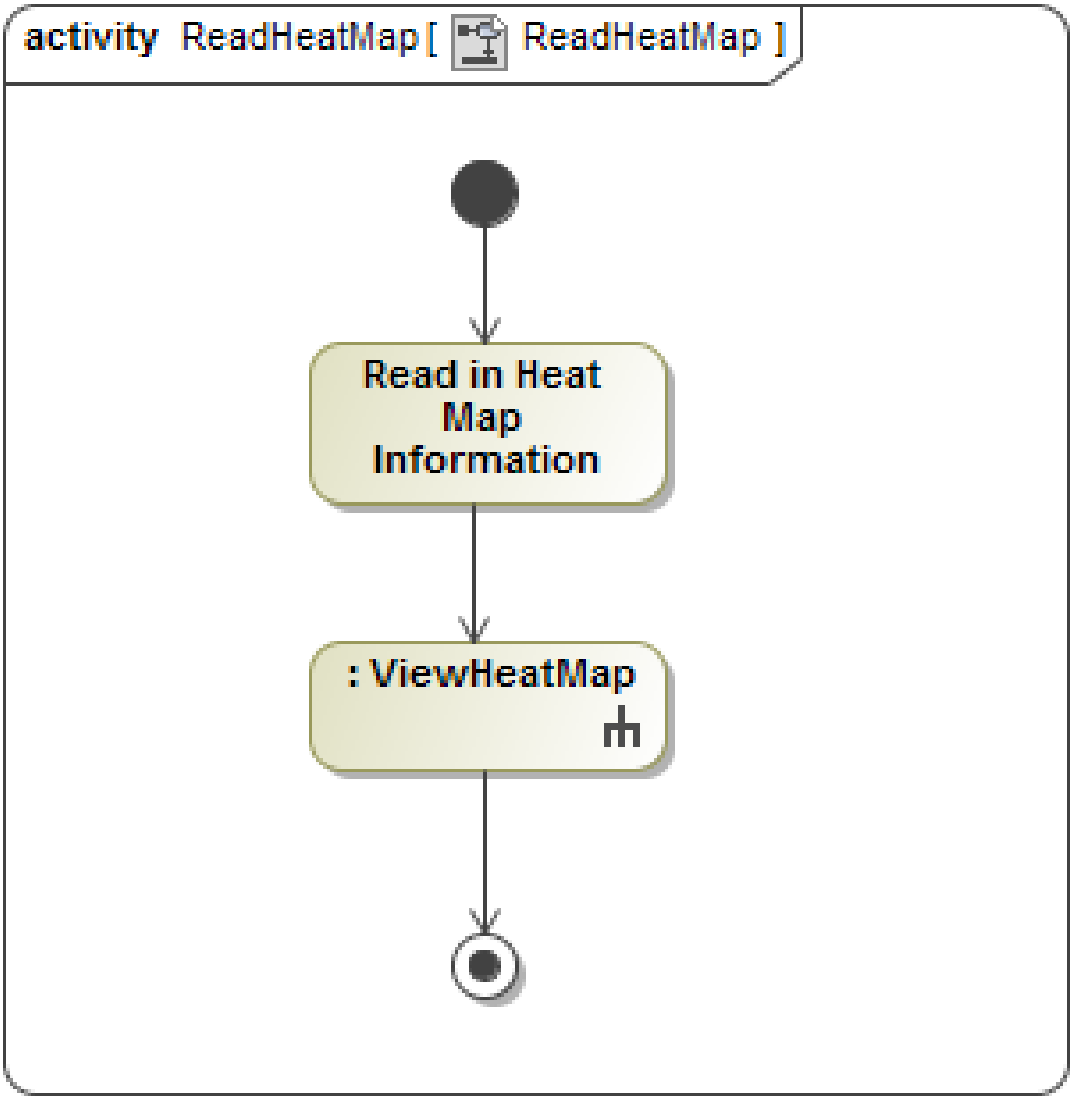

Figure 3: Read Heat Map Activity

2.2.1 SaveHeatMapOntoModel2D

The heat-map that is created for a two dimensional (2D) model media type will be saved and applied to the original 2D model to see the heat-map on top of the model. This then gives an indication of what has been viewed the most by the user(s) test participants.

- Pre-condition: Eye tracking data must have already been recorded.
- Post-condition: Heat map of the data is viewable on the 2D model as an overlay.
- Request Data Structure: Raw eye tracking data.
- Return Data Structure: Heat map of the data is placed over the 2D Model.

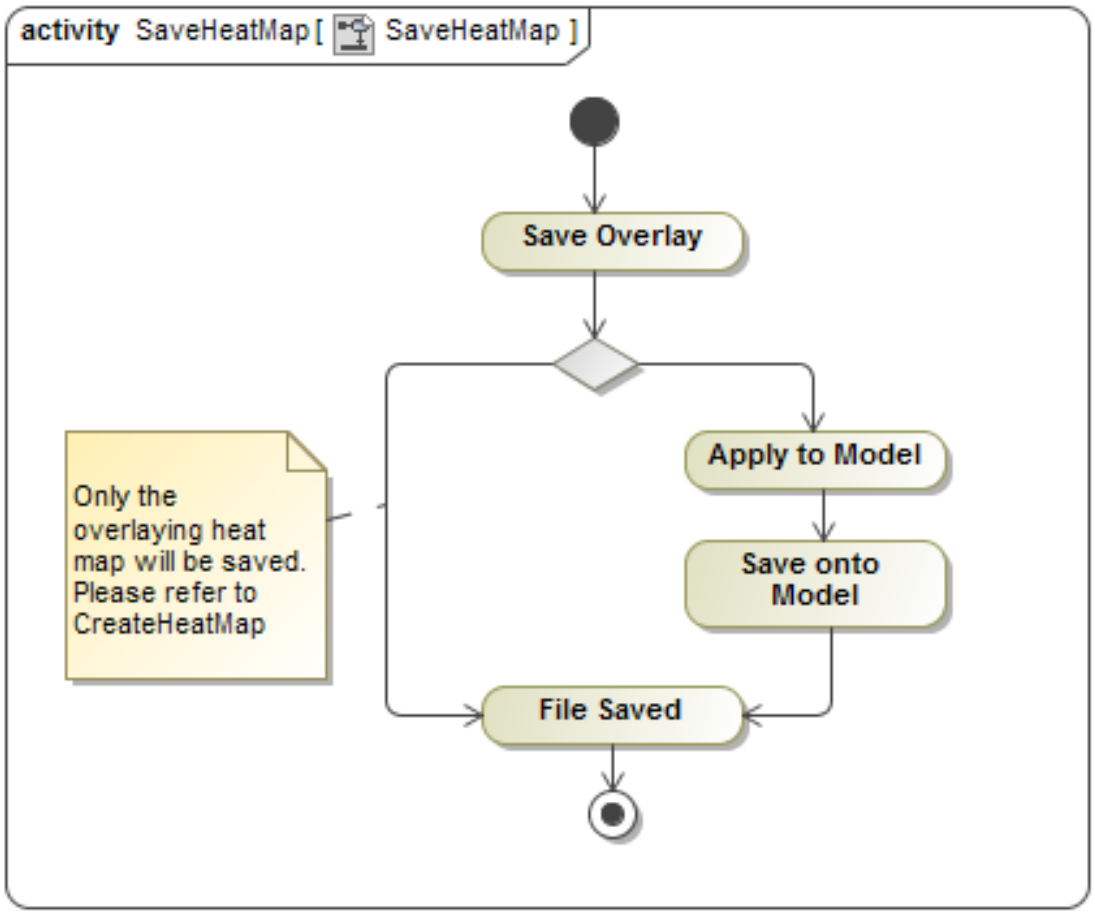

Figure 4: Save Heat Map Activity

2.2.2 SaveHeatMapOntoModelVideo

The heat-map that is created for a video media type will be saved and then applied to the video to see the heat-map over the video to indicate what has been viewed the most by the users'. The video model is spliced into images at either 30 or 60 frames per second of the video, the heat-maps can then be applied to the created images and recompiled into a new video.

- Pre-condition: Eye tracking data must have been recorded and video model must be available.
- Post-condition: New video is created with the heat-map on top of it.
- Request Data Structure: Video and raw information of recording.
- Return Data Structure: Heat-map of the data is placed over each frame of the video.

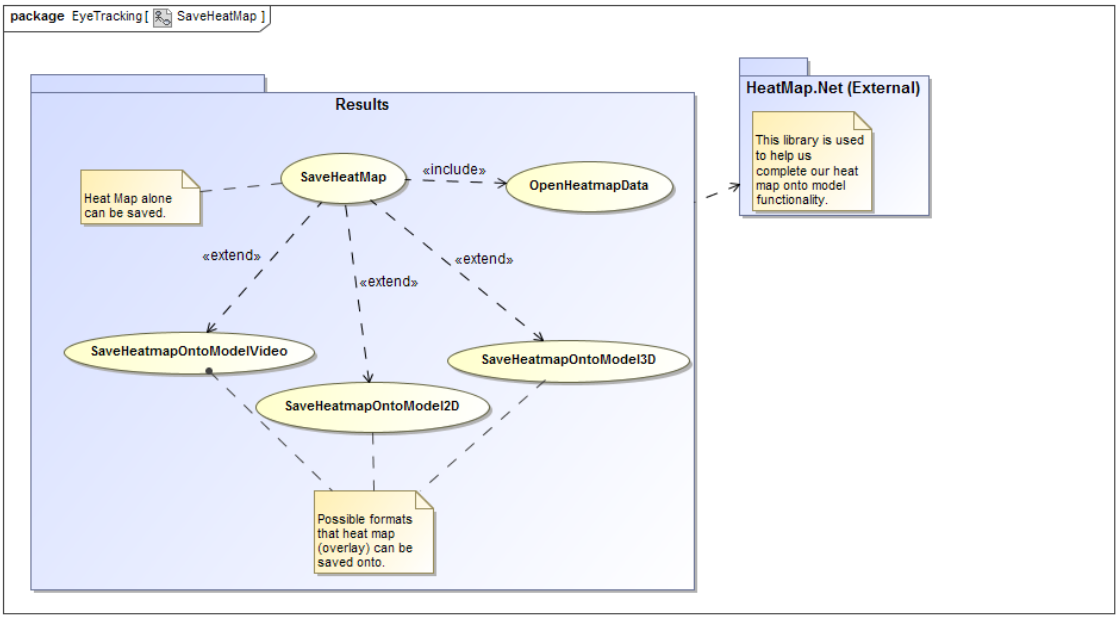

Figure 5: Save Heat Map Use Case

2.2.3 SaveHeatMapOntoModel3D

The 3D model will have multiple images of it created from different angles. Each of the created image can then have a heat-map placed over it.

- Pre-condition: Eye tracking data must have already been recorded and 3D model must be available for the creation of images.
- Post-condition: Heat-map of the data is viewable on the 3D model as an overlay.
- Request Data Structure: Recorded eye tracking information.
- Return Data Structure: Heat-map of the data is placed over each image of the 3D model.

2.2.4 SaveHeatMapOntoModel3DFlythrough

The 3D model will be rendered and allow the user to explore the model as they wish.The recording of the model is done and a video of the user experience is created.The video will then have the heatmap applied to it.

- Pre-condition: Eye tracking data must have already been recorded and 3D model must be available for the creation of the video.
- Post-condition: Heat-map of the data is viewable on the 3D model as an overlay.
- Request Data Structure: Recorded eye tracking information.

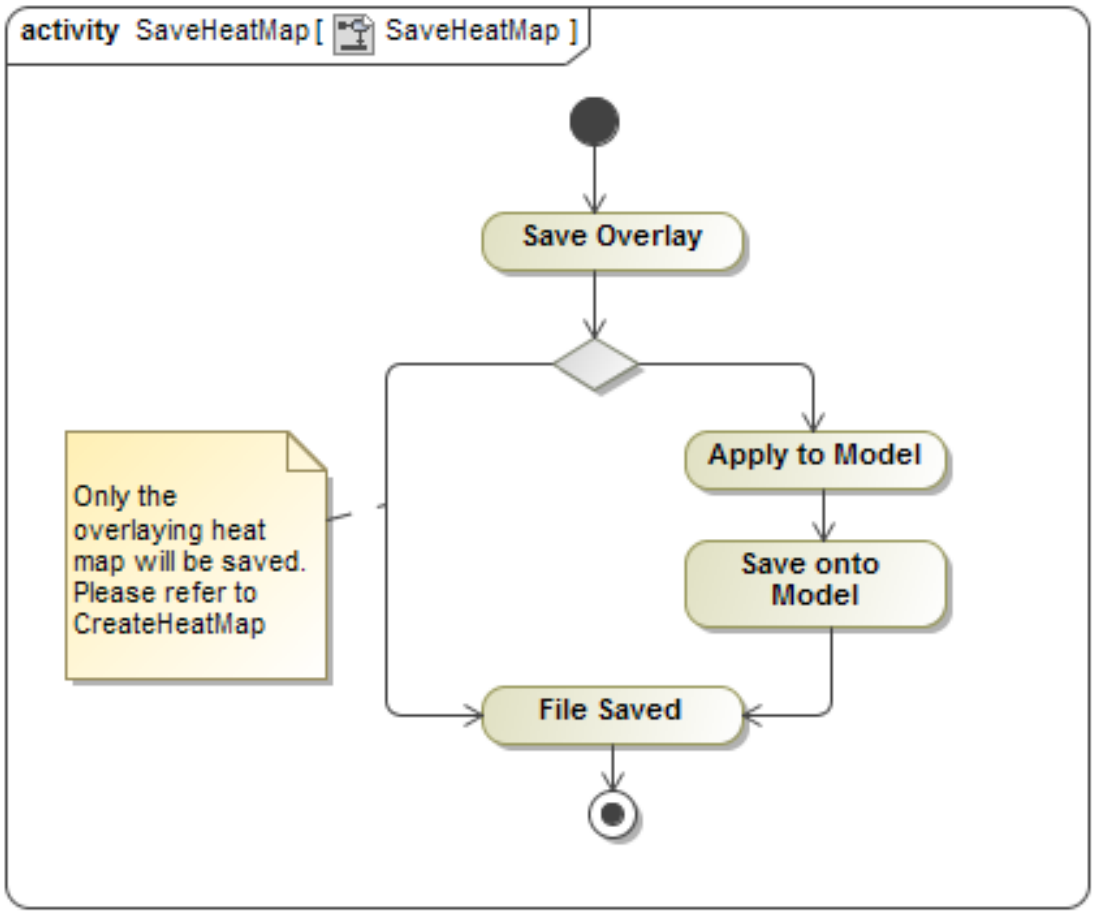

Figure 6: Save on 2D Model Activity

- Return Data Structure: Heat-map of the data is placed over the video of the 3D model.

