Universidad de las Ciencias Informáticas UCI

Trabajo de diploma para optar por el título de Ingeniero en Ciencias Informáticas

Título: *Componente para el control de las funcionalidades del ratón de la computadora haciendo uso de una cámara digital.*

Autores:

Miguel Osmai Miralles Pantoja

Miladys Vázquez Fernández

Tutores:

Ing. Abel Sánchez Alí

Ing. Yainier Labrada Nueva

Ciudad de La Habana, 2012

"Año 54 de la Revolución"

Declaración de Autoría

Declaramos ser los únicos autores de la tesis titulada: Componente para el control de las funcionalidades del ratón de la computadora haciendo uso de una cámara digital y autorizamos al Departamento de Biometría del Centro de Identificación y Seguridad Digital (CISED) de la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

Para que así conste firmamos la presente declaración a los ____ días del mes de _____ del año _____.

“Miladys Vázquez Fernández”

_____

Firma de autor

“Miguel Osmai Miralles Pantoja”

_____

Firma de autor

“Ing. Yainier Labrada Nueva”

_____

Firma de tutor

“Ing. Abel Sánchez Alí”

_____

Firma de tutor

Datos de Contacto

Nombre y Apellidos: Ing. Abel Sánchez Alí.

Correo: aesanchez@uci.cu

Situación laboral: Profesor, Instructor Facultad 1.

Años de graduado: 3 años.

Especialidad de graduación: Ingeniero en Ciencias Informáticas.

Institución a la que pertenece: Universidad de las Ciencias Informáticas (UCI).

Dirección: Carretera San Antonio de los Baños, Torrens, Municipio Boyeros, Ciudad de La Habana, Cuba, Código postal 19370.

Nombre y Apellidos: Ing. Yainier Labrada Nueva.

Correo: ylabrada@uci.cu

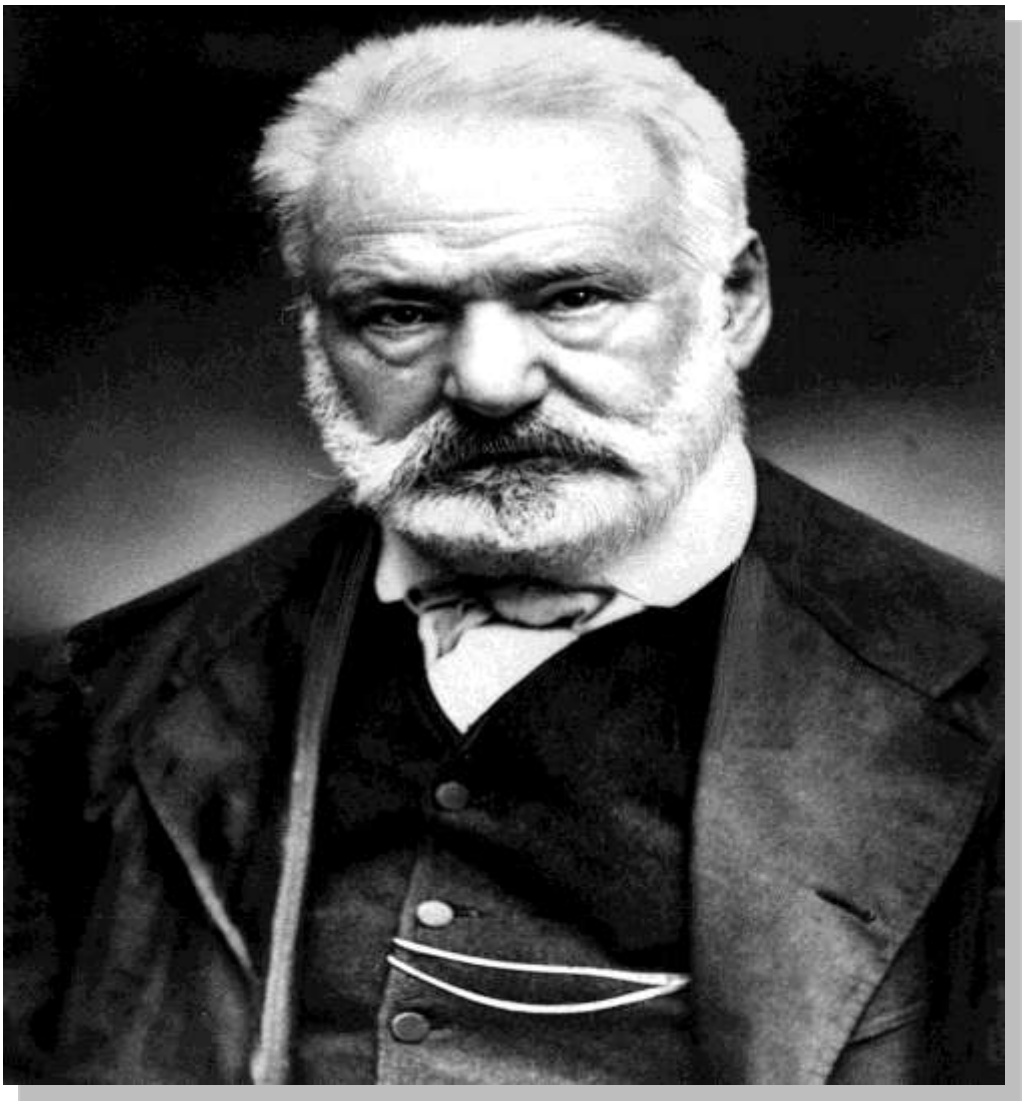
Situación laboral: Profesor, Instructor Facultad 1.

Años de graduado: 3 años.

Especialidad de graduación: Ingeniero en Ciencias Informáticas.

Institución a la que pertenece: Universidad de las Ciencias Informáticas (UCI).

Dirección: Carretera San Antonio de los Baños, Torrens, Municipio Boyeros, Ciudad de La Habana, Cuba, Código postal 19370.



El futuro tiene muchos nombres. Para los débiles es lo inalcanzable. Para los temerosos, lo desconocido. Para los valientes es la oportunidad.

Victor Hugo

Dedicatoria de Miladys

Después de cinco años de sacrificio y entrega el resultado se lo dedico a:

*A mis **padres** por su apoyo y entrega incondicional, por ser los máximos impulsores en mi vida.*

*A mi **familia** por ser la más linda del mundo y por darme su eterno amor.*

*A mi **esposo** por ser tan especial conmigo en todo momento.*

*A mis **amistades** por estar siempre ahí cuando los necesité.*

Dedicatoria de Miguel

*Esta tesis quiero dedicársela a toda la gente que de una forma u otra me han ayudado pero en especial a **mis padres** que siempre han estado presentes en cada momento de mi vida y sin ellos nada hubiese sido posible, hoy les debo todo lo que soy, los quiero mucho.*

*A mi **tío Pedro** por preocuparse por mí y por ayudarme.*

*A **mis sobrinitos** y **mis hermanas** los quiero mucho.*

Agradecimientos de Miladys

*A mi **mamita linda** por ser mi motor impulsor en todo momento, por ser esa luz en el camino que nunca se apaga, por enseñarme a ser mejor persona todos los días y por siempre tenerme como el centro de su vida.*

*A mi **papito** por su fuerza y seguridad, por siempre estar orgulloso de su negrita y su amor incondicional.*

*A mi **esposo** por ser tan paciente conmigo en estos cinco años, por ser el mejor compañero de tesis, por su amor y ternura en el día a día de nuestras vidas.*

*A mi **familia** que siempre me ha dado su apoyo, siempre desearme lo mejor y hacerme saber que puedo contar con ellos en todo momento.*

*A mis **tíos** por esas llamadas tan oportunas que me hacían cuando estaba triste, en especial a mi tía Reina, a Liz Marian, Adriana, a Rafa, a mi tía Crucita y a mi tío Rafelito por su confianza en mí.*

*A la **familia de mi esposo** por acogerme como una más de ellos, por su preocupación por nosotros.*

*A las **muchachitas** por ser únicas en todos estos años, Sashita, Adri, Dama, Dary, a todas gracias por su amistad, por apoyarme en todos los momentos que lo necesité y sobre todo por aguantar mis malacrianzas.*

*A **mis amistades de la universidad** a Pérez, Anita, Madison, Makira, Dayana, Trujillo a todos gracias por permitir que la vida en estos cinco años haya sido mejor.*

*A **mi gente** de las tunas por siempre estar preguntando por cómo iba la tesis y darme aliento.*

¡A todos gracias!

Agradecimientos de Miguel

*Primeramente que todo agradecerle a **mis padres** lo cuales me han ofrecido su apoyo en todas las dificultades que he afrontado, gracias por estar ahí en los momentos difíciles y por el amor que siempre me han ofrecido.*

*También a **mis hermanas** Milagro y Yunaika por darme ánimo para acabar la tesis y a mis queridos sobrinitos Héctor, Danitza y ahora con el nuevo que tengo Vismarcito otro colorao mas en la familia.*

*Gracias especiales a **mi esposa** Miladys por ser mi compañera, amiga y mucho más, por estar conmigo en todos los momentos malos y buenos, por haber compartido la tesis conmigo, te quiero mami.*

*Agradecer también a **mis suegros** Agustín y Milagro por todas las cosas que han hecho por mí y por ser como son de buenos conmigo.*

*A **mis primos** Yarima, Daili, Pedrito por ser lo máximo.*

*A **mis tíos** Pedro, Marielena, Margo por quererme tanto, yo también los quiero mucho.*

*A todos **los salvajes** del barrio; los míos, el Gato, Omar, Sergio, Alejandro, Radame, Arito, Osvaldito, Yoan que se metió a religioso.*

*A todos **los socios** que conocí en la UCI y por todas sus locuras El baby, Michel, El Micha, Erne, Trujillito, Omar, Yoandi, Angelón, Pedro, Roberto y muchos más.*

Resumen

El avance tecnológico alcanzado en la informática ha dado pie a que otras ciencias se desarrollen. Una de esas ciencias es la Biometría, algunas de las ramas de dicha ciencia son: el reconocimiento de iris, de huellas dactilares y la detección de rostro. En esta investigación la rama de la Biometría aplicada es la detección de rostros. Mediante la cual se desarrolló un componente para controlar las funcionalidades del ratón de la computadora haciendo uso de una cámara digital.

El objetivo fundamental de esta investigación es la implementación de un componente para controlar las funcionalidades del ratón de la computadora y obtener una mejor estabilidad en el movimiento del cursor.

Con el componente desarrollado las personas controlarán el desplazamiento del puntero del cursor del ratón de la computadora con el movimiento de los ojos y la nariz. Las acciones del clic se configurarán para ser ejecutadas mediante el pestañeo o por tiempo. Dicho componente puede ser utilizado por las personas discapacitadas, facilitándole la interacción con los ordenadores.

El componente debe su innovación a las maneras de controlar el desplazamiento del puntero del cursor, ya que entre los sistemas existentes de su tipo el control del desplazamiento del puntero solo lo realizan con la cabeza o el rostro en general. Para su desarrollo se utilizaron clasificadores para la detección del rostro y algoritmos como la Transformada de Hough para la detección de la pupila.

Palabras Clave: Biometría, rostro, cursor.

Índice de Contenidos

Índice de Figuras

Índice de Tablas

Introducción

Desde el surgimiento de los ordenadores su evolución ha sido a pasos agigantados. Las máquinas analíticas creadas en el siglo XIX ya se aproximaban a la idea del ordenador actual. En la década del cincuenta, se crearon los transistores los que hicieron posible reducir el tamaño de los ordenadores, en la década del sesenta aparecieron los circuitos integrados, los que posibilitaron integrar varios transistores en un solo saturado de silicio, llegando así al microprocesador en la década del setenta, permitiendo integrar aún más los transistores en los saturados de silicio.

La evolución de la computadora no fue solo internamente, los dispositivos con que cuenta para manejar la interacción del hombre con ella también han tenido su historia. Estos periféricos han mejorado en la forma de conectarse a la computadora y por las tecnologías usadas por los mismos, los más antiguos, conocidos y utilizados son el teclado y el ratón.

El teclado es el periférico de entrada más importante con que cuenta el ordenador. Su creación se debe a la semejanza con las máquinas de escribir mecánicas. Su estructura cuenta con cuatro bloques: el bloque funcional donde están las teclas que algunos programas utilizan para realizar funciones específicas; el bloque alfanumérico que se encuentra debajo del bloque funcional y es el que contiene las letras, números y otras teclas; el bloque especial que está, a la derecha del bloque alfanumérico, contiene teclas especiales y las flechas que permiten dar dirección al puntero del cursor del ratón de la computadora y el bloque numérico que está a la derecha del bloque especial y cuenta con los números, operadores matemáticos y la tecla que permite su activación.

El ratón es el segundo dispositivo de entrada más utilizado en la comunicación del hombre con la computadora. Por el surgimiento de la interfaz gráfica se hacía necesaria una mayor interacción entre el hombre y la máquina, lo que provoca la innovación del ratón. Fue creado a principio de los años sesenta por Douglas Engelbart y Bill English en la Universidad de Stanford. Su funcionamiento está centrado en la captura del movimiento realizado por el usuario con el cuando lo desplaza sobre una superficie plana o alfombrilla. Según la tecnología que se utilice para la captura del movimiento y la forma de comunicación de este con el ordenador es su clasificación.

Desde entonces ha tenido gran avance respecto a su apariencia, los hay que tienen un solo botón, hasta el más común que consta de dos botones que son los responsables de las acciones de clic derecho y clic izquierdo además de una rueda central, que facilita la forma de desplazarse por grandes cantidades de texto.

