# UNIVERSIDAD NACIONAL DE LOJA

*Área de la Energía, las Industrias y los Recursos Naturales No Renovables*

CARRERA DE INGENIERÍA EN SISTEMAS

**"Aplicación móvil para la detección de somnolencia de un conductor aplicando visión artificial"**

**Tesis previa a la obtención del título de Ingeniero en Sistemas**


**Autor:**

- Yeferson-Mauricio, Torres-Berrú

**Tutor:**

- Ing. Mario Andrés Palma Jaramillo. Mg. Sc


**LOJA-ECUADOR**

**2015**

## Certificación del Director

Ing. Mario Palma Jaramillo, Mgs

CERTIFICA

Haber dirigido, revisado y corregido en todas sus partes el desarrollo del Trabajo de Titulación de Ingeniería en Sistemas titulado: **"Aplicación móvil para la detección de somnolencia de un conductor aplicando visión artificial"**, con autoría del egresado Yeferson Mauricio Torres Berrú. En razón de que la misma reúne a satisfacción los requisitos de fondo y forma, exigidos para la investigación de éste nivel, autorizo su presentación, sustentación y defensa ante tribunal designado para el efecto.

Loja, 21 de diciembre del 2015.

.....

Ing. Mario Andrés Palma Jaramillo, Mgs

**DIRECTOR DEL TRABAJO DE TITULACIÓN.**

## Autoría.

Yo, **YEFERSON MAURICIO TORRES BERRÚ**, declaro ser autor del presente trabajo de tesis y eximo expresamente a la Universidad Nacional de Loja y a sus representantes jurídicos de posibles reclamos o acciones legales por el contenido de la misma.

Adicionalmente acepto y autorizo a la Universidad Nacional de Loja, la publicación de mi tesis en el Repositorio Institucional – Biblioteca Virtual.

**Firma:** .....

**Cédula:** 1104759665

**Fecha:** 21 de diciembre del 2015.

Aplicación móvil para la detección de somnolencia de un conductor aplicando visión artificial | 2015

**CARTA DE AUTORIZACIÓN DE TESIS POR PARTE DEL AUTOR, PARA LA CONSULTA, REPRODUCCIÓN PARCIAL O TOTAL Y PUBLICACIÓN ELECTRÓNICA DEL TEXTO COMPLETO.**

Yo, **YEFERSON MAURICIO TORRES BERRÚ**, declaro ser autor de la tesis titulada: **APLICACIÓN MÓVIL PARA LA DETECCIÓN DE SOMNOLENCIA DE UN CONDUCTOR APLICANDO VISIÓN ARTIFICIAL**, como requisito para optar el grado de **INGENIERO EN SISTEMAS**; autorizo al Sistema Bibliotecario de la Universidad Nacional de Loja para que con fines académicos, muestre al mundo la producción intelectual de la Universidad, a través de la visibilidad de su contenido de la siguiente manera en el Repositorio Digital Institucional:

Los usuarios pueden consultar el contenido de éste trabajo en el RDI, en las redes de información del país y del exterior, con las cuales tenga convenio la Universidad.

La Universidad Nacional de Loja, no se responsabiliza por el plagio o copia de la tesis que realice un tercero.

Para constancia de ésta autorización, en la ciudad de Loja, a los diez y nueve días del mes de diciembre del dos mil quince.

**Firma:** .....

**Autor:** Yeferson Mauricio Torres Berrú

**Cédula:** 1104759665

**Dirección:** Catamayo (24 de Mayo e Isabel Arias)

**Correo Electrónico**: ymtorresb@unl.edu.ec

**Teléfono:** 2 678 203 **Celular:** 09946682012

**DATOS COMPLEMENTARIOS**

**Director de Tesis:** Ing. Mario Andrés Palma Jaramillo, Mg. Sc

**Tribunal de Grado**: Ing. Pablo Fernando Ordóñez Ordóñez, Mg. Sc.

Ing. Waldemar Victorino Espinoza Tituana, Mg. Sc.

Ing. Ana Lucía Colala Troya, Mg. Sc.

# Dedicatoria

Dedico el presente trabajo a mis padres (Erguin y Rosana) a mis hermanos (Leandro, Maria y Katia) a mis tios (Byron y Maricel) y sobre todo a Monica Coronel por haber estado siempre ahí.

# Agradecimiento

Agradezco a Dios por permitirme conseguir cada uno de los objetivos que he planteado para mi vida y a mis padres (Erguin Torres, Rosana Berrú), por haberme inculcado valores y principios necesarios para poder cumplir mis metas.

También agradezco a mis hermanos por haber estado siempre para apoyarme y brindarme su apoyo en el transcurso de éste recorrido, y no podía dejar de agradecerte gracias amor por tu apoyo tus consejos y por formar parte del cumplimiento de esta meta.

Agradezco también a mis compañeros y amigos a la Universidad Nacional de Loja y a la carrera de Ingeniería en Sistemas en la persona de los Ingenieros por haberme dado la apertura para realizar éste Trabajo de Titulación.

## 1. Título

# APLICACIÓN MÓVIL PARA LA DETECCIÓN DE SOMNOLENCIA DE UN CONDUCTOR APLICANDO VISIÓN ARTIFICIAL

## 2. Resumen

El presente trabajo consiste en el desarrollo de una aplicación móvil para la detección de somnolencia de un conductor usando una técnica de inteligencia artificial conocida como visión artificial, basándose específicamente en el estado de los ojos.

Es orientada a conductores de vehículos sean profesionales o no pero con especial énfasis en el transporte público donde es más probable que un conductor debido a las horas laboradas presente somnolencia.

La aplicación tiene la capacidad de adaptarse a diferentes conductores, trabaja en tiempo real y genera un índice de somnolencia que de ser mayor al 80% emite una alerta sonora escogida por el usuario usa la cámara nativa del dispositivo móvil trabajando normalmente en condiciones de luminosidad naturales, es decir; no funciona en oscuridad.

# Summary

This work involves the development of a mobile application for detecting drowsiness of a driver using artificial vision, based specifically on the state of the eyes; the application uses the native mobile device camera normally working in conditions of natural light that is not working in darkness.

It is oriented vehicle drivers are professionals or not, but with special emphasis on public transport where it is more likely that a driver because of the hours worked this drowsiness.

The application has the ability to adapt to different drivers, works in real time and generates an index of sleepiness to be greater than 80% sounds an audible alert chosen by the user

# Índice de Contenidos

*Certificación del Director* ..... ***¡Error! Marcador no definido.***

*Autoría.* ..... ***¡Error! Marcador no definido.***

*Dedicatoria* ..... *V*

*Agradecimiento* ..... *VI*

*1. Título* ..... *VII*

*2. Resumen* ..... *VIII*

*Summary* ..... *IX*

*Índice de Contenidos* ..... *X*

*Índice de Figuras* ..... *XIII*

*Índice de Tablas y Diagramas* ..... *XV*

*3. Introducción* ..... *1*

*4. Revisión de Literatura* ..... *2*

1. SOMNOLENCIA ..... 2
   1.1. Somnolencia ..... 2
      1.1.1. Definición ..... 2
      1.1.2. Factores ..... 2
      1.1.3. Características visuales de la somnolencia al conducir ..... 3
      1.1.4. Consecuencias de tener somnolencia al conducir ..... 3
   1.2. Accidentes de tránsito debido a somnolencia ..... 3
   1.3. Casos de éxito de visión artificial para la somnolencia ..... 4
   1.4. Detección de somnolencia ..... 5
      1.4.1. Análisis basado en patrones de conducción. ..... 6
      1.4.2. Análisis basado en cambios físicos de los ojos y las expresiones faciales usando procesamiento de imágenes. ..... 6
         1.4.2.1. Perclos ..... 6
      1.4.3. Análisis basados en el cambio de las medidas fisiológicas ..... 7
2. DISPOSITIVOS MÓVILES INTELIGENTES. ..... 8
   2.1. Historia ..... 8
      2.1.1. Mercado ..... 8
   2.2. Sistemas Operativos Móviles ..... 10

2.2.1. Android ..... 10
2.2.1.1. Historia ..... 11
2.2.1.2. Estructura de Android ..... 11
2.2.1.3. Desarrollo de aplicaciones ..... 13
2.2.1.4. Ndk ..... 13
2.2.2. Blackberry ..... 14
2.2.2.1. Desarrollo de aplicaciones para Blackberry ..... 14
2.2.3. Ios ..... 14
2.2.3.1. Desarrollo de aplicaciones para Ios ..... 15
2.2.4. Windows Phone ..... 15
2.2.4.1. Desarrollo de aplicaciones para Windows Phone ..... 15

3. Visión artificial y metodologías de desarrollo móvil ..... 16

3.1. Historia de la visión artificial ..... 16
3.2. Herramientas de Visión Artificial ..... 16
3.2.1. OpenCv ..... 16
3.2.1.1. Estructura de librería OpenCv ..... 17
3.2.1.2. Clasificadores ..... 18
3.2.2. Matlab ..... 18
3.2.3. Labview ..... 19
3.3. Metodologías de desarrollo móvil ..... 20
3.3.1. Mobile D ..... 20
3.4. Fases para el reconocimiento digital de imágenes ..... 21

*5. Materiales y Métodos ..... 23*

1. Métodos de Investigación ..... 23

2. Técnicas ..... 23

3. Metodología de Desarrollo ..... 23

*6. Resultados ..... 25*

1. Analizar técnicas y herramientas de visión artificial para la detección de somnolencia en un conductor. ..... 25

1.1. Cálculo del Índice de Somnolencia. ..... 28

2. Desarrollar e implementar la aplicación móvil basada en visión artificial para la detección de somnolencia del conductor ..... 29

2.1. Entrenamiento de clasificadores en OpenCv ..... 29
2.1.1. Adquisición de imágenes ..... 29
2.1.2. Negativas ..... 29
2.1.3. Positivas ..... 29

2.1.4. Crear muestra....29
2.1.5. Entrenamiento....30
2.2. Aplicación Móvil....31
2.2.1. Exploración....31
2.2.1.1. Selección de los grupos de interés....31
2.2.1.2. Requerimientos....31
2.2.1.3. Módulos....32
2.2.2. Inicialización....32
2.2.2.1. Recursos....32
2.2.2.2. Limitaciones....33
2.2.2.3. Supuestos y dependencias....33
2.2.2.4. Configuración del ambiente de desarrollo....34
2.2.2.5. Apreciación global de la aplicación....34
2.2.3. Producción, estabilización, pruebas....35
2.2.3.1. Prototipos....35
2.2.3.2. Casos de uso....37
2.2.3.3. Diagrama de componentes....39
2.2.3.4. Diagrama de Paquetes....40
2.2.3.5. Codificación de la aplicación Android....41
2.2.3.6. Pruebas....45
2.2.3.7. Pruebas de Funcionalidad....46
2.2.3.8. Pruebas de Factibilidad....48
A. Diseño de Prueba....48
B. Análisis de las Pruebas de Validación....48

*7. Discusión.....52*

1. Desarrollo de la propuesta alternativa....52

*8. Conclusiones....55*

*9. Recomendaciones....56*

*10. Bibliografía....57*

*11. Anexos....60*

# Índice de Figuras

*Figura 1 Técnicas detección de somnolencia.* ..... 6

*Figura 2 Variables del PERCLOS* ..... 7

*Figura 3 Sistemas operativos en dispositivos móviles a nivel mundial* ..... 8

*Figura 4 Sistemas operativos en dispositivos móviles en latinoamerica* ..... 9

*Figura 5 Sistemas operativos en dispositivos móviles en ecuador* ..... 10

*Figura 6 Estructura del s.o. android* ..... 12

*Figura 7 Estructura de opencv* ..... 17

*Figura 8 Fase de sensado* ..... 21

*Figura 8 Fase de sengmentación* ..... 22

*Figura 10 Posición adecuada del dispositivo móvil en el vehículo* ..... 33

*Figura 11 Integrar opencv y android* ..... 34

*Figura 12 Apreciación global de la aplicación* ..... 35

*Figura 13 Pantalla de inicio de la aplicación* ..... 36

*Figura 14 Pantalla estado normal del conductor* ..... 36

*Figura 15 Pantalla estado somnolencia del conductor* ..... 37

*Figura 16 Pantalla alerta de somnolencia* ..... 37

*Figura 17 Diagrama de Componentes* ..... 40

*Figura 18 Diagrama de Paquetes* ..... 41

*Figura 19 Diagrama de Clases* ..... 42

*Figura 20 Método para cargar clasificadores* ..... 43

*Figura 21 Método onCameraFrame* ..... 44

*Figura 22 Método detectar somnolencia (Obtención de rostro y área de los ojos)* ..... 44

*Figura 23 Método detectar somnolencia (Comparación con porcentaje de Perclos y envió de alerta)* ..... *45*

*Figura 24 Pruebas Unitarias* ..... *46*

*Figura 22 Resultados plataforma apkudo* ..... *47*

**Índice de Tablas y Diagramas**

*TABLA 1 Comparativa métodos de análisis de detección de somnolencia* ..... *25*

*TABLA 2. Comparativa casos de éxito visión artificial somnolencia* ..... *26*

*TABLA 3. Comparativa sistemas operativos dispositivos móviles* ..... *27*

*TABLA 5. Comparativa herramientas de visión artificial* ..... *28*

*TABLA 6. Comparativa herramientas de visión artificial* ..... *28*

*TABLA 7. Recursos del proyecto* ..... *33*

*TABLA 8. Diseño de pruebas de factibilidad* ..... *48*

*TABLA 9. Recursos humanos* ..... *53*

*TABLA 10. Recursos técnicos y tecnológicos* ..... *54*

*TABLA 11. Presupuesto Total* ..... *54*

# 3. Introducción

La inteligencia artificial, presenta muchas ramas que pueden servir de mucha ayuda en el cotidiano vivir del ser humano una de ellas es la visión artificial, que en los últimos años ha sido usada en algunos trabajos para detectar la somnolencia y fatiga en conductores usando diferentes técnicas.

La somnolencia es la causa de muchos accidentes de tránsito en todo el mundo en Norteamérica el 55% de los accidentes por conductores somnolientos son causados por conductores menores de 25 años de edad [1], en Ecuador el escenario no es distinto según datos de la Agencia Nacional de Tránsito [2] hasta Agosto del 2015sucedieron 560 accidentes por somnolencia o malas condiciones físicas del conductor.

Se ha demostrado que el porcentaje de tiempo que los ojos de una persona están cerrados (PERCLOS) es el método más adecuado para la detección de somnolencia basada en el análisis de ojos [3], muchos sistemas de visión artificial detectan somnolencia en un conductor, pero en el campo móvil (usando un teléfono inteligente) las investigaciones son pocas.

La documentación inicia con el resumen del trabajo de titulación, una introducción que detalla a breves rasgos el contenido del informe una revisión literaria dividida en cuatro capítulos, el primero aborda la somnolencia, sus factores, los métodos de detección entre otras cosas, en el segundo se habla sobre los dispositivos móviles inteligentes los sistemas operativos principales y el desarrollo de aplicaciones en cada uno de ellos, el tercer capítulo trata sobre las herramientas para la visión artificial su funcionamiento así como también sobre metodologías para el desarrollo móvil y las fases para el reconocimiento de objetos con visión artificial. Posteriormente se revisan las técnicas y métodos aplicados en el desarrollo de la aplicación. Como último punto se plasman los resultados del proyecto; primero el entrenamiento de clasificadores para la visión artificial, luego la aplicación Android siguiendo las bases de la metodología. Finalmente se emiten conclusiones y recomendaciones en base al trabajo realizado.

# 4. Revisión de Literatura

## 1. SOMNOLENCIA

En el presente capitulo se detalla la somnolencia, sus caracteristicas visuales, factores, y métodos de detección de somnolencia.

### 1.1. Somnolencia

#### 1.1.1. Definición

La somnolencia es la sensación de pesadez y torpeza de los sentidos motivados por el sueño. [4] Estado intermedio entre el sueño y la vigilia en el que todavía no se ha perdido la conciencia

#### 1.1.2. Factores

Las causas de la excesiva somnolencia pueden ser primarias o secundarias. Dentro de las causas primarias, o también conocidas como de origen central, se encuentran la narcolepsia, la hipersomnia idiopática y otras hipersomnias primarias poco frecuentes como el síndrome Kleine- Levin.

Dentro de las causas secundarias podemos mencionar dos grandes grupos. El primero serían los trastornos que ocurren durante o relacionados con el sueño como por ejemplo los trastornos respiratorios siendo el más conocido el SAHS, comportamientos que lleven a privación del sueño como trabajar de noche u otros trastornos del sueño como el jet lag o cualquier otra alteración del ciclo circadiano, el síndrome de piernas inquietas o el movimiento periódico de miembros.

El segundo grupo que engloba a diversas condiciones médicas como traumas encefálicos, accidentes cerebrovasculares, cáncer, enfermedades inflamatorias, encefalitis o condiciones neurodegenerativas; o psiquiátricas, especialmente la depresión. Pero además se incluye los efectos de ciertos medicamentos como las benzodiacepinas [5].

### 1.1.3. Características visuales de la somnolencia al conducir

Entre las principales características tenemos.

- Dificultad para concentrarse, parpadeo frecuente, o pesados párpados.
- Soñar Despierto errante pensamientos desconectados.
- Dificultad para recordar los últimos kilómetros recorridos; las salidas o señales de tráfico falta.
- Bostezo repetidamente o frotarse los ojos.
- Dificultad para mantener la cabeza erguida.
- A la deriva de su carril, chupar rueda, o golpear una orilla del carril hombro.
- Inquieto e irritable Feeling, [6]

### 1.1.4. Consecuencias de tener somnolencia al conducir

La capacidad de respuesta de un conductor está íntimamente ligada a la información externa que recibe ya sea auditiva y en la mayoría de los casos visual, cuando el estado del conductor no es el adecuado la capacidad de respuesta disminuye lo que hace más probable un accidente de tránsito.