2.3 Gaze Point Maps

This section will cover all the aspects of the Gaze Point Map use case (Figure 10). Gaze point maps track where the user is looking but specifies the order in which their respective gaze moves from one location to another. This use case forms an added extra to the user and can be used to further analyse information and data that has been recorded. The data that is recorded will be used to create the gaze point that can then later be used to analyse the data and be used for future uses.

The following sub use cases are included in this use case:

2.3.1 CreateGazePointMap

The create gaze point activity deals with the creation of the gaze point maps from the recorded data. The gaze point map shows the order and length in which the subject

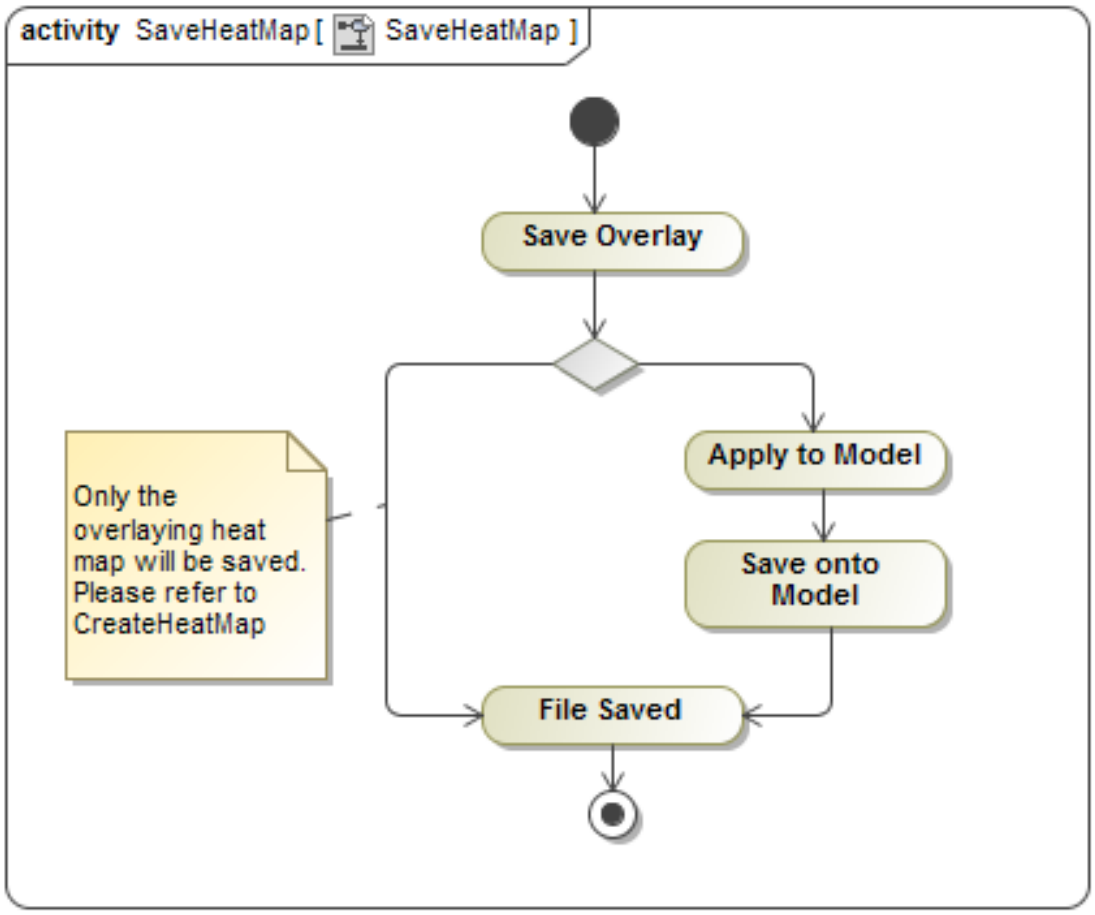

Figure 7: Save on Video Model Activity

views the model.

- Pre-condition: Eye tracking data must be already recorded.
- Post-condition: Gaze point map of the data is created.
- Request Data Structure: Raw information recorded by eye tracking camera.
- Return data Structure: Gaze point map is created

2.3.2 ReadGazePointMap

This activity deals with the viewing of gaze point maps from their recorded data. Once the gaze point maps have been created the users' will be able to see the gaze point maps and view the data.

- Pre-condition: Gaze point map must have been created.
- Post-condition: Gaze point map of the data is viewable and can be further analysed.

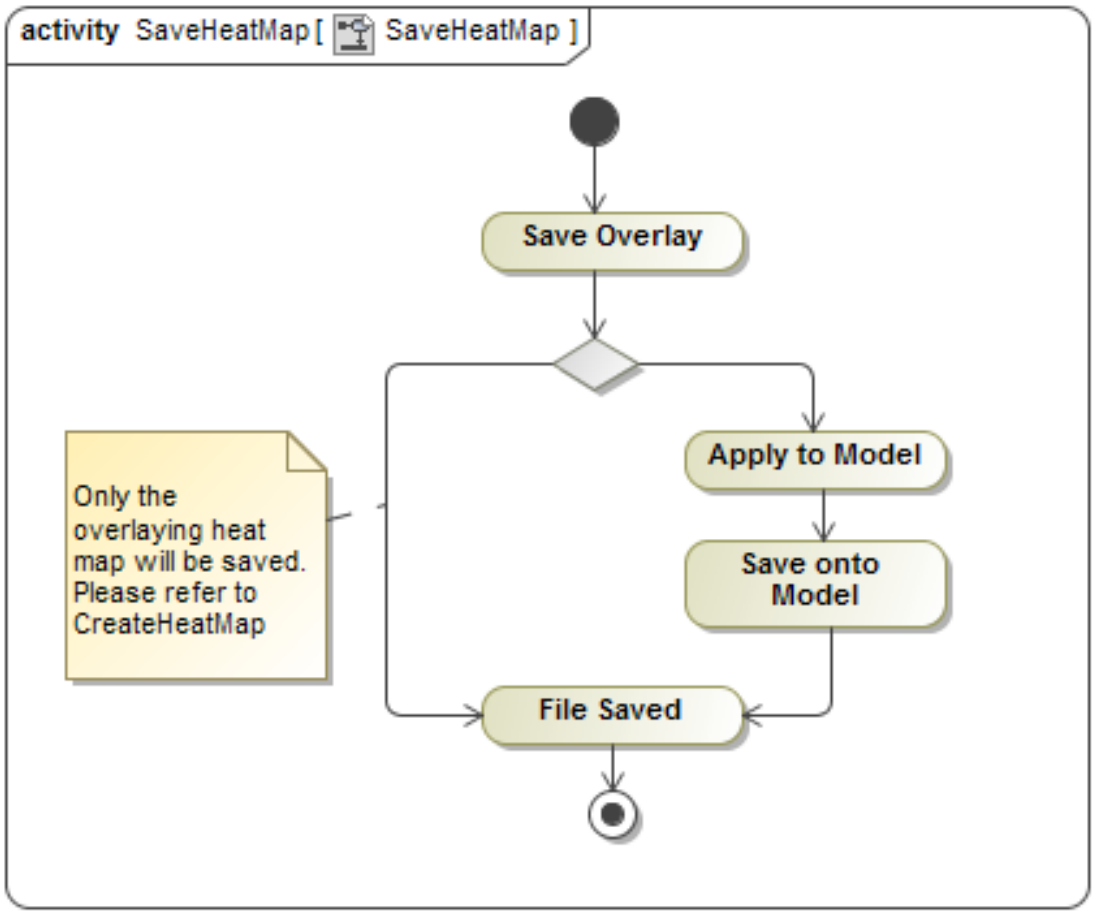

Figure 8: Save on 3D Model Activity

- Request Data Structure: The created gaze point map.
- Return Data Structure: Gaze point map is viewable to the user(s).

2.3.3 SaveGazePointMap

This activity deals with the saving of gaze point map that are created from their respective recorded data. The gaze point map will be saved on the users' computer to be viewed at a later date. This will save only the gaze point map which can later be saved onto a model at a later date.

- Pre-condition: Raw information must have already been recorded so that new gaze point map can be created.
- Post-condition: Gaze point map of the data is saved.
- Request Data Structure: Raw eye tracking information
- Return Data Structure: Gaze point map is saved.

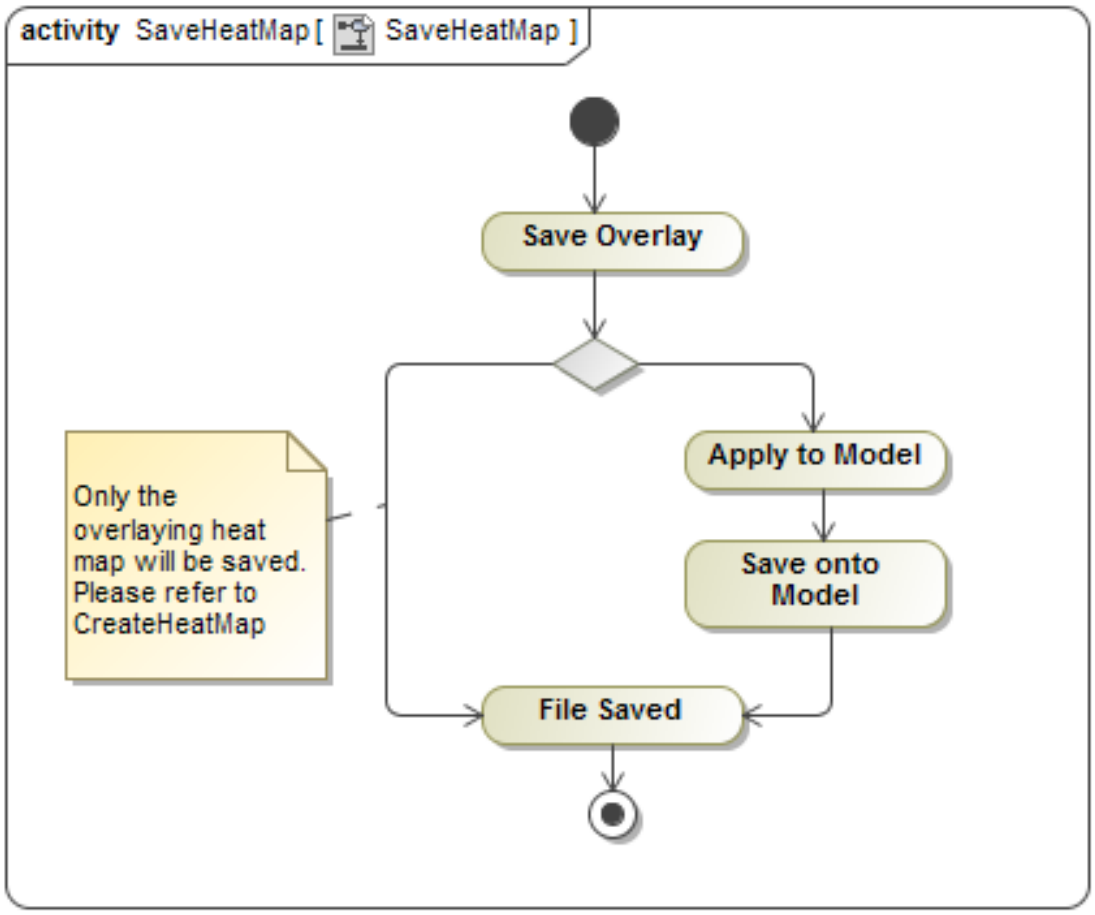

Figure 9: Save on 3D Model Activity

2.4 Save Gaze Point Maps

The saving of the gaze point map provides important functionality to the system as it will allow the users' to save the data which can later be used to either continue research or to compare data.The gaze point map will serve as an overlay for the media type and be placed over the media model. The saving of gaze point map can be divided into three subsections, namely SaveGazeMapOntoModel3D, SaveGazeMapOntoModelVideo and SaveGazeMapOntoModel2D.

2.4.1 SaveGazeMapOntoModel2D

The gaze point map that is created for a 2D model media type will be saved and applied to the 2D model to see the gaze point map on top of the model to indicate what has been viewed the most by the user(s) test participants.

- Pre-condition: Eye tracking data must have already been recorded.
- Post-condition: Gaze point map of the data is viewable on the 2D model as an overlay.