El avance tecnológico alcanzado en la informática ha dado pie a que otras ciencias se desarrollen, una de esas ciencias es la Biometría. El termino biometría viene del griego "bio" que significa vida y "metría" que significa medida o medición. De acuerdo al diccionario de la real academia de la lengua española, biometría es el estudio mensurativo o estadístico de los fenómenos o procesos biológicos. En el caso de la informática la biometría es el conjunto de métodos automatizados que analizan determinadas características humanas para identificar o autentificar personas. [1]

Con el desarrollo de la informática, la biometría ha tenido un gran auge en todas sus ramas como: el reconocimiento de huellas dactilares, firma, iris, ADN, detección y reconocimiento del rostro. La técnica de detección de rostro consiste en detectar en una imagen el rostro de alguna persona. Esta técnica se puede aplicar en dos enfoques diferentes: basados en rasgos faciales y basados en la imagen.

Para efectuar la detección del rostro se necesita una imagen, en la cual será donde se detecte el rostro y un dispositivo de captura, con el cual se podrá obtener la imagen para su procesamiento. Este dispositivo de captura puede ser una cámara digital.

Las cámaras digitales fueron creadas en 1991 por Quentin Stafford-Fraser y Paul Jardetzky, lanzaron su producto al mercado en 1992. Las cámaras digitales se conectan a la computadora por el puerto USB o pueden estar integradas a la computadora como en las *laptop* o computadoras portátiles. Están formadas por un lente, un sensor de imagen y los circuitos necesarios para conectarlos y manejarlos. Las mismas pueden ser usadas en sistemas para la vigilancia, chatear vía internet, impartir vídeo conferencias y en aplicaciones para controlar las funcionalidades del ratón de la computadora. Su uso es difundido progresivamente por las facilidades que brinda como: la comunicación en tiempo real y la ayuda en la seguridad.

Introducción

Para el desarrollo del componente que permita controlar las funcionalidades del ratón de la computadora se cuenta con una cámara digital. La cámara digital es el dispositivo de entrada que permite la captura de imágenes y transmisión de vídeo, con la cámara digital se podrá obtener la imagen a procesar, en la cual se realiza la extracción de las partes de interés.

El principal beneficio que traerá el componente es la nueva forma que las personas tendrán de interactuar con las computadoras y que será de gran utilidad a los discapacitados que no puedan utilizar sus manos.

Cuba a pesar de ser un país subdesarrollado cuenta con centros de investigación en diferentes campos como la salud y la informática. Uno de esos centros es la Universidad de las Ciencias Informáticas (UCI), donde se desarrollan softwares informáticos para la exportación y uso nacional. La UCI tiene varios centros productivos entre los que está el Centro de Identificación y Seguridad Digital (CISED) y este a la vez cuenta con departamentos entre los que se encuentra el departamento de Biometría.

En el departamento de Biometría es donde se desarrollan aplicaciones biométricas que son usadas en la identificación de personas, haciendo uso de su firma, huellas dactilares y el rostro. En el centro se puede desarrollar un componente que permita controlar las funcionalidades del ratón mediante el uso de una cámara digital, el mismo ofrecerá una nueva forma de interactuar con la computadora y una mayor explotación de los conocimientos y técnicas biométricas con que cuentan sus miembros. El poder contar con un componente de este tipo permitirá que el departamento de Biometría respalde una vez más sus trabajos y permitirá que las personas interactúen con la computadora de una forma diferente y amena, además que lo puedan usar la personas discapacitadas facilitándole el uso de la computadora, al controlar las funcionalidades del ratón sin utilizar las manos, por lo que se tiene como **problemática:** que actualmente el Centro de Identificación y Seguridad Digital carece de un componente que permita controlar las funcionalidades del ratón de la computadora haciendo uso de una cámara digital.
Por lo que se hace necesario para realizar la investigación plantear el siguiente **problema de investigación:** ¿Cómo lograr controlar el desplazamiento del puntero del cursor del ratón de la computadora a través del uso de una cámara digital?

El **objeto de estudio** del presente trabajo está enmarcado en: el proceso de desarrollo de sistemas informáticos para la detección del rostro.

El **objetivo general** de este trabajo: es desarrollar un componente que permita controlar el desplazamiento del cursor del ratón de la computadora interactuando con una cámara digital, para obtener una mejor estabilidad del cursor del ratón.

El **campo de acción** queda delimitado en: componentes para controlar las funcionalidades del ratón de la computadora.

Tareas investigativas:

Tareas comunes:

- Análisis del estado del arte referente a las aplicaciones que permitan controlar las funcionalidades del cursor del ratón de la computadora con el rostro.
- Definición de las tecnologías, metodologías y herramientas a utilizar para el desarrollo del componente.
- Definición de la arquitectura del software.
- Definición de los patrones del diseño.

Tareas llevadas a cabo por Miladys Vázquez Fernández:

- Confección del modelo de dominio del componente.
- Levantamiento de los requisitos funcionales y no funcionales del componente.
- Confección del diagrama de los casos de uso del sistema.
- Descripción de los casos de uso del sistema.
- Confección del diagrama de clases del análisis.
- Confección del diagrama de colaboración.
- Confección del diagrama de clases del diseño del componente.
- Confección del diagrama de componente.
- Diseño de los casos de prueba.

Tareas llevadas a cabo por Miguel Osmai Miralles Pantoja:

- Análisis de las técnicas para la detección del rostro.
- Análisis de la librería OpenCV.
- Implementación del algoritmo para la detección del pestañeo.
- Implementación del algoritmo para detección de la pupila.
- Diseño de las interfaces gráficas del componente.

Los métodos investigativos se clasifican en métodos teóricos y métodos empíricos. En las investigaciones se usan estos métodos para que el conocimiento expuesto sobre el tema sea claro y preciso, los métodos utilizados en la misma son:

Métodos Teóricos:

- **Analítico–Sintético:** este método se aplica con el aprovechamiento de toda la información bibliográfica que se consulta para profundizar y aclara el conocimiento ya acumulado. Se usa específicamente en la adquisición de información sobre los métodos de detección de imagen, los sistemas existentes y en la determinación de las herramientas a utilizar, abriendo así el camino para la puesta en práctica.
- **Inductivo - Deductivo:** se refleja su uso en el estudio del arte realizado, este método posibilitó a través de la obtención de conocimientos generales, deducir conocimientos particulares para la solución de la investigación, aplicando conocimientos que estuvieran más acorde con el objetivo de la investigación.
- **Modelación:** está representado mediante la elaboración de diagramas, que permiten hacer una reproducción simplificada del dominio del problema, facilitando el entender cómo funciona el componente.

Método Empírico:

- **Observación:** este método se aplica en la explotación del funcionamiento de las aplicaciones ya realizadas, permitiendo definir lo que debe ser mejorado.

El **posible resultado** de esta investigación es la obtención de un componente que permita el control de las funcionalidades del ratón de la computadora, pensado en ofrecerle una nueva forma de interactuar a las personas con la computadora, ofreciéndole más suavidad en el desplazamiento del puntero del cursor y una nueva forma de controlar el movimiento del cursor del ratón utilizando una cámara digital.

La **justificación de la investigación** está dada por su relevancia social, porque brinda una manera fácil, amena y diferente de interacción de las personas con la computadora, indistintamente de su condición física. Además de ser conveniente su realización para abarcar más sobre las técnicas biométricas existentes lo cual sería de gran utilidad para el departamento de Biometría del centro CISED, logrando que este producto tenga la calidad

requerida en los indicadores: suavidad en el movimiento del cursor por el área de trabajo, la configuración de la acción del clic y estabilidad del puntero del cursor, ya que son los que permiten que el componente tenga un impacto positivo en la sociedad. Teniendo como aporte principal que el desplazamiento del cursor del ratón de la computadora será realizado con el movimiento de los ojos y la nariz. Además se pretende mejorar la estabilidad del puntero del cursor.

El desarrollo del tema tratado en la investigación queda reflejado en este documento a través de tres capítulos.

Capítulo 1 "Fundamentación teórica": es donde se expone el origen del tema que se trata en la investigación, desde el nivel más general al más específico, reflejando las observaciones alcanzadas en la profundización del tema. Los conceptos y términos que serán importantes conocer para un debido entendimiento de lo desarrollado en el mismo. Se explicarán las herramientas y tecnologías empleadas para alcanzar el objetivo.

Capítulo 2 "Características del sistema": es donde se describe el funcionamiento interno del componente, se definirán los requisitos funcionales y no funcionales, se realizará la propuesta del modelo del dominio y el modelo de casos de uso del sistema, se plasmará el análisis y diseño del componente y será donde se explicará la arquitectura a usar.

Capítulo 3 "Resultado y prueba": se reflejará la implementación de la solución al problema planteado y las interfaces del componente desarrollado, además se contará con las pruebas requeridas para demostrar que el funcionamiento del componente se corresponde con lo esperado.

Capítulo 1: Fundamentación Teórica

Introducción

En este capítulo se abordará sobre la situación actual de los componentes que permiten manipular las funcionalidades del ratón de una computadora mediante el rostro de una persona haciendo uso de una cámara digital. Se analizarán los componentes existentes. Se tratan conceptos referentes a la captura y procesamiento de imágenes y se analizarán la metodología y herramientas que serán usadas en el desarrollo de la investigación.

1. Adquisición de imágenes. Vídeo

La historia del vídeo se remonta a 1824, con el fenómeno del estroboscopio que no era más que una lámina de cartón que contenía dibujos a ambos lado de las caras y cuando la imagen se giraba se sobreponía dando la ilusión de movimiento. Los estroboscopios eran uno de los elementos previos al proyector cinematográfico que se basa en este efecto para la percepción del movimiento continuo entre fotogramas. El vídeo surge en la década del cincuenta como una tecnología estrechamente relacionada con la televisión ya que permitía la emisión de programas y retransmisiones en diferido.[2] A continuación se exponen algunos conceptos que facilitan comprender su funcionamiento y uso en esta investigación.

1.1 Imagen

Imagen es un término que proviene del latín imāgo y que se refiere a la figura, representación, semejanza o apariencia de algo. Es la representación visual de un objeto a través de técnicas de la fotografía, la pintura, el diseño, el vídeo u otras disciplinas.[3]

1.2 Vídeo

Un vídeo es una sucesión de imágenes presentadas a cierta frecuencia. El ojo humano es capaz de distinguir aproximadamente 20 imágenes por segundo. De este modo, cuando se muestran más de 20 imágenes por segundo, es posible engañar al ojo y crear la ilusión de una imagen en movimiento.

1.2.1 Vídeo análogo y vídeo digital

- Vídeo análogo: representa la información como un flujo continuo de datos análogos, para mostrarse en una pantalla de TV (en base al principio de escaneo).

- Vídeo digital: consiste en mostrar una sucesión de imágenes digitales. Dado que estas imágenes digitales se muestran a una frecuencia determinada.[4]

1.3 Cámara digital

La cámara digital es el dispositivo de captura de vídeo o imágenes en forma digital. A diferencia de las tradicionales cámaras analógicas que convierten las intensidades de luz en señales infinitamente variables, las cámaras digitales convierten las intensidades de luz en números discretos.

La cámara digital descompone la imagen de la figura en un número fijo de píxeles (puntos), verifica la intensidad de luz de cada píxel y la convierte en un número. En una cámara digital de color, se crean tres números, que representan la cantidad de rojo, verde y azul en cada píxel.[5]

1.3.1 RGB (Rojo, Verde, Azul)

RGB espacio de color nativo de las computadoras y de los sistemas para la captura y muestra de imágenes en color de forma electrónica o digital. Es un modelo de color basado en la síntesis aditiva, con el que es posible representar un color mediante la mezcla por adición de los tres colores de luz primarios.

2. Procesamiento de vídeo

El procesamiento de vídeo se lleva a cabo a través de la cámara digital. En el vídeo se puede detectar el rostro de la persona permitiendo detectar información útil para su procesamiento.

2.1 Detección del rostro

La detección del rostro es el proceso que tiene como fin detectar o señalar el rostro de una persona teniendo en cuenta ciertas características físicas de la misma. Esas características pueden ser los ojos, la boca, la nariz o el globo facial en general.

3. Métodos para la detección del rostro

La detección del rostro es un proceso importante en cualquier aplicación donde se realice algún tipo de análisis facial como por ejemplo su reconocimiento, vigilancia de multitudes, seguridad doméstica entre otros. El objetivo de esta etapa consiste en detectar y localizar la posición de un número indefinido de rostros en una imagen. En general, la detección del rostro es un problema muy complejo ya que los objetos a detectar pueden ser de diferentes colores, expresiones, poses, tamaños relativos o tener condiciones de iluminación muy dispares.

La detección del rostro se inició en la década de los setenta que fue cuando surgieron los primeros algoritmos basados en técnicas heurísticas y antropométricas. Estos algoritmos eran muy lentos y no tenían un buen por ciento de acierto y si se cambiaba algo en el rostro, como por ejemplo, si llevaba unos espejuelos se tenía que cambiar por completo el algoritmo. Estas investigaciones se detuvieron debido a que la tecnología tampoco era la óptima, es decir no había mucha capacidad de procesamiento y poca memoria.

En la actualidad existen variados métodos para llevar a cabo la detección del rostro en una imagen, ya sea esta un vídeo o una foto indiferentemente. Dentro de estos métodos para efectuar la detección del rostro se encuentran: las técnicas basadas en rasgos y las técnicas basadas en la imagen.

2.1 Técnicas basadas en rasgos

En este tipo de técnicas el conocimiento previo se hace completamente explícito y se sigue una metodología clásica de detección en la que los rasgos de más bajo nivel se extraen de un análisis a priori basado en el conocimiento. En diferentes niveles del sistema se explotan propiedades aparentes de la cara tal como el color de la piel y la geometría facial. En estas técnicas la detección de la cara se resuelve manipulando medidas de distancia, ángulos y áreas de los rasgos visuales en la escena. La cuestión determinante en este tipo de métodos es decidir qué rasgos de la cara o de la imagen interesan para su estudio. Con esta técnica se pueden realizar diferentes análisis a la imagen como pueden ser el análisis de bajo nivel, el análisis de rasgos y análisis de formas activas. A continuación se explica en qué consisten dichos análisis y en lo factores que se basan en su puesta en práctica.

- Análisis de bajo nivel

Bordes: uno de los rasgos más primitivos que tiene cualquier figura es su contorno. Los trabajos que utilizaban esta idea extraían bordes de la cara tanto externos como internos.