- Perjudica el tiempo de reacción.
- Disminuye la capacidad de actuación, de concentración y de vigilancia.
- Incrementa el comportamiento agresivo y el malhumor, etc.

## 1.2. Accidentes de tránsito debido a somnolencia

En Norteamérica el 55% de los accidentes por conductores somnolientos son causados por conductores menores de 25 años de edad. [1]

Estar despierto durante 18 horas es igual a una concentración de alcohol en sangre (BAC) de 0.08%, que es legalmente ebrio y te deja el mismo riesgo de un accidente. El riesgo de tener un accidente debido a la conducción somnolencia no se distribuye uniformemente a través de la población. La investigación ha identificado hombres jóvenes, los trabajadores por turnos, conductores profesionales y personas con trastornos del sueño sin tratar o con corto plazo o la falta de sueño crónica como estar en mayor riesgo de tener un accidente de caída dormido.

En Ecuador el escenario no es distinto según datos de la Agencia Nacional de Tránsito en el año 2014 en los meses de Septiembre a Diciembre del Año mencionando hubieron 226 accidentes por Conducir en estado de somnolencia o malas condiciones físicas (Sueño, cansancio y fatiga) .En el presente año hasta el mes de abril han ocurrido 291 accidentes por las condiciones anteriormente descritas [2].

### 1.3. Casos de éxito de visión artificial para la somnolencia

Se detallarán 3 casos de éxito de distintas universidades a nivel de ingeniería, maestría y doctorado que son los siguientes.

**"Detección de somnolencia, como asistencia para conductores, utilizando técnicas de procesamiento digital de imágenes"**

Tesis de Ingeniería desarrollada por Gilbert Obed Arriaga Saavedra de la Universidad Tecnológica del Perú en el año 2012. Esta tesis presenta el desarrollo de un algoritmo para la detección de somnolencia en conductores durante el día, basado en el procesamiento de imágenes y fue diseñado para funcionar en vehículos de carga pesada o de transporte interprovincial. [7]

Usa específicamente el análisis del estado de los ojos del conductor para determinar la somnolencia, las pruebas se realizaron solamente en un laboratorio, en condiciones estándares e ideales de manejo. Por ello la adquisición de las imágenes fueron realizadas con una cámara web instalada en un computador y el algoritmo fue diseñado, ejecutado y comprobado con la herramienta de software matemática MATLAB.

**"Análisis del conductor: estimación de la distracción y somnolencia mediante visión por computador e inteligencia artificial usando tecnología TOF"**

Tesis de Ingeniería desarrollada por Fernando Alejandro Guevara y Osvaldo Sebastián Valencia de la Escuela Politécnica del Ejército en el año 2013, se presenta como una alternativa a los sistemas desarrollados hasta ese momento por el hecho de usar inteligencia artificial y el modelo CANDIDE que es un modelo similar a una malla de alambre con una textura mapeada que paramétrica el rostro humano. Se desarrolló usando el lenguaje de programación C# y el Facetraking SDK que permite crear

aplicaciones que puedan hacer seguimiento en tiempo real al rostro humano usando una cámara Kinect. [8]

Este algoritmo fue creado específicamente para pruebas de escritorio así como sus pruebas por lo que su implementación en un vehículo y condiciones reales depende de su resolución.

**"Sistema avanzado de asistencia a la conducción mediante visión por computador para la detección de la somnolencia"**

Esta tesis doctoral presenta un sistema para la detección de la somnolencia del conductor, basado en el análisis de los ojos. El sistema tiene la capacidad de adaptarse a cualquier persona, trabaja en tiempo real, bajo condiciones variables de iluminación y reales de conducción, generando en cada instante un índice de somnolencia que mide el estado de vigilia del conductor desarrollado en Matlab.

El sistema desarrollado para el día utiliza una cámara estándar que captura imágenes con iluminación natural, mientras que el sistema desarrollado para la noche utiliza un complejo sistema de visión basado en la iluminación infrarroja del tipo cercano, de esta manera, se aprovechan las propiedades espectrales para resaltar las pupilas de los ojos del conductor en condiciones de baja iluminación, como es el caso de la noche.

Esta forma parte del sobre el vehículo experimental IvvI que es una propuesta de un sistema avanzado de ayuda a la conducción y se ha presentado magníficos resultados respecto a los objetivos planteados. Los experimentos fueron desarrollados en dos escenarios: situaciones reales de conducción, y condiciones de laboratorio [9].

## 1.4. Detección de somnolencia

Para la detección de somnolencia podemos usar algunos métodos, pueden ser clasificados en tres categorías, enumeradas a continuación e ilustradas en la Figura 1 [6]

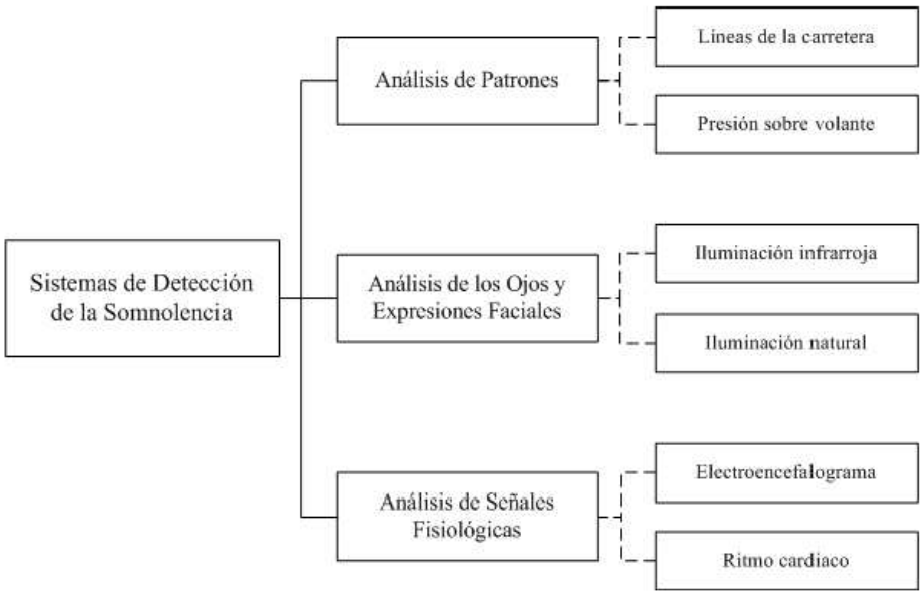



*FIGURA 1 TÉCNICAS DETECCIÓN DE SOMNOLENCIA.*

### 1.4.1. **Análisis basado en patrones de conducción.**

Se construyen patrones de conducción en base a diferentes experimentos obteniendo de ellos parámetros medibles sobre el conductor, el vehículo y su entorno, parámetros como la velocidad, ángulo de inclinación que han demostrado ser buenos indicadores de la vigilia del conductor pese a sus limitaciones, además puede ser necesario un modelo por cada tipo de vehículo [3]

### 1.4.2. **Análisis basado en cambios físicos de los ojos y las expresiones faciales usando procesamiento de imágenes.**

Se utilizan las técnicas de procesamiento de imágenes para detectar cambios físicos en el rostro del conductor. Esta técnica es altamente fiable además de rápida y no genera molestias en el conductor con el procesamiento de imágenes se pueden utilizar técnicas para la detección de somnolencia como.

#### 1.4.2.1. **Perclos**

El porcentaje de tiempo que los ojos de una persona están cerrados PERCLOS por siglas en ingles de (Percentage of the Time Eyelids are Closed). Mide el porcentaje de

cerrado de ojos en un intervalo de tiempo sin tomar en cuenta el tiempo de cerrado normal. En varias investigaciones desarrolladas por la National Highway Traffic Safety Administration (NHTSA) [3]se ha demostrado que es el más adecuado para caracterizar la somnolencia del conductor a través del análisis de los ojos.

Se ha establecido un criterio de somnolencia en los ojos cuando el porcentaje de estar cerrado es mayor a 80%. La métrica resultante es la proporción de tiempo en que los ojos se encontrarán con criterio de cierre de 80% a más. [11]

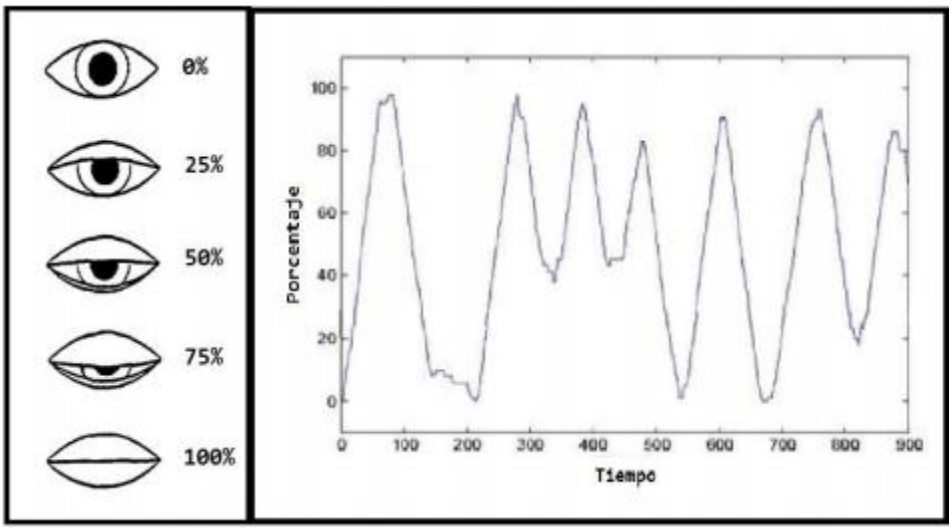



*FIGURA 2 VARIABLES DEL PERCLOS*

Por ejemplo si se toma como tiempo base 1 segundo y como valor de umbral porcentajes mayores o iguales al 80% se obtendrá un índice de somnolencia positivo con un tiempo de cierre de 0.8 segundos.

$$PERCLOS = \frac{TIEMPO\ DE\ OJOS\ CERRADOS * 100}{TIEMPO\ BASE\ DE\ CÁLCULO}$$

### 1.4.3. Análisis basados en el cambio de las medidas fisiológicas

Este método está basado en medir los cambios fisiológicos tales como la variabilidad del ritmo cardiaco, el electromiograma y electroencefalograma del conductor, como un medio para detectar los estados cognitivos humanos. Las ventajas que presentan es la precisión en el diagnóstico del estado. Mientras que uno de los mayores inconvenientes está en el uso de sensores conectados al cuerpo del conductor.

# 2. DISPOSITIVOS MÓVILES INTELIGENTES.

En el presente capitulo se tratara sobre dispositivos móviles inteligentes su historia, su porcentaje en el mercado, los distintos sistemas operativos asi como tambien herramientas de desarrollo para cada uno de ellos.

## 2.1. Historia

En el año 2007 Apple Inc., introduce su primera generación de dispositivos iPhone, que sería uno de los primeros Smartphones que permitían manipularse íntegramente desde su pantalla táctil. Este dispositivo marcaría un punto de inflexión en este segmento de mercado. Durante estos últimos años, Apple ha sacado nuevas versiones de su iPhone con soporte 3G y descarga de aplicaciones desde su propia comunidad denominada App Store. En el 2008 sale a la luz Android, una plataforma de código abierto basada en una modificación del kernel de Linux y específico para Smartphones. El primer dispositivo en utilizar la plataforma Android como base fue el HTC Dream. [12]

### 2.1.1. Mercado

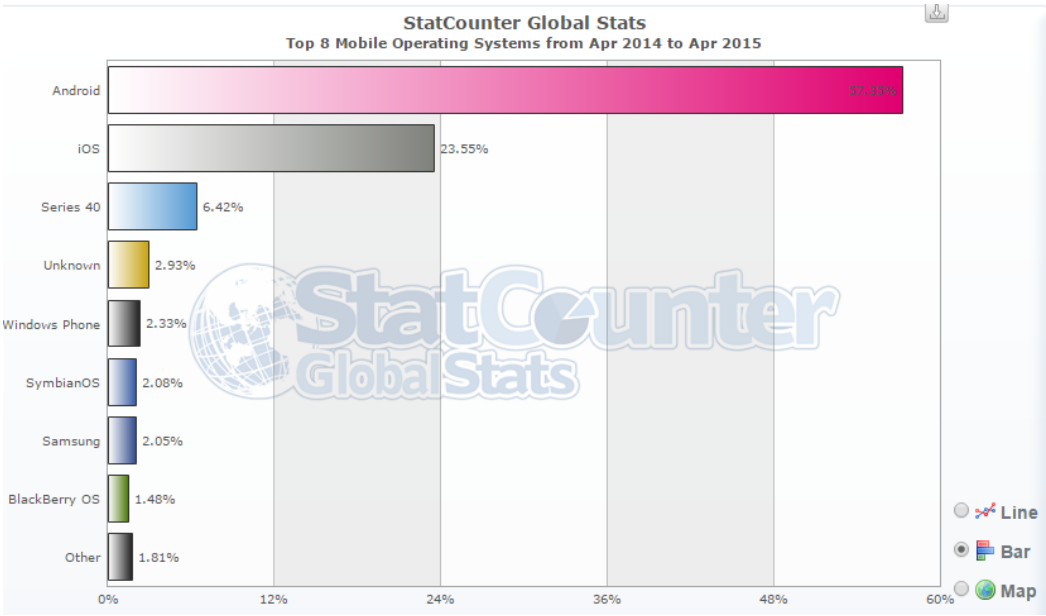



*FIGURA 3 SISTEMAS OPERATIVOS EN DISPOSITIVOS MÓVILES A NIVEL MUNDIAL*

En el sitio web de la encuestadora [13] statCounter se puede visualizar a nivel mundial: La porción de uso de los distintos sistemas operativos para dispositivos móviles inteligentes esto se basa en la amplia disponibilidad de aplicaciones, costos y estabilidad que existen para cada una de las plataformas, las más destacadas son: [13]

- Android (57,35% )
- IOS (23,55%)
- Series 40 (6,42%)
- Windows Phone (2,33%)
- BlackBerry OS (1,48% ) [13]

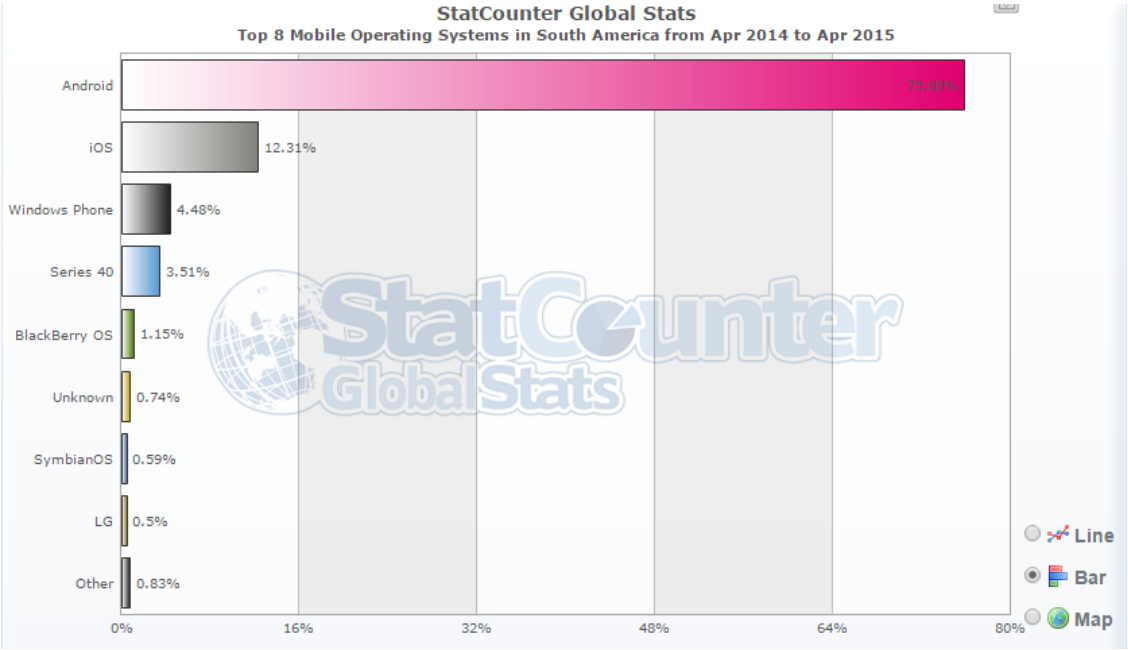



*FIGURA 4 SISTEMAS OPERATIVOS EN DISPOSITIVOS MÓVILES EN LATINOAMERICA*

En Latinoamérica las plataformas, las más destacadas son:

- Android (75,88% )
- IOS (12,31%)
- Windows Phone (4,48%)
- Series 40 (3,51%)