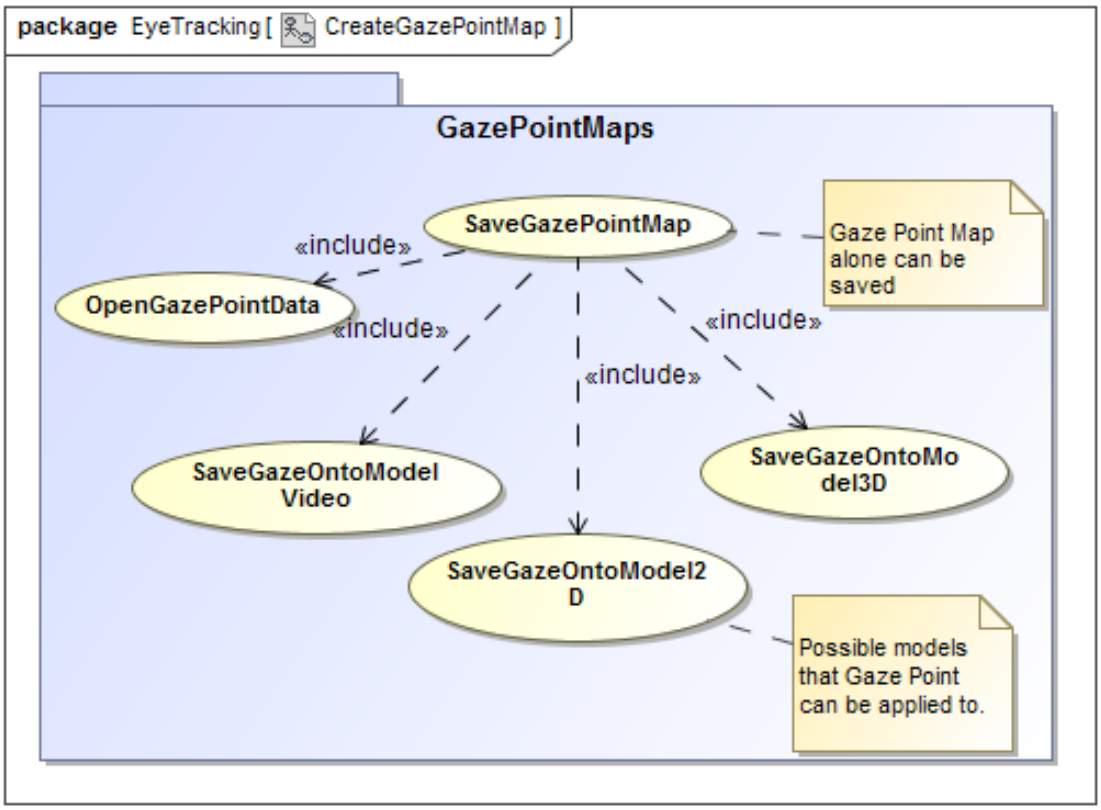

Figure 10: Gaze Point Use Case

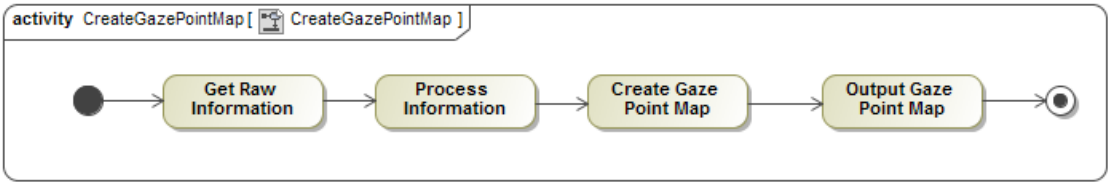

Figure 11: Create Gaze Point Map Activity

- Request Data Structure: Raw eye tracking data.
- Return Data Structure: Gaze point map of the data is placed over the 2D Model.

2.4.2 SaveGazeMapOntoModelVideo

The gaze point map that is created for a video media type will be saved and then applied to the video to see the gaze point map over the video to indicate what has been viewed the most by the users'. The video model is spliced into images at either 30 or 60 frames per second of the video, the gaze point map can then be applied to the created images and recompiled into a new video.

- Pre-condition: Eye tracking data must have been recorded and video model must be available.

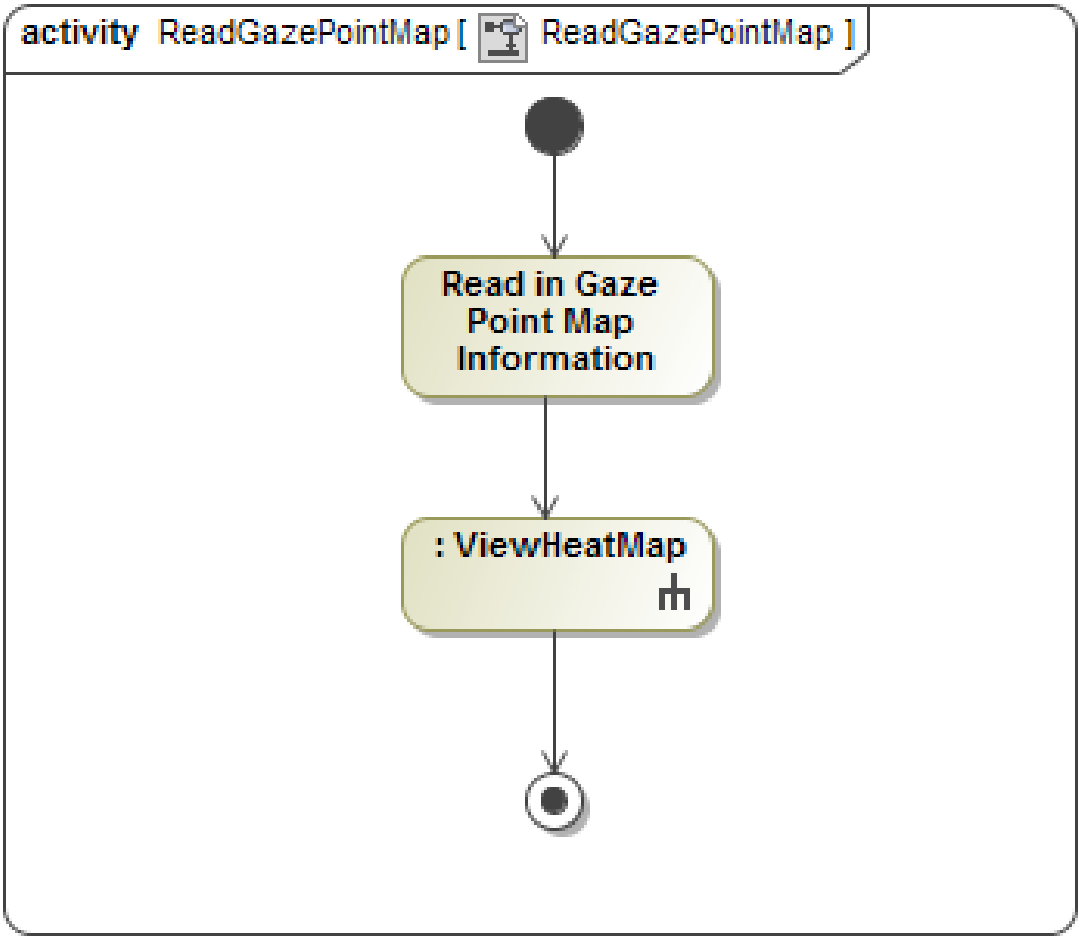

Figure 12: Read Gaze Point Map Activity

- Post-condition: New video is created with the gaze point map on top of it.
- Request Data Structure: Video and raw information of recording.
- Return Data Structure: Gaze point map of the data is placed over each frame of the video.

2.4.3 SaveGazeMapOntoModel3D

The 3D model will have multiple images of it created from different angles Each created image can then have a gaze point map placed over it.

- Pre-condition: Eye tracking data must have already been recorded and 3D model must be available for the creation of images.
- Post-condition: Gaze point map of the data is viewable on the 3D model as an overlay.
- Request Data Structure: Recorded eye tracking information.

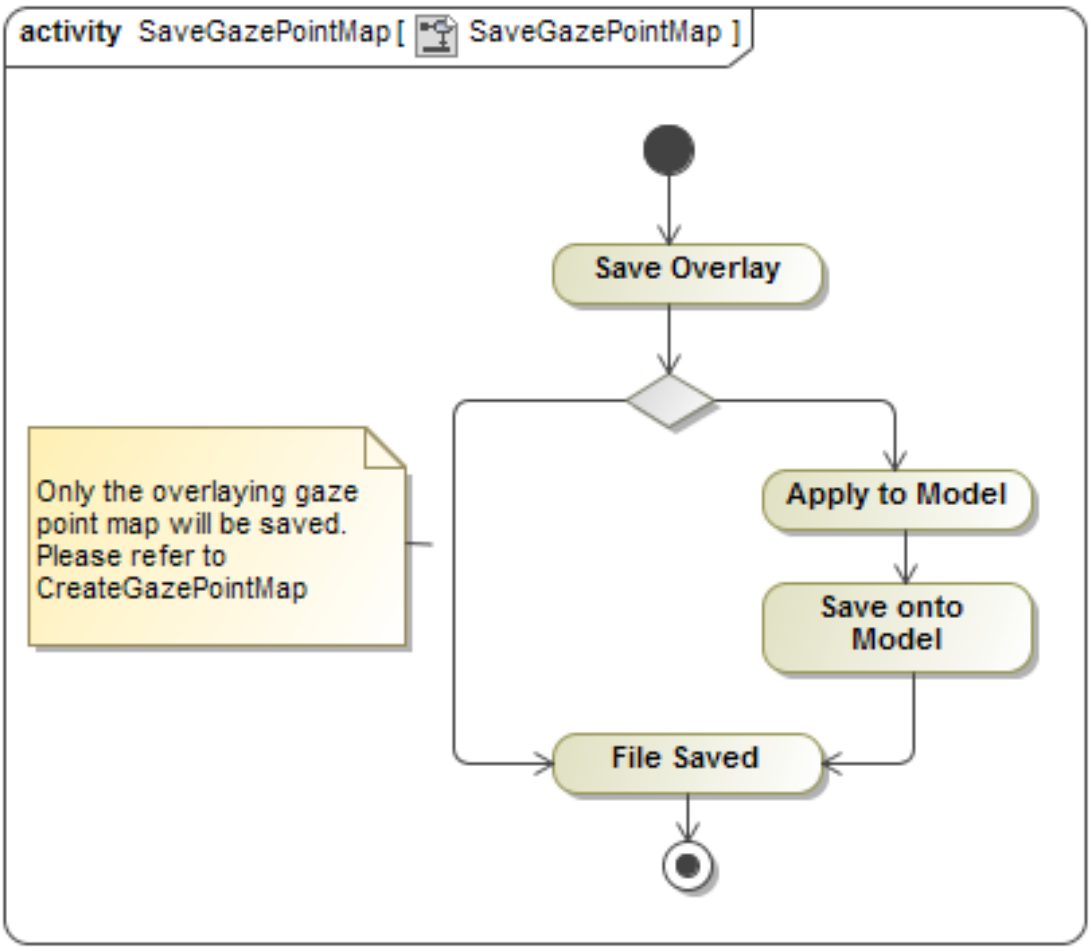

Figure 13: Save Gaze Point Map Activity

- Return Data Structure: Gaze point map of the data is placed over each image of the 3D model.

2.5 Models

This section details all the different types of media or models that heat-maps can be created for and saved onto. There are three types of models, namely Video, 2D and 3D. The heat maps are created based on the models and their types.

2.5.1 3D Models

3D Models will be opened and rendered by the system, and will then be made viewable to the user. The model will have images created of it at multiple angles. The heat-map can then be saved onto the models individual images as well as opened as the stand alone heat maps of each image.

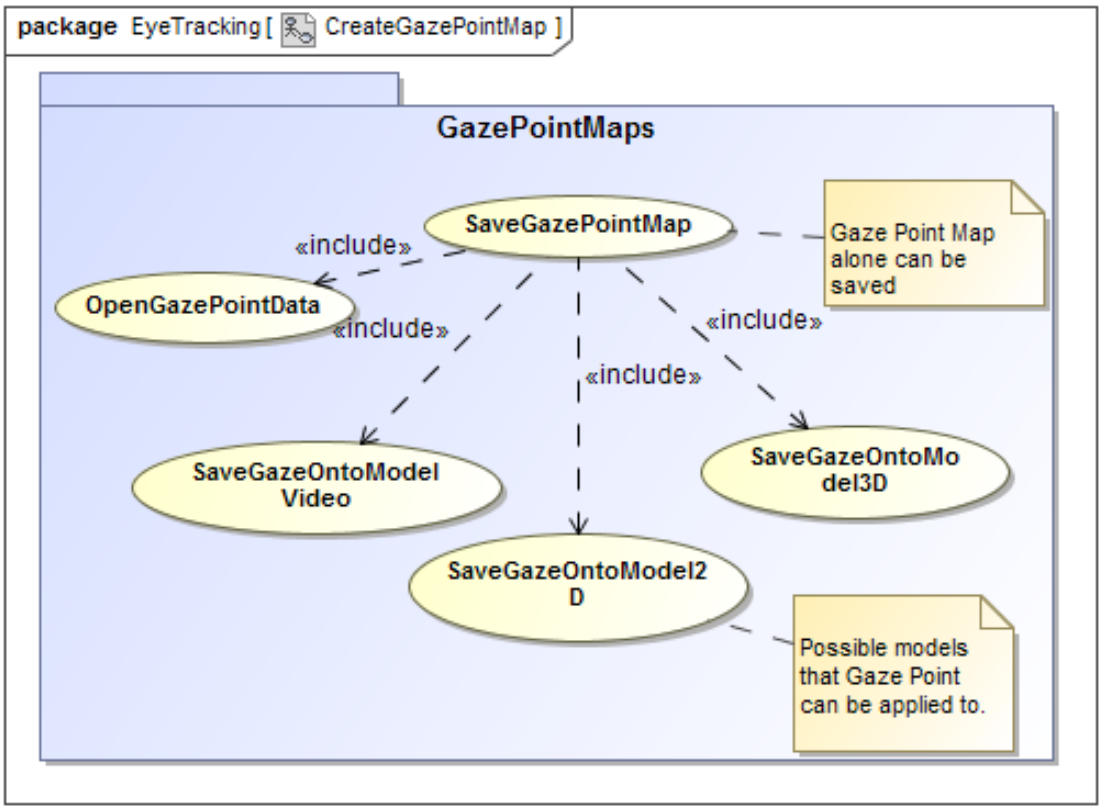

Figure 14: Save Gaze Point Map Use Case

1. ImportModel:
   This will allow for a 3D model media type to be imported into the system for further processing. The imported model can be used for recording data once the relevant images have been created and creating a relevant heat-map based on it which can then be applied to it at a later stage.