Luego eran sometidos a análisis de forma y posición. También se puede utilizar estos métodos para detectar si el individuo lleva gafas. Una vez extraídos los bordes pueden usarse diferentes operadores. Existen técnicas más recientes que utilizan estas estrategias, en las que se alcanzan tasas de acierto del 76% con dos falsos positivos por imagen.

Niveles de gris: los rasgos faciales como las cejas, pupilas y los labios aparecen generalmente más oscurecidos que las regiones de su alrededor. Esta propiedad puede ser explotada para diferenciar partes. Algunos algoritmos de extracción de rasgos faciales buscan mínimos locales dentro de regiones faciales segmentadas. En dichos algoritmos a las imágenes se les realza previamente el contraste y se les aplica rutinas morfológicas de escalas de grises para mejorar la calidad de las regiones oscuras y facilitar su detección. La posición relativa de los ojos y su detección también puede ser descubierta con este tipo de métodos.

Color: si el blanco y negro permite una representación básica de la imagen para extraer propiedades, el color es una herramienta mucho más poderosa, entre otras cosas porque al triplicarse la dimensionalidad se tiene más información. Dos formas que en una imagen de blanco y negro aparecen iguales pueden ser diferentes en una imagen con color. El color de la piel humana (en sus diferentes variantes) ha permitido desarrollar algunas técnicas que detectan la raza. Como una de las maneras más habituales de trabajar con fotografía digital es la representación RGB5, en este tipo de técnicas se suele trabajar con un RGB normalizado que vendría dado por las ecuaciones:

$$r = \frac{R}{R+G+B}$$

$$g = \frac{G}{R+G+B}$$

$$b = \frac{B}{R+G+B}$$

A partir de las cuales siempre ocurrirá r + g + b = 1. Pudiéndose averiguar cualquiera de los tres componentes conociendo los otros, la componente azul por ejemplo será: b = 1 − r − g. Al analizar el color de la piel, un histograma basado en r y g muestra el color de la cara como una pequeña región. Al comparar luego un píxel con dicha región se puede averiguar si dicho píxel pertenece o no a la cara.

Relacionado con la detección de razas, se utiliza también la representación YIQ en la que el componente I representa un rango de color desde el naranja hasta el cyan, lo que permite realzar la imagen para el caso de individuos asiáticos.
En este tipo de trabajos y aprovechando la representación RGB (normalizada) se pueden encontrar también técnicas que detecten labios, ojos y cejas, para poder usar esquemas de segmentación.

Vídeo: si se dispone de una secuencia de vídeo, la localización de objetos en la imagen es más factible. Una de las mejores formas es mediante diferencia de fotogramas. Existen técnicas que miden variaciones verticales y horizontales para encontrar los ojos. Detectar contornos de una escena en movimiento es más sencillo. Se utilizan filtros espaciotemporales de gaussiana para encontrar los bordes de la cara y el cuerpo. También existen técnicas que trabajan con la velocidad a la que se mueven ciertas regiones de la escena. En estas técnicas se completa la localización de la cara con un algoritmo que construye una elipse ajustándola. En otros trabajos se parte de la concavidad y convexidad de las regiones (pómulos, ojos) para utilizar la derivada de la medida del gradiente.

- Análisis de rasgos

El problema de los métodos anteriores es que sus medidas son demasiado ambiguas. Detectar la cara sobre la base del color permite que se cuelen como falsas caras cualquier objeto del fondo. Utilizar el conocimiento que ya se tiene de la geometría facial puede servir para caracterizar y verificar muchos rasgos a priori confusos. Existen básicamente dos estrategias:

Búsqueda de rasgos: estas técnicas buscan rasgos prominentes que permiten localizar rasgos menos prominentes partiendo de hipótesis geométricas. Por ejemplo, una pequeña área sobre un área alargada puede corresponderse al escenario (cabeza sobre los hombros) y un par de regiones oscuras encontradas en el área facial incrementa la probabilidad de que aquello sea realmente una cara. Los rasgos más usados son los ojos, contorno de la cabeza y el cuerpo (bajo la cabeza).

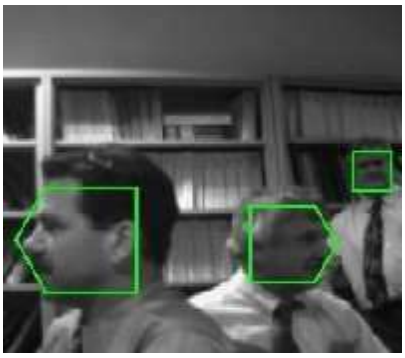


Figura 1: Detección de caras dentro de una imagen

Análisis de constelación: la mayoría de los algoritmos que se basan en búsqueda de rasgos dependen fuertemente de su heurística. Aplicados a casos más generales (diferentes posturas, primer plano sobre fondos complejos) pueden volverse demasiado rígidos. Existen otros métodos posteriores a aquellos que agrupan rasgos faciales en constelaciones (*facial-like*) y utilizan métodos más robustos de modelado como análisis estadístico.

- Análisis de formas activas

Se basan en representar la imagen en rasgos de alto nivel para posteriormente interactuar con rasgos locales de la imagen (ojos, brillo) y gradualmente deformarla hasta adaptarla a la forma de los rasgos.

Modelos de distribuciones de puntos (PDM): se trata de una descripción compacta parametrizada de las formas de la cara basada en estadísticas. La arquitectura y ajuste de este método es diferente a los anteriores. El contorno del PDM se discretiza en un conjunto etiquetado de puntos. Las variaciones de dichos puntos son primero entrenadas con diferentes posturas y tamaños. Durante el proceso de deformación solo se permite variar a la forma en una de las posibilidades del conjunto de entrenamiento.

2.2 Técnicas basadas en la imagen

En estas técnicas, por el contrario, el objeto de estudio es la imagen misma. El conocimiento previo se incorpora implícitamente en esquemas de entrenamiento. Se trabaja directamente con una representación de la imagen a la que se le aplican algoritmos de entrenamiento y análisis.

Métodos basados en subespacios: se basan en los trabajos de Sirovich y Kirby y los desarrollos posteriores de Turk y Pentland. Consideran las imágenes de caras humanas como un subespacio lineal de un espacio mayor (de todas las imágenes). Al representarlas así pueden utilizarse muchos métodos para tratar los datos como: análisis estadístico multivariante y redes neuronales.

Redes neuronales: las redes neuronales se han convertido en una técnica popular para el reconocimiento de patrones. Ciertas técnicas basadas en redes neuronales han utilizado ventanas de 20 × 20 píxel que reciben como entrada una imagen preprocesada. Se entrenan las redes neuronales con imágenes con caras e imágenes sin caras. Se resuelven problemas de solapamiento de regiones ya que estas técnicas son usadas para verificar si una foto es de una cara o no. También se han utilizado este tipo de técnicas para la detección de rasgos.

Métodos estadísticos: hacen referencia a métodos basados en teoría de la información. También se incluyen aquí técnicas que utilizan máquinas de vectores soporte y reglas de decisión de bayes.[6]

2.3 Método usado en la detección del rostro en el componente. Clasificadores

Los clasificadores o *haarcascade* son los que detectan ciertas características de la imagen digital que se utilizan en el reconocimiento de objetos, en este caso rostros. Estos se utilizaron en el primer detector de caras en tiempo real.

Históricamente, el trabajo con la intensidad de la imagen (es decir, los valores de los píxeles RGB en todos y cada uno de píxeles de la imagen) hizo que la tarea de cálculo computacionalmente sobre la imagen fuera costosa. Una de las características *haar-like* es que resume las intensidades de los píxeles en estas regiones y calcula la diferencia entre ellos. Esta diferencia se utiliza para clasificar las subsecciones de una imagen. Por ejemplo, se cuenta con una base de datos de imágenes con los rostros humanos. Es una observación común que entre todas las caras la región de los ojos es más oscuro que la región de las mejillas, por lo tanto una característica *haar* común para la detección de caras es un conjunto de dos rectángulos adyacentes que se encuentran por encima del ojo y la región de la mejilla.

La posición de estos rectángulos se define en relación a una ventana de detección que actúa como un cuadro de límite al objeto de destino (la cara en este caso).
Los clasificadores facilitan el proceso de detección del rostro. La ventaja clave de una función de *haar-like* en la mayoría de otras características es su velocidad de cálculo.[7]

Los clasificadores serán usados en la implementación del componente, ya que con su uso la detección del rostro es más fácil, las partes a detectar por los mismo serán: la nariz, los ojos y el rostro en sí .Los clasificadores son una gran herramienta a explotar en investigaciones como la que se está realizando, donde el objetivo fundamental no es el proceso de detección del rostro, sino usarlo para lograr controlar el cursor de la computadora.

4. Sistemas existentes

La facilidad en la comunicación hombre-computadora y la utilización de medios que no tienen que ser los tradicionales son uno de los pasos que se han ido dado en la actualidad con respecto al tema. Existen aplicaciones capaces de controlar las funcionalidades del ratón usando una cámara digital. La idea central de estos proyectos no es solo tener una manera diferente que permita la interacción hombre-computadora, sino facilitarles el uso de las tecnologías a las personas con discapacidades físicas. Algunos de estos sistemas se explican a continuación.

4.1 Head Mouse

Fue desarrollado en la Universidad de Lleida en España, permite controlar el desplazamiento del cursor del ratón con movimientos de la cabeza y el rostro. En el mercado se encuentran algunas versiones realizadas a esta aplicación en las que se han mejorado: el consumo de CPU, la versión 4.0 es la que menos CPU utiliza de todas las versiones realizadas hasta el momento. Se ha optimizado el consumo de memoria RAM. Se ha incluido un nuevo menú desplegable con macros configurables por el usuario, las macros pueden contener secuencias de teclas y ordenes de ejecución de programas, archivos y páginas web.[8]

4.2 Camera Mouse

Este *software* fue desarrollado por investigadores de la Universidad de Boston y Boston College. El fin buscado es el mismo, ayudar a las personas con discapacidades a usar las

tecnologías y por ende mejorar su estilo de vida y a las personas sin problemas físicos brindarle una nueva alternativa de controlar el cursor del ratón.

Existen otras alternativas que siguen la misma línea que ya se mencionaron, una de estas es el programa *eViacam*. Este presenta una ligera ventaja en comparación al *Camera Mouse* en relación al punto seleccionado para controlar el cursor. En el *Camera Mouse*, identificar el punto que controlará el cursor se determina dando clic sobre la región con que se desea controlar el movimiento. El *eViacam* registra automáticamente el punto y el usuario no lo tiene que seleccionar, haciendo más fácil el inicio de la sesión al no tener que activar de forma manual la acción del ratón.[9]

4.3 Ratón Facial

Mueve el cursor del ratón con cualquier parte del cuerpo, pero como forma fundamental usa la cabeza haciendo uso de una cámara digital, la acción del clic se puede configurar para realizarla por tiempo o la voz. Con la voz interpreta un sonido corto como el clic izquierdo y uno largo como el clic derecho, el nivel de sonido es configurable para no tener problemas con el sonido de fondo. No necesita de la calibración inicial como otros sistemas, la velocidad de desplazamiento del cursor es variable. Tiene entre sus características que es rápido, fácil de usar, cómodo y su precio es asequible.[10]

4.4 Enable Viacam (eViacam)

Enable Viacam permite controlar las funcionalidades del ratón mediante una barra que tiene en la parte inferior de la pantalla, el usuario puede activar o desactivar la acción del clic, dar clic izquierdo, clic derecho o doble clic y puede arrastrar un objeto. Es compatible con la mayoría de las cámaras, tolera los cambios de luz y de posición del usuario, cuenta con varias opciones de configuración para el movimiento del ratón, brinda la posibilidad de acceder a un teclado virtual si el usuario lo desea, es compatible con sistemas operativos *Windows* y *Linux*.[11]

4.5 Sistemas de sensores ópticos

Además de los sistemas mencionados anteriormente en este capítulo existen otros que permiten el movimiento del cursor del ratón de la computadora mediante sensores ópticos. Entre estos sistemas se encuentran: *IG-30 de Alea Technologies*, *Eyegaze Edge de LC Technologies*, *Quick Glance de EyeTech Digital Systems*, *PCEyes de Tobii Technology*.

Estos sistemas cuentan con cámaras especiales externas al ordenador, con sistemas de iluminación que permiten capturar la imagen en condiciones óptimas y un *hardware* que se encarga de procesar el movimiento de los ojos del usuario y traducirlos en acciones interactivas igual que las que tiene un ratón convencional. Estos sistemas en general permiten un buen control del puntero del ratón con una buena interacción y usabilidad sin perder las funcionalidades principales que brindan los sistemas operativos más comunes. Entre sus inconvenientes tiene que no son gratis, su precio puede estar entre los 4000 y 15000 euros, por lo que no todo tipo de personas pueden contar con ellos y requieren de *software* concretos para traducir el movimiento de los ojos en acciones específicas del ratón.[12]

4.6 Análisis referente a los sistemas investigados

Los sistemas que realizan el control de las funcionalidades del ratón haciendo uso de una cámara digital tienen como ventaja que son de distribución gratuita por lo que pueden ser accedidos por un gran número de personas en el mundo. Pero tienen inconvenientes en su uso entre los cuales está presente: el control del cursor, que no es el mejor en el mayor de los casos y su configuración puede llegar a hacer un poco compleja.

En la siguiente tabla se refleja el análisis cualitativo (bien, mal, regular) del grado de deficiencia que presentan los sistemas más usados en sus funcionalidades generales como: la estabilidad del movimiento del cursor del ratón de la computadora y configuración de la acción del clic. La evaluación se le dio a cada sistema estudiado basado en parámetros ya definidos mediante pruebas de usabilidad realizadas a los mismos.

