Willburg Outdoor PTY(ltd.)

Smart Image Identifier

Version 1.0

Application Requirements Specifications and Design

Author:	
Stephen Swanepoel	u11032091
Dian Veldsman	u12081095

September 12, 2016

1 Background

Willburg Outdoor is company that is passionate about South Africa. With this passion comes the need to protect its farmers and it animals from those who intend to harm them. The client intends to use the system to assist in fighting the following problems: animal poaching and farm attacks. Willburg provides its' clients with a camera system which is interfaced with a dashboard as well as a mobile application. The cameras snap pictures when movement is detected and sends them to the Willburg server. The images are then pushed to the respective users. Willburg intend on providing its' users with real-time push notifications when a human has been detected in an image captured by a camera on their property. Smart Image Identifier will play an influential role in the prevention of both farm attacks as well as poaching.

1.1 Smart Image Identification System

The Smart Image Identifier will consist of 4 modules:

- Retrieve Image.
- Image Manipulation.
- Test Image.
- Response.

The above mentioned module will consist of the following:

1.1.1 Retrieve Image

This module consists of:

- Retreiving an image from the Willburg server.
- Assign image for testing.

1.1.2 Image Manipulation

This module consists of:

- Resizing of images.

1.1.3 Test Image

This module consists of:

- Human detection.

1.1.4 Response

This module consists of:

- Sending a response back to the Willburg server for further processing.

2 Vision and Objectives

2.1 Vision

The client for this project, Willburg Outdoor PTY(ltd.), has called for the design of an application that will assess an image and identify any human(s) present in the image. If a human has been identified by the system, a response will be sent to the Willburg server afterwhich a notification is pushed through to the user. The main idea behind the project is to be able to alert users of Willburg's software that a human has been identified and action can be taken if there is an intruder on the premesis.

2.2 Objectives

Objectives of the Smart Image Identifier are:

- Identify the presence of a human in the image.
- Assesses, store and sort images into a database based on their content.