   - Pre-condition: 3D model must exist.
   - Post-condition: Images can be created and recording can be done on the model.
   - Request Data Structure: The 3D model that user wishes to preform eye tracking on.
   - Return Data Structure: Images of model will be created and returned.

2. ViewModel: Once the model is imported the model and images created, the will be available for viewing and further processing decided by the user.

   - Pre-condition: 3D model must have been imported and images created.
   - Post-condition: The 3D model images are viewable and can be recorded on.
   - Request Data Structure: The 3D models images.
   - Return Data Structure: Viewable 3D model images.

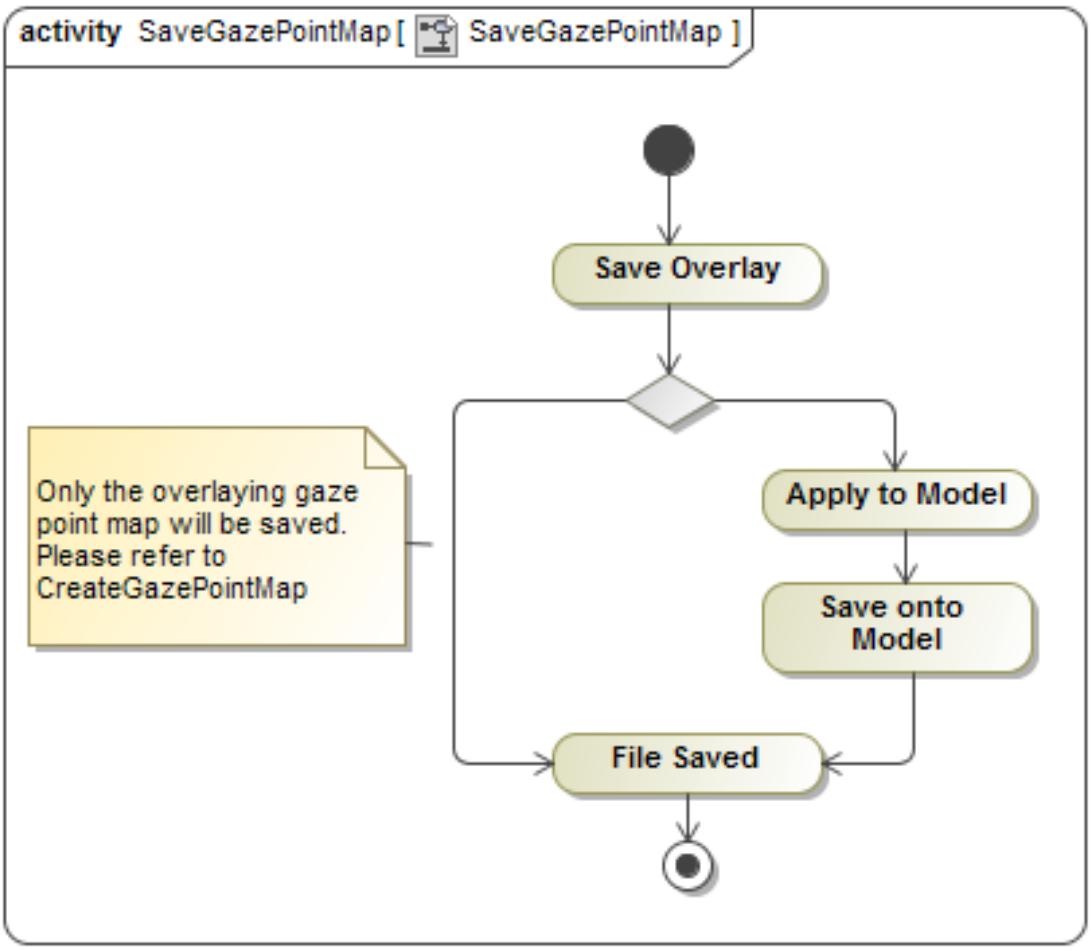

Figure 15: Save on 2D Model Activity

3. ExportHeatMapModel: The exported model will have the heat-map applied to it which can then be exported as a new images of the 3D model so that it can be viewed or used at a later date by the user.

    - Pre-condition: Heat-map must be created and Model Imported.
    - Post-condition: Exported model With heat-map on it to be viewed.
    - Request Data Structure: The eye tracking data must have been recorded and 3D model available
    - Return Data Structure: Model with heat-map applied.

4. Export3DModelImages: The model will have images created from various angles of the model. These images can then be used for recording.

    - Pre-condition: Selected model must be available.
    - Post-condition: Images will be created of model.
    - Request Data Structure: The selected model.
    - Return Data Structure: Images of 3D model.

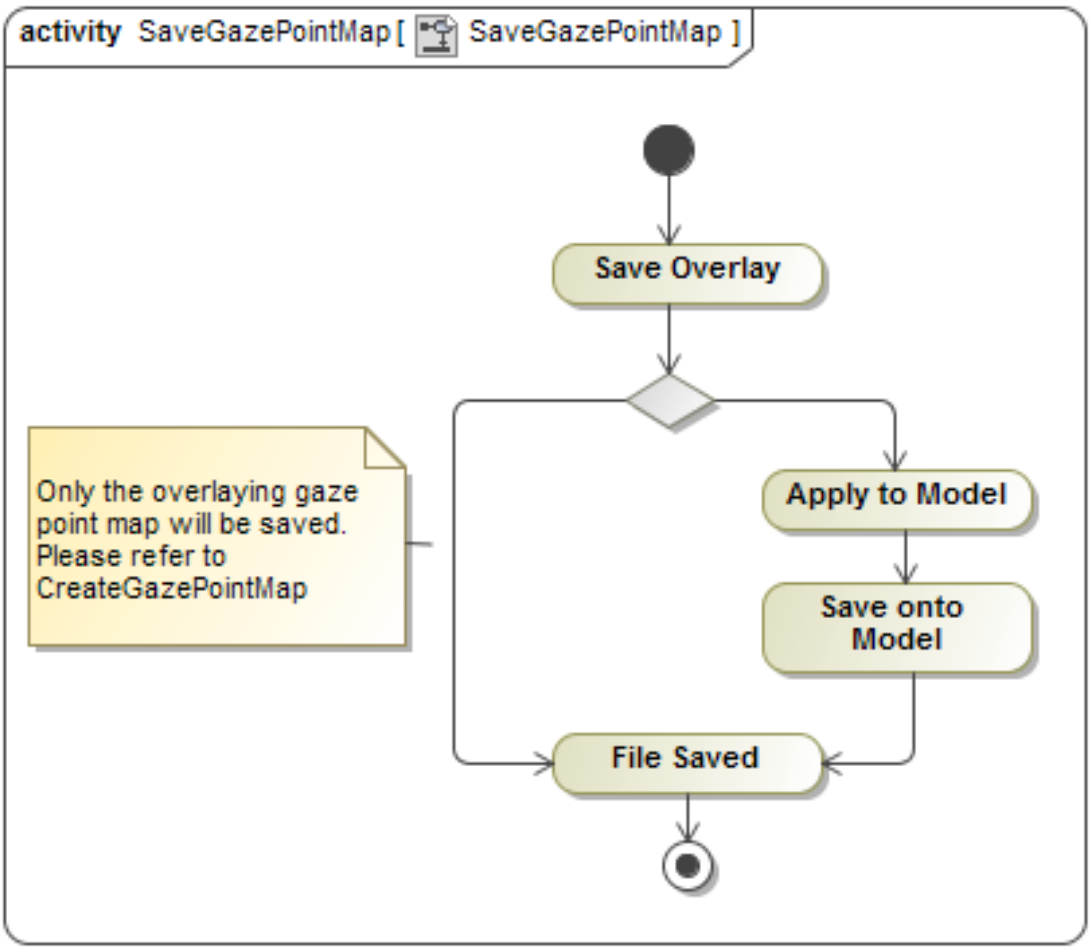

Figure 16: Save on Video Model Activity

2.5.2 3D Model Fly through

3D Models will be opened and rendered by the system, and will then be made viewable to the user. The model will have a window opened for the user to explore the model. The heat-map can then be saved onto the models video as well as opened as the stand alone heat map in the form of a video.

1. ImportModel:
   This will allow for a 3D model media type to be imported into the system for further processing. The imported model can be used for recording data once the video has been created and creating a relevant heat-map based on it which can then be applied to it at a later stage.
   - Pre-condition: 3D model must exist.
   - Post-condition: 3D model is rendered and is made interactive and recording can be done on the model.
   - Request Data Structure: The 3D model that user wishes to preform eye tracking on.