Aplicaciones	Acción del clic	Estabilidad en el movimiento del cursor	Movimiento del cursor con los ojos	Movimiento del cursor con la nariz
eViacam	regular	regular	no posee	no posee
Head mouse	bien	regular	no posee	no posee
Ratón facial	regular	bien	no posee	no posee

Cámara mouse	regular	regular	no posee	no posee

Tabla 1: Comparación de los sistemas existentes por funcionalidades generales
Fuente: Elaboración propia.

Realizado el análisis de los valores de la tabla anterior se determinó que el componente a desarrollar debe contar con una nueva forma para controlar el movimiento del cursor, la cual puede ser mediante la detección de la pupila y la detección de la nariz. Además debe mejorar en cuanto a la estabilidad del movimiento del cursor.

5. Detección de la pupila

Con la detección y localización de la pupila dentro de la imagen el componente será capaz de poder mover el cursor por toda la pantalla. Por lo que se analizan los ojos por separados para aplicarle el algoritmo y así detectar la pupila. El algoritmo que se explica a continuación brinda la posibilidad de detectar la pupila de manera simple y eficaz, además en comparación de otros algoritmos es más fácil de implementar y matemáticamente no posee una gran complejidad.

5.1 Algoritmo de Transformada de Hough

La transformada de Hough es una técnica utilizada para aislar características de forma particular dentro de una imagen. La idea básica es encontrar curvas que puedan ser parametrizada como líneas rectas, polinomios y círculos.[13]

Detección de círculos:

El proceso comienza partiendo de la imagen completa del ojo la cual se lleva a escala de grises, una vez filtrada la imagen se le aplica un algoritmo de detección de contornos como por ejemplo el detector *canny*. A partir de la imagen de contornos, se procede a crear el llamado espacio de Hough, que contiene círculos que pasan sobre cada punto del contorno.

Figura 2: Detección de círculos

Estos círculos se definen con la ecuación general: $x^2+y^2-r^2=0$. Siendo (x, y) las coordenadas del centro del círculo, y r el radio. Una vez creados todos los círculos posibles, el máximo dentro del espacio de Hough proporciona el círculo mejor definido.[14]

6. Algoritmo para la detección del pestañeo

Con el análisis de varios algoritmos para la detección del pestañeo se llegó a la conclusión de que los mismos trabajaban de cierta forma de la misma manera, por lo cual se eligió el de menor complejidad matemática y uno de los más robusto, es decir la unión de estos algoritmos y con la ayuda de la librería *OpenCV* se implementó el algoritmo a usar. Los algoritmos estudiados se pueden encontrar en la bibliografía consultada [5] [7] [8].

Para llevar a cabo la detección del pestañeo se toman dos fotogramas; el fotograma actual y el anterior los cuales serán procesados para ser filtrados a escala de grises, estos dos fotogramas una vez procesados se restan para obtener una imagen binarizada es decir en blanco y negro.

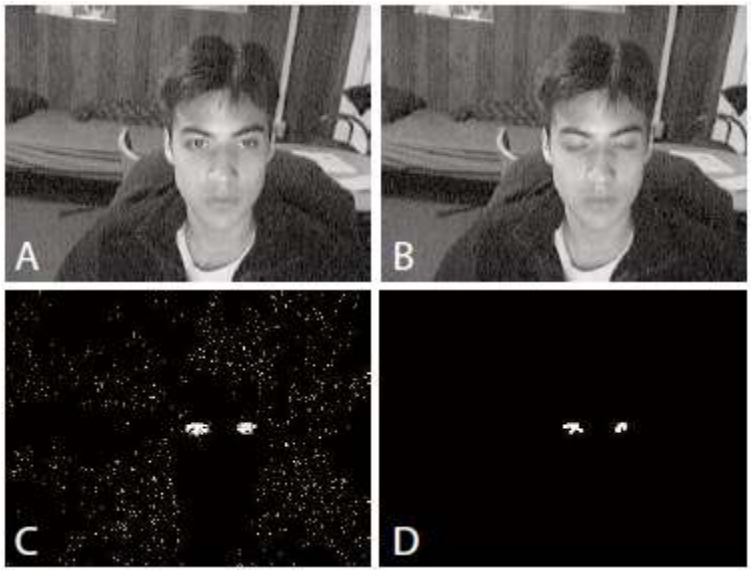


Figura 3: Fases del análisis del pestañeo:
(A) Imagen del usuario en el marco f. (B) Imagen del usuario en el marco f + 1, que acaba de parpadear. (C) Diferencias iniciales de los dos marcos F y F +1. (D) Diferencia de imagen que se utiliza para localizar los ojos

A esta imagen binarizada se le aplican los operadores morfológicos de apertura y clausura, el primero consiste en aplicar una erosión seguida de una dilatación aplicando la misma forma estructurante, como resultado esta tiende a “abrir pequeños huecos”. La clausura es el la aplicación de las operaciones básicas en el sentido inverso, y resulta en “cerrar los huecos”.

Una vez ya aplicado los operadores, a través de la librería *OpenCV* utilizando la funcionalidad ***FindContours()*** se detectan los cambios de intensidad en los fotogramas es decir la acción del pestañeo.[15]

7. Metodologías y herramientas empleadas

Las metodologías son las que proporcionan una organización lógica en el proceso de desarrollo del *software*, apoyándose de las herramientas necesarias y más convenientes para lograr llegar al resultado esperado, con la eficacia requerida. Por lo que el estudio de las mismas permitirá seleccionar las correctas para el desarrollo del componente.

7.1 Metodología

Una metodología es un conjunto de procedimientos utilizados para guiar el proceso de desarrollo de *software* generando disimiles artefactos. Las metodologías se dividen en dos grupos, en ágiles y tradicionales, esto se debe a la forma de registrar los artefactos creados en el proceso de desarrollo del *software*. Las tradicionales son más rígidas, son efectivas principalmente en proyectos que tienen gran cantidad de recursos y cuenta con suficiente tiempo. En entornos donde el proceso es cambiante y el tiempo con el que se cuenta es realmente poco se recomienda usar metodologías ágiles, ya que se generan solo los artefactos necesarios y se ajustan a equipos de trabajo pequeños.

En este epígrafe quedarán recogidas brevemente las principales características de la metodología Proceso Unificado de Rational (RUP), que aunque es una metodología tradicional será la usada para guiar el proceso de desarrollo en esta investigación por el control que se logra tener del desarrollo del *software* al aplicarla.

7.1.1 RUP

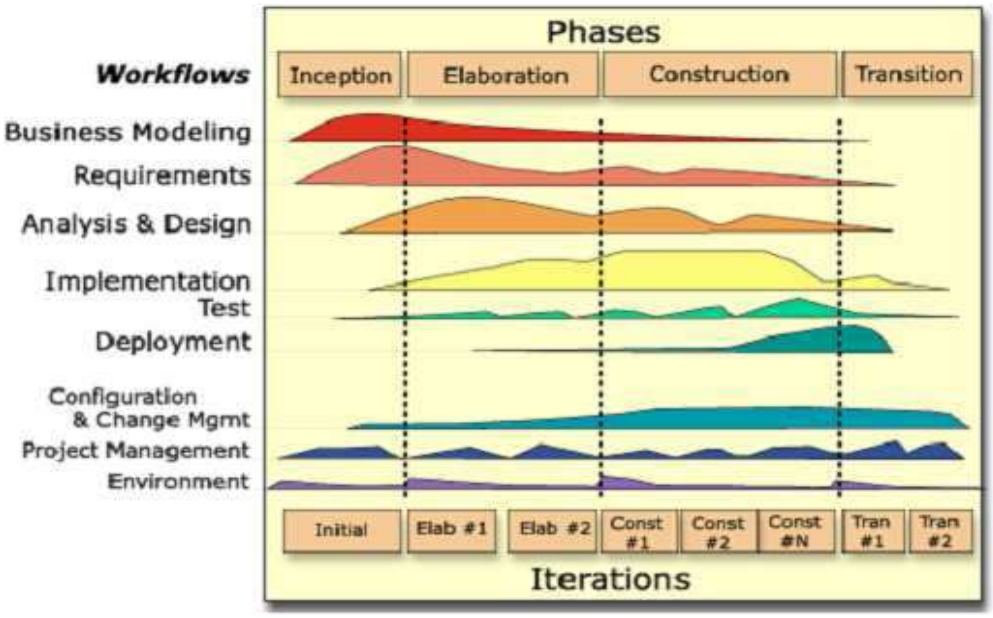


Figura 4: Fases, iteraciones y flujos de trabajos de RUP

Proceso Unificado de Rational (RUP) realiza el proceso de desarrollo de forma bidimensional, en fases y flujo de trabajos. Las fases con que cuenta son cuatro: Inicio, Elaboración, Construcción y Transición. Las mismas consisten en:

- Inicio: su hito fundamental es definir la visión del proyecto a desarrollar, definiendo el modelo de negocio del mismo.
- Elaboración: el objetivo de esta fase es definir el problema a resolver, definiendo una arquitectura candidata para el desarrollo del componente y planificar las actividades a desempeñar por el equipo de trabajo.
- Construcción: en esta fase se lleva a cabo la implementación del componente, apoyado en los requisitos definidos en las fases anteriores, al final de esta fase el componente debe estar listo para su utilización.
- Transición: esta es la fase en que el componente es llevado e instalado en el entorno del usuario, al terminar esta fase el desarrollo del componente debe estar documentado y el mismo debe satisfacer la necesidad del usuario.[16]

Los flujos de trabajo que se van desarrollando en las diferentes fases son: Modelado del negocio, Requisitos, Análisis y Diseño, Implementación, Pruebas, Despliegue, Gestión del

proyecto, Configuración y control de cambios y Entorno, de los mismo los seis primeros son de ingeniería y los tres restantes de apoyo al ambiente de trabajo, ningún flujo es más importante que el otro, todos contribuyen al desarrollo adecuado del software. Los mismos consisten en:

- Modelamiento del Negocio: identifica y detalla los procesos de negocios a través de casos de uso, así como quién inicia o participa, y cuáles son las actividades necesarias a ser automatizadas.
- Requerimientos: establece qué es lo que el sistema debe hacer, incluyendo funcionalidades y restricciones.
- Análisis y Diseño: describe cómo será realizado el sistema a partir de las funcionalidades previstas y las restricciones impuestas, por lo que indica con precisión lo que se debe programar.
- Implementación: define la organización de las clases, objetos en componentes y los nodos, así como la estructura de capas de la aplicación.
- Pruebas: es el encargado de evaluar la calidad del producto que se está desarrollando. Busca los defectos del *software* durante el ciclo de vida.
- Despliegue: produce el *release* del producto.
- Gestión y Configuración de cambios: el control de los cambios en el proyecto permite mantener la integridad de todos los artefactos que se crean en el proceso de desarrollo, así como proveer información acerca del proceso evolutivo que van experimentando los mismos.
- Administración del proyecto: realiza actividades con las que se busca que el producto satisfaga al cliente. Es el arte de lograr un balance al gestionar objetivos, riesgos y restricciones para desarrollar un producto que sea acorde a los requisitos de los clientes y los usuarios.
- Entorno: el objetivo de este flujo es dar soporte al proyecto con las herramientas, procesos y métodos adecuados. Para lograr este objetivo se debe realizar una selección y adquisición de las herramientas, además de instalarlas y configurarlas de forma tal que se ajusten a la institución.[16]

El objetivo fundamental de la metodología es asegurar la calidad del *software* y que el mismo satisfaga lo previsto por el cliente.

RUP tiene definido un grupo de roles, los que realizando las actividades que le corresponden generan diferentes artefactos. RUP es una metodología tradicional, se aplica cuando se va llevar el control completo del proceso de desarrollo y su calidad es de gran importancia. Las características que rigen a RUP son:

- Centrado en la arquitectura: porque busca que la arquitectura del *software* sea sencilla y recoja todos los elementos del sistema que está en desarrollo para que todos los involucrados puedan comprenderlo.
- Iterativo e incremental: divide el trabajo en fases para lograr hacerlo más manejable, dentro de estas fases se pueden realizar varias iteraciones según el proyecto.
- Regido por casos de uso: ya que los casos de uso son los que constituyen la guía fundamental establecida para las actividades a realizar durante todo el proceso de desarrollo incluyendo el diseño, la implementación y las pruebas del sistema.

Entre las ventajas principales que tiene están:

- Evaluación en cada fase que permite cambios de objetivos.
- Funciona bien en proyectos de innovación.
- Es sencillo, ya que sigue los pasos intuitivos necesarios a la hora de desarrollar el software.
- Seguimiento detallado en cada una de las fases.[17]

RUP es una metodología robusta que pretende abarcar y guiar todo el desarrollo del software, por lo que es la que se va utilizar en el desarrollo del componente que tiene como fin esta investigación, para poder contar con un expediente de proyecto claro y fácil de entender en el momento de ser consultado.

7.2 Herramientas CASE

Las herramientas *CASE* (*Computer Aided Software Engineering*, Ingeniería de Software Asistida por Ordenador) son las que permiten la automatización, reutilización, portabilidad y estandarización del *software*. Las herramientas *CASE* tienen como ventajas fundamentales el aumento de la calidad de los productos desarrollados y la productividad. Entre las herramientas *CASE* se encuentran el *Rational Rose* y el *Visual Paradigm*.

7.2.1 Visual Paradigm

Visual Paradigm es otra de las herramientas que permite y facilita la modelación del negocio sin crear grandes conflictos entre diseñadores, analistas y desarrolladores, ya que es un puente visual entre ellos. Entre los beneficios con que cuenta están: la navegación intuitiva entre el código y el modelo visual, poderoso generador de informes en formato de documento portátil (PDF), tiempo real en la demanda, un ambiente modelador visual superior y cuenta con sofisticado diseño de diagramas.[18]

Por todas estas ventajas que tiene y su fácil manejo esta es la herramienta de modelado seleccionada para el desarrollo del producto que se desea obtener.