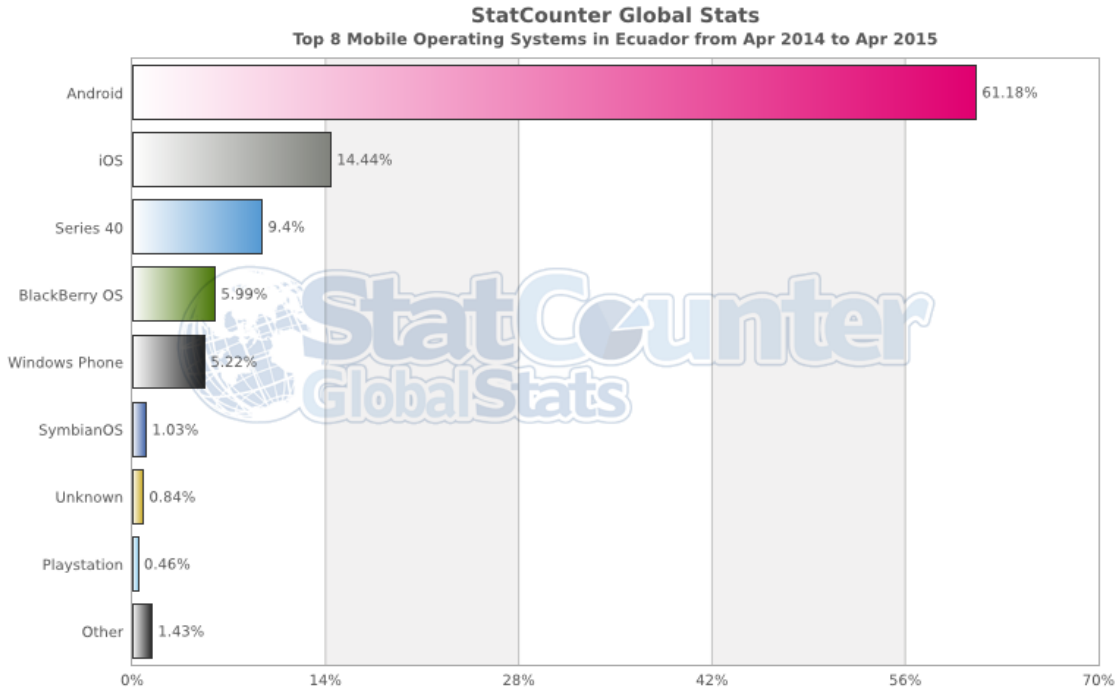



*FIGURA 5 SISTEMAS OPERATIVOS EN DISPOSITIVOS MÓVILES EN ECUADOR*

Finalmente en Ecuador las plataformas, las más destacadas son: [13]

- Android (61,18% )
- IOS (14,44%)
- Series 40 (9,4%)
- BlackBerry OS (5,99% )
- Windows Phone (5,22%)

## 2.2. Sistemas Operativos Móviles

### 2.2.1. Android

Android es un sistema operativo y una plataforma software, basado en Linux para teléfonos móviles una de las claves de la popularidad de Android es que, como Linux, es una plataforma de código abierto, lo que permite a fabricantes, operadores y desarrolladores dar mayor funcionalidad a sus Smartphone. Además Android es un sistema gratuito y multiplataforma; por multiplataforma entendemos que el sistema operativo puede ser usado en distintas plataformas, y por plataforma entendemos que es

una combinación de hardware y software usada para ejecutar aplicaciones; en su forma más simple consiste únicamente de un sistema operativo, una arquitectura, o una combinación de ambos

#### 2.2.1.1. **Historia**

En octubre del año 2003, Andy Rubin, Rich Miner, Nick Sears y Chris White daban forma a Android Inc. En sus inicios, únicamente trascendió que la actividad de la empresa se centraba en "el desarrollo de software para teléfonos móviles". Android Inc. pasó casi dos años trabajando "en la sombra", hasta que Google se fijó en la compañía la fecha clave para llegar a entender mejor el éxito de Android es el 5 de noviembre de 2007. Ese día se fundaba la OHA (Open Handset Alliance), una alianza comercial de 35 componentes iniciales liderada por Google, que contaba con fabricantes de terminales móviles, operadores de telecomunicaciones, fabricantes de chips y desarrolladores de software. El mismo día se dio a conocer por vez primera lo que hoy conocemos como Android, una plataforma de código abierto para móviles que se presentaba con la garantía de estar basada en el sistema operativo Linux [14]

#### 2.2.1.2. **Estructura de Android**

Android utiliza como base el kernel de Linux pero los dos sistemas no son lo mismo, Android no cuenta con un sistema nativo de ventanas de Linux ni tiene soporte para glibc (librería estándar de C); tampoco es posible utilizar la mayoría de aplicaciones de GNU de Linux.

Además de todo lo ya implementado en el kernel de Linux, Android agrega algunas cosas específicas para plataformas móviles como la comunicación entre procesos (lograda a través del binder), la forma de manejar la memoria compartida (ashmem) y la administración de energía (con wakelocks).

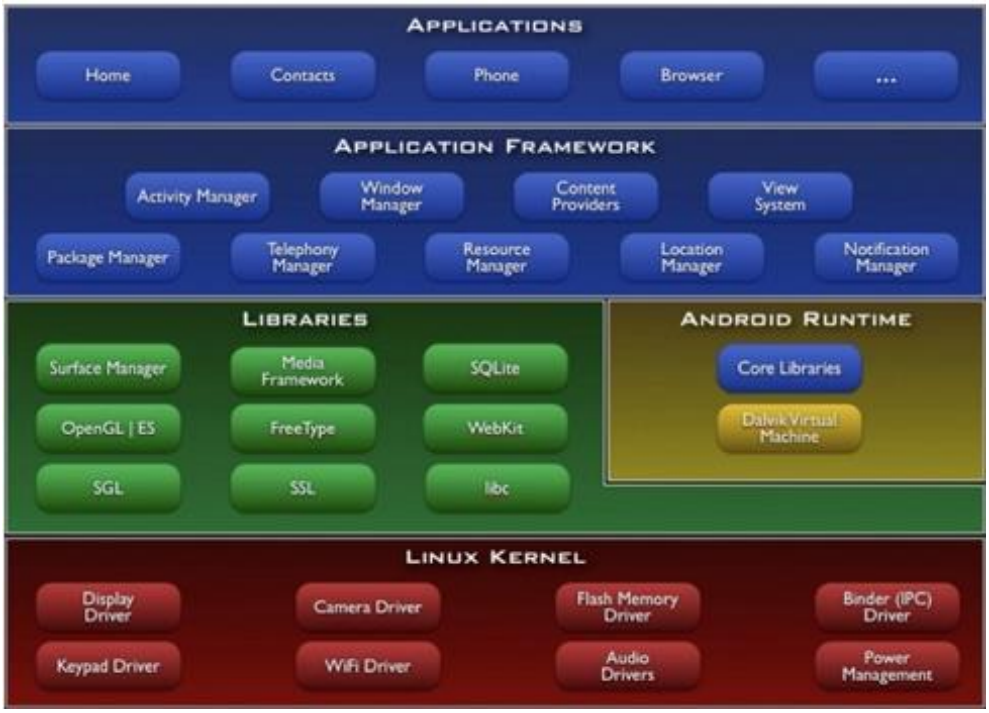



*FIGURA 6 ESTRUCTURA DEL S.O. ANDROID*

[14] Aquí tenemos el corazón de Android: el manejo de memoria, procesos, drivers, etc. Aquí es donde se da la comunicación con el hardware. Esto nos sirve para no estar peleando con los fabricantes de cada móvil, nos ayuda a solo usar la "cámara" y no tener que saber cómo funciona la cámara del fabricante X, fabricante Y; solamente hacemos lo que nos interesa, que sería usar la cámara y listo. Además de eso, aquí se administran los recursos del celular, memoria, energía.

- **Capa Librerías (Verde)** Esta capa tiene las librerías nativas de Android, están escritas en C o C++ y tienen tareas específicas. Gestión del acceso a la pantalla, Reproducción de imágenes, audio y vídeo, base de datos, navegador, gráficos 2D, gráficos 3D.
- **Android Runtime (Capa Amarilla)** Esta capa amarilla no se considera al 100% una capa. Lo que es muy importante comentar es que aquí se encuentra Dalvik, la máquina virtual de Android, que no es lo misma que la Java Virtual Machine. Esto quiere decir que cuando compilamos en Java lo que se genera solamente va a funcionar en la JVM, porque Dalvik es una máquina virtual, pero de Android, así que el ByteCode que genera Java es inservible para Dalvik [15].

- **Aplicación Framework (Capa azul)** Esta capa es la más visible para el desarrollador, ya que la mayoría de los componentes que forman parte del desarrollo los vamos a encontrar aquí. Las vistas de elementos que son parte de la interfaz gráfica, como los mapas, cuadros de texto, etc. Administra las notificaciones, administra los paquetes y nos permite el uso de archivos en otros paquetes. Administra recursos de la aplicación, como los xml, imágenes, sonido. Se gestiona la posición geográfica y administra la cámara.
- **Aplications Capa Azul última)** Aquí tenemos las aplicaciones que vienen en el dispositivo, por ejemplo: el gestor de correos, los mensajes, el google play, etc.

#### 2.2.1.3. **Desarrollo de aplicaciones**

La comunidad de desarrollo de Android tiene como objetivo hacer el desarrollo de aplicaciones de manera sencilla y accesible al mayor número de sistemas posibles. Por ello, todas las herramientas, entornos, demos e instrucciones que se ofrecen son gratuitas y se pueden descargar de su página web, http://developer.android.com. Las aplicaciones base incluyen un teléfono, cliente de email, programa de envío de SMS, calendario, mapas, navegador, contactos... que pueden a su vez ser usados por otras aplicaciones.

**Entornos de desarrollo.**

En los últimos años, Android ha alentado y ayudado a los desarrolladores a emplear el Paquete ADT (Herramientas de Desarrollo de Android) para Eclipse como entorno de desarrollo de aplicaciones. Android ofrece un plugin para Eclipse que extiende la funcionalidad de éste y facilita el desarrollo de aplicaciones El nuevo entorno de desarrollo de aplicaciones para Android Studio, que en el futuro reemplazará al Paquete ADT* para Eclipse.

#### 2.2.1.4. **Ndk**

El Ndk de Android es un conjunto de herramientas que le permite implementar partes de su aplicación utilizando lenguajes de código nativo, como C y C ++. Para ciertos tipos de aplicaciones, esto puede ayudar a volver a utilizar las bibliotecas de código existentes escritos en esos idiomas; hacer programación de bajo nivel hacia el hardware, o diferenciar sus aplicaciones aprovechando características que no es posible o práctico acceder con otros mecanismos. [16].

### 2.2.2. Blackberry

El sistema operativo BlackBerry OS2, nace de la mano de la empresa canadiense RIM (Research In Motion) y comparte nombre con su línea de teléfonos inteligentes lanzada al mercado en 1999.

Las principales características de este sistema operativo se centran en el mercado de los negocios. Tiene aplicaciones como soporte para MIDP (perfiles informativos del terminal), pantallas táctiles y el resto de elementos que, en general, podemos encontrar en sus competidores [17].

En la actualidad una de las características que hacen a este sistema operativo único reside en que los servicios de datos pasan por los servidores de RIM, permitiendo así la compresión de la información que, además, viaja cifrada por la red, obteniendo velocidad y seguridad. Una de las desventajas de este sistema operativo es que es exclusivo para dispositivos de la marca BlackBerry.

#### 2.2.2.1. Desarrollo de aplicaciones para Blackberry

Todo lo que ofrece RIM a los desarrolladores es abierto y gratuito, sin importar el sistema operativo o el entorno de desarrollo con numerosos plugins para Eclipse o Visual Studio.

Java Plug-in para Eclipse es la herramienta de desarrollo preferida a la hora de desarrollar aplicaciones Java de BlackBerry. Sin embargo, puede utilizar, de forma alternativa, Visual Studio o NetBeans como su entorno de desarrollo.

El complemento incluye BlackBerry Java SDK, BlackBerry Smartphone Simulator y numerosas aplicaciones de ejemplo. El complemento también incluye la siguiente herramienta de desarrollo:

RAPC: puede utilizar este compilador de símbolo de sistema para compilar archivos .java y .jar en archivos .cod. Las aplicaciones del Smartphone deben ser archivos .cod.

### 2.2.3. Ios

Es un sistema operativo móvil de Apple desarrollado originalmente para el iPhone, siendo después usado en el iPod Touch e iPad. Es un derivado de Mac OS X, que a su vez está basado en Darwin BSD. El iOS tiene cuatro capas de abstracción: la capa del núcleo del sistema operativo, la capa de "Servicios Principales", la capa de "Medios de

comunicación" y la capa de "Cocoa Touch" iOS es un derivado de Mac O X y es software propietario y usado exclusivamente en productos Apple.

#### 2.2.3.1. Desarrollo de aplicaciones para Ios

La principal herramienta de desarrollo para Mac OS X y iOS es este kit, **Xcode** es un IDE (entorno integrado de desarrollo) que permite escribir, compilar, ejecutar y depurar el código de la aplicación que se está desarrollando. También permite organizar los ficheros por proyectos y su importación y exportación de manera sencilla.

**Interface Builder.** Herramienta que nos permite realizar la interfaz gráfica de las aplicaciones cómodamente. A partir de Xcode 4.0 está integrado con Xcode. La interfaz gráfica de las aplicaciones, además de poder crearse programáticamente, se puede crear en un fichero XML con la extensión .xibque se compila en un .nib. Cuando se ejecuta una aplicación se carga en memoria este fichero en base al cual se crea la interfaz gráfica [18].

### 2.2.4. Windows Phone

Windows Phone (abreviado WP) es un sistema operativo móvil desarrollado por Microsoft, como sucesor de Windows Mobile, es de uso exclusivo para telefonos Nokia adquirido por Windows y bajo licencia privativa.

#### 2.2.4.1. Desarrollo de aplicaciones para Windows Phone

Las siguientes son algunas herramientas de desarrollo y productos relacionados que puedes utilizar para crear aplicaciones.

Visual Studio es el IDE de programación por excelencia de la plataforma .NET y, por ende, de las aplicaciones que corren en los dispositivos del ecosistema de Microsoft.

Crea aplicaciones de la Tienda de Windows Phone y Windows en Visual Studio 2013, un completo entorno de desarrollo. Visual Studio incluye plantillas de proyectos, un editor de código, un diseñador visual y un cuadro de herramientas de controles. También incluye características de prueba integradas.

Ignite UI permite a los desarrolladores crear experiencias móviles y de escritorio que son táctiles (toque de pantalla) y que están basadas en el uso del navegador. Ignite UI es una serie de núcleos (core) jQuery, jQuery UIs y librerías móviles de jQuery que pueden ser utilizadas con HTML5 o ASP.NET.

# 3. Visión artificial y metodologías de desarrollo móvil

## 3.1. Historia de la visión artificial

El inicio de la visión artificial, desde el punto de vista práctico, fue marcado por Larry Roberts, el cual, en 1961 creó un programa que podía ver una estructura de bloques, analizar su contenido y reproducirla desde otra perspectiva, demostrando así a los espectadores que esa información visual que había sido mandada al ordenador por una cámara, había sido procesada adecuadamente por él.

Más adelante surgieron muchos programas y cámaras para implementar la visión artificial en muchas áreas diferentes, hasta la actualidad donde llegaron a tener objetivos tales como:

- La detección, segmentación, localización y reconocimiento de ciertos objetos en imágenes (por ejemplo, caras humanas).
- Registro de diferentes imágenes de una misma escena u objeto, y hacer
- Concordar un mismo objeto en diversas imágenes.
- Seguimiento de un objeto en una secuencia de imágenes.

## 3.2. Herramientas de Visión Artificial

### 3.2.1. OpenCv

OpenCv (Open Source Computer Vision) es una biblioteca informática de código abierto desarrollado originalmente por la visión de Intel. La librería OpenCv está dirigida fundamentalmente a la visión por computador en tiempo real. Entre sus muchas áreas de aplicación destacarían: interacción hombre-máquina; segmentación y reconocimiento de objetos; reconocimiento de gestos; seguimiento del movimiento; estructura del movimiento y robots móviles. [19]

3.2.1.1. **Estructura de librería OpenCv**

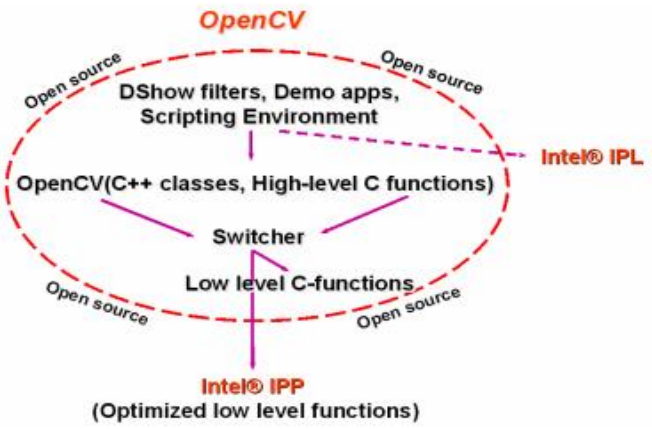



*FIGURA 7 ESTRUCTURA DE OPENCV*

[20]Todas estas herramientas de alto nivel hacen uso de un paquete de clases C++ y funciones C de alto nivel que utilizan a su vez funciones muy eficientes escritas en C. Concretamente, el conjunto de funciones suministradas por la librería OpenCv se agrupan en los siguientes bloques:

• Estructuras y operaciones básicas: matrices, grafos, árboles, etc.

- Procesamiento y análisis de imágenes: filtros, momentos, histogramas, etc.
- Análisis estructural: geometría, procesamiento del contorno, etc.
- Análisis del movimiento y seguimiento de objetos: plantillas de movimiento, seguidores (i.e. Lucas-Kanade), flujo óptico, etc.
- Reconocimiento de objetos: objetos propios (eigen objects), modelos HMM, etc.
- Calibración de la cámara: morphing, geometría epipolar, estimación de la pose (i.e. POSIT), etc.
- Reconstrucción tridimensional (funcionalidad experimental): detección de objetos, seguimiento de objetos tridimensionales, etc.
- Interfaces gráficos de usuarios y adquisición de video.

#### 3.2.1.2. **Clasificadores**

OpenCv propone diferentes clasificadores ya entrenados, entre ellos un clasificador de caras o un clasificador de ojos. Si queremos crear nuestro propio clasificador utilizamos una herramienta proporcionada por OpenCv llamada opencv_haartraining usando nuestras imágenes. Para probar estos clasificadores también propone una aplicación que evalúa el clasificador denominada opencv_performance, esta aplicación evalúa la calidad del clasificador utilizando diferentes imágenes positivas. [21]

Para poder crear nuestro clasificador necesitamos cientos o incluso miles de imágenes del objeto que queramos clasificar. Partiendo de estas imágenes generamos los casos positivos. Además necesitamos imágenes arbitrarias donde no aparezca el objeto a clasificar, estas imágenes se utilizaran como casos negativos

### 3.2.2. **Matlab**

MATLAB dispone en la actualidad de un amplio abanico de programas de apoyo especializados, denominados Toolboxes, que extienden significativamente el número de funciones incorporadas en el programa principal. Estos Toolboxes cubren en la actualidad prácticamente casi todas las áreas principales en el mundo de la ingeniería y la simulación, destacando entre ellos los “toolboxes” de adquisición y procesamiento de imágenes y el Matlab Guide que nos permitirá realizar nuestro programa en ambiente visual.