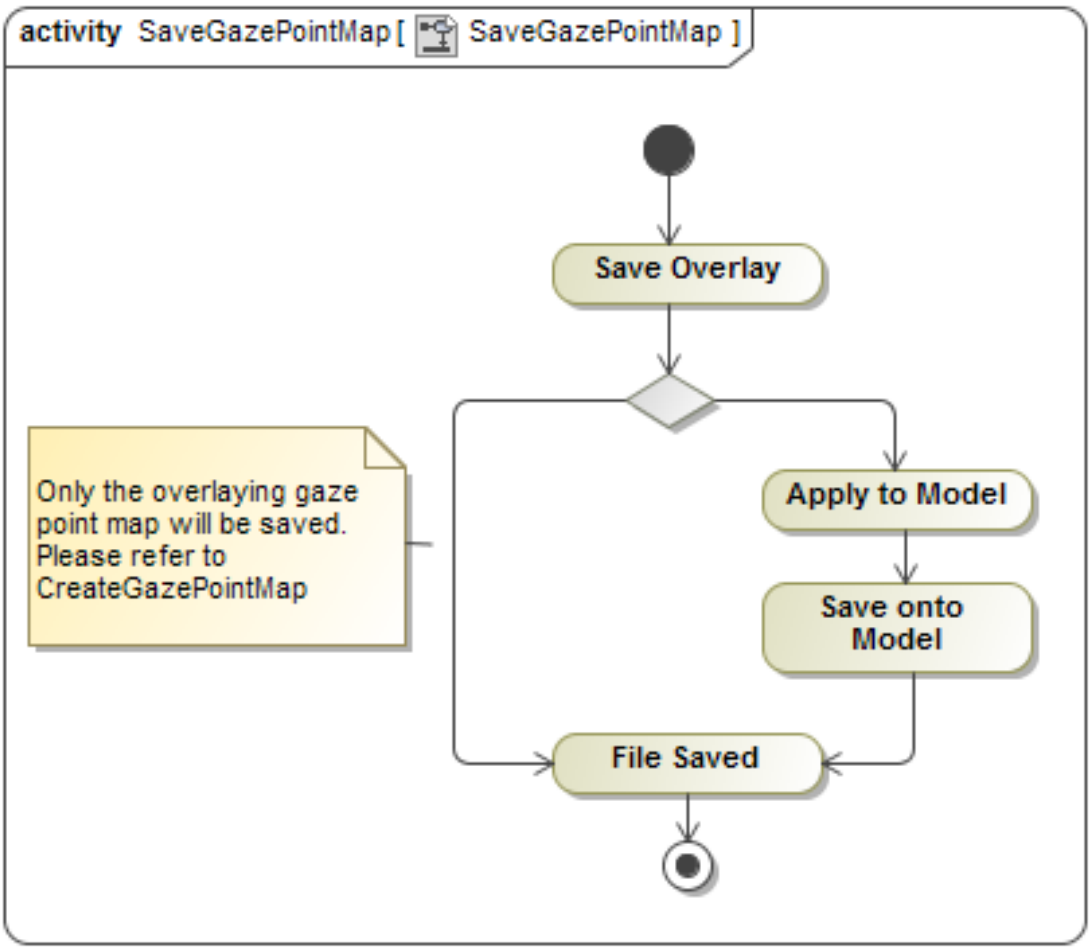

Figure 17: Save on 3D Model Activity

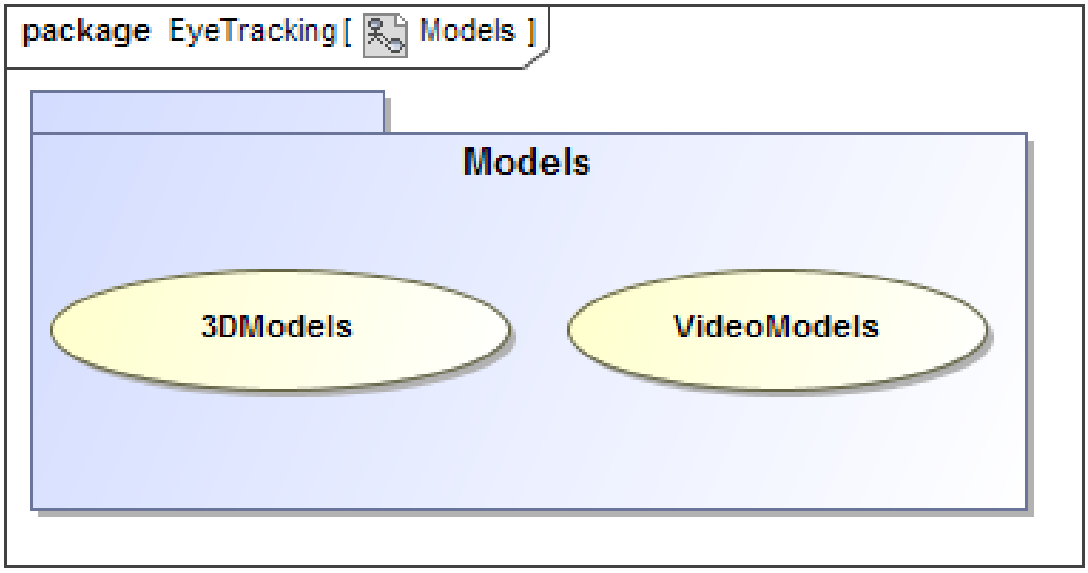

Figure 18: Models Use Case

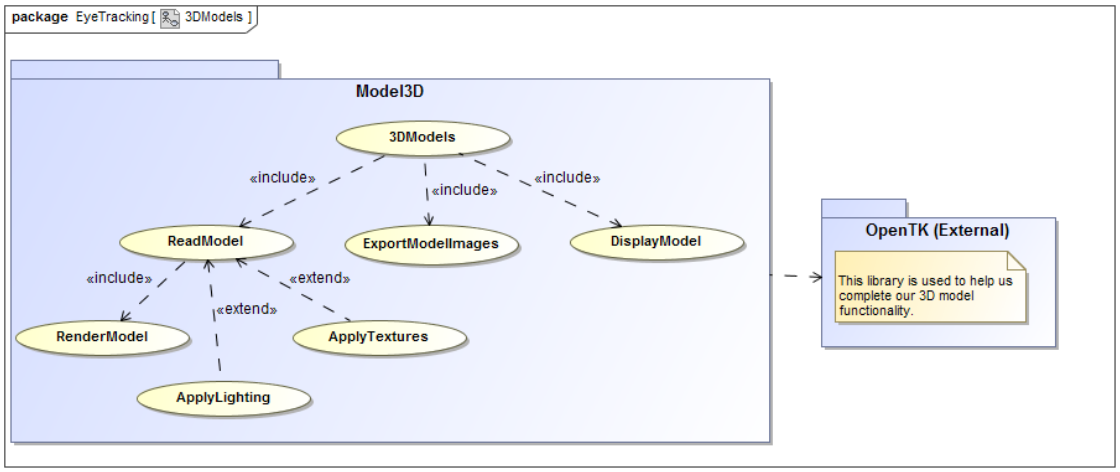

Figure 19: 3D Models Use Case

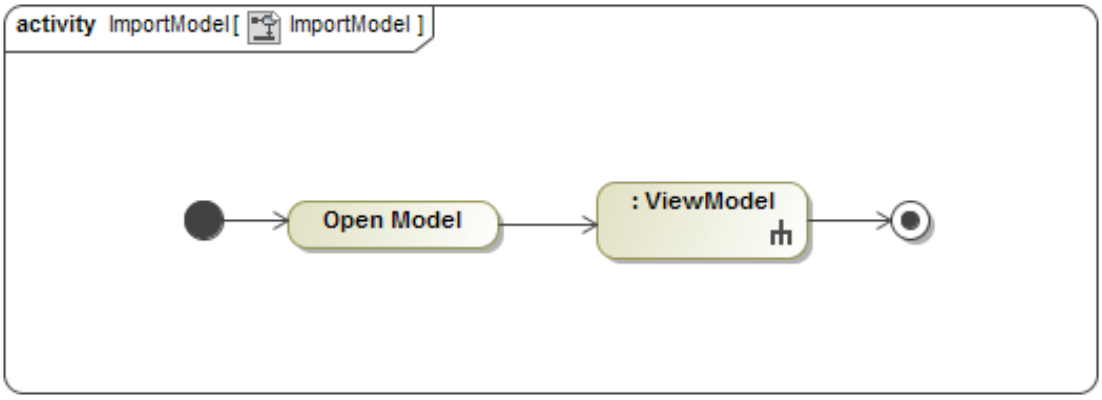

Figure 20: Import 3D Model Activity

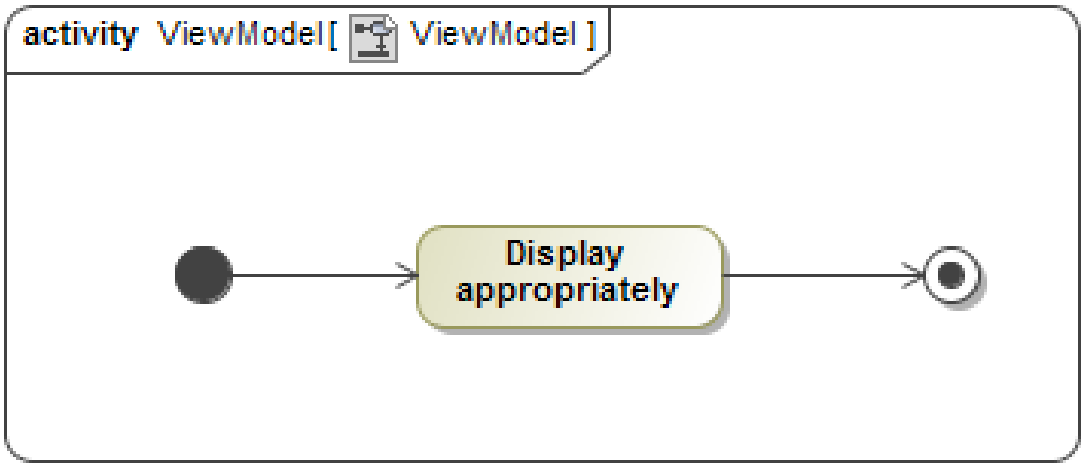

Figure 21: View 3D Model Activity

- Return Data Structure: Video of model exploration will be created and returned.

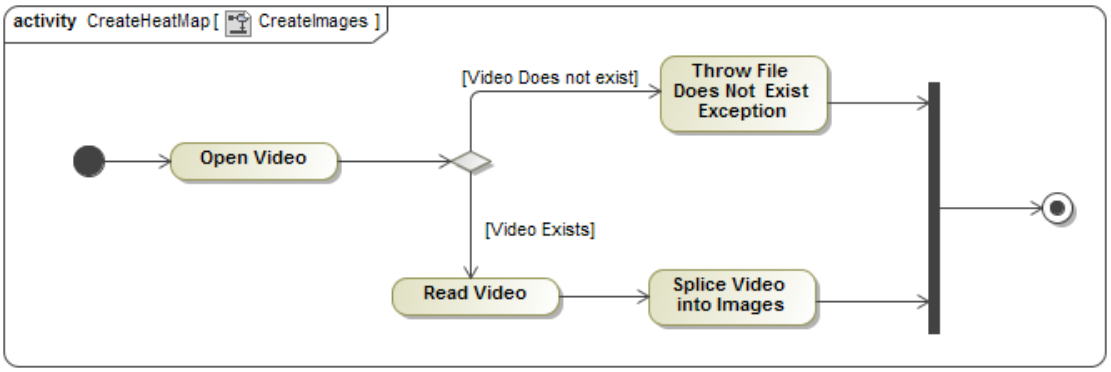

Figure 22: Create 3D Model Images Activity

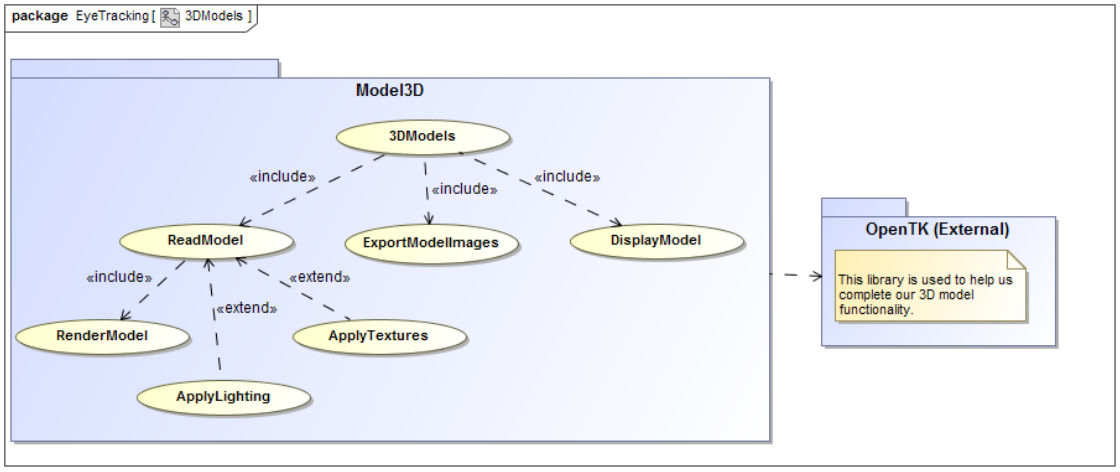

Figure 23: 3D Models Use Case

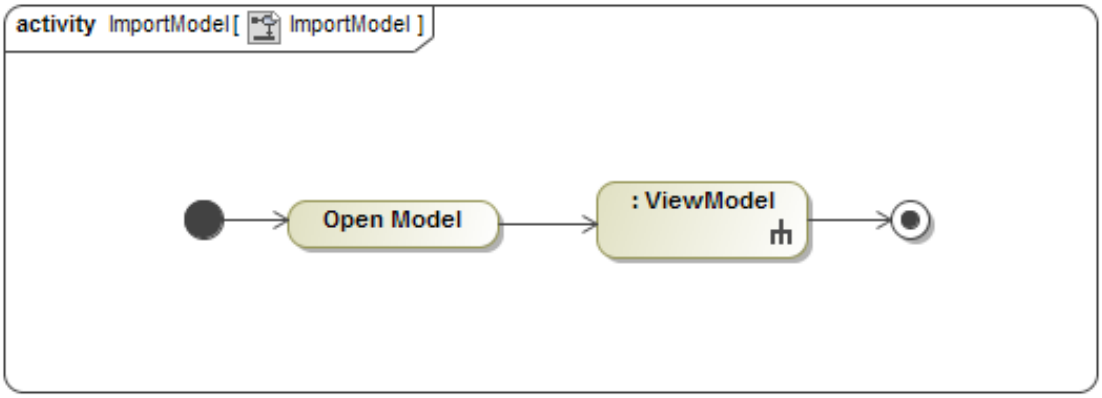

Figure 24: Import 3D Model Activity

2. ViewModel: Once the model is imported the model is rendered, this will be available for viewing and exploration and further processing decided by the user.

   - Pre-condition: 3D model must have been imported and rendered.
   - Post-condition: The 3D model render is explorable and can be recorded on.

- Request Data Structure: The 3D models render.
- Return Data Structure: Viewable 3D model render.

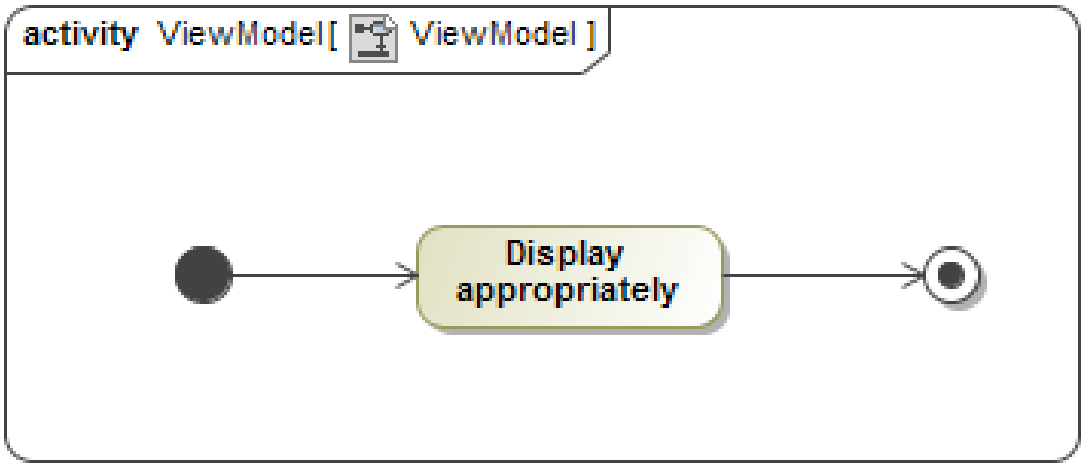

Figure 25: View 3D Model Activity

3. ExportHeatMapModel: The exported model will have the heat-map applied to it which can then be exported as a new videp of the 3D model exploration so that it can be viewed or used at a later date by the user.
    - Pre-condition: Heat-map must be created and Model Imported.
    - Post-condition: Exported model With heat-map on it to be viewed.
    - Request Data Structure: The eye tracking data must have been recorded and 3D model available
    - Return Data Structure: Model with heat-map applied.
4. Export3DModelFlythroughVideo: The model will have a render created. These render can then be explored and used for recording.
    - Pre-condition: Selected model must be available.
    - Post-condition: Render will be created of model.
    - Request Data Structure: The selected model.
    - Return Data Structure: Render of 3D model.

2.5.3 2D Models

2D models will typically refer to images but can also refer to any media that is does not show a third dimension of movement but for the scope of this project will only refer to images. The heat-map can then be created and applied to the image.