3 Overview of Smart Image Identifier

3.1 UML Diagram

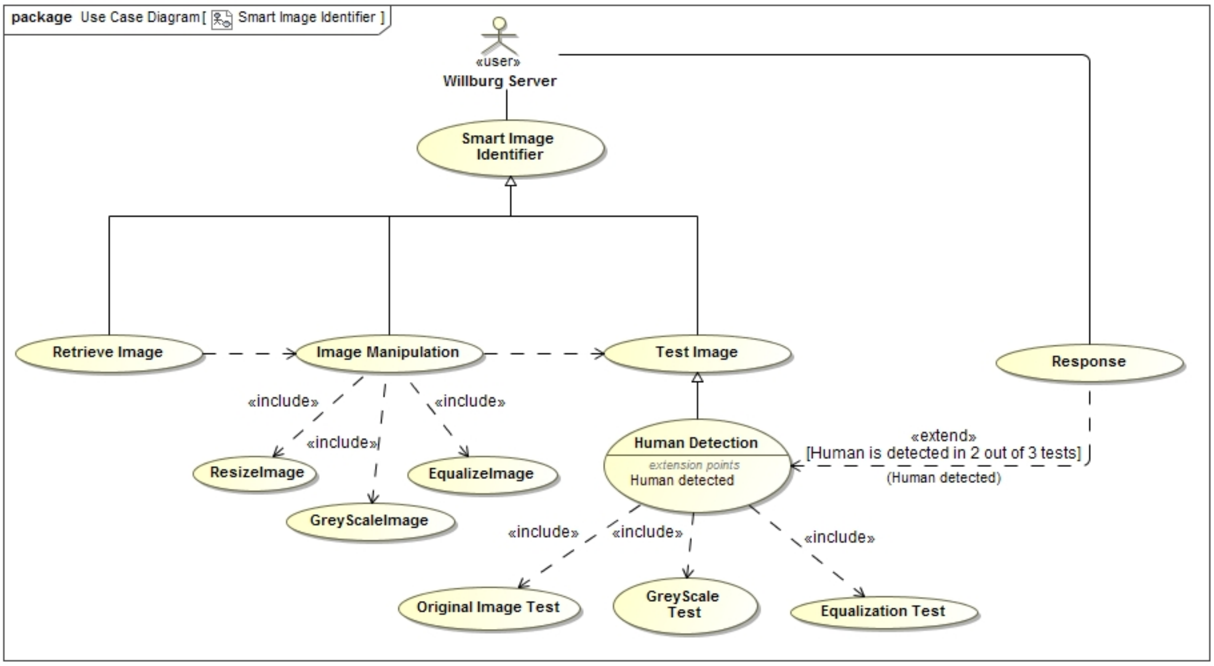
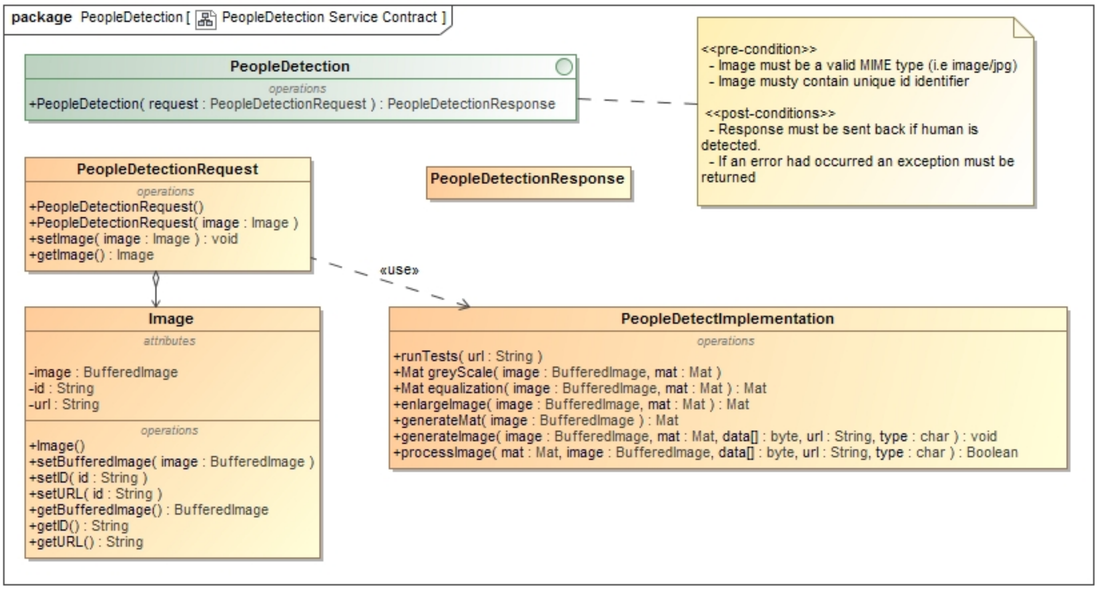


Figure 1: UML diagram of Smart Image Identifier

3.2 Service Contract


package PeopleDetection [PeopleDetection Service Contract]
PeopleDetection
operations
+PeopleDetection(request : PeopleDetectionRequest) : PeopleDetectionResponse
PeopleDetectionRequest
operations
+PeopleDetectionRequest()
+PeopleDetectionRequest(image : Image)
+setImage(image : Image) : void
+getImage() : Image
PeopleDetectionResponse
«use»
Image
attributes
-image : BufferedImage
-id : String
-url : String
operations
+Image()
+setBufferedImage(image : BufferedImage)
+setID(id : String)
+setURL(id : String)
+getBufferedImage() : BufferedImage
+getID() : String
+getURL() : String
PeopleDetectImplementation
operations
+runTests(url : String)
+Mat greyScale(image : BufferedImage, mat : Mat)
+Mat equalization(image : BufferedImage, mat : Mat) : Mat
+enlargeImage(image : BufferedImage, mat : Mat) : Mat
+generateMat(image : BufferedImage) : Mat
+generateImage(image : BufferedImage, mat : Mat, data[] : byte, url : String, type : char) : void
+processImage(mat : Mat, image : BufferedImage, data[] : byte, url : String, type : char) : Boolean

<<pre-condition>>
- Image must be a valid MIME type (i.e image/jpg)
- Image musty contain unique id identifier

<<post-conditions>>
- Response must be sent back if human is detected.
- If an error had occurred an exception must be returned