El Toolbox de adquisición de imágenes es una colección de funciones que sirven, entre otras cosas, para adquirir imágenes de diversos dispositivos (desde frame grabbers profesionales hasta USB webcams), para visualizar videos en vivo, para la adquisición de imágenes mediante triggers, para importar datos hacia el entorno de trabajo de MATLAB, etc. En otras palabras, es la herramienta que nos permitirá realizar la aplicación en tiempo real. El Toolbox de Procesamiento de Imágenes proporciona a MATLAB un conjunto de funciones que amplía las capacidades del producto para realizar desarrollo de aplicaciones y de nuevos algoritmos en el campo del proceso y análisis de imágenes. El entorno matemático y de creación de MATLAB es ideal para el procesado de imágenes, ya que estas imágenes son, al fin y al cabo, matrices. [22]

Este toolbox incorpora funciones para:

- Diseño de filtros.
- Mejora y retocado de imágenes.
- Análisis y estadística de imágenes.
- Operaciones morfológicas, geométricas y de color.
- Transformaciones 2D, etc.

### 3.2.3. **Labview**

Labview es un entorno grafico de programación. El lenguaje utilizado para programar denominado "Lenguaje G, donde la "G" simboliza que es un lenguaje de tipo gráfico.

Los programas desarrollados en Labview se llaman VI´s (Virtual Instruments), su origen provenía del control de instrumentos, pero hoy en día su uso se ha expandido más allá. Labview tiene un entorno de programación gráfico, por lo que los programas no se escriben, sino que se dibujan, una labor facilitada gracias a que Labview consta de una gran cantidad de bloques prediseñados.

Los programas se dividen en dos partes bien diferenciadas, una llamada "Panel Frontal", y otra "Diagrama de Bloques".

- Panel Frontal: es la interfaz con el usuario, la utilizamos para interactuar con el usuario cuando el programa se está ejecutando. En esta interfaz se definen los controles (los usamos como entradas, pueden ser botones, marcadores etc...) e indicadores (los usamos como salidas, pueden ser gráficas, etc…..).
- Diagrama de Bloques: es el programa propiamente dicho, donde se define su funcionalidad, aquí se colocan íconos que realizan una determinada función y se interconectan (el código que controla el programa). Suele haber una tercera parte icono/conector que son los medios utilizados para conectar un VI con otros VIs.

Con Labview, no se programa el software a escribir líneas de códigos con una sintaxis compleja. La programación se hace con iconos los cuales representan funcionalidades,

ligados entre ellos por cables quienes representan los flujos de datos (un poco a la manera de una tarjeta electrónica con sus componentes y circuitos integrados). [23]

Esta representación muy adornada del código es cerca de la concepción como uno hace: con esquemas; eso, por supuesto, facilita mucho el trabajo que necesita la programación del concepto. Esta abstracción del lenguaje gráfico no requiere de ser un experto en programación para desarrollar software sencillo. De igual forma, para software más complejos, el cliente "profano" en programación pero experto en su dominio, podrán leer y entender la idea y así guiar el ingeniero experto en programación.

## 3.3. Metodologías de desarrollo móvil

### 3.3.1. Mobile D

Mobile-D es la metodología para el desarrollo ágil de software. Además del desarrollo de software para móviles, es conveniente para los varios contextos, por ejemplo, la seguridad, financieros, logística y aplicaciones de simulación de productos. Esta metodología usa las prácticas de desarrollo de Extreme Programming (XP), la escalabilidad de la metodología Crystal y el ciclo de vida del Rational Unified Process (RUP).

El ciclo del proyecto se divide en cinco fases: exploración, inicialización, producción, estabilización y prueba del sistema. En general, todas las fases (con la excepción de la primera fase exploratoria) contienen tres días de desarrollo distintos: planificación, trabajo y liberación.

La fase de exploración, siendo ligeramente diferente del resto del proceso de producción, se dedica al establecimiento de un plan de proyecto y los conceptos básicos. Por lo tanto, se puede separar del ciclo principal de desarrollo. En esta fase es importante la comunicación, en este caso, con el tutor del proyecto.

Durante la fase de inicialización se preparan e identifican todos los recursos necesarios. Se preparan los planes para las siguientes fases y se establece el entorno técnico.

En la fase de producción se repite la programación de tres días (planificación-trabajo-liberación). Se repite iterativamente hasta implementar todas las funcionalidades. Primero se planifica la iteración de trabajo en términos de requisitos y tareas a realizar. Las tareas se llevaran a cabo durante el día de trabajo desarrollando e integrando el

nuevo código con el anterior. Durante el último día se lleva a cabo la integración del sistema seguida de las pruebas de aceptación.

En la fase de estabilización, se llevan a cabo las últimas acciones de integración para asegurar que el sistema completo funciona correctamente. En esta fase, los desarrolladores realizarán tareas similares a las que debían desarrollar en la fase de producción, aunque en este caso todo el esfuerzo se dirige a la integración del sistema. Adicionalmente se puede considerar en esta fase la producción de documentación. [24]

La última fase (prueba y reparación del sistema) tiene como meta la disponibilidad de una versión estable y plenamente funcional del sistema. El producto terminado e integrado se prueba con los requisitos de cliente y se eliminan todos los defectos encontrados

## 3.4. Fases para el reconocimiento digital de imágenes

Dentro del reconocimiento digital de imágenes tenemos las siguientes fases que ayudaron en el proceso detección de somnolencia. [25] Se hará una explicación en base al funcionamiento de la aplicación

- **Sensado:** Es el proceso que nos lleva a la obtención de una imagen visual.

*FIGURA 8 FASE DE SENSADO*

En la Figura 8 se observa la imagen que se obtiene de la cámara del dispositivo

- **Pre procesamiento**: Trata de las técnicas de reducción de ruido y enriquecimiento de detalles en la imagen, provistas por la librería de visión artificial además se obtienen detalles como el pixelaje de la imagen y los frames por segundo de la cámara.

- **Segmentación:** Es el proceso que parte una imagen en objetos de interés.

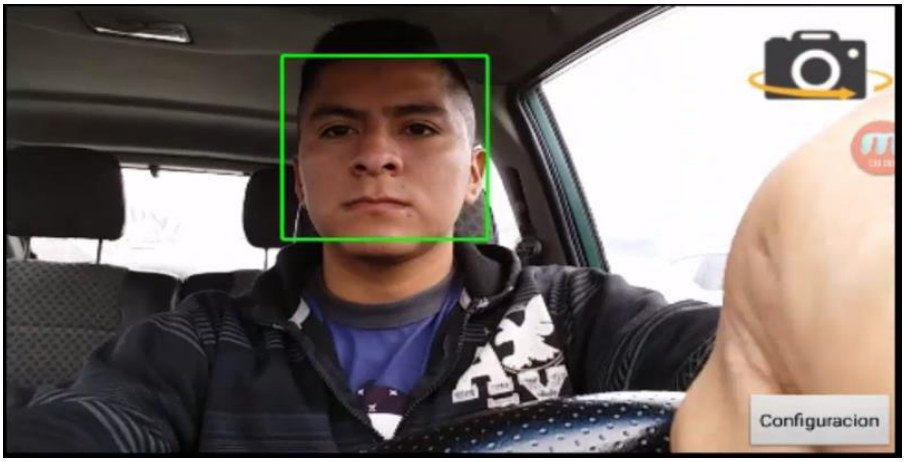



*FIGURA 9 FASE DE SEGMENTACIÓN*

En la Figura 8 se observa el área de interés que se usa en la aplicación en este caso el rostro del conductor.

- **Descripción:** Trata con el cómputo de características útiles para diferenciar un tipo de objeto de otro, se obtienen las matrices en RGB que sirven para tratar y diferenciar objetos.
- **Reconocimiento:** Es el proceso que identifica esos objetos.

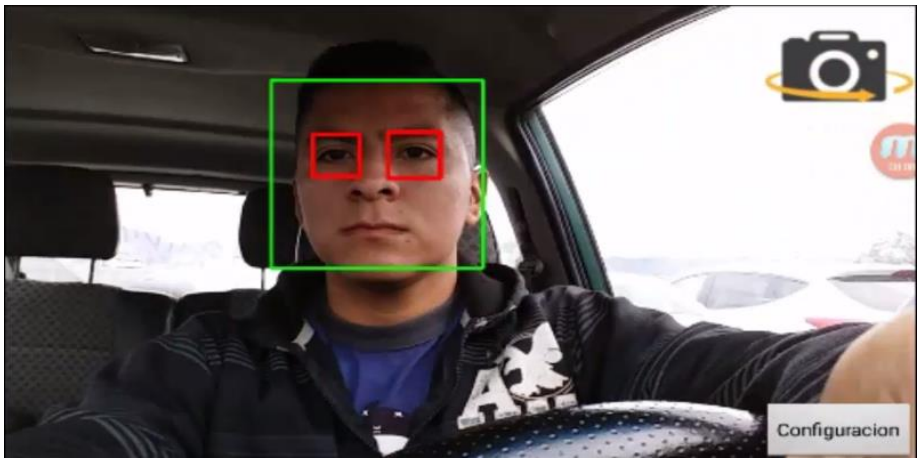



*FIGURA 9 FASE DE RECONOCIMIENTO*

Como se observa en la figura se reconocen distintos tipos de objetos en este caso el rostro y los ojos del conductor

- **Interpretación:** Asigna un significado a un conjunto de objetos reconocidos, dentro de este proceso en el marco del proyecto en esta fase se realiza la obtención del índice de somnolencia en base a los objetos reconocidos.

# 5. Materiales y Métodos

## 1. Métodos de Investigación

- **Método Deductivo:** Tomando en cuenta las soluciones y proyectos realizados por otros autores permitió mejorar en aspectos específicos de la aplicación realizada.
- **Método Inductivo:** Permitió identificar factores, tomar muestras y plantear escenarios para la resolución del problema general.

## 2. Técnicas

- **Revisión Bibliográfica.-** Mediante esta técnica se sustenta el desarrollo del proyecto en base a consultas de libros, artículos científicos, tesis doctorales y casos de éxito.

- **Observación:** Permitió identificar factores previos a la somnolencia para generar un algoritmo de detección adecuado.

- **Encuesta:** Permitió hacer las pruebas de factibilidad con los posibles usuarios finales de la aplicación

## 3. Metodología de Desarrollo

La metodología usada es MOBILE-D el objetivo de esta es conseguir ciclos de desarrollo muy rápidos en equipos muy pequeños esto debido al tiempo y que fue desarrollado solamente por una persona. Se compone de distintas fases: exploración, inicialización, fase de producto, fase de estabilización y la fase de prueba.

**Exploración**: Se definió el alcance de proyecto, supuestos, limitaciones, posibles usuarios finales, se levantaron requerimientos basados en los entregables de la metodología

**Inicialización:** Se prepararon los recursos, físicos, materiales, tecnológicos (NDK, Android, OpenCv) y se configuraron e integraron cada uno de ellos.

**Fase de producto**: Se realizaron casos de uso en base a los requerimientos, planificación de las iteraciones, codificación de la solución y pruebas en cada sprint

**Fase de Prueba:** Se realizaron pruebas para determinar si la aplicación cumple con lo pedido en la fase de exploración, entre las pruebas realizadas destacan pruebas de compatibilidad, unitarias y de factibilidad.

Esta metodología nos presentó muchas ventajas entre ellas facilidad para detectar y resolver tempranamente problemas técnicos, baja densidad de defectos en las liberaciones de los productos, se basa en el desarrollo basado en pruebas que es una de las mejores formas de asegurar la calidad

# 6. Resultados

## 1. Analizar técnicas y herramientas de visión artificial para la detección de somnolencia en un conductor.

Se inicio con un estudio de los métodos de análisis de detección de somnolencia en base a factores escogidos por varios autores como son el costo de implementación, la comodidad para el conductor; la dificultad de aplicación es decir que tan complicado es aplicar ese metodo, la adaptibilidad que se refiere a si el método se adapta a distintos conductores,vehiculos y finalmente la tecnología usada

| Método | Costo Implementación | Comodidad para Conductor | Adaptabilidad | Tecnologías |
|---|---|---|---|---|
| **Análisis Basado en patrones de conducción.** | ALTO (Por tiempo en realización de estudios) | SI | NO | Mucha tecnología en vehículo |
| **Análisis Basado en cambios físicos de los ojos y las expresiones faciales usando procesamiento de imágenes** | MEDIO(Se necesita una máquina para el procesamiento de imágenes) | SI | SI | Fácil adquirir un dispositivo de imagen |
| **Análisis basados en el cambio de las medidas fisiológicas** | ALTO (Sensores y máquinas para la medición) | NO | SI | Mucha tecnología en conductor |

*TABLA 1 COMPARATIVA MÉTODOS DE ANÁLISIS DE DETECCIÓN DE SOMNOLENCIA*

Partiendo de los factores previamente descritos a partir de investigación bibliográfica (Ver Tabla 1), en base a todos estos parámetros y como se detalla en la tabla se podría decir que el método que se ajusta a las necesidades del proyecto es **Análisis Basado en cambios físicos de los ojos y las expresiones faciales usando procesamiento de imágenes,** tanto por su costo de implementación, la comodidad para el conductor y su adaptabilidad.

Se evaluaron 3 casos de éxito de uso de visión artificial para la detección de somnolencia con el fin de conocer el trabajo de otros autores en el tema, hasta donde llego su investigación, que tecnología usaron la plataforma en la que fue desarrollado que tipo de tecnología así como también si la detección se realizaba en condiciones de poca luminosidad (modo nocturno)

| Caso de éxito | Plataforma | Modo Nocturno | Hardware | Software |
|---|---|---|---|---|
| **Detección de somnolencia, como asistencia para conductores, utilizando técnicas de procesamiento digital de imágenes** | Escritorio | No | Web Cam. | Matlab |
| **Análisis del conductor: estimación de la distracción y somnolencia mediante visión por computador e inteligencia artificial usando tecnología tof** | Escritorio | No | Cámara y Kinect | Candide C# |
| **Sistema avanzado de asistencia a la conducción mediante visión por computador para la detección de la somnolencia** | Escritorio | Si | Cámara IP y Cámara Infrarroja | Matlab |

*TABLA 2. COMPARATIVA CASOS DE ÉXITO VISIÓN ARTIFICIAL SOMNOLENCIA*

En el análisis de casos de éxito (Ver Tabla 2), se pudo observar que todas las investigaciones fueron desarrolladas para ser usadas en un computador de escritorio; usando cámaras externas, usando software privativo, también solo una de ellas que es una tesis doctoral incluye un algoritmo para el reconocimiento de somnolencia en condiciones de poca luminosidad (modo nocturno) con ayuda de una cámara infrarroja; en base a esto se puede concluir que la solución propuesta al ser para una plataforma móvil y utilizando software libre presenta un grado de innovación.

Se han evaluado 4 principios para la comparación de los sistemas operativos usados por un dispositivo móvil detallados en la revisión de literatura, el primero es la portabilidad es decir en que marcas de teléfonos pueden ser usados estos sistemas operativos, el siguiente parámetro fue la licencia bajo la cual son liberados, el porcentaje de utilización en el mercado es el tercer parámetro así como también la curva de aprendizaje de las diferentes herramientas para el desarrollo

| Sistema Operativo | Portabilidad | Licencia | Porcentaje en mercado |
|---|---|---|---|
| **ANDROID** | N Marcas | Open Source | 61,18% |
| **IOS** | Apple | Privativa | 14,44% |
| **BLACKBERRY** | Blackberry | Privativa | 5,99% |
| **WINDOWS PHONE** | NOKIA | Privativa | 5,22% |

*TABLA 3. COMPARATIVA SISTEMAS OPERATIVOS DISPOSITIVOS MÓVILES*

Se descartó el desarrollo de una aplicación híbrida debido a que para el desarrollo en visión artificial se requiere componentes a bajo nivel y estos no son provistos por ningún framework de desarrollo híbrido. En base a esto se eligió Android como el sistema operativo para el desarrollo del proyecto, debido a que es software libre es compatible para distintas marcas de dispositivos además que tiene el mayor porcentaje de usuarios en el mercado.

Se evaluó también las herramientas de visión artificial que se pueden aplicar en el proyecto en base a cuatro parámetros escogidos en base al criterio del autor, para ello se inició con los sistemas operativos en lo que es compatible la herramienta, así como también la licencia bajo la cual fue liberada la herramienta y el lenguaje de programación en el cual se debe programar para hacer uso de la misma; finalmente se evaluó el costo de la licencia de las herramientas (Ver Tabla 5).

| Herramienta | Sistemas Operativos | Licencia | Lenguaje de Programación | Costo Licencia |
|---|---|---|---|---|

| | | | | |
|---|---|---|---|---|
| **OPENCV** | Windows, Linux, Mac, Android | Libre | C, Java | $0 |
| **MATLAB** | Windows | Privada | Matlab | $625 |
| **LABVIEW** | Windows ,Linux, Mac | Privada | Labview | $3300 |

*TABLA 5. COMPARATIVA HERRAMIENTAS DE VISIÓN ARTIFICIAL*

Se eligió OpenCv debido a que es multiplataforma, es software libre así como también el costo es gratuito.