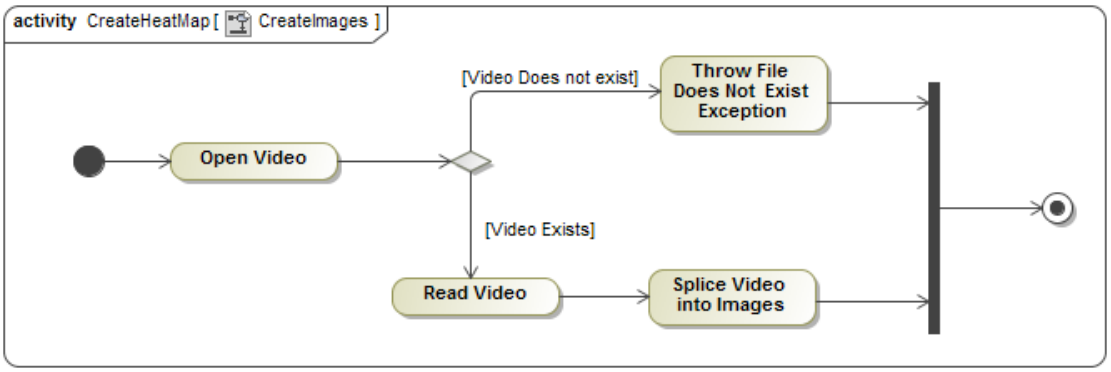

Figure 26: Create 3D Model Images Activity

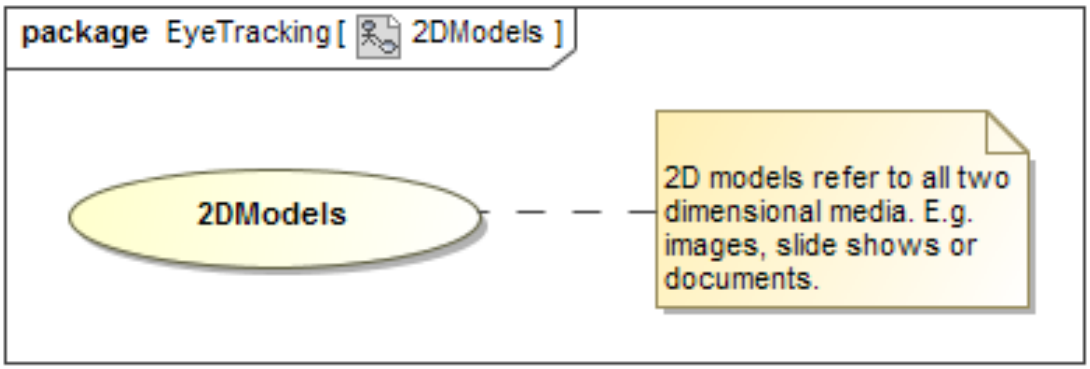

Figure 27: 2D Models Use Case

1. ImportModel:
   The 2D models can be imported into the system which can then be process further by the user. The imported model can then have recordings done and heat maps created for it.

   - Pre-condition: The 2D model must exist, a image preferable.
   - Post-condition: Recording can be done on the model.
   - Request Data Structure: The desired 2D model.
   - Return Data Structure: Rendered image is imported.

2. ViewModel: Once the model is imported the users' can view the model and eye tracking can then be done on the model. The model can the be processed further as the user sees fit.

   - Pre-condition: The 2D model (image) must have been imported.
   - Post-condition: Image is viewable and can be recorded on.
   - Request Data Structure: The selected model image.
   - Return Data Structure: Viewable image.

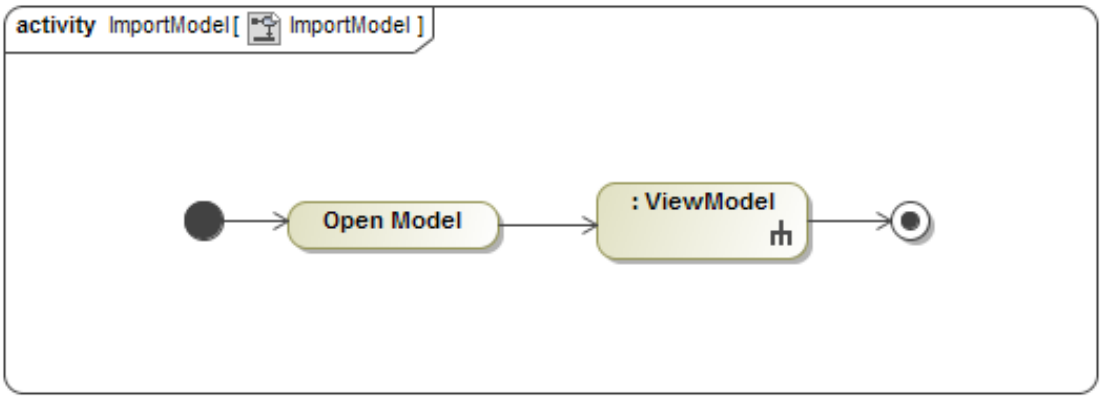

Figure 28: Import 2D Model Activity

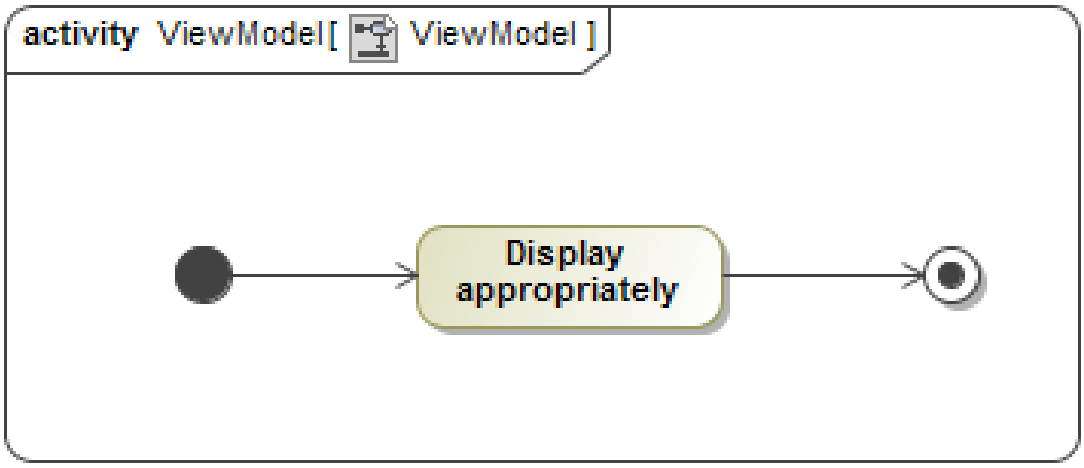

Figure 29: View 2D Model Activity

3. ExportHeatMapModel: The exported model will have the heat map applied to it which can then be exported as a new 2D model so that it can be viewed or used at a later date by the user.

    - Pre-condition: The eye data must have been recorded and model imported.
    - Post-condition: Exported model with heat-map on it to be viewed.
    - Request Data Structure: Recorded eye data and imported model.
    - Return Data Structure: Model with heat map.

2.5.4 Video Models

A video can be easily imported into the system. Eye tracking can then be done to the video model and the recorded data can be stored. The data can then be used to create a heat-map that can then be applied to the video second by second.

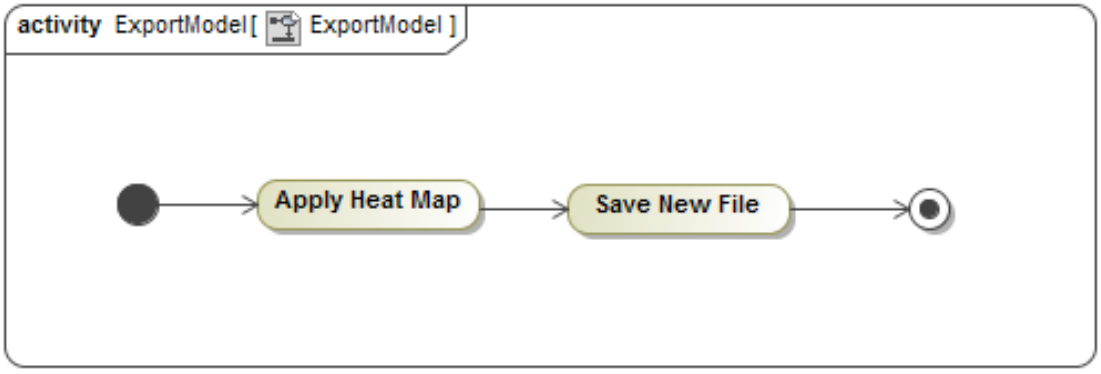

Figure 30: Export 2D Model Activity

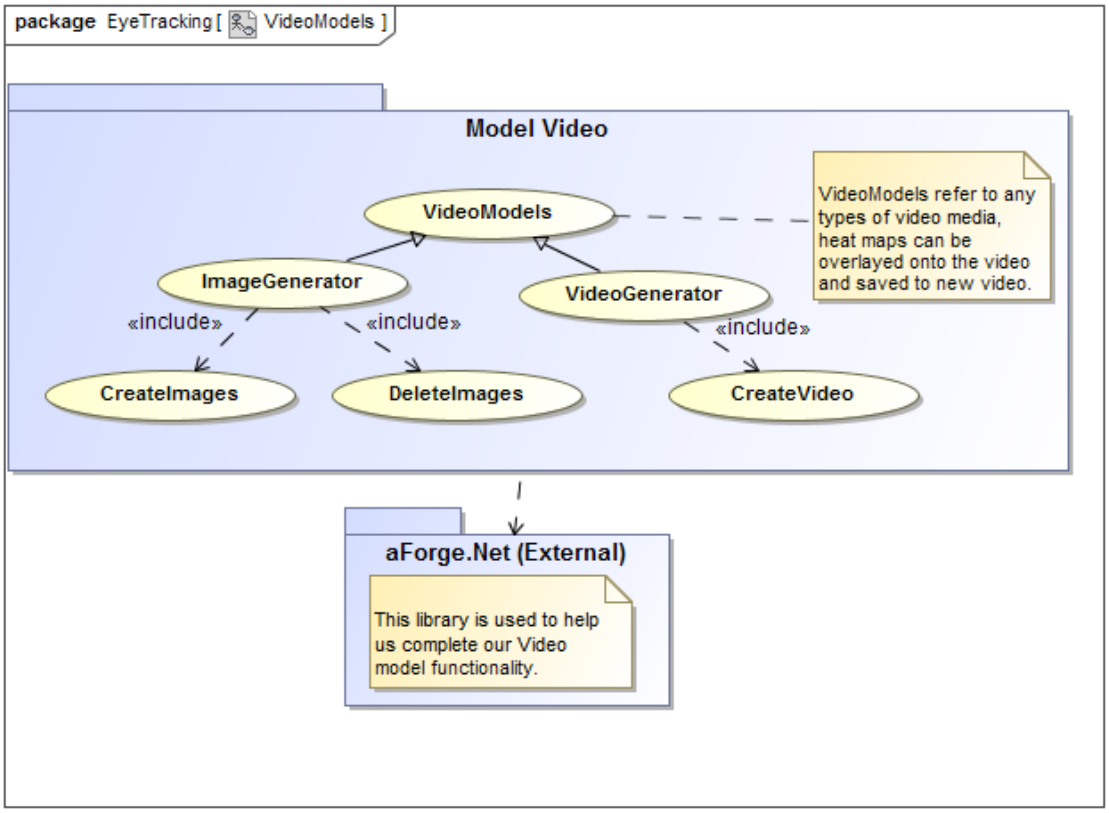

Figure 31: Video Models Use Case

1. ImportModel:
   The video models can be imported into the system which can then be processed further by the user. The imported model can then have recordings done and heat maps created for it. Heat maps that will be recorded will show a second by second representation of the heat map data.

   - Pre-condition: Desired video must exist.
   - Post-condition: Recording can be done on the video.
   - Request Data Structure: The video model.

- Return Data Structure: Rendered video is imported.

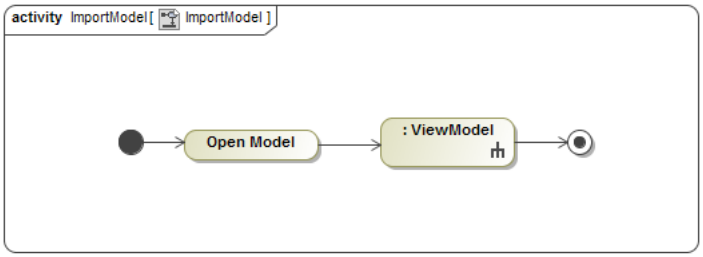

Figure 32: Import Video Model Activity

2. ViewModel: Once the model is imported the users' can view the model and eye tracking can then be applied on the model. The model can the be processed further as the user sees fit.

   - Pre-condition: Desired video model must be available.
   - Post-condition: Video can be played and recordings taken.
   - Request Data Structure: Requested video.
   - Return Data Structure: Playable video.

3. ExportHeatMapModel: The exported model will have the heat-map applied to it which can then be exported as a new video model so that it can be viewed or used at a later date by the user. The exported video model will have the heat map recorded onto the original video and saved in the appropriate format.

   - Pre-condition: Eye tacking data must have been recorded and desired video imported.
   - Post-condition: New video with heat-map overlay will be saved..
   - Request Data Structure: Eye tracking data as well as the desired model.
   - Return Data Structure: Video with heat map is saved.

2.6 Raw Information

When recording happens on the system the raw information is saved. The raw information is important as it forms the basis for creating heat maps and statistics for specific models. The raw information is can be saved on the system and processed further by the user. The raw information can then be read as the raw data or put into a statistical report.