7.3 Lenguaje de modelado

El Lenguaje Unificado de Modelado (UML) es un lenguaje de modelado que se utiliza para definir y documentar artefactos en el proceso de desarrollo del *software*. Este lenguaje no define un estándar para procesos específicos pero si está pensado para procesos donde el desarrollo es iterativo, por lo que es muy útil dando apoyo a metodologías como RUP. Contiene una organización en paquetes, lo que facilita dividir el trabajo en piezas más pequeñas y controlar así la dependencia entre paquetes. Su objetivo fundamental es que es un lenguaje de modelado general y que todos los modeladores lo puedan usar.[19]

8. Lenguaje de programación

Los lenguajes de programación surgieron un poco después de los ordenadores, su creación vino dada a la gran necesidad de programarle tareas fundamentales a los ordenadores. Los primeros eran muy difíciles de desarrollar, en la actualidad han evolucionado logrando facilitar su aprendizaje, uso y compilación. Entre los más usados se encuentran PHP, Java y C Sharp. Este último será el lenguaje en el que se va a desarrollar el Componente para controlar las funcionalidades del ratón de la computadora haciendo uso de una cámara digital.

8.1 C Sharp

El C Sharp (C#) es un lenguaje de programación orientado a objeto, el cual ha transitado un largo camino desde que salió a luz C, seguido por el C++ hasta llegar al C#. El C# es un híbrido entre el C++ y Java, mezclando la combinación de operadores del primero y la plena orientación a objetos del segundo. Actualmente C# se encuentra entre los diez lenguajes más

utilizados. A pesar de su corta historia, ha recibido la aprobación de estándar de dos organizaciones: en el 2001 se aprueba el ECMA y en el 2003 el ISO.

Las ventajas que ofrece respecto a otros lenguajes son varias, la declaración de datos tiene un rango más amplio y mejor definido que otros lenguajes como el C++ o el Java, cada miembro de las clase tiene atributos lo que pueden estar en diferentes niveles como: público, privado o protegido, permite el pase de parámetro, de forma predeterminada, el pase de parámetros es por valor, a menos que se use la palabra reservada *ref*, la cual indica que el pase es por referencia, permite el control de versiones, lo que facilita versiones nuevas y anteriores de *software* puedan ejecutarse en forma simultánea.

8.2 Microsoft Visual Studio 2010 Ultimate

Microsoft Visual Studio 2010 Ultimate incluye potentes herramientas que simplifican todo el proceso de desarrollo de aplicaciones, de principio a fin. Los equipos pueden observar una mayor productividad y ahorro de costes al utilizar características de colaboración avanzadas, así como herramientas de pruebas y depuración integradas que le ayudarán a crear siempre un código de gran calidad. Entre las características del mismo están:

Compatibilidad con la plataforma de desarrollo

Tanto si crea soluciones nuevas como si quiere mejorar las aplicaciones ya existentes, *Visual Studio 2010 Ultimate* le permite hacer realidad su idea en una gran variedad de plataformas, entre las que se incluyen *Windows*, *Windows Server*, *Web*, *Cloud*, *Office y SharePoint*, entre otras, todo en un único entorno de desarrollo integrado.

Entorno de desarrollo integrado

Visual Studio 2010 Ultimate le permite ponerse al mando, aprovechándose de las características personalizables como, por ejemplo, compatibilidad con varios monitores, de modo que pueda organizar y administrar su trabajo como quiera. También puede dar rienda suelta a su creatividad utilizando los diseñadores visuales para mejorar las últimas plataformas, incluido *Windows* 7.[20]

8.2.1 OpenCV

OpenCV "Open Source Computer Vision Library", una librería de tratamiento de imágenes, destinada principalmente a aplicaciones de visión por computador en tiempo real.[21], desarrollado por *Intel*. Esta librería proporciona un alto nivel de funciones para el procesado de imágenes. Permite a los programadores crear aplicaciones poderosas en el dominio de la visión digital. *OpenCV* ofrece muchos tipos de datos de alto-nivel como árboles, gráficos y matrices, es de código abierto y multiplataforma. Implementa una gran variedad de herramientas para la interpretación de la imagen. Es compatible con *Intel Image Processing Library* (IPL) que implementa algunas operaciones en imágenes digitales. Posee primitivas como binarización, filtrado, estadísticas de la imagen y pirámides. *OpenCV* es principalmente una librería que implementa algoritmos para las técnicas de la calibración (calibración de la cámara), detección de rasgos, para rastrear (flujo óptico), análisis de la forma (geometría, contorno que procesa), análisis del movimiento (plantillas del movimiento, estimadores), reconstrucción 3D (transformación de vistas), segmentación de objetos y reconocimiento (histograma).[22] Además, tiene otras funciones referidas a capturas en tiempo real desde dispositivos de entrada como puede ser una cámara digital.

La principal característica de esta biblioteca, además de su funcionalidad y calidad, es su rendimiento. Los algoritmos están basados en estructuras de datos altamente flexibles, y más de la mitad de las funciones han sido optimizadas aprovechando la arquitectura de *Intel*.
Esta librería consta con distintos tipos de versiones o *wrappers* para distintos tipos de lenguaje. Ejemplo para C# esta *Emgu Cv* la cual se utiliza para el desarrollo de la aplicación.

9. Conclusiones parciales

El análisis realizado sobre los sistemas ya existentes permitió determinar las principales deficiencias que enfrentan estos tipos de componentes en la actualidad, la utilidad que se le da a los mismos y cuáles son los más usados. Lo que conllevó a determinar cuál sería la solución a estas principales deficiencias, para que los usuarios se sientan cómodos con su uso.

Como solución a las principales deficiencias detectadas se determinaron algunos aportes que se pueden lograr con el desarrollo de un nuevo componente, estos aportes pueden ser: controlar el desplazamiento del puntero del cursor con el movimiento de los ojos y la nariz.

Utilizando para ello la técnica de detección de rostro basada en imagen y los clasificadores, permitiendo así que el proceso de desarrollo se centre en el control de las funcionalidades del ratón de la computadora y no el proceso de detección de rostro en sí. Concluyendo que las herramientas, tecnología y metodología a utilizar van a facilitar el proceso de desarrollo del componente.

Capítulo 2: Características del sistema

Introducción

En este capítulo se recogen las diferentes características del componente que permitirá controlar las funcionalidades del ratón de la computadora haciendo uso de una cámara digital, los requisitos funcionales y no funcionales con que debe contar el mismo. Será donde se describan los diagramas que se modelan en proceso de desarrollo y las descripciones de los casos de uso del sistema y donde quede definida la arquitectura a seguir en el desarrollo del componente.

1. Descripción del sistema

El componente que se propone debe ser capaz de facilitarle al usuario controlar el cursor de la computadora haciendo uso de una cámara digital. Con el componente a desarrollar en esta investigación el desplazamiento del cursor del ratón debe ser suave y estable.

La interfaz con que el usuario va a trabajar debe ser intuitiva y fácil de entender. Debe contar con la opción de poder configurar la acción del clic a realizar (clic derecho, clic izquierdo), estas opciones de configuración son: por tiempo y con los ojos con la acción del pestañeo. Otra opción con que debe contar es con la de calibración, la misma permitirá entrenar la mirada del usuario siguiendo diferentes puntos con distintas localizaciones en la pantalla.

El funcionamiento del componente parte de la cámara digital, la cámara digital captura la imagen para su procesamiento, a la imagen se le aplica los *haarcascade* (clasificadores) para detectar el rostro, del rostro se extraen las partes de interés, que pueden ser la nariz o los ojos, después se usa el algoritmo para la detección de las parte de interés y así obtener su localización, la cual se señalará en la imagen. A partir de la parte detectada, la nariz o la pupila, se podrá controlar el desplazamiento del puntero del cursor del ratón de la computadora.

Para realizar la acción del clic a través del pestañeo se aplica un algoritmo para detectar los cambios de intensidad de los píxeles en la imagen. Para esta acción se parte de la imagen, se procesa la misma aplicándole el algoritmo para detectar la acción del pestañeo y finalmente se ejecuta la acción del clic si este detectó algún cambio en la imagen.

A continuación se muestran tres gráficos donde se refleja el flujo que se lleva a cabo en las acciones de mover el cursor con los ojos o la nariz y ejecutar la acción del clic.

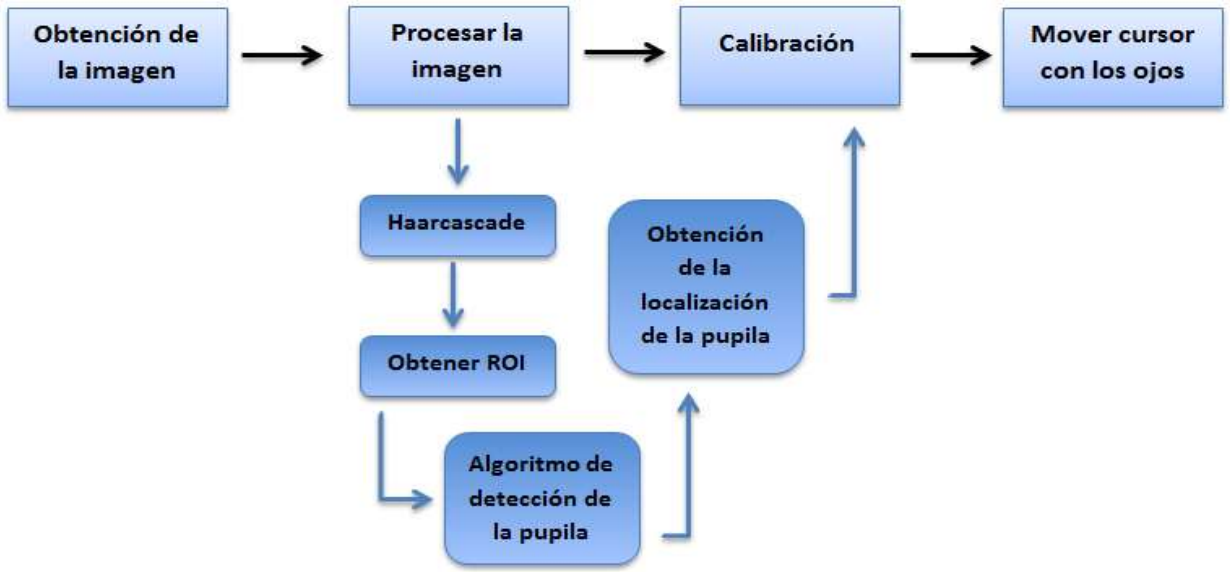


Figura 5: Descripción del sistema (acción de mover el cursor con los ojos)

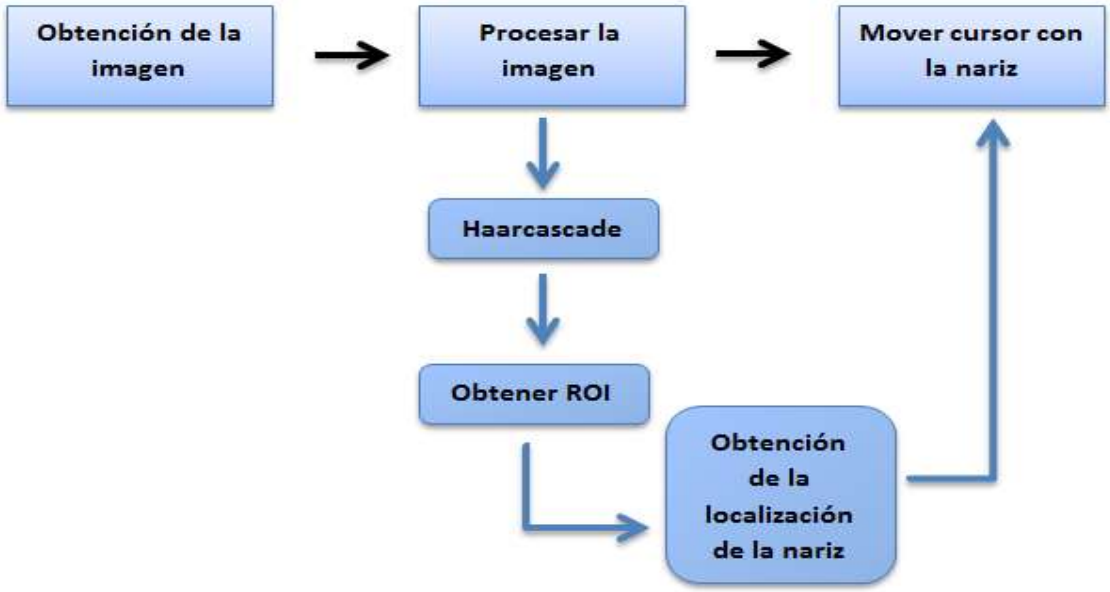


Figura 6: Descripción del sistema (acción de mover el cursor con la nariz)

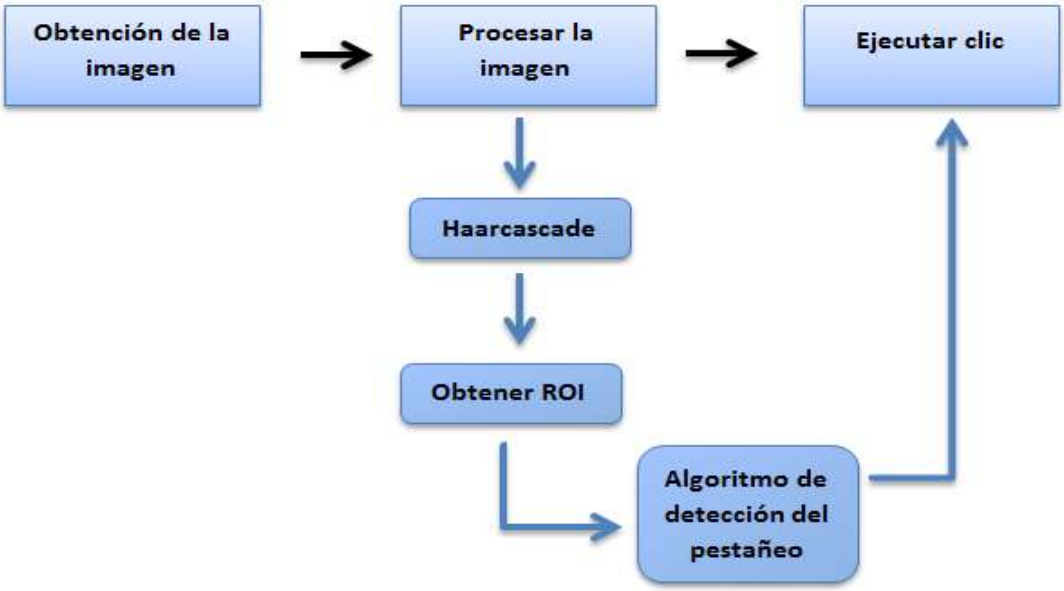


Figura 7: Descripción del sistema (acción de ejecutar clic con el pestañeo)

1.1 Cálculo del movimiento de la pupila

Una vez encontrada la localización de la pupila, se hicieron varios cálculos para reflejar los movimientos de la pupila en el cursor de la computadora. Para esto se utilizaron vectores de dirección y fórmulas de geometría plana. Para lograr darle dirección al cursor primero se trazaron líneas que forman un triángulo, es decir se trazó una línea de la pupila del ojo izquierdo a la pupila del ojo derecho y luego dos líneas más de las pupilas respectivamente hacia la nariz, quedando formado un triángulo. A continuación se muestra como queda formado dicho triángulo.