### 1.1. Cálculo del Índice de Somnolencia.

Para el cálculo del índice de somnolencia se puede escoger dos métodos los cuales fueron implementados para luego realizar el testeo de los mismos

| FACTOR | PERCLOS | HORN Y DONG |
|---|---|---|
| **TEORIA** | Indica que al sobrepasar un determinado umbral detecta síntomas de somnolencia en este caso el umbral es 80% | Implica que si durante cinco fotogramas consecutivos se detecta que los ojos están cerrados se emite una alarma de alerta ante un posible estado de adormecimiento. |
| **NUMERO DE PRUEBAS** | 50 | 50 |
| **DETECCIÓNES** | 45 (90%) | 40 (80%) |
| **FALSOS POSITIVOS** | 2 (4%) | 6 (12%) |
| **FALSOS NEGATIVOS** | 3 (6%) | 4 (8%) |

*TABLA 6. COMPARATIVA HERRAMIENTAS DE VISIÓN ARTIFICIAL*

Para analizar los resultados tendremos en cuenta la somnolencia detectada correctamente, la que no se ha detectado de ninguna manera (falsos negativos) y que detecciones no corresponden con la somnolencia (falsos positivos).Debido a esto se decidió usar el método de PERCLOS debido a que el número de detecciones es mayor reduciendo de esta manera los falsos positivos y negativos

**2. Desarrollar e implementar la aplicación móvil basada en visión artificial para la detección de somnolencia del conductor.**

Para cumplir con este objetivo se tuvieron que entrenar clasificiadores que al usarlos en la aplicación Android ayudaron a identificar distintos rasgos como los ojos y el rosto, que resultaron utiles para el proceso de detección de somnolencia .

## 2.1. Entrenamiento de clasificadores en OpenCv

Para ello es necesario tener instalada la librería OpenCv, el objetivo de este proceso fue obtener los archivos xml que posteriormente fueron usados en la aplicación Android para el reconocimiento en este caso del rostro.

### 2.1.1. Adquisición de imágenes

Se requiere adquirir dos tipos de imágenes positivas y negativas, las positivas son imágenes donde está presente el objeto en distintos ángulos niveles de iluminación etc. a detectar.

### 2.1.2. Negativas

Las imágenes negativas son cualquier grupo de imágenes en las cuales no aparece el objeto. Para estas imágenes hay que generar un archivo (negativas.txt) que sirve de índice para las etapas posteriores, con la siguiente estructura:

negativas/img1.jpg
negativas/img2.jpg
negativas/imgn.jpg

### 2.1.3. Positivas

Para estas imágenes también debe generarse un archivo (positivas.txt), en este archivo se especifica donde se encuentra el objeto dentro de la imagen positiva. Siendo las 2 primeras coordenadas x e y (respectivamente) y las siguientes 2 ancho y largo de la imagen.

positivas/img1.jpg 100 100 25 25
positivas/img2.jpg 120 110 27 28
positivas/imgn.jpg 140 105 22 23

### 2.1.4. Crear muestra.

Para crear la muestra de imágenes positivas se usó el siguiente comando

**opencv-createsamples -info positivas.txt -vec data/muestra.vec-num 1500 -w 20 -h 20**

Donde

- **info:** es la ubicación del archivo con el índice las imágenes positivas.
- **vec:** es el nombre del archivo de salida con la muestra generada.
- **num:** es la cantidad de imágenes positivas.
- **w:** ancho de la muestra de salida.
- **h:** alto de la muestra de salida.

Se puede verificar la correcta generación del archivo con el siguiente comando:

**opencv-createsamples -vec positivas.vec -w 20 -h 20**

### 2.1.5. Entrenamiento

Para esta etapa se usó, el archivo índice generado con las imágenes negativas y el archivo de muestras generado en la etapa anterior de esta manera.

***opencv-haartraining -data data/cascade -vec data/muestra.vec* -bg negativas.txt -nstages 30 -nsplit 2 -minhitrate 0.999 -maxfalsealarm 0.5 -npos 1500 -nneg 5000 -w 20 -h 20 -mem 1300 -mode ALL**

- **data:** directorio de salida para la generación del entrenamiento.
- **vec:** archivo con muestras generado en la etapa anterior.
- **bg:** archivo índice con las imágenes negativas.
- **nstages:** número de etapas de entrenamiento. Es directamente proporcional a la calidad del clasificador a generar.
- **nsplit:** debilidad del clasificador.
- **minhitrate:** mínimo rango de cada etapa.
- **npos:** número de imágenes positivas indexadas.

- **nneg:** número de imágenes negativas indexadas.
- **w:** ancho.
- **h:** alto.
- **mem:** memoria a utilizar en el proceso.

Como resultado de este proceso obtuvimos dos clasificadores xml que serán usados en la aplicación para la detección del rostro y los ojos.

## 2.2. Aplicación Móvil

Para el desarrollo de la aplicación móvil se siguieron las etapas de la metodología de desarrollo a continuación se detalla cada una de ellas.

### 2.2.1. Exploración

Los objetivos de esta etapa son:

- Establecer los grupos de interés y definir el alcance para la aplicación.

Se desarrolló una aplicación Android que permite detectar somnolencia usando la cámara del dispositivo y aplicando redes neuronales para la determinación, se creará una alerta en caso de encontrar somnolencia.

Se inició con el análisis de los posibles casos de éxito, herramientas a utilizar, seleccionar los requerimientos, se procederá con la codificación y finalmente se realizaran pruebas para determinar la funcionalidad de la aplicación así como para encontrar su margen de error y sus limitaciones.

#### 2.2.1.1. Selección de los grupos de interés

La aplicación es orientada a conductores de vehículos sean profesionales o no pero con especial énfasis en el transporte público donde es más probable que un conductor debido a las horas laboradas presente somnolencia.

#### 2.2.1.2. Requerimientos

| Identificación del requerimiento | REQ 1. |
|---|---|

| | |
|---|---|
| **Nombre del Requerimiento** | Detección de somnolencia |
| **Descripción del requerimiento** | Detectar Somnolencia en un conductor utilizando Cámara del Teléfono y emitir alerta de emergencia |
| **Prioridad del requerimiento** | Alta |

| **Identificación del requerimiento** | **REQ 2.** |
|---|---|
| **Nombre del Requerimiento** | Escoger Sonido |
| **Descripción del requerimiento** | Permitir al Usuario elegir el sonido de la alerta de emergencia. |
| **Prioridad del requerimiento** | Media |

#### 2.2.1.3. **Módulos**

En la aplicación tiene un módulo que cumple con el propósito de mantener activa la cámara, detectando el rostro si se encuentra en estado de somnolencia y procediendo a mostrarnos una alerta.

### 2.2.2. **Inicialización**

Los objetivos de esta etapa son:

- Preparar los recursos, físicos, humanos y técnicos del proyecto

#### 2.2.2.1. **Recursos**

| RECURSO | TIPO | DESCRIPCIÓN |
|---|---|---|
| Smartphone. | **HADWARE** | Se utilizó un celular para realizar las pruebas de funcionamiento |
| Computador | **HADWARE** | Se utilizó Para el desarrollo de la aplicación móvil. |

| | | |
|---|---|---|
| Android | **SOFTWARE** | Sistema operativo en el que se desarrolló la aplicación. |
| OpenCv | **SOFTWARE** | Herramienta para el desarrollo de visión artificial. |
| Android Studio | **SOFTWARE** | IDE utilizado para codificar la aplicación |
| NDK. | **SOFTWARE** | Nos permite compilar aplicaciones C en Android. |

*TABLA 7. RECURSOS DEL PROYECTO*

#### 2.2.2.2. **Limitaciones**

- La aplicación móvil solo podrá ser ejecutada en la plataforma Android.
- Versión de Android 4.0 en adelante.
- La aplicación necesita condiciones de iluminación naturales

#### 2.2.2.3. **Supuestos y dependencias**

- Los usuarios tienen conocimiento básico en el manejo de Android.
- Las condiciones de luz en el ambiente son las adecuadas.
- El usuario tendrá que despejar el área de sus ojos, sin gafas o cualquier otra cosa que dificulte el reconocimiento de los ojos
- El usuario coloca su teléfono en una posición adecuada dentro del vehículo (Ver Figura 10).

*FIGURA 10 POSICIÓN ADECUADA DEL DISPOSITIVO MÓVIL EN EL VEHÍCULO*

### 2.2.2.4. Configuración del ambiente de desarrollo

- **NDK.** Descargar e instalar NDK para Android (NDK-R10E).
- **OpenCv.** Descargar OpenCv para Android.
- **Android Studio.** Descargar Android Studio desde la página oficial de Android
- **Integrar Android Studio y OpenCv.** Para integrar OpenCv y Android es necesario crear un nuevo proyecto de Android.

Una vez creado el proyecto accedemos al directorio procedemos a crear una carpeta llamada libs en esta carpeta procedemos a copiar OpenCv para Android previamente descargado.

Lo siguiente será modificar el gradle para poder incluir la librería quedando como se observa en la figura 11 siendo libs la carpeta que creamos y OpenCv la librería incluida.

```
}

dependencies {
    compile fileTree(include: ['*.jar'], dir: 'libs')
    compile 'com.android.support:appcompat-v7:22.2.0'
    compile project(':libs:opencv')
}
```

*FIGURA 11 INTEGRAR OPENCV Y ANDROID*

### 2.2.2.5. Apreciación global de la aplicación.

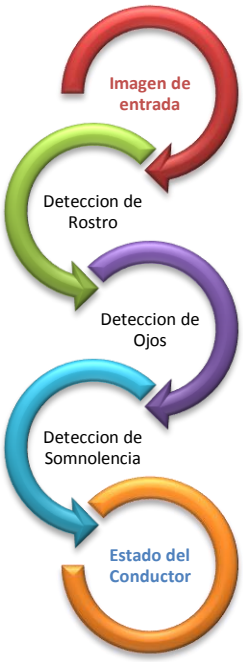



*FIGURA 12 APRECIACIÓN GLOBAL DE LA APLICACIÓN*

Se detalla el flujo normal a seguir durante la ejecución de la aplicación

**Detector del rostro**: encargado de detectar la posición del rostro del conductor dentro de la imagen.

**Detector de los ojos:** localiza la posición de los ojos en la imagen

**Estado del conducto**: analizar el estado de los ojos y generar el índice de somnolencia.

### 2.2.3. **Producción, estabilización, pruebas**

Los objetivos de estas etapas son:

- Implementar las funcionalidades.
- Finalizar la implementación del producto.
- Mejorar y Asegurar la calidad del producto.
- Corregir los defectos.
- Producir una aplicación libre de errores en lo posible.

#### 2.2.3.1. **Prototipos**

*FIGURA 13 PANTALLA DE INICIO DE LA APLICACIÓN*

En la figura 13 se observa la pantalla de bienvenida que se muestra a los usuarios de la aplicación

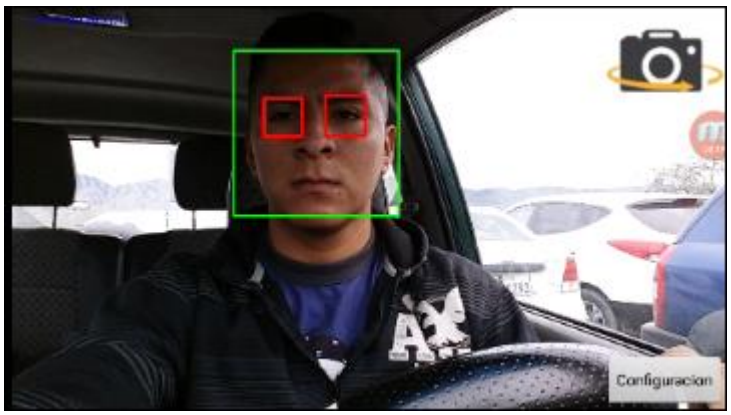



*FIGURA 14 PANTALLA ESTADO NORMAL DEL CONDUCTOR*

En la figura 14 se muestra la pantalla principal de la aplicación que consta de dos botones el primero permite cambiar de cámara principal a secundaria del teléfono y viceversa, el botón de configuración permite elegir un sonido de alerta de acuerdo al gusto del usuario.

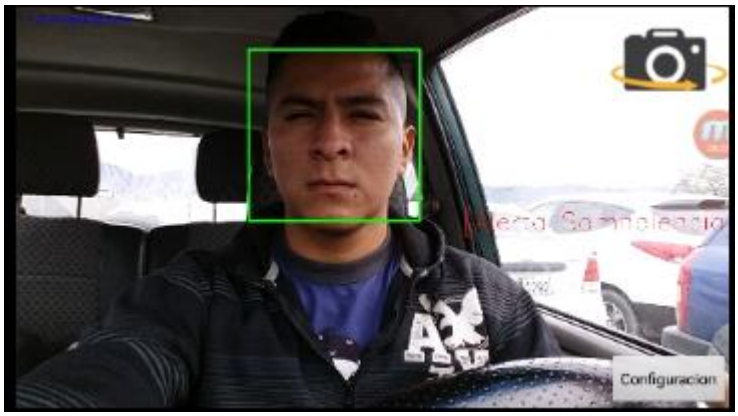



*FIGURA 15 PANTALLA ESTADO SOMNOLENCIA DEL CONDUCTOR*

En la figura 15 se muestra la pantalla principal de la aplicación cuando el conductor presenta síntomas de somnolencia en este caso presenta un mensaje en letras rojas con las palabras Alerta de Somnolencia.

*FIGURA 16 PANTALLA ALERTA DE SOMNOLENCIA*

En la figura 16. Se muestra la pantalla de alerta de la aplicación que es mostrada cuando se detecta somnolencia que debe ser tocada para regresar al proceso normal de detección.

2.2.3.2. **Casos de uso**

| REQ 1. DESCRIPCIÓN DEL CASO DE USO |
|---|
| ACTOR (es) del caso de uso |
| Usuario: **Persona que utilizara la aplicación móvil para detectar somnolencia** |
| Aplicación: **Aplicación móvil que detecta la somnolencia de un conductor.** |

| |
|---|
| Identificador del Caso de uso: **CU001: • Detectar Somnolencia en un conductor utilizando Cámara del Teléfono.** Se detalla el proceso necesario para la detección de somnolencia del conductor utilizando la cámara del teléfono. |
| Secuencia de eventos |
| Flujo Normal - Detectar Somnolencia en un conductor utilizando cámara del teléfono. |
| 1. **El usuario coloca su teléfono de manera que su rostro quede cerca de la cámara del vehículo** |
| 2. **La aplicación detecta su rostro.** |
| 3. **La aplicación detecta sus ojos.** |
| 4. **La aplicación calcula el índice de somnolencia y es mayor al 80%** |
| 5. **La aplicación emite una alerta sonora.** |
| 6. **El usuario toca la pantalla del teléfono.** |
| 7. **La aplicación regresa al evento 2.** |
| Flujo Alterno 1. A partir del Evento 4 |
| 4. **La aplicación calcula el índice de somnolencia y es menor al 80%** |
| 5. **La aplicación vuelve al evento 3** |
| Flujo Alterno 2. A partir del Evento 5 |
| De no realizar el evento 6 por parte del usuario la aplicación seguirá en el evento 5 |

| REQ 2. DESCRIPCIÓN DEL CASO DE USO |
|---|
| ACTOR (es) del caso de uso |
| Usuario: **Persona que utilizara la aplicación móvil para detectar somnolencia** |
| Aplicación: **Aplicación móvil que detecta la somnolencia de un conductor.** |

| |
|---|
| Identificador del Caso de uso **CU002: •Permitir al Usuario elegir el sonido de la alerta de emergencia.** Se detalla el proceso necesario para que el usuario pueda elegir el sonido de la alerta de emergencia. |
| Secuencia de eventos |
| Flujo Normal - Permitir al Usuario elegir el sonido de la alerta de emergencia. |
| **1. El usuario presiona el botón de configuración** |
| **2. La aplicación muestra una lista de sonidos.** |
| **3. El usuario elige un sonido** |
| **4. La aplicación guarda el sonido escogido.** |

2.2.3.3. **Diagrama de componentes**

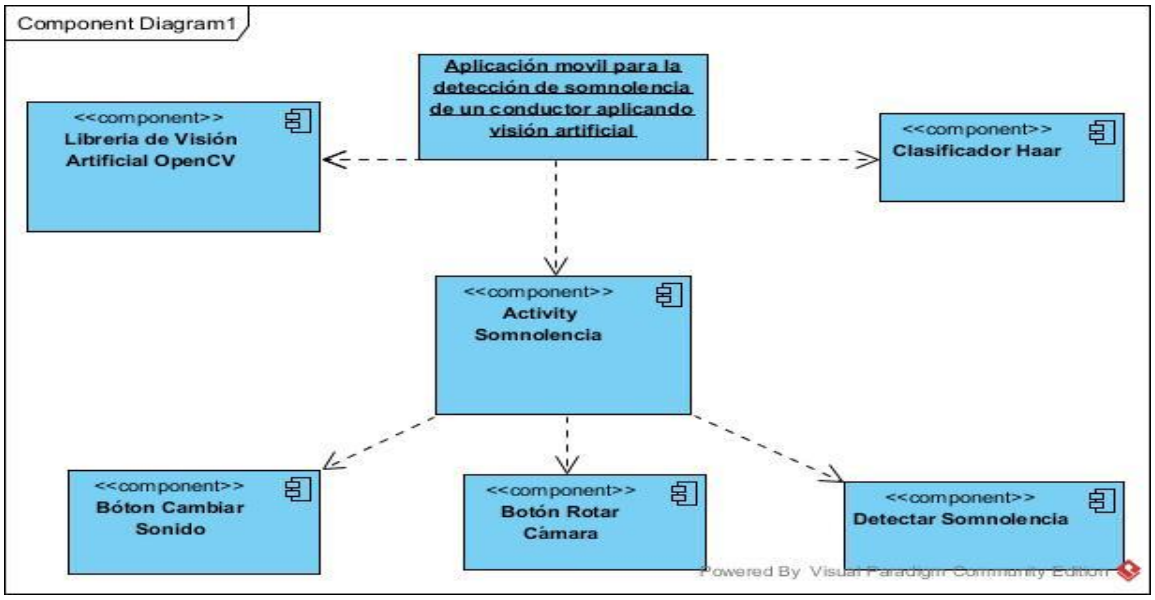



*FIGURA 17 DIAGRAMA DE COMPONENTES*

Cada uno de los componentes de la aplicación (Figura 17) tendrá la siguiente relación:

- **Librería de visión artificial OpenCv:** Utilizada para el procesamiento digital de imágenes.
- **Clasificador Haar:** Archivo previamente entrenado por una red neuronal que sirve para encontrar rostro y ojos dentro de la aplicación
- **Activity Somnolencia:** Es el medio de comunicación entre los métodos. la librería y clasificadores.
- **Botón cambiar sonido:** El medio con el que el usuario va a interactuar para cambiar el sonido de alerta de la Aplicación.
- **Botón rotar cámara:** El medio con el que el usuario va a interactuar para cambiar la cámara del teléfono a principal o secundaria.
- **Detectar somnolencia:** Es el core de la aplicación aquí se realiza el proceso de detección en base al índice de Perclos

### 2.2.3.4. **Diagrama de Paquetes**

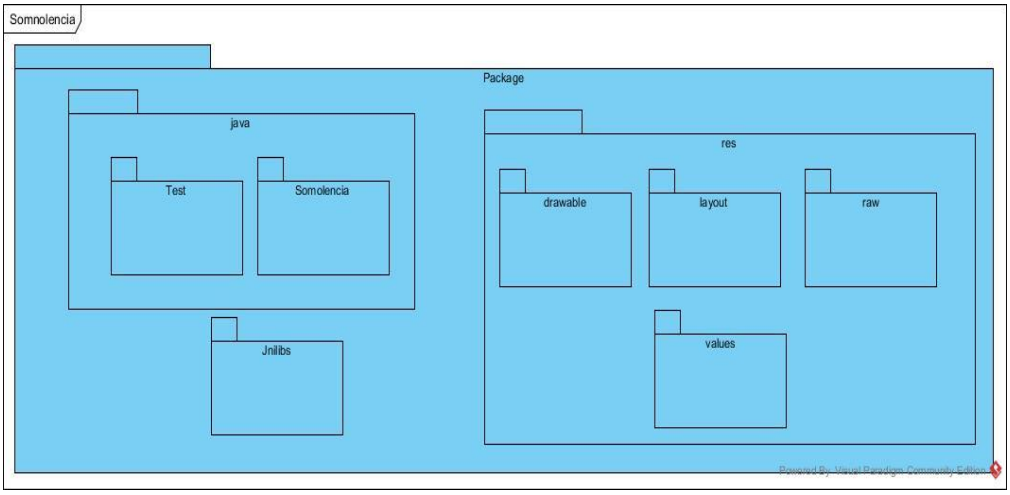



*FIGURA 18 DIAGRAMA DE PAQUETES*

Se detalla los paquetes principales de la aplicación

- **Java** Código Java de la aplicación
  - **Test** Paquete con las clases de pruebas Unitarias.
  - **Somnolencia** Paquete con el código core de la aplicación.
- **Res** Recursos de la Aplicación.
  - **Drawable** Estilos de botones.
  - **Layout** Vistas de la aplicación.
  - **Raw** Recursos como los clasificadores xml.
- **Jnilibs** Librerías en C para el trabajo directo con el dispositivo de acuerdo a la arquitectura de cada uno.

### 2.2.3.5. Codificación de la aplicación Android.

Se detalla las clases y los métodos principales de la aplicación. [26]

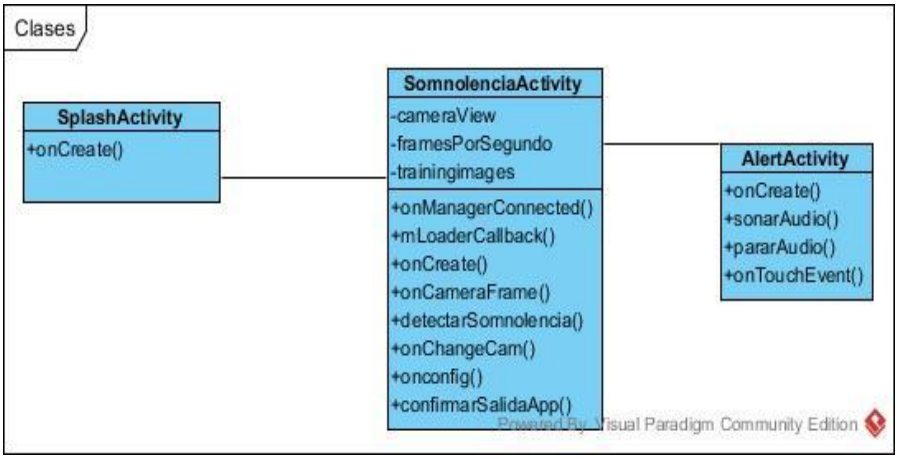



*FIGURA 19 DIAGRAMA DE CLASES*

**Clases**

**SPLASHACTIVITY** Esta Clase nos muestra una pantalla de Bienvenida y después de unos segundos pasa a la pantalla principal de la aplicación.

**SOMNOLENCIAACTIVITY** Clase donde se realiza la detección de la somnolencia

**Métodos**

Se detallan los métodos de la clase SomnolenciaActivity.

**onManagerConnected** Método provisto por OpenCv que verifica si OpenCv manager está instalado en el teléfono.

**mLoaderCallback** Aquí se cargan los archivos XML previamente entrenados en OpenCv para usarlos en nuestra aplicación en este caso se ha usado:

- **haarcascade_frontalface.xml** Archivo que nos permite detectar un rostro.
- **haarcascade_eye_tree_eyeglasses.xml** Archivo para la detección de ojos.

```
public void onManagerConnected(int status) {
    switch (status) {
    case LoaderCallbackInterface.SUCCESS: {
        Log.i(TAG, "OpenCV loaded successfully");
        try {
            // Cargar clasificador 1
            InputStream is = getResources().openRawResource(
                    R.raw.haarcascade_frontalface);
            File cascadeDir = getDir("cascade", Context.MODE_PRIVATE);
            mCascadeFile = new File(cascadeDir,
                    "haarcascade_frontalface.xml");
            FileOutputStream os = new FileOutputStream(mCascadeFile);

            byte[] buffer = new byte[4096];
            int bytesRead;
            while ((bytesRead = is.read(buffer)) != -1) {
                os.write(buffer, 0, bytesRead);
            }
            is.close();
            os.close();
            // Cargar clasificador 2
            InputStream iser = getResources().openRawResource(
                    R.raw.haarcascade_eye_tree_eyeglasses);
            File cascadeDirER = getDir("cascadeER",
                    Context.MODE_PRIVATE);
            File cascadeFileER = new File(cascadeDirER,
                    "haarcascade_eye_tree_eyeglasses.xmL");
            FileOutputStream oser = new FileOutputStream(cascadeFileER);

            byte[] bufferER = new byte[4096];
            int bytesReadER;
            while ((bytesReadER = iser.read(bufferER)) != -1) {
                oser.write(bufferER, 0, bytesReadER);
```

*FIGURA 20 MÉTODO PARA CARGAR CLASIFICADORES*

**onCreate** Método ejecutado al iniciar la actividad aquí asignamos el layout para nuestro activity además de iniciar la cámara del dispositivo.

**onCameraFrame** Obtenemos la entrada de la cámara y lo asignamos a una matriz la enviamos al método detectar.

```
public Mat onCameraFrame(CvCameraViewFrame inputFrame) {
    mRgba = inputFrame.rgba();
    mGray = inputFrame.gray();
    if (mAbsoluteFaceSize == 0) {
        int height = mGray.rows();
        if (Math.round(height * mRelativeFaceSize) > 0) {
            mAbsoluteFaceSize = Math.round(height * mRelativeFaceSize);
        }
    }
    MatOfRect faces = new MatOfRect();
            if (mJavaDetector != null)
            mJavaDetector.detectMultiScale(mGray, faces, 1.1, 2,
                    2, // TODO: objdetect.CV_HAAR_SCALE_IMAGE
                    new Size(mAbsoluteFaceSize, mAbsoluteFaceSize),
                    new Size());
    Rect[] facesArray = faces.toArray();
    detectarSomnolencia(facesArray);
    return mRgba;
}
```

*FIGURA 21 MÉTODO ONCAMERAFRAME*

**detectarSomnolencia** Detectamos el rostro y en base a la matriz con el índice de Perclos que se obtiene sacando los frames por segundo de la cámara y viendo en cuántos de ellos los ojos están cerrados utilizando el clasificador, promediándolas y si el índice de Perclos es mayor al 80% se emite un alerta de somnolencia.

```
public void detectarSomnolencia(Rect[] facesArray){
    //Deteccion de rostro
    for (int i = 0; i < facesArray.length; i++) {
        Imgproc.rectangle(mRgba, facesArray[i].tl(), facesArray[i].br(),
                FACE_RECT_COLOR, 3);
        Rect r = facesArray[i];
        //obtencion del area de los ojos en rgb y en gris
        Mat areaojo=mGray.submat(r);
        Mat areaojorgb=mRgba.submat(r);
        if (learn_frames < 10) {
            learn_frames++;
        } else {
            MatOfRect eye = new MatOfRect();
            if (mAbsoluteEyeSize == 0) {
                int height = areaojo.rows();
                if (Math.round(height * mRelativeEyeSize) > 0) {
                    mAbsoluteEyeSize = Math.round(height * mRelativeEyeSize);
                }
            }
            if (mJavaDetectorEye!= null)
                    mJavaDetectorEye.detectMultiScale(areaojo, eye, 1.1, 2,
                            2, // TODO: objdetect.CV_HAAR_SCALE_IMAGE
                            new Size(mAbsoluteEyeSize, mAbsoluteEyeSize),
                            new Size());
            Rect[] eyeArray = eye.toArray();
            if(eyeArray.length>0){
                for (int j = 0; j < eyeArray.length; j++) {
                    Imgproc.rectangle(areaojorgb, eyeArray[j].tl(), eyeArray[j].br(),
                            new Scalar(255, 0, 0, 255), 6);
```

*FIGURA 22 MÉTODO DETECTAR SOMNOLENCIA (OBTENCIÓN DE ROSTRO Y ÁREA DE LOS OJOS)*

```
if (contadoralertas>=porcentajePeclos){
    Imgproc.putText(mRgba, "[Alerta Somnolencia]",
            new Point(800, 400),
            Core.FONT_HERSHEY_SIMPLEX, 1.5, new Scalar(255, 0, 0, 255));
    contadoralertas=0;
    learn_frames=0;
    Intent mainIntent = new Intent().setClass(
            SomnolenciaActivity.this, AlertActivity.class);
    startActivity(mainIntent);
}
```

*FIGURA 23 MÉTODO DETECTAR SOMNOLENCIA (COMPARACIÓN CON PORCENTAJE DE PERCLOS Y ENVIÓ DE ALERTA)*

**onChangeCam** Se cambia la cámara del teléfono para la detección ya sea delantera o posterior.

**ALERTACTIVITY** Es mostrada una vez se detecta somnolencia y emite un sonido para ayuda del usuario

**Métodos**

Se detallan los métodos de la clase AlertActivity.

**sonarAudio** Reproduce un audio utilizando todo el volumen del dispositivo.

**pararAudio** Detiene la reproducción del Audio.

**onTouchEvent** Llama al método de detener el audio y nos regresa a la actividad de detección de somnolencia.

### 2.2.3.6. Pruebas.

**S**e pretende detallar las pruebas necesarias para evaluar la aplicación las mismas que se dividen en dos pruebas de funcionalidad, y pruebas de factibilidad a continuación se detalla el plan para cada una de ellas.

- **Funcionalidad.**
  - **Pruebas Unitarias** Pruebas al correcto funcionamiento de los métodos principales de la aplicación con el fin de evitar errores.
  - **Pruebas de Compatibilidad**. Dispositivos en los que puede ser ejecutada la aplicación.
  - **Pruebas en un dispositivo.**

Pruebas de la aplicación en dispositivos reales con sistema operativo Android, celulares, tablets, etc.

- **Factibilidad.**
  - **Validación** Pruebas con sujetos reales para determinar el margen de error de la aplicación.

### 2.2.3.7. Pruebas de Funcionalidad

**Unitarias**

Las pruebas se aplicaron para validar el correcto funcionamiento del código estas se aplicaron a las 3 clases y a los 9 métodos de la aplicación para ello se usó JUnit junto con el Gradle de Android Studio se preparó un conjunto de Test en una Clase llamada Tests del paquete creado AndroidTest.testsomnolencia.

*FIGURA 24 PRUEBAS UNITARIAS*

De acuerdo a la figura 24 las pruebas fueron exitosas siendo los métodos más importantes de la aplicación.

- mLoaderCallback
- detectarSomnolencia
- onChangeCam

**Compatibilidad.**

Para esta fase de pruebas se ha hecho uso de la plataforma web apkudo que proporciona una solución robusta que permite conocer de forma acertada en que plataformas de Android se ejecutara de manera adecuada la aplicación.

En esta página procedimos a subir nuestro archivo apk (ejecutable en Android) para que realice el análisis de la misma, el análisis estará listo dentro de unas horas y se pudieron obtener los siguientes resultados

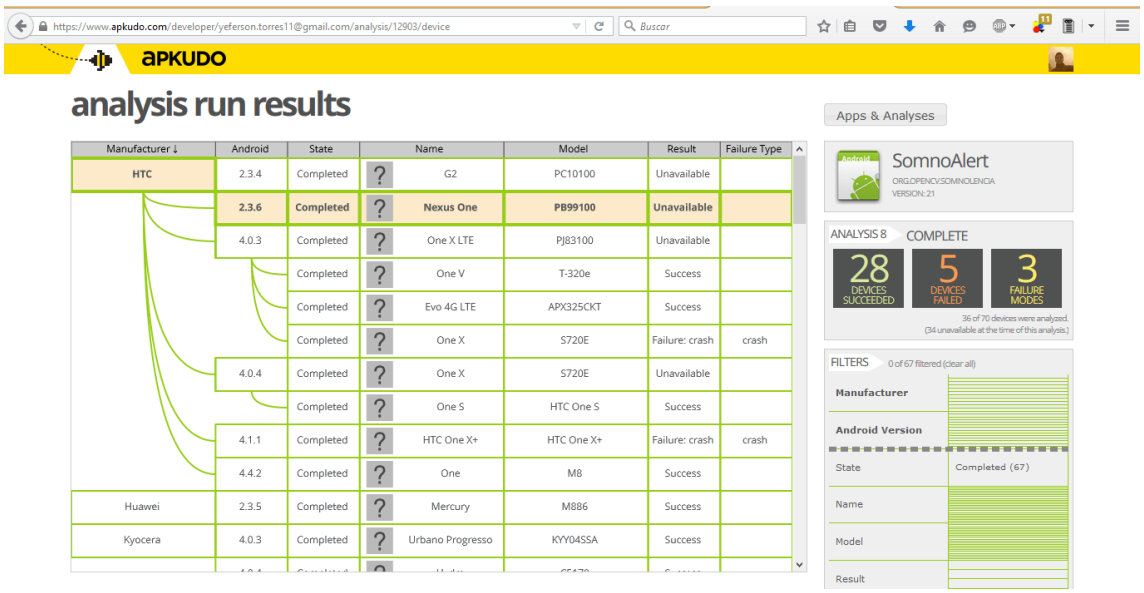



*FIGURA 25 RESULTADOS PLATAFORMA APKUDO*

La plataforma filtra por marca, luego plataforma de Android, nombre del dispositivo, resultado obtenido y modelo del dispositivo.

Como podemos observar la aplicación se ejecuta en 28 de los dispositivos, en 5 no pudo ser ejecutada y en 3 se ejecutó con errores, esto debido a que la versión mínima para la ejecución de la aplicación es Android 4.0.

**Pruebas en un dispositivo.**

Las pruebas realizadas en los dispositivos físicos dieron como resultado, éxito en él ya que su ejecución fue satisfactoria.

Las pruebas se realizaron en dos dispositivos de las siguientes características.

Numero de modelo: Samsung GT-19300

Versión de Android: 4.2.2 JELLY BEAN

Numero de modelo: Samsung J-5000

Versión de Android: 5.1.1 LOLILLOP

### 2.2.3.8. Pruebas de Factibilidad

**A. Diseño de Prueba**

| | |
|---|---|
| **Alcance** | Aplicación móvil para la detección de somnolencia de un conductor aplicando visión artificial |
| **Ítems a probar** | • Cambio de sonido<br>• Detección de somnolencia |
| **Estrategia** | Manejo de la aplicación por parte de los usuarios.<br>Obtener información a partir de las encuestas realizadas |
| **Recursos** | Smartphone<br>Computador |

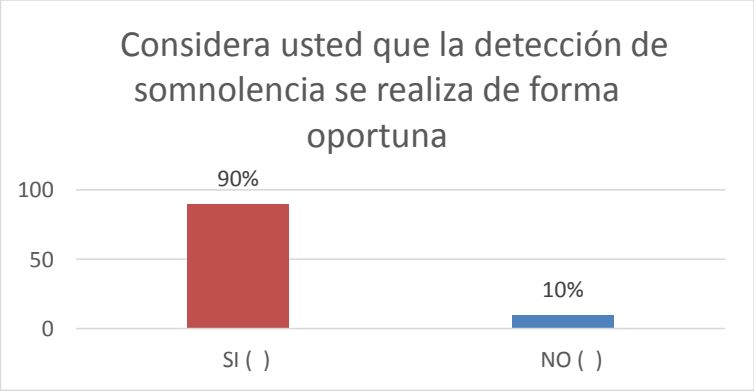
*TABLA 8. DISEÑO DE PRUEBAS DE FACTIBILIDAD*

**B. Análisis de las Pruebas de Validación**

Se realizaron pruebas (Ver anexo 2) en distintas condiciones de luminosidad el resultado de las mismas es el siguiente.