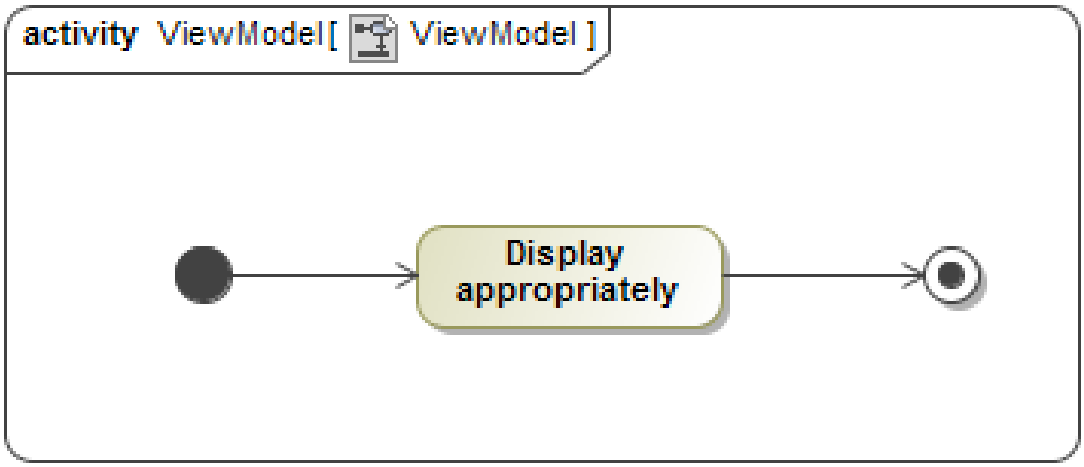

Figure 33: View Video Model Activity

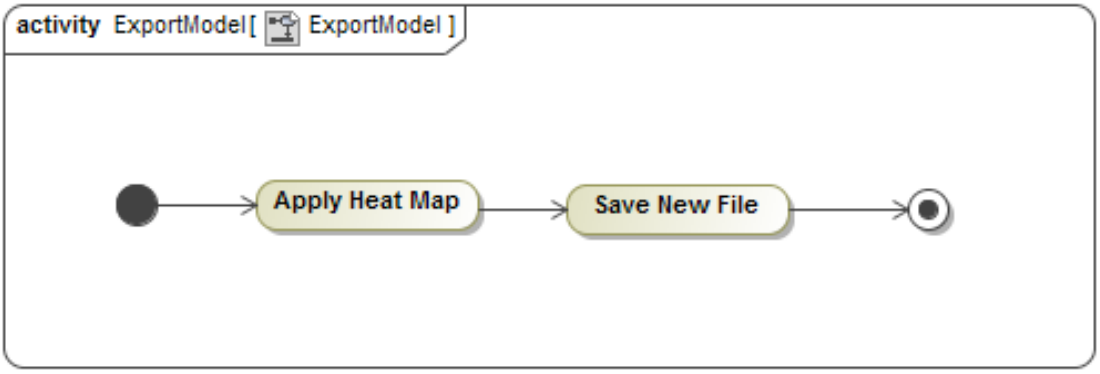

Figure 34: Export 3D Model Activity

2.6.1 SaveRawInformation

When recording is done on the media the raw information will need to be saved so that it can be used at a later stage. This raw information can then be reopened later for the relevant model and used by user for further processing and interpretation.

- Pre-condition: Recording on a media type must have already taken place.
- Post-condition: Data is saved and can be used later.
- Request Data Structure: Incoming x- and y-axis data coming form the Eye Tribe recording server.
- Return Data Structure: The recorded eye tracking data text file is created.

2.6.2 ReadRawInformation

Data can be read from a previously saved file. This will allow the user to import the raw information for the relevant model which can then be used for comparison or further

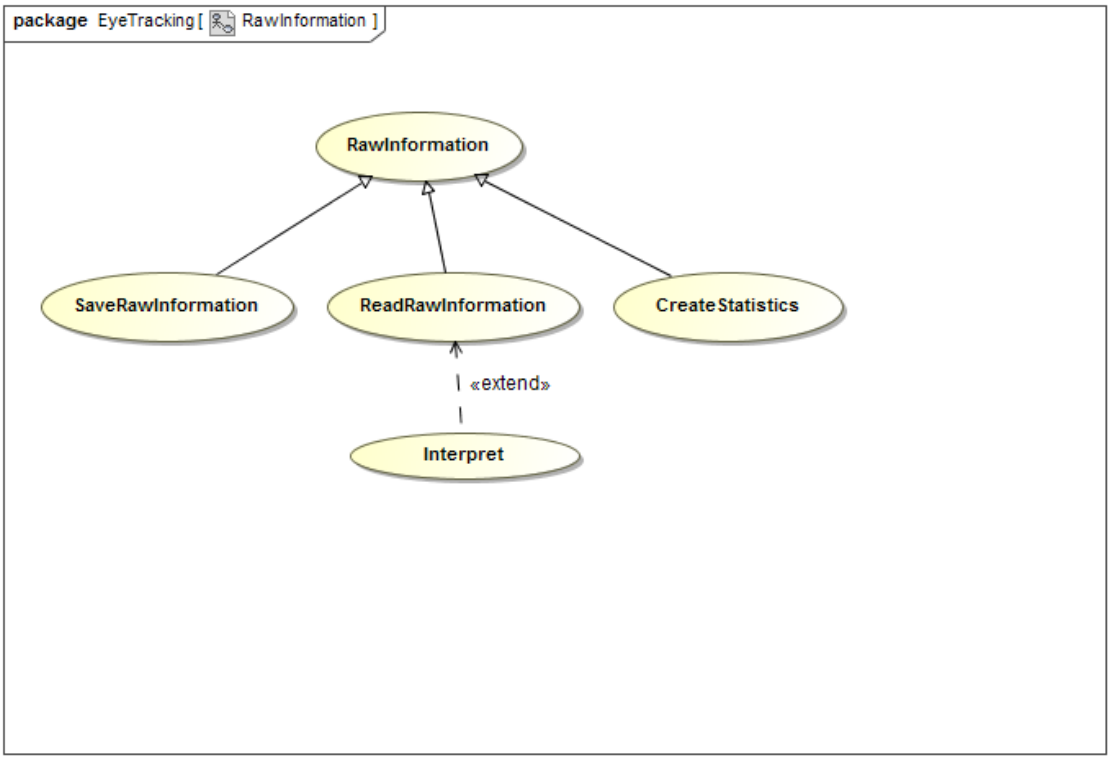

Figure 35: Raw Information Use Case

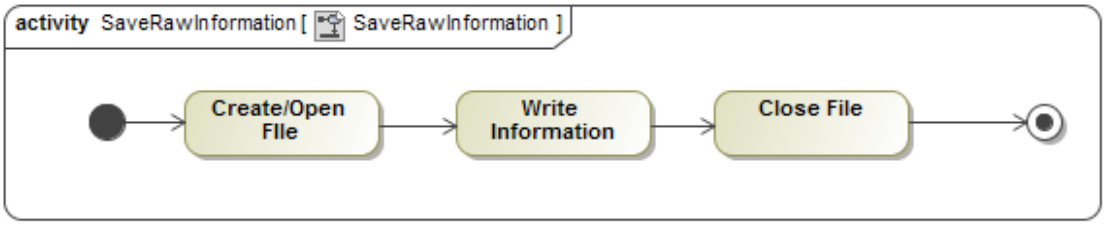

Figure 36: Save Raw Information Activity

processing.

- Pre-condition: Recorded data must have been saved in a text file.
- Post-condition: Data can now be used to create heat-maps and statistics.
- Request Data Structure: The recorded eye tracking data text file.
- Return Data Structure: Data array with all data points that can then be further processed by application.

2.6.3 CreateStatistics

The writing of statistics is part of the Statistics use-case and all information can be found there regarding the sub use case

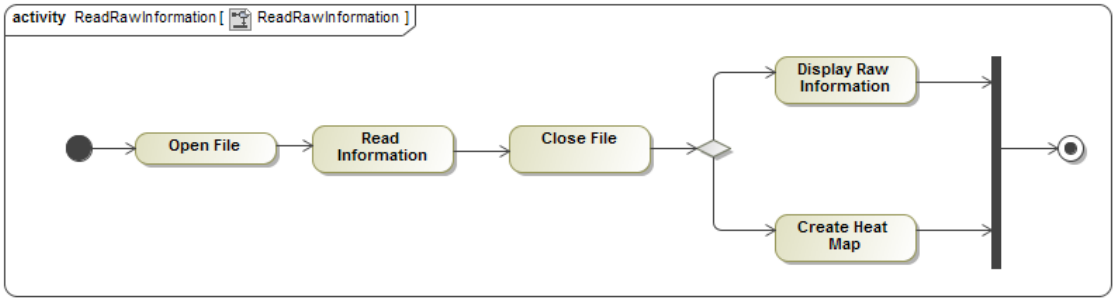

Figure 37: Read Raw Information Activity

2.7 Statistics

Statistics on all the data will be collected and will then be used to make a statistical page based off the model that can be analysed and then it can be easily compared at any time.

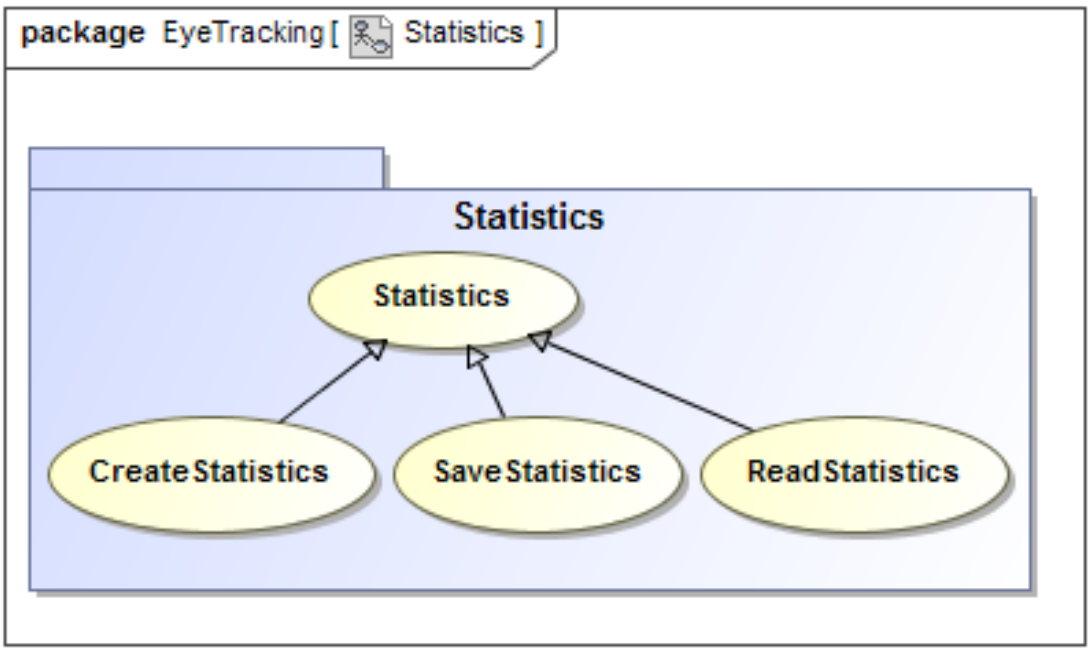

Figure 38: Statistics Use Case

2.7.1 CreateStatistics

Using the recorded data we can create statistical data which can then be used for analysis. This data can then be viewed or used to compared data.

- Pre-condition: Recorded data must exist.
- Post-condition: Data is turned into statistical data.

- Request Data Structure: Text file containing recorded data.
- Return Data Structure: Statistics will be created.

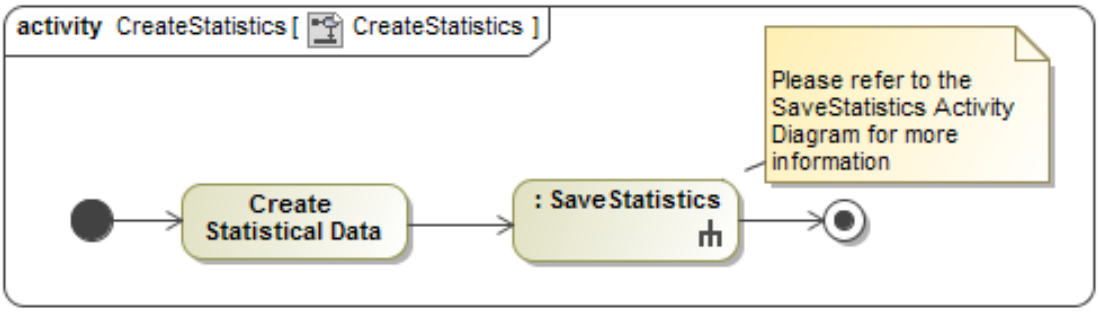

Figure 39: Create Statistics Activity

2.7.2 SaveStatistics

The statistical data can be saved into a file so that they can be printed out and be used further by users'. This will take all created statistical data and save it into a pdf, excel or csv file.

- Pre-condition: Statistical data must exist.
- Post-condition: Data is put into a report.
- Request Data Structure: Created statistical data.
- Return Data Structure: Save report in pdf, excel or csv format.

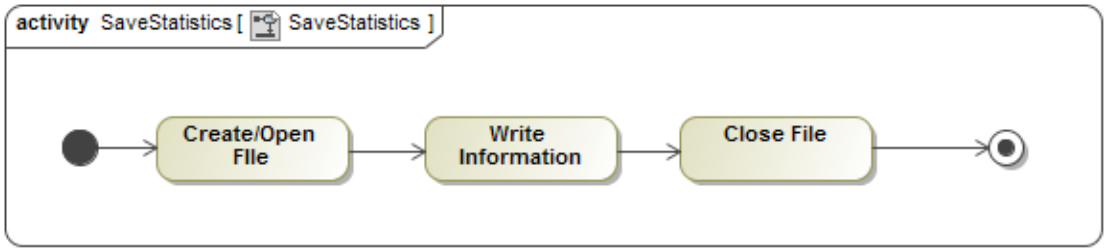

Figure 40: Save Statistics Activity

2.8 Record Eyes

The recording of the eyes will be done using the Eye Tribe camera or any other (compatible) eye tracking camera. The data recorded in this process will allow heat maps and statistics to be generated. The heat maps can then be turned into overlays for

the appropriate media types. The record eye use case (Figure 41) makes use of sub use cases from other use cases that are listed below, such as SaveRawInformation. The recording of the information is important as this is used throughout the project and its functionality.