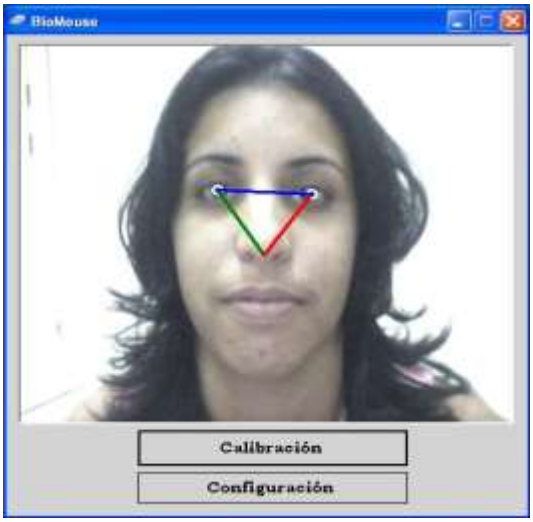


Figura 8: Triángulo formado para la detección de la pupila

Una vez trazado el triángulo se buscan los ángulos internos, ya que nuestra mirada en dependencia para donde se esté mirando los ángulos que se forman en las pupilas varían, pueden ser más grande o más pequeños, en dependencia de la dirección de las pupilas; ejemplo si se mira para la izquierda el ángulo formado por la pupila derecha va a ser más grande que el de la pupila izquierda y viceversa. Para encontrar los ángulos internos se utilizó la ley de los cosenos la cual se representa en dichas fórmulas, a través de los despejes correspondientes se obtienen los ángulos respectivamente:

$$c^2 = a^2 + b^2 - 2ab \cos C.$$

$$a^2 = b^2 + c^2 - 2bc \cos A$$

y

$$b^2 = c^2 + a^2 - 2ca \cos B$$

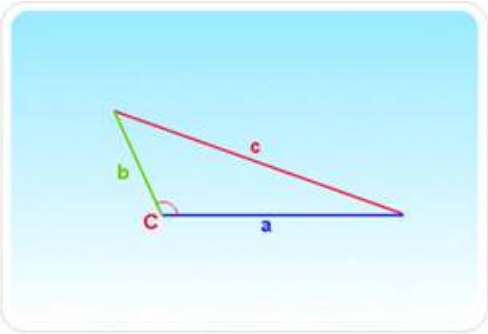


Figura 9: Fórmulas de la ley de los cosenos

Siendo (a, b, c) las longitudes de los lados del triángulo formado. Por lo cual al obtener dichos ángulos se pudo mover el puntero del cursor para el lugar en donde se mira, pero solo en el eje x (de un lado a otro).

Para resolver el movimiento de forma vertical se trazó un rectángulo enmarcado en la región de los ojos y otro que parte de la localización de la pupila, es decir que el movimiento de este rectángulo es controlado por las pupilas. Si el rectángulo que está contenido dentro del que enmarca los ojos se acerca a la parte superior o inferior de este, se podría detectar cuando las pupilas se mueven de arriba hacia abajo. A continuación se muestra como quedan formados dichos rectángulos.

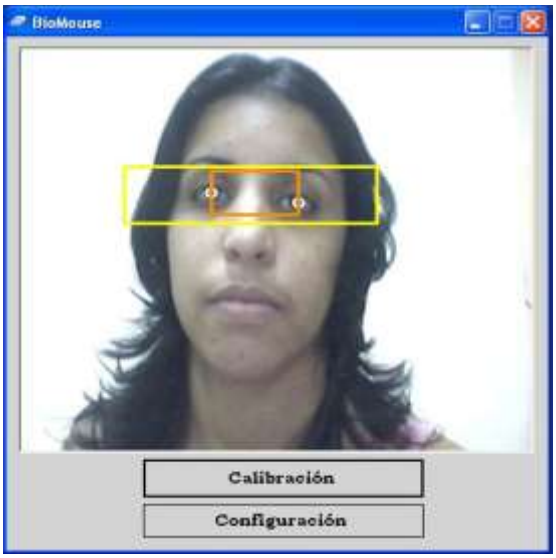


Figura 10: Rectángulo formado para el control del movimiento de la pupila

Al obtener la dirección de la mirada se le suma a la posición actual del cursor unos deltas pequeños (deltax=10 y deltay=10) para que este se desplace en la dirección que el usuario esté mirando.

1.2 Cálculo del movimiento de la nariz

Al obtener la localización de la nariz dentro de la imagen se trazó un rectángulo en el centro de la imagen a mostrar, el cual equivaldría a las longitudes del monitor, dentro de esta área es donde el cursor podrá moverse. Para ello se hicieron cálculos de proporción para que el movimiento dentro del área señalada se reflejara por toda la pantalla (Figura 22). Para encontrar la posición en el eje ***x*** y el eje ***y*** se aplicaron dicha fórmulas respectivamente:

Fórmula del eje X	**X=wm*(Xnc - Xar)/wr**
Fórmula del eje Y	**Y=hm*(Ync-Yar)/hr**

Tabla 2: Fórmulas del cálculo del movimiento de la nariz

Siendo wm el ancho del monitor, Xnc eje x del centro de la nariz, Xar el eje x del rectángulo, wr el ancho del rectángulo, hm alto del monitor, Ync eje y del centro de la nariz, Yar eje y del rectángulo y hr el alto del rectángulo. Pudiéndose así desplazar el cursor por toda la pantalla.

2. Especificación de la arquitectura

Arquitectura - IEEE 1471-2000:

Capítulo 2

La Arquitectura de *Software* es la organización fundamental de un sistema, encarnada en sus componentes, las relaciones entre ellos, el ambiente y los principios que orientan su diseño y evolución.

Las arquitecturas de *software* son agrupadas por estilos o patrones arquitectónicos. Los estilos reúnen un grupo de arquitecturas que tienen en común varias características, además de ser muy importantes en el cumplimiento de los requerimientos de calidad. Entre los estilos más comunes se encuentran: *Peer to peer*, Flujo de datos, Centrado en datos y el de Llamada y retorno. El estilo utilizado como guía en el desarrollo del componente es el de Llamada y retorno, la arquitectura a usar de las agrupadas en el mismo es la Arquitectura en capas.

2.1 Arquitectura en capas

La arquitectura en capas define un conjunto de capas o niveles, donde cada capa se comunica solamente con la capa siguiente. Esta arquitectura brinda varias ventajas entre las que se encuentra el desarrollo paralelo en cada capa; mantenimiento y soporte más sencillo, ya que es más fácil cambiar una funcionalidad que modificar todo un componente. La arquitectura en capas es un muy buen modelo para sistemas donde su solución puede estar inmersa en problemas de cambios.

La arquitectura sirve de guía para desarrolladores, analistas y diseñadores; favoreciendo el cumplimiento de los objetivos previstos para el desarrollo del componente. Normalmente se utiliza dividida en tres capas, presentación, negocio o dominio y acceso a dato. En el desarrollo de este componente no se cuenta con base de datos ya que no se requiere de ninguna identidad persistente para el funcionamiento del componente, por lo que la arquitectura de este componente queda dividida en dos capas: presentación y negocio.

A continuación se describe el funcionamiento de ambas capas:
Presentación: es la capa donde están las interfaces de usuario, mediante las cuales el usuario podrá interactuar con el componente para controlar las funcionalidades del ratón de la computadora mediante el uso dado al *framework* 4.0.

Negocio: es la capa donde se gestiona todo el proceso de controlar las funcionalidades del ratón de la computadora, es la encargada de recoger todos los cambios que el usuario pueda hacer en la configuración de la acción del clic y de la opción del control del puntero del cursor.

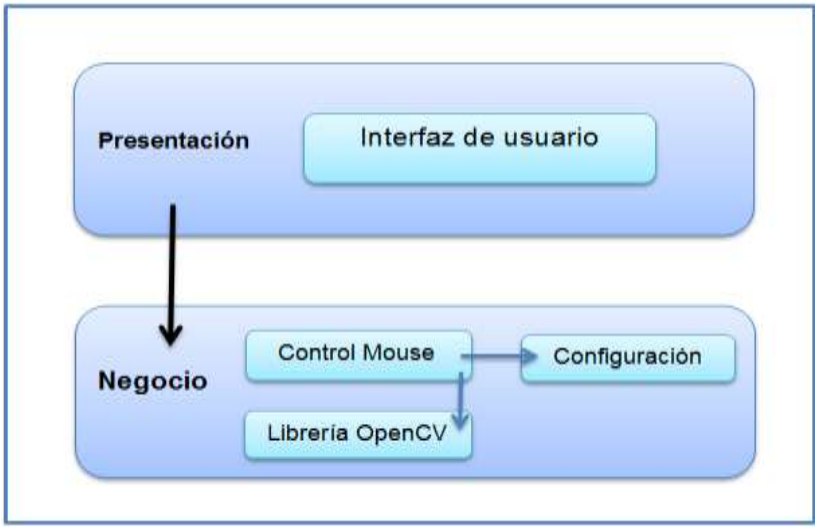


Figura 11: Arquitectura en capas

3. Modelo de dominio

El Modelo de dominio es una representación visual de clases conceptuales o de objetos del dominio. Se realiza cuando los términos del negocio no están bien definidos. Su representación es mediante diagrama UML (especialmente mediante diagramas de clases), donde se identifican las clases conceptuales y se les añade relación y atributos. Con la modelación del dominio se pretende entender los conceptos con que trabaja el componente.[23]

Seguidamente se explican los conceptos tratados en el proceso de desarrollo del componente para controlar las funcionalidades del ratón de la computadora haciendo uso de una cámara digital y a continuación estará la representación gráfica del Modelo de dominio.

- Persona: es el individuo que necesite controlar las funcionalidades del ratón de la computadora.
- Webcam: dispositivo mediante el cual se captura el vídeo, del cual se obtiene la imagen para su procesamiento.
- Imagen: es en la que se detecta la pupila o la nariz, para con la misma poder controlar el movimiento del cursor de la computadora.
- Región_intrés: es la parte del rostro detectada con la que se controlará el movimiento del cursor.

- Movimiento_cursor: es el objetivo a controlar con el movimiento de los ojos o la nariz.
- Clic: es la acción que se puede ejecutar partiendo de la región de interés.
- Configuración: es la persistencia de los cambios que puede realizar el usuario sobre los valores ya predefinidos.

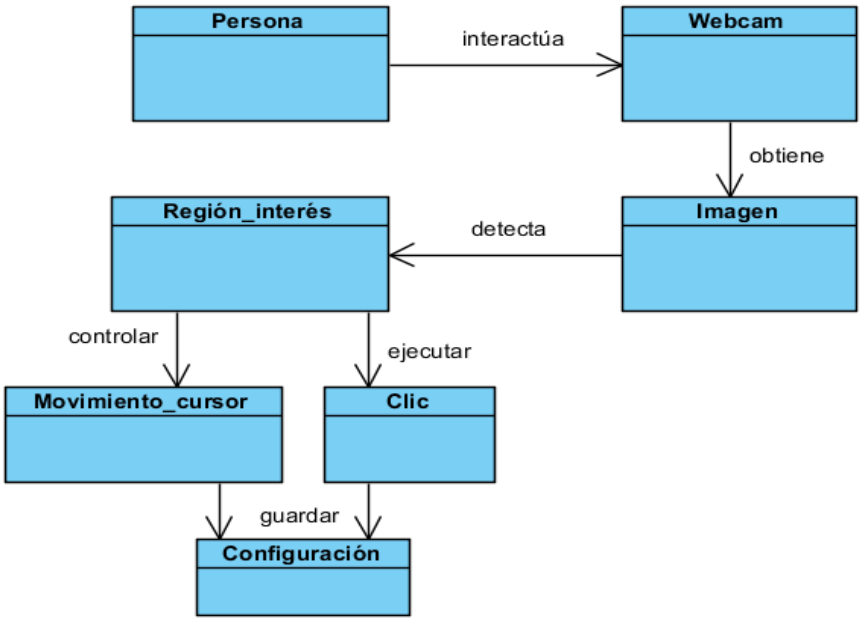


Figura 12: Modelo de Dominio

4. Requisitos

Según la IEEE *Standard Glossary of Software Engineering Terminology*, los requisitos se definen como una "condición o capacidad que necesita un usuario para resolver un problema o lograr un objetivo". Los requisitos es la manera de comprender con el cliente sobre las especificaciones que debe tener el producto. Los requisitos pueden ser funcionales y no funcionales. La captura de los requisitos es una tarea de comunicación con todas las fuentes involucradas en el proceso: el cliente, los analistas y los diseñadores, dado a la misma se descubre que es lo que realmente necesita el componente. Los mismos deben ser fáciles de entender y explicar.