Figure 2: Service contract of Smart Image Identifier

3.3 Domain model

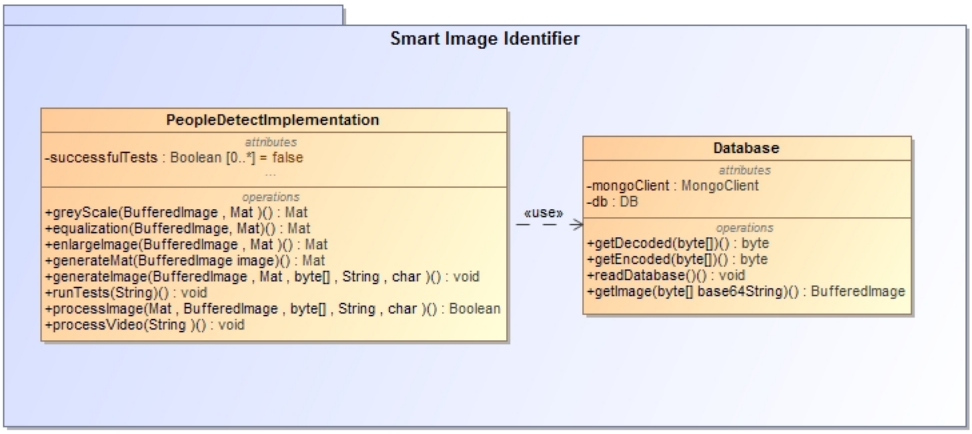

Smart Image Identifier
PeopleDetectImplementation
attributes
-successfulTests : Boolean [0..*] = false
...
operations
+greyScale(BufferedImage , Mat)() : Mat
+equalization(BufferedImage, Mat)() : Mat
+enlargeImage(BufferedImage , Mat)() : Mat
+generateMat(BufferedImage image)() : Mat
+generateImage(BufferedImage , Mat , byte[] , String , char)() : void
+runTests(String)() : void
+processImage(Mat , BufferedImage , byte[] , String , char)() : Boolean
+processVideo(String)() : void
«use»
Database
attributes
-mongoClient : MongoClient
-db : DB
operations
+getDecoded(byte[])() : byte
+getEncoded(byte[])() : byte
+readDatabase()() : void
+getImage(byte[] base64String)() : BufferedImage

Figure 3: Domain model of Smart Image Identifier

4 Retrieve Image

The Retrieve Image Service module is responsible for pulling images from the Willburg server for the system to process. The module will retrieve these image using an port connection to the server. Once these images are retrieved they are assigned for testing.

4.1 Scope

The scope for the Retrieve Image Service module is shown in Figure4.

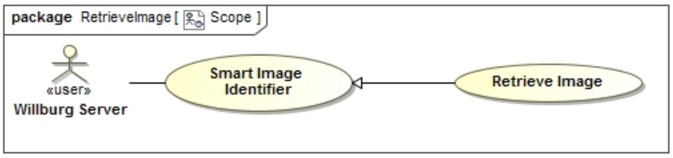


Figure 4: Retrieve Image scope

The scope of the Retrieve Image module include:

- Retrieve image from the Willburg server.
- Assign image for testing.

4.2 Retrieve Image service model conditions

4.2.1 Pre-conditions

- System must have a valid connection to the server.
- Images must be of valid MIME type image

4.2.2 Post-Conditions

- Image is pulled from server and assigned for processing.

If there is an invalid connection to the to the server a NoConnection exception is thrown. If an error occurs during retrieval of the image an IOException is thrown.

4.3 Functional requirements

The lower level services required by the retrieve image service to either check the pre-conditions or address the post-conditions is shown 2.

4.4 processDesign

The images are retrieved from the Willburg server in a base64 encoded string. The images are then decoded into a BufferedImage object. A Image object is created to store the id of the image and the BufferedImage object. The Image object is then sent as request to identify the presence of a human using the PeopleDetectionRequest object.