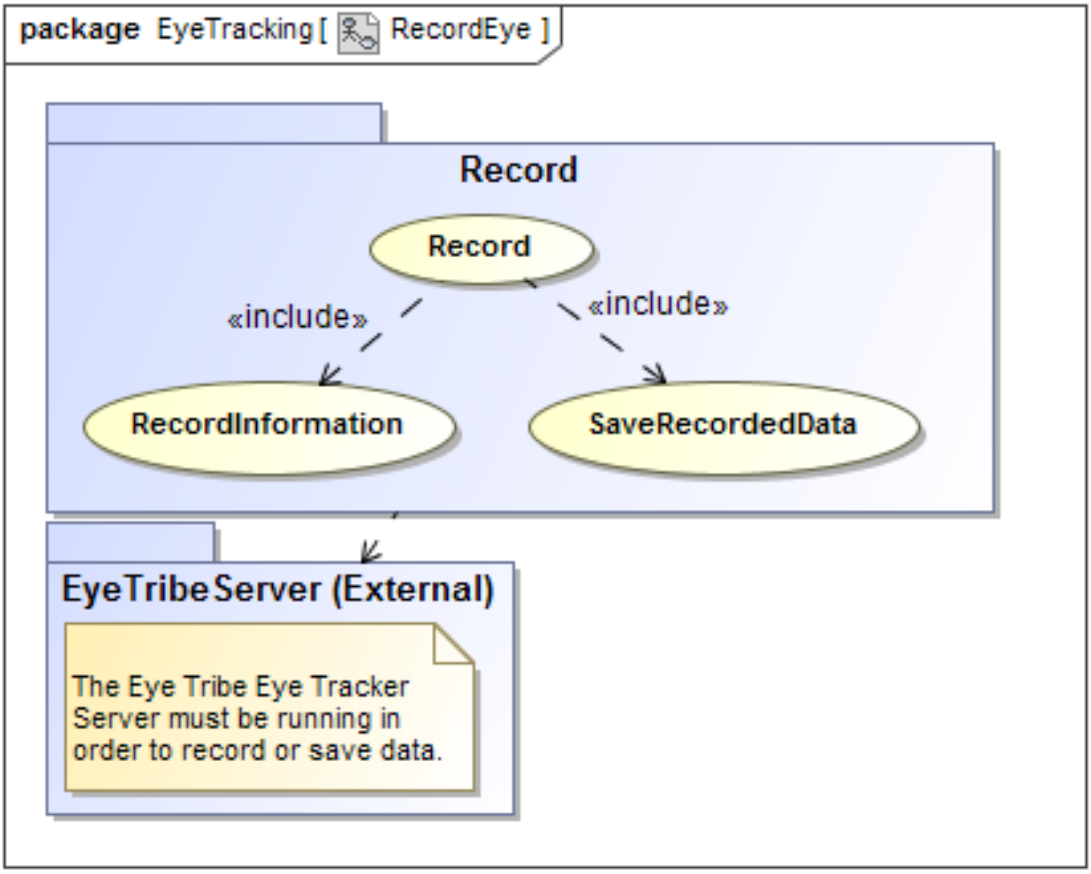

Figure 41: Record Eye Use Case

2.8.1 RecordInformation

The data will be recorded and saved into a text file which can be further processed and interpreted, which will then return the relevant information which can then be used to carry out functions in the system.

- Pre-condition: Calibration must have been previously completed and user must be viewing the specified media type, so that recording of data can take place.
- Post-condition: Data is recorded about the eye movements and saved by another function.
- Request Data Structure: N/A.
- Return Data Structure: Raw data in x and y co-ordinates will be returned.

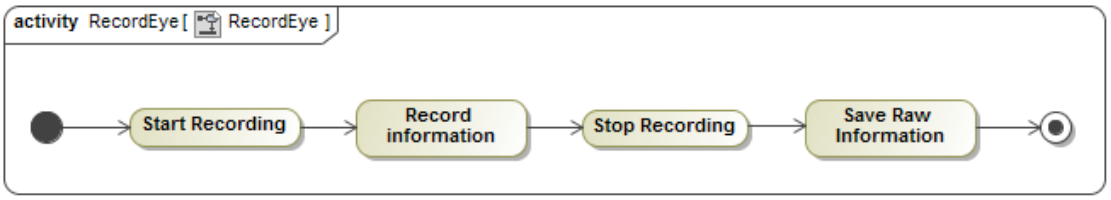

Figure 42: Record Eye Activity

2.8.2 3D models and graphics

Our system required a three dimensional conversion and rendering module to be created. We created this module by making use of OpenTK software and libraries. We use the created modules to take in an "OBJ" file (".obj") and parse it into a structure we can use to begin rendering the model. As multiple different three dimensional objects models can be imported all with different structures and objects, we make use of the builder design pattern (described further in the architectural specification) to separate the construction of each game-object so that each object can be built dynamically based on its requirements. A three dimensional object consists of a buffer, vertex buffer objects, materials and a list of children objects. When parsing the parsing the ".obj" file all objects are created, followed by the parsing of the ".mtl" file where all materials are created and attached to the objects. The root object and its tree structure is then generated and returned.

- Pre-condition: 3D Model file format must be that of an OBJ (".obj").
- Post-condition: model can be converted and rendered.
- Request Data Structure: Object file.
- Return Data Structure: Model images with be created or model will be rendered.

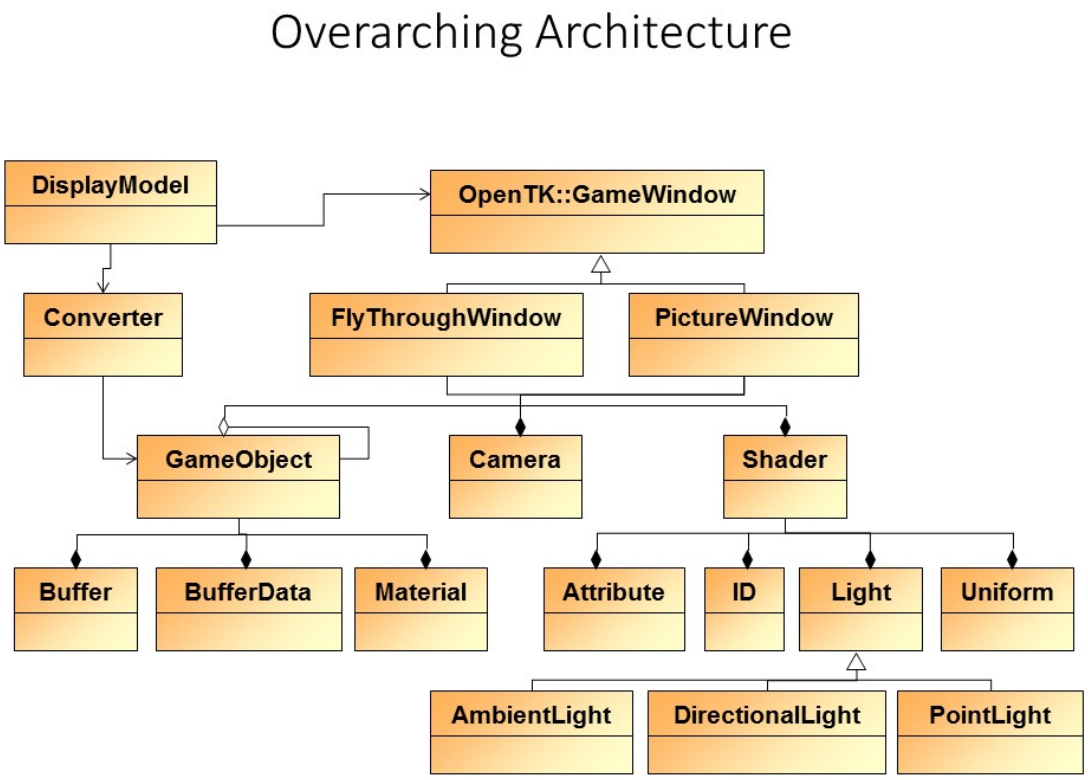

Figure 43: Record Eye Activity

3 Required Functionality

3.1 Render 3D scene

Priority: Critical
Given a 3D scene, in a format such as a BLEND file or an OBJ file, the scene must be able to be rendered to a display. The rendered scene will be placed in a window for viewing so that the eye tracker can "see" the scene.
Pre-condition: Open the 3D rendering software with the 3D scene file.
Post-condition: Have a window open that has the scene loaded and rendered. Images of this scene can then be taken for later viewing and recordings.

3.2 Eye-tracking on a 3D scene

Priority: Critical
The eye tracking software must be used or augmented in such a way that specific areas in the 3D scene can capture viewing information. This is will be done given information from the eye tracker itself.
Pre-condition: Have Neo Tandem Technology eye tracking software and eye tracking equipment set-up and running.
Post-condition: Captured information is written into a file.

3.3 Create heat mapped texture

Priority: Critical
From the eye tracking information a heat map in the form of a texture map must be produced. The format of the texture will depend on the 3D software used.
Pre-condition: Have captured information from Neo Tandem Technology eye tracking software and eye tracking equipment.
Post-condition: Captured information can be used to create a heat-map image.

3.4 Map heat mapped texture back onto scene

Priority: Critical
The heat mapped texture of the object must be able to be mapped correctly back onto the scene, that is the placement must be as was the original texture.
Pre-condition: Have heat mapped texture..
Post-condition: 3D scene now has new heat mapped texture applied.

3.5 Real-time heat mapping

Priority: Nice-to-have
As the scene is being viewed it will also be coloured according to the information gathered by the eye tracking device during the recording.
Pre-condition: Have Neo Tandem Technology eye tracking software and eye tracking equipment set-up and running.
Post-condition: Captured information used to colour the scene.

3.6 Interactive Maps

Priority: Important
This function involves overlaying a map with data captured by the Eye tribe eye tracking technology and stored using the Neo Tandem Technology eye tracking software. The data will be retrieved from a database which has already been integrated into the Neo Tandem Technology eye tracking software that will be used for this project. Through the use of the heat-map creation provided by the Heatmap.net library, its functionality will be extended into the visualization of the information it has stored. This includes the attention maps, scan paths and replay functions.
Pre-condition: Have the Eye tribe eye tracking technology running during the activity.
Post-condition: Output a 2D representation on the interactive map with the necessary data overlaying it.

3.7 In-Video Eye Tracking

Priority: Nice-to-have
This function overlays video frames with eye tracking attention maps while watching a

video.
Pre-condition: Have the Eye tribe eye tracking technology running during the activity.
Post-condition: Run time editing of video frames overlaid with attention maps and scan paths.

3.8 Post-video Eye Tracking

Priority: Nice-to-have
This function involves rendering a video with an attention map and scan path overlaid after it has been watched. This function is more achievable than the In-video eye tracking because the Neo Tandem Technology eye tracking software is able to record eye gaze and movement on image slide shows thus it can be used on a slide show of frames from the video to render a video coupled with this analysed data overlaying it.
Pre-condition: Have the Eye tribe eye tracking technology running during the activity.
Post-condition: Output a video with its frames overlaid with attention maps and scan paths.

3.9 Suitable Video Output Formats

Priority: Nice-to-have
This function aims allow the post-video eye tracking video to be outputted in more than one video format. The default video output provided by aForge.net is AVI, but it is possible to output videos in more popular formats such as MP4, WMV, MKV etc. This allows the product to be used by many more devices, some of which have limited memory or only support specific video formats.
Pre-condition: Video should be created and ready for saving.
Post-condition: Output a video in the format that the user has selected.

4 Domain Model

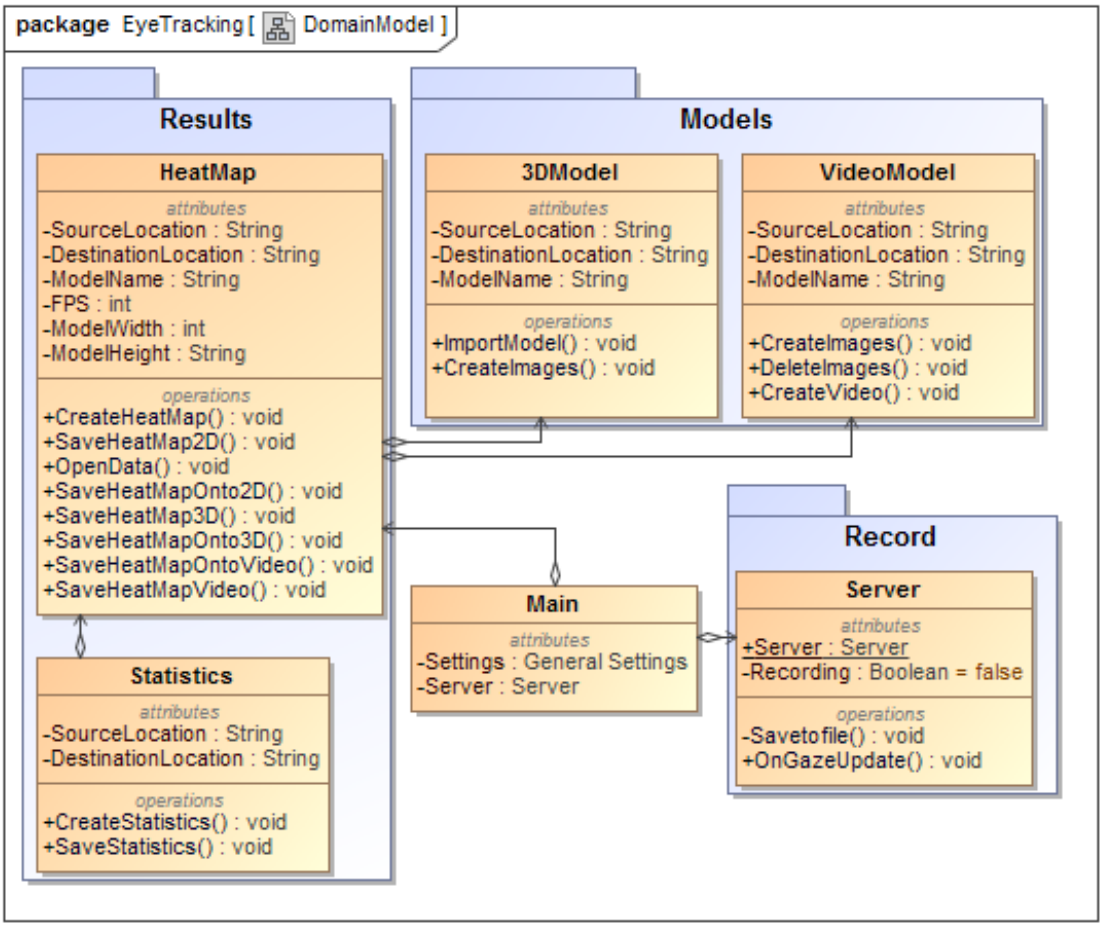

Figure 44: Domain Model