4.1 Requisitos Funcionales

Los requisitos funcionales (RF) son capacidades o condiciones que el componente debe cumplir. Se mantienen invariables sin importar con que propiedades o cualidades se relacionen. A continuación se muestran los requisitos funcionales definidos para el componente.

Número	Requisitos Funcionales
RF 1	Configurar la opción del clic por tiempo.
RF 1.1	Realizar la acción de clic izquierdo (por tiempo).
RF 1.2	Realizar la acción de clic derecho (por tiempo).
RF 2	Configurar la opción del clic con los ojos.
RF 2.1	Realizar la acción de clic izquierdo (con los ojos).
RF 2.2	Realizar la acción de clic derecho (con los ojos).
RF 3	Controlar el movimiento del puntero del cursor.
RF 3.1	Mover el puntero del cursor con los ojos.
RF 3.2	Mover el puntero del cursor con la nariz.
RF 4	Calibrar la mirada.

Tabla 3: Requisitos Funcionales

4.2 Requisitos no Funcionales

Los requisitos no funcionales (RNF) son las cualidades que debe tener el componente pero que no son una funcionalidad en específico; los requisitos no funcionales se pueden clasificar en varios tipos como: usabilidad, soporte, *hardware*, *software*, seguridad, diseño e implementación, apariencia e interfaz externa. A continuación se muestran los requisitos no funcionales definidos para el componente.

Número	Requisitos no Funcionales
Diseño e implementación	
RNF 1	Utilizar la herramienta IDE de desarrollo Visual Studio 2010.
RNF 2	Utilizar C Sharp como lenguaje de programación.

RNF 3	Utilizar la herramienta CASE Visual Paradigm for UML 8.0 Standard Edition.
RNF 4	Realizar el modelado con UML.
Software	
RNF 5	Sistema operativo WindowsXP/Vista7.
Hardware	
RNF 6	En el cliente se requiere una máquina con 256Mb de RAM y un microprocesador de 1GHz como mínimo.
RNF 7	La PC debe contar con al menos un puerto de conexión USB.
RNF 8	La cámara con que cuenta el cliente debe tener como mínimo 2.0mpx.
Usabilidad	
RNF 9	El sistema podrá ser utilizado por cualquier usuario con conocimientos básicos sobre el uso de una computadora.
RNF 10	El usuario debe tener una postura correcta delante de la cámara para poder lograr correctamente la detección del rostro.
RNF 11	El local donde se vaya a utilizar el componente debe contar con un buen nivel de iluminación.
Apariencia o interfaz externa	
RNF 12	La interfaz del componente debe ser sencilla y amigable.
RNF 13	Los colores a usar en la interfaz deben se agradables a la vista.
Rendimiento	
RNF 14	Los tiempos de respuesta del componente deben ser rápidos y precisos.

Tabla 4: Requisitos no Funcionales

5. Modelo de Caso de Uso del Sistema

El Modelo de Caso de Uso es la representación en forma de diagrama de las funcionalidades del componente. Con el mismo se pueden representar conceptos como el de la Generalización/Especialización. Está formado por actores, caso de uso y la relación entre ellos. Un caso de uso tiene la descripción de una unidad significativa de trabajo de una funcionalidad del componente además puede incluir o extender funcionalidades de otro caso de uso con su propio comportamiento.

5.1 Diagrama de Caso de Uso del Sistema (CUS)

El Diagrama de Caso de Uso del Sistema se realiza generalmente partiendo de los requisitos funcionales ya definidos. En el mismo todos los casos de uso deben estar relacionados con los actores y un caso de uso solo puede ser inicializado por un único actor, mientras que un actor puede inicializar más de un caso de uso. Un caso de uso es una funcionalidad que el componente puede realizar, la misma en su interior sigue un grupo de pasos para ser ejecutada. El siguiente diagrama es el propuesto para el componente que permitirá controlar las funcionalidades del ratón de la computadora haciendo uso de una cámara digital.

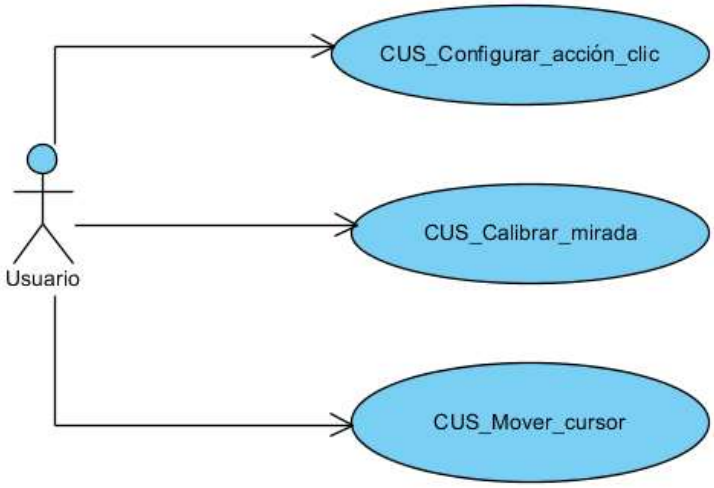


Figura 13: Diagrama de Casos de Uso del Sistema

5.2 Descripción de los Casos de Uso del Sistema

Con la descripción de los casos de uso se podrá entender de forma individual cada uno de ellos, ya que con el diagrama solo se tiene una visión general del funcionamiento del componente. La misma se puede realizar de dos maneras extendida y breve, la realizada para comprender de forma específica todo el proceso que se recoge en una funcionalidad es la

extendida. Seguidamente aparece la descripción del caso de uso Configurar_acción_clic, la de los casos de uso restantes se pueden encontrar en los anexos.

Nombre del caso de uso: Configurar_acción_clic.	
Actores	Usuario
Propósito	Configurar las acciones que se pueden de realizar con el cursor.
Resumen	El caso de uso inicia cuando el usuario selecciona la opción de configuración, el usuario puede configurar las acciones de realizar el clic.
Referencia	RF-1, RF-2
Precondiciones	

Flujo normal de eventos	
Acción del actor	**Respuesta del sistema**
1. El usuario selecciona la opción de configuración de la acción del clic.	
	2. El sistema muestra una nueva interfaz con las opciones de configuración a escoger por el usuario. Las opciones de configuración son: a) La acción de realizar el clic. Que puede ser: - Por tiempo. - Con los ojos.
3. El usuario realiza la siguiente acciones: a) Configurar la acción de realizar el clic. Que puede ser:	

3.1 Por tiempo. 3.2 Con los ojos.	
	4. 4.1 El sistema cuenta el tiempo establecido por el usuario para ejecutar la acción del clic. 4.2 El sistema trabaja con el algoritmo necesario para identificar si el pestañeo es más fuerte de lo normal para ejecutar la acción del clic.
5. El usuario da clic en el botón aceptar de la interfaz de configuración. En caso contrario dirigirse a la sección de flujo alterno.	
	6. El sistema guarda en un fichero la configuración realizada por el usuario para realizar las acciones del clic.
Flujo Alterno: “ Cancelar configuración realizada por el usuario”	
5.1 El usuario desea cancelar la acción de configuración que estaba realizando.	6.1 El sistema cierra la ventana de configuración y no guarda los cambios que el usuario haya realizado.
Pos condiciones	Se tenía ya predefinida una opción para realizar la acción del clic y controlar el desplazamiento del cursor.

Tabla 5: Descripción del CUS_Configurar_acción_clic

7. Análisis

El análisis consiste en obtener una visión del componente que se preocupa de ver qué hace, de modo que solo se interesa por los requisitos funcionales.[16] Su objetivo es comprender de forma precisa los requisitos, y refinarlos para que permitan estructurar el sistema entero.

7.1 Modelo de análisis

El modelo de análisis está descrito con el lenguaje del desarrollador, es la vista del sistema y está estructurado por clases estereotipadas, es la manera del desarrollador entender como deber ser implementado el componente. Los estereotipos manejados en el análisis son:

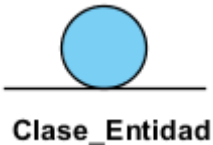
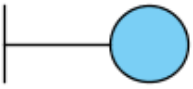
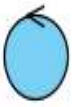
Nombre	Estereotipo	Descripción
Entidad		Estas clases son utilizadas para modelar información que posee larga vida y que es a menudo persistente. Poseen la información acerca de algún fenómeno (persona, objeto del mundo real o suceso).[23]
Interfaz		Se utilizan para modelar la interacción entre el sistema y sus actores (usuarios, y sistemas externos). Modelan las partes del sistema que dependen de sus actores.[23]
Control		Los aspectos dinámicos del sistema se modelan con clases de control, debido a que ellas manejan y coordinan las acciones y los flujos de control principal, y delegan trabajo a otros objetos (es decir objetos de interfaz y de entidad).[23]

Tabla 6: Descripción de los estereotipos del modelo de análisis
Fuente: Elaboración propia

7.2 Diagrama de clases del análisis

Con el diagrama de clases del análisis se representan los conceptos en un dominio del problema. Representa las cosas del mundo real, no la implementación automatizada. Cada diagrama representa un único caso de uso.

El diagrama de clases del análisis que se muestra a continuación se corresponde con el caso de uso Configurar_acción_clic, los demás diagramas aparecen en los anexos.

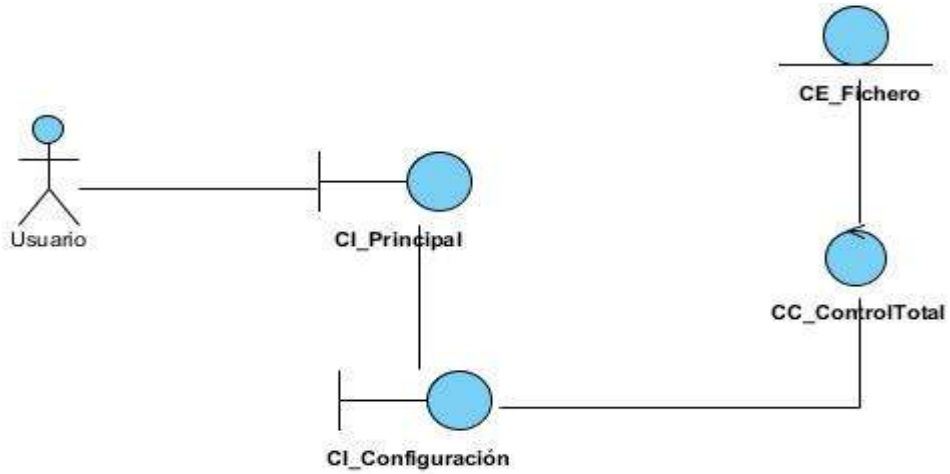


Figura 14: Diagrama de clases del análisis del CUS_Configurar_acción_clic

7.3 Diagrama de colaboración

Con el diagrama de colaboración se destaca la organización de los objetos que participan en una interacción. Estos enlaces se realizan mediante mensajes que envían y reciben los objetos. Para indicar la ordenación temporal de un mensaje, se precede de un número (comenzando con el mensaje número 1), que se incrementa secuencialmente por cada nuevo mensaje en el flujo de control (2, 3 sucesivamente).

El diagrama de colaboración que se muestra a continuación se corresponde con el caso de uso Configurar_acción_clic, los demás diagramas se pueden apreciar en los anexos.

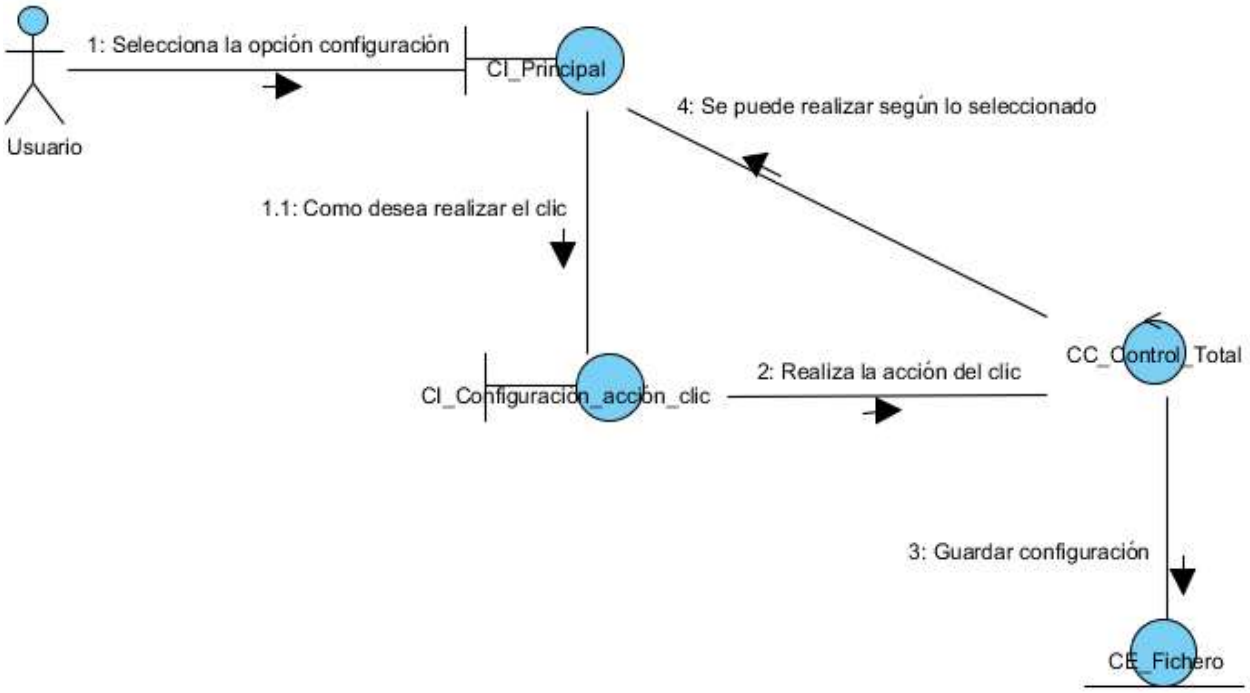


Figura 15: Diagrama de colaboración del CUS_Configurar_acción_clic

7.4 Diagrama de secuencia

Los diagramas de secuencia representan la interacción de los objetos del componente a través del tiempo. Se elabora un diagrama de secuencia por cada caso de uso. Un diagrama de secuencia muestra los objetos que intervienen en el escenario con líneas discontinuas verticales, y los mensajes pasados entre los objetos como flechas horizontales. Existen dos tipos de mensajes: sincrónicos y asincrónicos. Los mensajes sincrónicos se corresponden con llamadas a métodos del objeto que recibe el mensaje. El objeto que envía el mensaje queda bloqueado hasta que termina la llamada. Este tipo de mensajes se representan con flechas con la cabeza llena. Los mensajes asincrónicos terminan inmediatamente, y crean un nuevo hilo de ejecución dentro de la secuencia. Se representan con flechas con la cabeza abierta. También se representa la respuesta a un mensaje con una flecha discontinua. A continuación se refleja en diagrama de secuencia del caso de uso Configurar_acción_clic, los de los demás diagramas se pueden apreciar en los anexos.