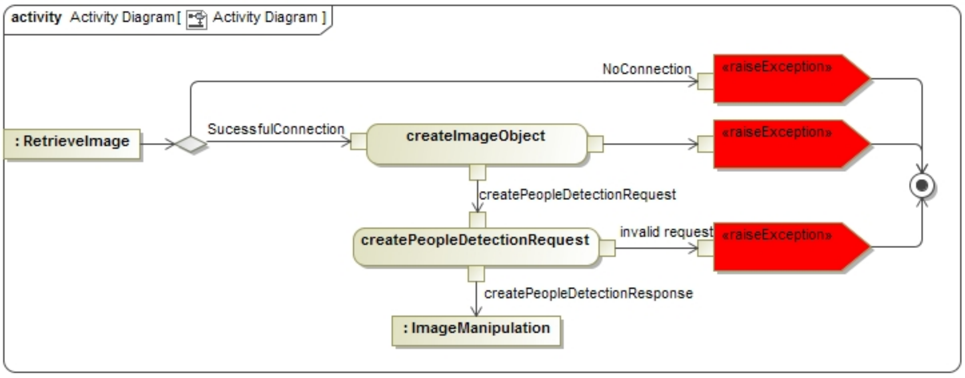


Figure 5: Process image process design

5 Image Manipulation

The Image Manipulation Service module is responsible for manipulating images inorder to better identify human. The module will manipulate each image by resizing the image, grayscaling and equalization. Once these images are manipulated they are then assigned for testing.

5.1 Scope

The scope for Image Manipulation Service module is shown is Figure 6.

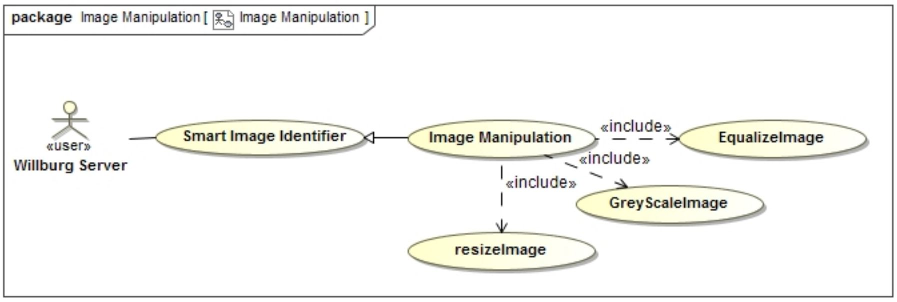


Figure 6: Image Manipulation scope

5.2 processDesign

The BufferedImage data is retrieved from the Image object, that was sent through in the request. The BufferedImage is then used to create three Mat obejcts. The first mat object is the original image itself. This mat object isthen resized and used to create the last two mat object. The second mat object is a grayscaled mat object and the third is the grayscaled mat object that was equalised. After these three mat objects are created they are sent on to the Test Image Service model for testing.

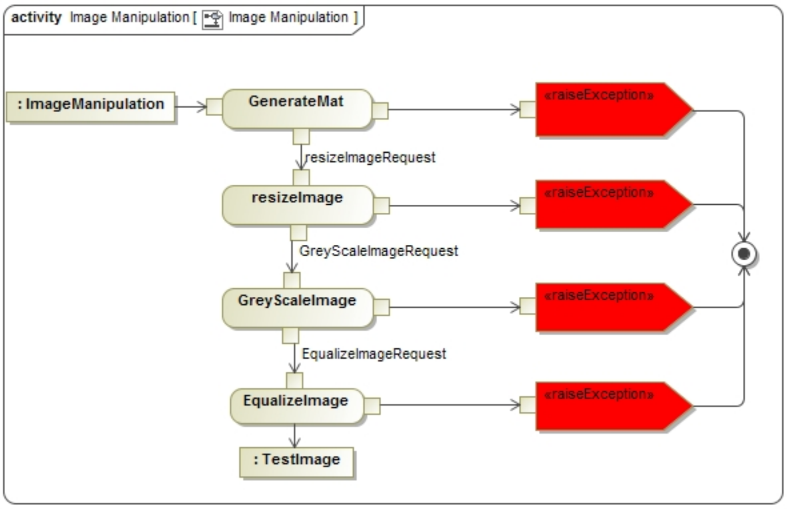


Figure 7: Image Manipulation process design

6 Test Image

The Image Manipulation Service module is responsible for human detection. The module will use the manipulated image and the prerform human detection. The module will use the manipulated mat objects to perform three individual tests. These tests include performing human detection on the original image, grayscaled image and the equalised image.

6.1 Scope

The scope for Test Image Service module is shown is Figure 8.