**1. ¿Considera usted que la detección de somnolencia se realiza de forma oportuna?**

**SI ( )** **NO ( )**



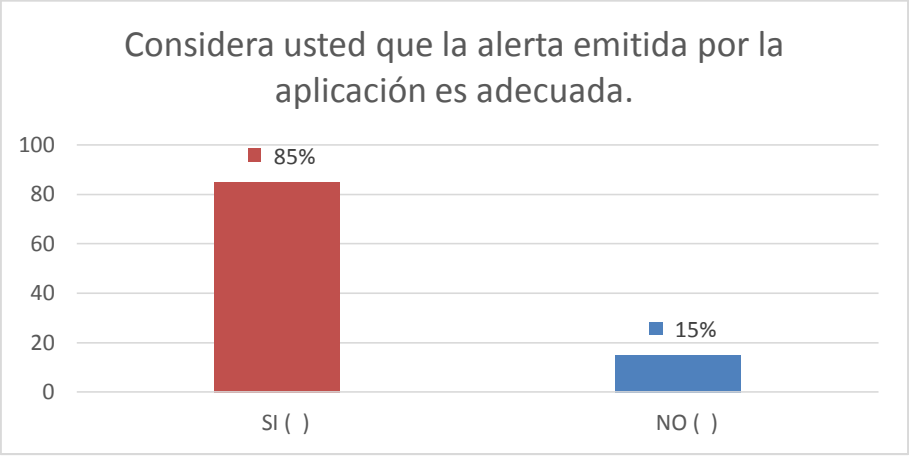

**Análisis**

Se puede observar que el 90% de usuarios considera que la detección de somnolencia es oportuna debido a que la aplicación respondió correctamente y en el caso del 10% que

manifestó lo contrario es debido a que las condiciones de luminosidad no eran las adecuadas.

**2. ¿Considera usted que la alerta emitida por la aplicación es adecuada?**

**SI ( )** **NO ( )**

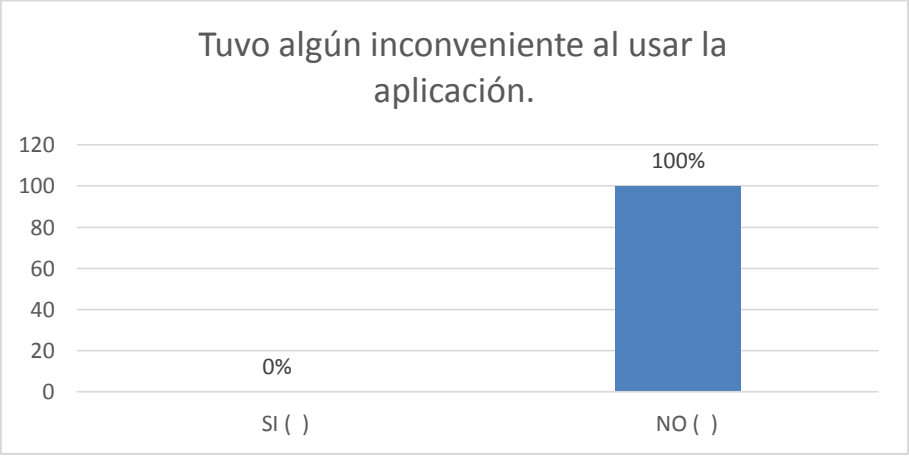



**Análisis**

Se puede observar que el 85% de usuarios considera que la alerta emitida por la aplicación es adecuada debido a que el sonido emitido es alto y en el caso del 15% que manifestó que la alerta no es adecuada y sugirieron que debería ser más alta puesto que cuando se está en estado de somnolencia algunos sonidos no se escuchan.

**3. ¿Tuvo algún inconveniente al usar la aplicación?**

**SI ( )** **NO ( )**

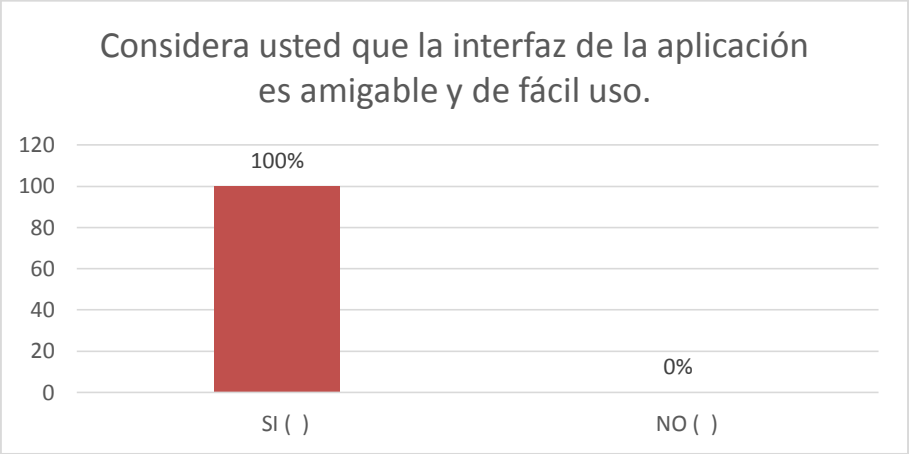


**Análisis**

Se puede observar que el 100% de usuarios no tuvo inconveniente al usar la aplicación esto debido a que la interacción el usuario es poca.

**4. ¿Considera usted que la interfaz de la aplicación es amigable y de fácil uso?**

**SI ( )** **NO ( )**

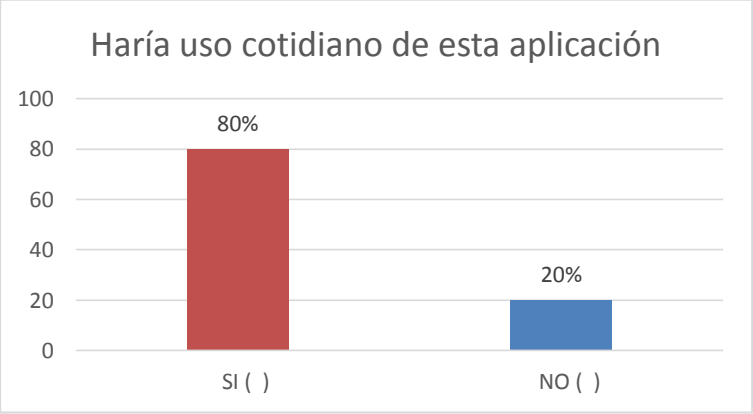



**Análisis**

Se puede observar que el 100% considera que la interfaz de la aplicación es amigable y de fácil uso.

**5. ¿Haría uso cotidiano de esta aplicación?**

**SI ( )** **NO ( )**

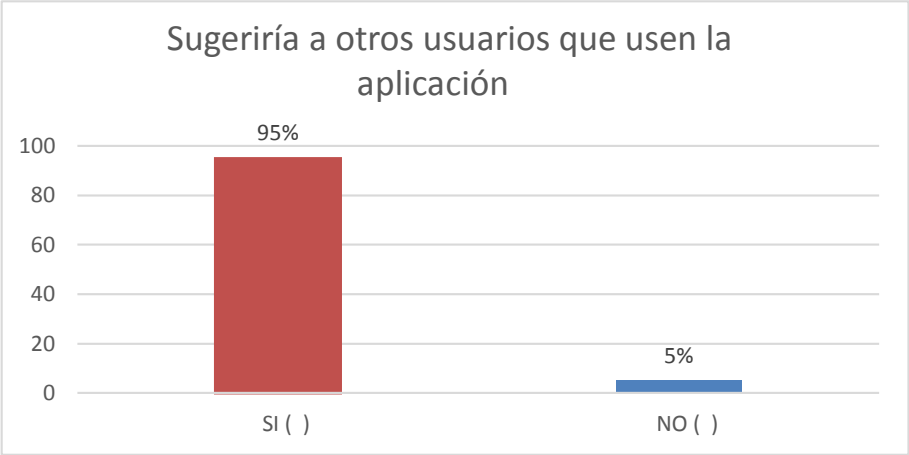


**Análisis**

Se puede observar que el 80% haría uso cotidiano de la aplicación el 20% manifestó que no la usaría cotidianamente en especial por que no cuentan con un dispositivo.

**6. ¿Sugeriría a otros usuarios que usen la aplicación?**

**SI ( )** **NO ( )**




**Análisis**

Se puede observar que el 95% de usuarios recomendaría a otros usar la aplicación y un 5% manifestó que no lo haría.

En base a las pruebas de validación se obtuvo que el margen de acierto de la aplicación es 90% así como también que la mayoría de usuarios piensa que la aplicación ayudara a prevenir accidentes de tránsito que la interfaz de la misma es amigable y que además de hacer uso cotidiano de la aplicación recomendarían a otras personas usarla.

# 7. Discusión.

## 1. Desarrollo de la propuesta alternativa.

Para llevar a cabo la realización del presente proyecto de titulación se evaluaron cada uno de los objetivos específicos planteados, indicando la forma en la que se fueron cumpliendo. Los cuales se detallan a continuación.

- **Analizar técnicas, herramientas y casos de éxito de visión artificial para la detección de somnolencia en un conductor.**

Para cumplir el objetivo planteado se visitó varias páginas webs de universidades, bibliotecas virtuales, repositorios digitales con el fin de encontrar las mejores herramientas y técnicas de visión artificial también se evaluaron cada una de ellas de acuerdo a los requerimientos del proyecto; se revisó casos de éxito con el objetivo de ver hasta donde ha llegado la investigación de los autores, que herramientas usaron, cuáles fueron sus limitaciones. En base a estos factores se decidió utilizar el método de detección de somnolencia basado en el Análisis de los Ojos y Expresiones Faciales la Aplicación fue desarrollada para el Sistema Operativo Android y como método de cálculo del Índice de somnolencia de uso PERCLOS debido a que en la experimentación comparando con el método de Horn y Dong se obtuvo mejores resultados, Además se eligió la librería de Vision Artificial OpenCv debido a que es multiplataforma y Open Source frente a otras opciones como Matlab y Labview siendo software privativo y no compatible con dispositivos móviles.

- **Desarrollar e implementar la aplicación móvil en Android basada en visión artificial para la detección de somnolencia del conductor.**

Se entrenaron Clasificadores para OpenCv recolectando para ellos 3000 imágenes para obtener un índice de detección de rostros y ojos alto. Se siguió los pasos de la metodología MOBILE-D entre otras cosas por la facilidad para detectar y resolver tempranamente problemas técnicos, baja densidad de defectos en las liberaciones de los productos, se basa en el desarrollo basado en pruebas.

Se levantaron requerimientos, establecieron grupos de interés, supuestos, dependencias; además se redactaron casos de uso, diagramas de componentes, paquetes y finalmente se codifico la Aplicación usando Android Studio.

- **Aplicar pruebas de funcionalidad y factibilidad al prototipo desarrollado de la aplicación móvil Android**

Para ello se realizó pruebas de Compatibilidad usando la plataforma APKUDO, pruebas unitarias con Junit en los métodos principales de la Aplicación, además pruebas con los usuarios finales de la aplicación en el Consorcio de transportistas Urbanos Ciudad de Loja mediante el uso de la aplicación y encuestas de esta manera se obtuvo el margen de error de la aplicación que en este caso es 90%.

## 2 VALORACIÓN TÉCNICA – ECONÓMICA- SOCIAL

Para el desarrollo del Trabajo de Titulación se utilizaron recursos técnicos, económicos, humanos que contribuyeron al cumplimiento de cada uno de los objetivos planteados.

### VALORACIÓN ECONÓMICA

### RECURSOS HUMANOS

| Descripción | Cantidad | Nº de Horas | Valor unitario | Valor Total |
|---|---|---|---|---|
| • Investigador del trabajo de titulación | 1 | 420 | 5 | $ 2100,00 |
| • Director del trabajo | 1 | -- | -- | -- |
| | | | SUBTOTAL | $ 2100,00 |

*TABLA 9. RECURSOS HUMANOS*

### RECURSOS TÉCNICOS Y TECNOLÓGICOS

| Descripción | Cantidad | Valor unitario | Valor Total |
|---|---|---|---|
| **HARDWARE** | | | |

| | | | |
|---|---|---|---|
| • Computadora Dell Inspiron | 1 | $ 600,00 | $ 600,00 |
| • Celular Galaxy S3 I9300 | 1 | $ 300,00 | $ 260,00 |
| • Celular Galaxy J5 | 1 | $ 400,00 | $ 400,00 |
| **SOFTWARE** | | | |
| • Plataforma Java | 1 | -- | -- |
| • OpenCv | 1 | -- | -- |
| • Android Studio | 1 | -- | -- |
| • NDK Android | 1 | -- | -- |
| | | SUBTOTAL | $ 2035,00 |

*TABLA 10. RECURSOS TÉCNICOS Y TECNOLÓGICOS*

| **Recursos** | **Subtotales** |
|---|---|
| Recursos Humanos<br>Recursos Técnicos y tecnológicos<br>Recursos Materiales | $ 1452,00<br>$ 2035,00<br>$ 194,20 |
| **Subtotal** | **$ 3681,00** |
| Imprevistos 10% | $ 368,12 |
| **Total** | **4049,12** |

*TABLA 11. PRESUPUESTO TOTAL*

# 8. Conclusiones

- La elección del PERCLOS (Porcentaje de tiempo que los ojos de una persona están cerrados) frente a otros métodos de obtención del índice de somnolencia permitió obtener un método robusto y eficiente para la detección de somnolencia.
- Se desarrolló una aplicación nativa Android ya que de esta manera se podía acceder directamente a los recursos de bajo nivel del dispositivo móvil a través del NDK en tiempo real.
- El uso de los métodos de la librería OpenCv y clasificadores ayudo a mejorar el proceso de detección de imágenes puesto que sus métodos y algoritmos son eficientes gracias a ello se pudo obtener un margen de error del 10%.
- La aplicación propuesta es un método de detección de somnolencia económico, fácil de usar para el usuario final y adaptable a cualquier vehículo o conductor.

# 9. Recomendaciones

- Usar una cámara infrarroja para adaptar la aplicación a condiciones con poca iluminación (Nocturno).
- Desarrollar futuros proyectos en otros sistemas operativos móviles partiendo como base las herramientas usadas y el índice de Perclos.
- Analizar el tiempo de conducción desde que el conductor aborda del vehículo para tener un factor más de análisis y poder hacer una combinación de técnicas de detección.
- Hacer análisis de datos en base a las alertas enviadas por la aplicación para de esta manera obtener horas en las que es propenso un conductor a sufrir somnolencia, sitios peligrosos, etc.

# 10. Bibliografía

## Referencias Bibliográficas


[1] NHTSA, «National highway trafic safety administration,» [En línea]. Available: http://www.nhtsa.dot.gov/.. [Último acceso: 02 07 2015].

[2] A. N. d. Transito, «ANT Estadisticas,» [En línea]. Available: http://www.ant.gob.ec/index.php/noticias/estadisticas. [Último acceso: 23 07 2015].

[3] NHTSA, «Evaluation of techniques for ocular measurement as an index of fatigue and the basis for alertness management.,» *Final Report DOT HS 808 762.* .

[4] R. A. Española, «Real Academia Española,» vigésima segunda edición, 2008. [En línea]. Available: http://www.rae.es. [Último acceso: 20 3 2015].

[5] G. J. L. H. Rey de Castro J, «Cansancio y somnolencia en conductores de ómnibus y accidentes de carretera en el Perú: estudio cuantitativo.,» *Rev Panam Salud Pública,* 2004.

[6] L. J. E. K. H. J. a. S. B. H. Chang B. C, «A study of classification of the level of sleepiness for the drowsy driving prevention. IEEE,» *SICE Annual Conference,* pp. 3084-3089, September 2007..

[7] G. O. A. Saavedra, «Universidad Tecnológica del Perú,» 2012. [En línea]. Available: DETECCIÓN DE SOMNOLENCIA, COMO ASISTENCIA PARA CONDUCTORES, UTILIZANDO TÉCNICAS DEPROCESAMIENTO DIGITAL DE IMÁGENES. [Último acceso: 27 07 2015].

[8] O. S. Fernando Alejandro Guevara, «ANALISIS DEL CONDUCTOR: ESTIMACIÓN DE LA DISTRACCIÓN Y SOMNOLENCIA MEDIANTE VISION POR COMPUTADOR E INTELIGENCIA ARTIFICIAL USANDO TECNOLOGIA TOF,» [En línea]. Available: http://repositorio.espe.edu.ec/bitstream/21000/6522/1/T-ESPE-047086.pdf.

[9] M. J. Flores Calero, «SISTEMA AVANZADO DE ASISTENCIA A LA CONDUCCIÓN MEDIANTE VISIÓN POR COMPUTADOR PARA LA DETECCIÓN DE LA SOMNOLENCIA,» [En línea]. Available: http://orff.uc3m.es/handle/10016/5678.

[10] J. C. G. Crespín Luis, «Sistema detector de somnolencia en secuencias de video de conductores manejando usando visión computacional,» 2014. [En línea]. Available: http://www.inf.unitru.edu.pe/revistas/2014/4.pdf.

[11] C. N. C. d. CyberSeguridad, «Malware en Smartphones,» 2014.

[12] StatCounter, «StatCounter,» 2015. [En línea]. Available: http://gs.statcounter.com/#mobile_os-ww-monthly-201409-201509.

[13] K. J. S. S. C. F. Carvajal Valdivieso, «Análisis comparativo entre las plataformas de desarrollo de aplicaciones móviles para los sistemas operativos Android y Ios.,» 22 Julio 2013. [En línea]. Available: http://dspace.e.

[14] D. Web, «Introducción a Android,» 2015. [En línea]. Available: http://www.desarrolloweb.com/articulos/introduccion-android.html.

[15] A. Inc, «Developers Android NDK,» 2015. [En línea]. Available: http://developer.android.com/ndk/reference/index.html.

[16] RIM, «Blackberry,» 2015. [En línea]. Available: http://www.rim.com/ BLACKBERRY.