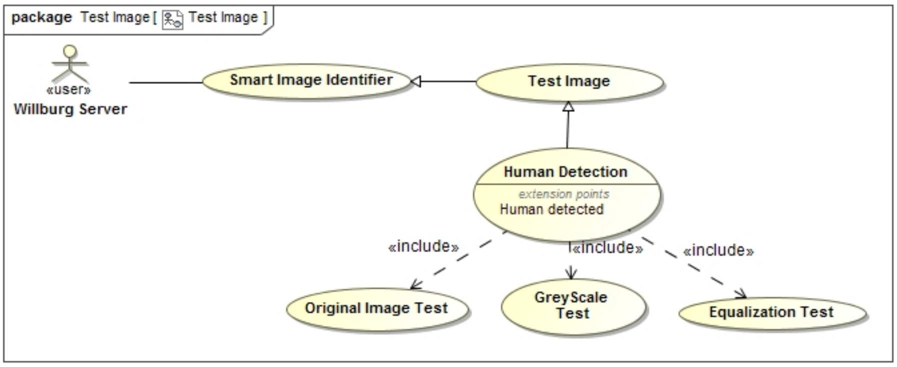


Figure 8: Test Image scope

6.2 processDesign

The three manipulated mat objects are used to in a three phase test system inorder to improve the accuracy of human detection. Humand detection will be performed on each manipulated image. If two out of the three test are positive then the system has detected a human and a reponse will be sent to the server.

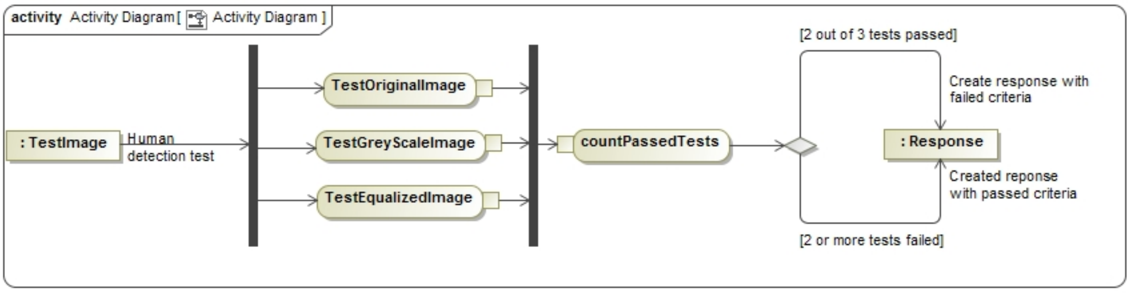

activity Activity Diagram [Activity Diagram]
: TestImage
Human detection test
TestOriginalImage
TestGreyScaleImage
TestEqualizedImage
countPassedTests
[2 out of 3 tests passed]
Create response with failed criteria
: Response
Created reponse with passed criteria
[2 or more tests failed]

Figure 9: Test Image process design

7 Response

The Response Service module is creating a response object to send to the Willburg server.

7.1 Scope

The scope for Response Service module is shown is Figure 10.

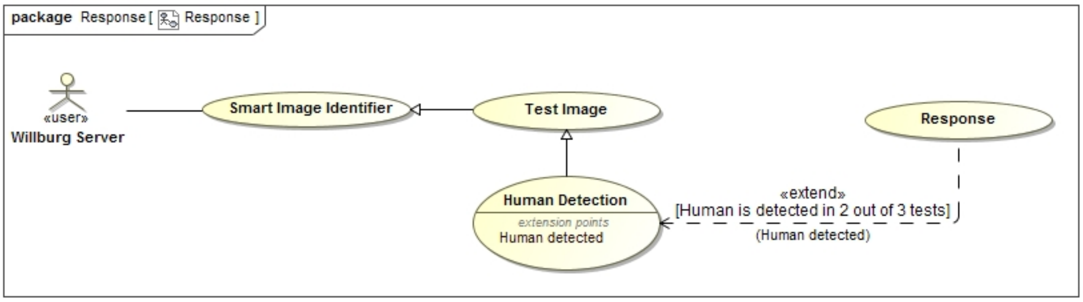


Figure 10: Response scope

7.2 processDesign

A response object is created and sent back to the server.

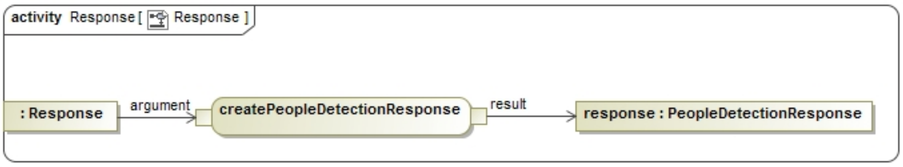

activity Response [Response]
: Response
argument
createPeopleDetectionResponse
result
response : PeopleDetectionResponse

Figure 11: Response process design