[17] P. D. Barron, «Aplicacion Grafica para dispositivos IOS,» [En línea]. Available: https://riunet.upv.es/bitstream/handle/10251/18226/Memoria.pdf?sequence=1.

[18] E. “. p. todos”., «OpenCv,» Mayo 2014. [En línea]. Available: Disponible en: http://www.ecured.cu/index.php/OpenCV.

[19] V. M. Arévalo, J. González y G. Ambrosio, «Dpto. De Ingeniería de Sistemas y Automática, Universidad de Málaga. España,» [En línea]. Available: LA LIBRERÍA DE VISIÓN ARTIFICIAL OPENCV. [Último acceso: 07 07 2015].

[20] OpenCv, «Open CV Classifiers,» [En línea]. Available: Opencv.

[21] I. T. D. C. Madero., «Matlab e Interfaces Gráficas.,» 2012. [En línea]. Available: http://proton.ucting.udg.mx/~cheko/pdf/matv.pdf.

[22] Estuelectronic., «LabView,» 2012. [En línea]. Available: https://estuelectronic.wordpress.com/2012/08/06/que-es-y-para-que-sirve-labview/.

[23] A. S. t. R. Programme., «MobileD,» [En línea]. Available: http://agile.vtt.fi/mobiled.html.

[24] U. P. d. Madrid, «Introducción a la Visión Artificial,» [En línea]. Available: http://www.elai.upm.es/webantigua/spain/Asignaturas/MIP_VisionArtificial/ApuntesVA/cap1IntroVA.pdf. [Último acceso: 29 07 2015].

[25] Y. T. Berrú, «Aplicación Movil para la detección de somnolencia de un conductor aplicando visión artificial,» 12 08 2015. [En línea]. Available: https://bitbucket.org/ymtorresb/somnoalert. [Último acceso: 15 12 2015].

[26] O. H. a. N. T. Kaneda M, «Adaptability to ambient light changes for drowsy driving detection using image processing.,» pp. (98-S2-P-15).

[27] S. B. K. y. S. Harishankar, «Applying Agile Methodology in Mobile Software».

[28] V. R. y. R. Ramsin, «Designing an agile methodology for mobile software development: A hybrid method engineering approach».

[29] D. Astels, Test-driven development a practical, Londres: Prentice , 2003.

# 11. Anexos

## Anexo 1: Plantilla Requerimientos

**REQ 1. DESCRIPCIÓN DEL REQUIERIMIENTO**

**ACTOR (es) del caso de uso**

Se detalla los actores del sistema

**Identificador del Caso de uso**

**CU001: •**

Se detalla rápidamente el caso de uso

**Secuencia de eventos**

**Flujo Normal -**

**Flujo Alterno-**

## Anexo 2: Encuesta para validación de la aplicación

# UNIVERSIDAD NACIONAL DE LOJA

***Área de la Energía, las Industrias y los Recursos Naturales No Renovables***

CARRERA DE INGENIERÍA EN SISTEMAS

**TEMA:** Aplicación móvil para la detección de somnolencia de un conductor aplicando visión artificial.

# ENCUESTA

1. **Considera usted que la detección de somnolencia se realiza de forma opo rtuna**

   **SI ( )** **NO ( )**

   **Si respuesta es no diga el por que**_____

   _____

   _____

2. **Considera usted que la alerta emitida por la aplicación es adecuada.**

   **SI ( )** **NO ( )**

   **Si respuesta es no diga el por que**_____

   _____

   _____

   _

3. **Tuvo algún inconveniente al usar la aplicación**

**SI ( )** **NO ( )**

**Si respuesta es sí diga Cual**_____

_____

_____

4. **Considera usted que la interfaz de la aplicación es amigable y de fácil uso .**

**SI ( )** **NO ( )**

5. **Haría uso cotidiano de esta aplicación**

**SI ( )** **NO ( )**

6. **Sugeriría a otros usuarios que usen la aplicación**

**SI ( )** **NO ( )**

# Anexo 3: Certificación de Pruebas de Validación

Loja 16 de Noviembre del 2015

Economista Vinicio Reinoso

**ADMINISTRADOR DEL CONSORCIO DE TRANSPORTISTA URBANOS CIUDAD DE LOJA**

**CERTIFICA:**

Que el señor, **YEFERSON MAURICIO TORRES BERRÚ** realizo pruebas de validación, mediante la aplicación de encuestas para conocer el grado de satisfacción y fiabilidad de su Trabajo de Titulación denominado "Aplicación móvil para la detección de somnolencia de un conductor aplicando visión artificial", realizado por el, previo a la obtención del título de INGENIERO EN SISTEMAS, y considero que la aplicación es funcional y util.

Atentamente,

.....

Economista Vinicio Reinoso

C.I: 1102335

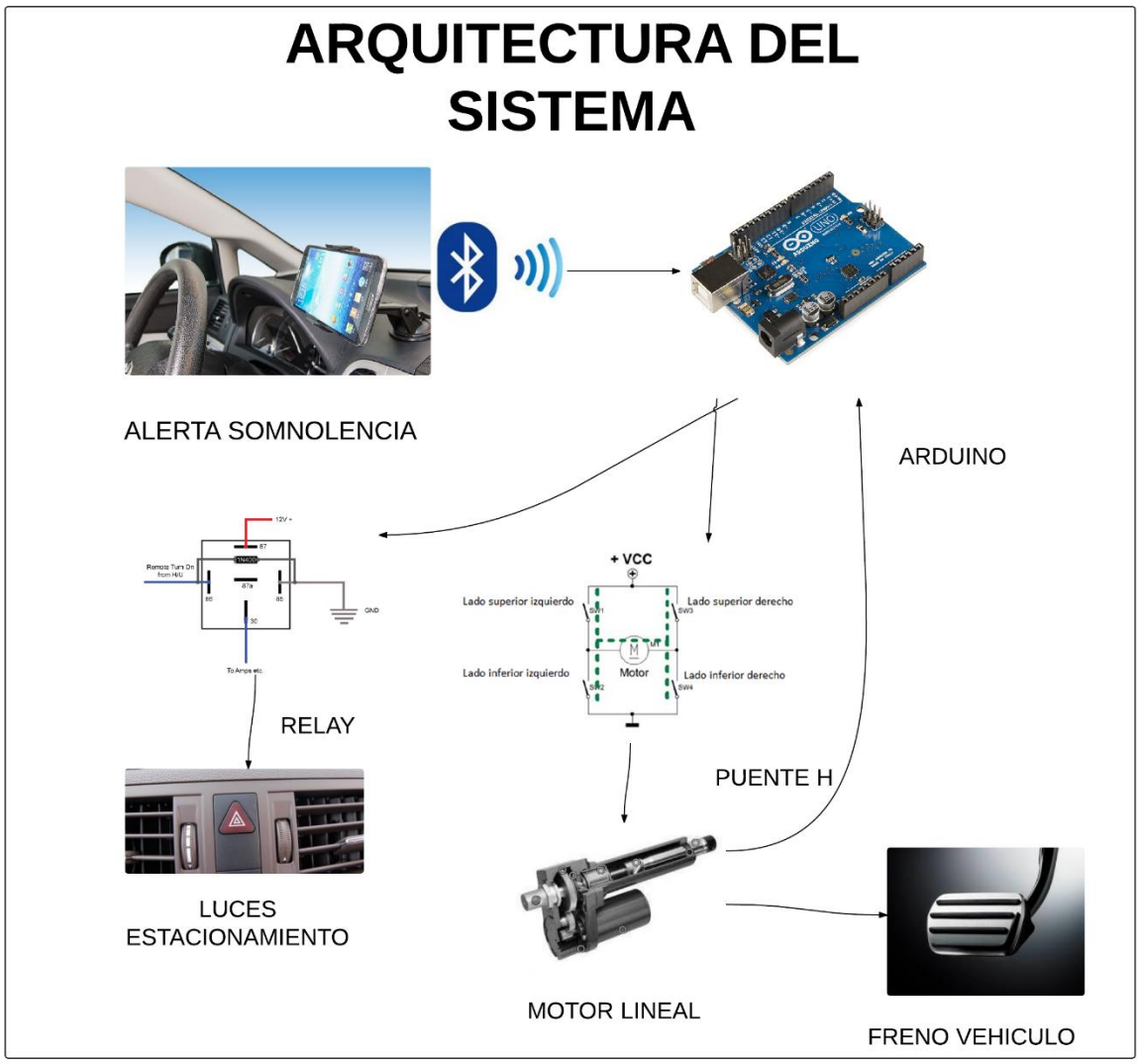
CONSORCIO TRANSPORTISTAS URBANOS "CIUDAD DE LOJA" ADMINISTRACIÓN Loja - Ecuador

# Anexo 4: Licencia del Trabajo de Titulación

Trabajo de Titulación Yeferson Mauricio Torres by is licensed Under a

Creative Commons Reconocimiento- NoComercial-CompartirIgual 3.0 Unported License.

## Anexo 5: Arquitectura Trabajo Futuro



# Anexo 6: Certificado Revisión de Estilo y Ortografía

Loja, 31 de Octubre del 2014

**Nombre:**

**Título**: Doctora en lengua y literatura

**CERTIFICA:**

Haber revisado y corregido el estilo y ortografía del Trabajo de Titulación denominado "Estándar de usabilidad para la interfaz gráfica de usuario en los proyectos de desarrollo de software", realizado por el Señor egresado **YEFERSON MAURICIO TORRES BERRÚ**, previo a la obtención del título de INGENIERO EN SISTEMAS, el mismo que cumple con las reglas establecidas por la RAE, y por el normativo de presentación dispuesto por la carrera de Ingeniería en Sistemas de la Universidad Nacional de Loja.

Por lo cual autorizo su presentación, sustentación y defensa.

Atentamente,

…….

C.I: 1101837811

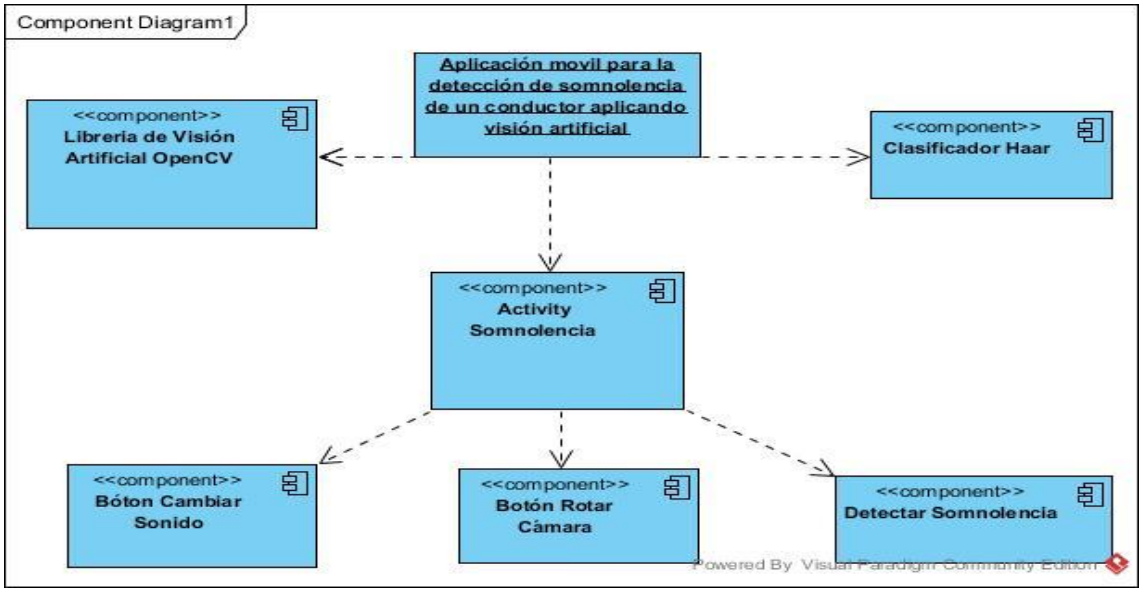
# Anexo 7: Aplicación en IOS.

En el presente anexo se detallarán los pasos necesarios para replicar la aplicacíon en Ios con el fin de se pueda utilizar en futuros proyectos.

## 1. Herramientas a usar

- CMake 2.8.8 o superior
- Xcode 4.2 o superior
- OpenCv

## 2. Diagrama de Componentes




- **Librería de visión artificial OpenCv:** Utilizada para el procesamiento digital de imágenes.
- **Clasificador Haar:** Archivo previamente entrenado por una red neuronal que sirve para encontrar rostro y ojos dentro de la aplicación
- **Activity Somnolencia:** Es el medio de comunicación entre los métodos. la librería y clasificadores.
- **Botón cambiar sonido:** El medio con el que el usuario va a interactuar para cambiar el sonido de alerta de la Aplicación.

- **Botón rotar cámara:** El medio con el que el usuario va a interactuar para cambiar la cámara del teléfono a principal o secundaria.
- **Detectar somnolencia:** Es el core de la aplicación aquí se realiza el proceso de detección en base al índice de Perclos

### 3. Integración OpenCv Ios

Para desarrollar en Ios primeramente debemos integrar OpenCv e Ios en una aplicación, para ello es necesario tener instalado

Usando la terminal acceder al directorio donde se guardara la librería y ahí clonar la librería OpenCv de la siguiente manera.

```
cd ~/<directorio_de_aplicación>
git clone https://github.com/Itseez/opencv.git
```

Hacer un enlace simbólico con Xcode

```
cd /
sudo ln -s /Applications/Xcode.app/Contents/Developer Developer
```

Construir el entorno de trabajo OpenCv

```
cd ~/< directorio_de_aplicación >
python opencv/platforms/ios/build_framework.py ios
```

Finalmente incluir el entorno de trabajo en la Aplicación Ios

Debemos también añadir los clasificadores en el proyecto y finalmente pasar cada uno de los métodos desarrollados para Android.

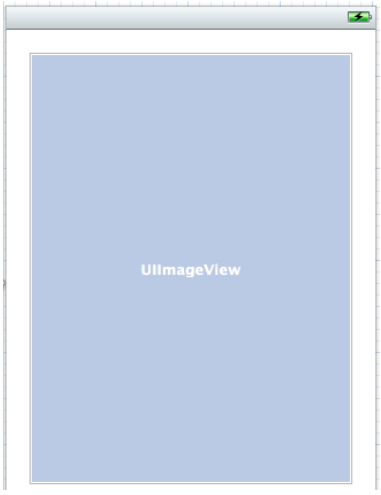
## 4. Importar OpenCv a tu aplicación en Xcode

Para ello se debe importar antes del inicio de la clase con las siguientes lineas de código.

```
#ifdef __cplusplus
#import <opencv2/opencv.hpp>
#endif
#ifdef __OBJC__
    #import <UIKit/UIKit.h>
    #import <Foundation/Foundation.h>
#endif
```

## 5. Ejemplo procesamiento de imágenes

Primeramente se crea la vista donde se va a mostrar las imágenes obtenidas por la cámara




Luego como se había indicado anteriormente importamos la librería OpenCv y en este caso la asignamos a la variable cv.

```
#import <opencv2/highgui/cap_ios.h>
using namespace cv;
@interface ViewController : UIViewController
{
    CvVideoCamera* videoCamera;
```

```
}
@property (nonatomic, retain) CvVideoCamera* videoCamera;
@end
```

El siguiente paso será inicializar el objeto tipo cámara aquí se asigna el objeto tipo cámara al ImageView para la presentación de imágenes además se define la cámara a usar en este caso la frontal así como también la orientación y los frames por segundo por detecto de la aplicación.

```
self . videoCamera = [[ CvVideoCamera alloc ] initWithParentView : imageView ];
self . videoCamera . defaultAVCaptureDevicePosition = AVCaptureDevicePositionFront ;
self . videoCamera . defaultAVCaptureSessionPreset = AVCaptureSessionPreset352x288 ;
self . videoCamera . defaultAVCaptureVideoOrientation =
AVCaptureVideoOrientationPortrait ;
self . videoCamera . defaultFPS = 30 ;
self . videoCamera . grayscale = NO ;
```

También podemos obtener el frame de la cámara y tratarlo como una matriz con el código siguiente.

```
#pragma mark - Protocol CvVideoCameraDelegate

#ifdef __cplusplus
- (void)processImage:(Mat&)image;
{
}
#endif
```

También podemos cargar un clasificador en nuestra aplicación de la siguiente manera

```
const char* cascade_name =
"/Users/chet/OpenCV-2.3.1/data/haarcascades/haarcascade_frontalface_alt.xml";
/*    "haarcascade_profileface.xml";*/
```

**Anexo 8: Artículo Científico**

## Anexo 9: Certificado de Traducción

FINE-TUNED ENGLISH
LANGUAGE INSTITUTE

Sr. Carlos Velastegui
PROFESOR DEL INSTITUTO
"FINE-TUNED ENGLISH"

CERTIFICA:

Que el documento aquí compuesto es fiel traducción del idioma español al idioma inglés del resumen para el trabajo de titulación denominado: ***"APLICACIÓN MÓVIL PARA LA DETECCIÓN DE SOMNOLENCIA DE UN CONDUCTOR APLICANDO VISIÓN ARTIFICIAL"***, del alumno YEFERSON MAURICIO TORRES BERRÚ, egresado de la Carrera de Ingeniería en Sistemas, de la Universidad Nacional de Loja.

Lo certifica en honor a la verdad y autoriza al interesado hacer uso del presente en lo que a sus intereses convenga.

Loja, 20 de noviembre de 2015

Sr. Carlos Velastegui
PROFESOR DE F.T.E.

FINE-TUNED ENGLISH CIA. LTDA. LOJA-ECUADOR

Fine-Tuned English Cía. Ltda.
LOJA: Macará entre Miguel Riofrío y Rocafuerte * 2578899 * 2563224 * 2574702
CATAMAYO: Av. 24 de Mayo 08-21 y Juan Montalvo * 2678442
ZAMORA: García Moreno y Pasaje 12 de Febrero * 2608169
www.finetunedenglish.edu.ec