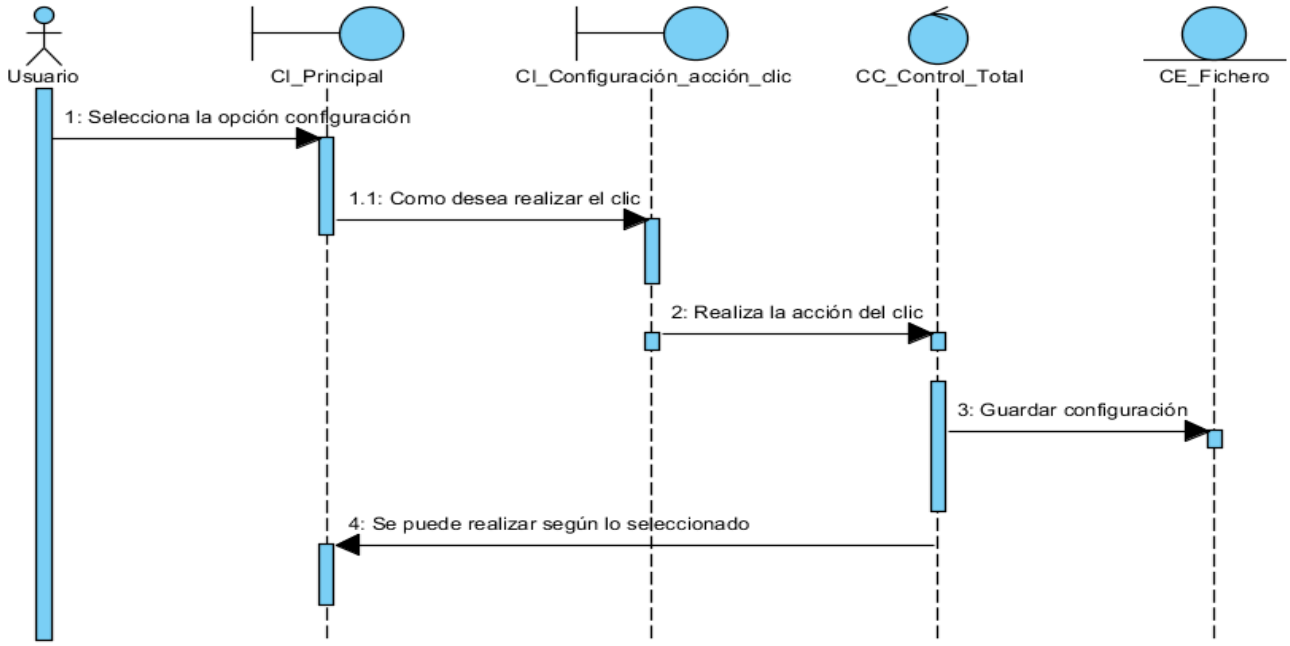


Figura 16: Diagrama de secuencia del CUS_Configurar_acción_clic

8. Diseño

El diseño es el encargado del refinamiento del análisis, tiene en cuenta los requisitos no funcionales. El diseño debe ser suficiente para que el componente sea desarrollado sin ambigüedad.

8.1 Patrones de diseño

Los patrones de diseño son el dúo problema-solución a inconvenientes del diseño. Los patrones de diseño son los que brindan una solución ya probada y documentada a los problemas que pueden aparecer en el diseño del *software*.

Los patrones de diseño pretenden: proporcionar catálogos de elementos reusables en el diseño de sistemas, evitar la reiteración en la búsqueda de soluciones a problemas ya conocidos y solucionados anteriormente, formalizar un vocabulario común entre diseñadores, estandarizar el modo en que se realiza el diseño, facilitar el aprendizaje de las nuevas generaciones de diseñadores condensando conocimiento ya existente. Entre los patrones existentes los utilizados en el diseño de este componente son los explicados a continuación:

8.1.1 Patrones GRASP (General Responsibility Assignment Software Patterns)

Los patrones GRASP describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en forma de patrones. Con estos patrones se puede entender el diseño del componente. Los patrones GRASP se dividen en seis, ellos son: experto, creador, controlador, fachada, alta cohesión y bajo acoplamiento, a continuación se explican los usados y donde se manifiestan.

- Alta cohesión: la cohesión es una medida de la fuerza con la que se relacionan las clases y el grado de focalización de las responsabilidades de un elemento, cada elemento del diseño debe realizar una labor única dentro del sistema, no desempeñada por el resto de los elementos y auto-identificable, una clase con baja cohesión hace muchas cosas no relacionadas o hace demasiado trabajo.

  En este caso las clases cuentan con una alta cohesión ya que hacen las actividades que les corresponden sin hacer muchas tareas. Ejemplo de esto es la clase *MouseDriver,* ya que la misma solo se encarga de ejecutar las acciones del clic.
- Bajo acoplamiento: este patrón es un principio que asigna la responsabilidad de controlar el flujo de eventos del sistema, a clases específicas; esto facilita la centralización de actividades. El controlador no realiza estas actividades, las delega en otras clases con las que mantiene un modelo de alta cohesión. Un error muy común es asignarle demasiada responsabilidad y alto nivel de acoplamiento con el resto de los componentes del sistema.

  Este patrón tiene su uso en las clases que se implementan para desarrollar el componente, ya que una clase no dependen de muchas otras clases, un ejemplo específico de donde se pone de manifiesto el mismo es en la clase controladora, *MouseControl* ya que no es la encargada de realizar todas las funcionalidades del componente, transmitiéndole responsabilidades a otras clases como a *MouseDriver*.
- Experto: este patrón se aplica en todas las clases que cuentan con la información necesaria para cumplir una responsabilidad.[24] Un ejemplo de este patrón está en la clase *ConfigurationClass*, la misma solo cuenta con la información necesaria para guardar los cambios que realiza el usuario.

Los patrones alta cohesión y bajo acoplamiento se aplican en todas las clases de la implementación del componente, ya que cada clase se relaciona con un número pequeño de clases y solo contienen las funcionalidades requeridas para su uso en el componente.

8.2 Modelo de diseño

El modelo de diseño se centra en cómo cumple el componente sus objetivos, cómo los requisitos funcionales y no funcionales, junto con otras restricciones relacionadas en el entorno de la implementación, tienen impacto en el componente a considerar. El modelo de diseño sirve de abstracción de la implementación del componente.

8.3 Diagrama de diseño

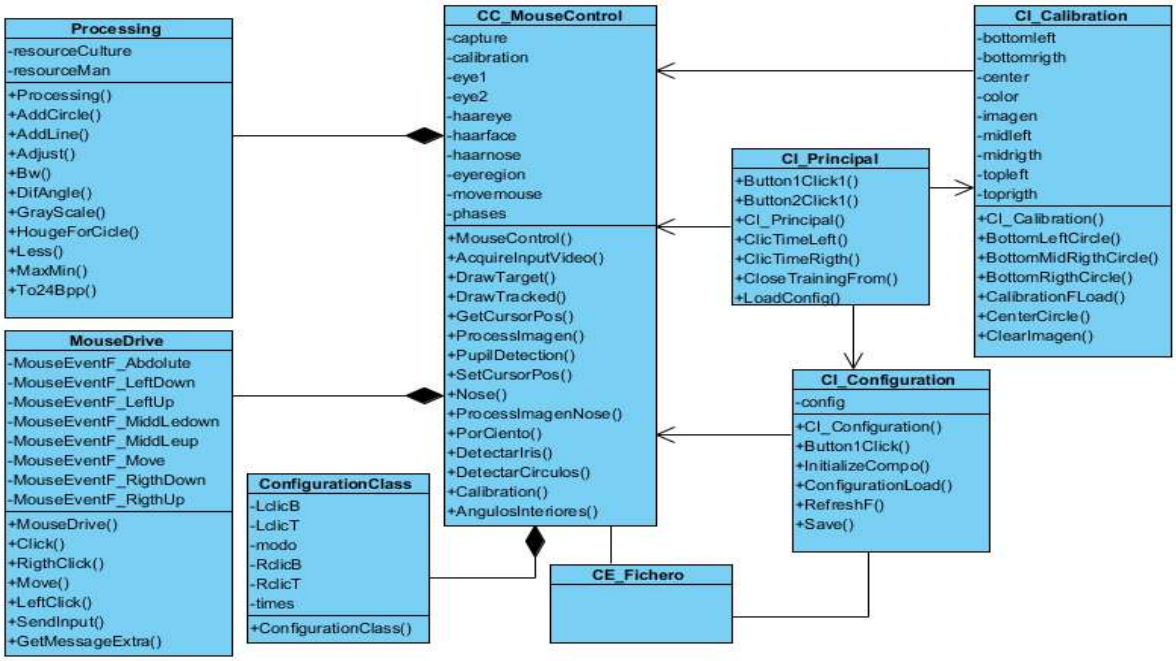


Figura 17: Diagrama de clases del diseño

9. Conclusiones parciales

Se elaboró la propuesta de solución del componente a desarrollar y se realizó una descripción de la misma mediante diagramas. Con la confección de los diagramas de la descripción del funcionamiento del componente se pudo definir las funcionalidades con que contará el mismo.

Permitiendo definir los requisitos funcionales y no funcionales con que debe contar el componente.

Se determinó que los patrones de diseño a utilizar serán los de asignación de responsabilidad y que la arquitectura a implementar será en dos capas. Se concluyó en todo lo antes expuesto por las particularidades que tendrá el componente a implementar.

Capítulo 3: Implementación y prueba

Introducción

En este capítulo será donde se documente la implementación del componente. Se le realizarán las pruebas de calidad al componente, las mismas serán las de caja blanca y caja negra. En el mismo se mostrarán las interfaces de usuario y se explicará su funcionamiento.

1. Modelo de despliegue

Es un modelo de objetos que describe la distribución física del componente en términos de cómo se distribuye la funcionalidad entre los nodos de cómputo. Los nodos poseen relaciones que representan medios de comunicación entre ellos. La funcionalidad de un nodo se define por los componentes que se distribuyen en ese nodo.

Todos los componentes no necesitan de un modelo de despliegue, ya que el mismo es la representación de cómo se gestionan los servicios. En el caso del componente que se está desarrollando en esta investigación no se realiza ya que los dispositivos usados son gestionados desde una sola computadora.[25]

1.1 Diagrama de componentes

Los diagramas de componentes representan la distribución física del sistema y sus relaciones. Los componentes que forman parte del diagrama son los que estuvieron presentes en la implementación del mismo.

Para la confección del diagrama de componentes se trabaja con cinco tipos de estereotipo, que son: ejecutables (.exe) son los que especifican un componente ejecutable, librerías (.dll) especifican bibliotecas de objetos, tablas (.db) especifica una tabla de la base de dato, archivos (.fl) especifica un componente que representa un documento que contiene código fuente o datos y documento (.doc) que especifica un componente que especifica un documento.[26]
Los estereotipos utilizados en el diagrama de componente, del Componente para controlar las funcionalidades del ratón de la computadora haciendo usa de una cámara digital, fueron los de

ejecutable y librerías, se usaron para la representación del componente a ejecutar y las librerías utilizadas en el proceso de desarrollo.

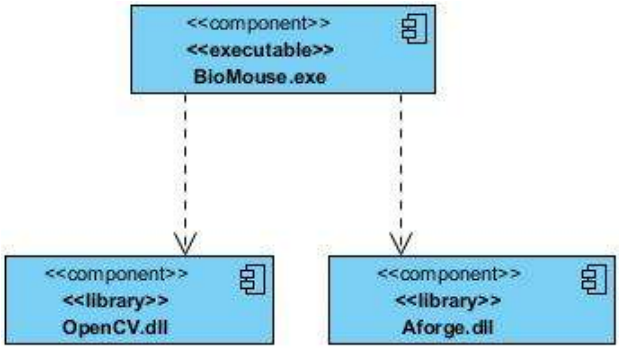


Figura 18: Diagrama de componente

2. Interfaz de usuario

Las interfaces son el vínculo directo que tiene el usuario con el componente. Mediante las interfaces el usuario puede explotar las funcionalidades que brinda el componente. Las interfaces del componente son fáciles de usar y agradables a la vista del usuario, tienen bien distribuida las opciones que el usuario necesita para controlar las funcionalidades del ratón de la computadora.

Las interfaces con que cuenta el componente se mostrarán a continuación dando una breve descripción de las mismas:

- Interfaz principal es donde se refleja la imagen del usuario y a la vez se detecta la región de interés que pueden ser la pupila o la nariz. En la misma se cuenta con las opciones de configuración y calibración, las que llevan a nuevas interfaces. Además es donde se encuentra la ayuda del componente, en la misma se le explica al usuario el funcionamiento del componente.

Figura 19: Interfaz principal

- Interfaz de configuración es donde el usuario puede modificar la opción de realizar la acción del clic, en la misma puede decidir que clic desea realizar y de qué forma desea realizarlo si por tiempo o pestañeo. Además puede seleccionar de qué forma desea desplazar el movimiento del puntero del cursor.

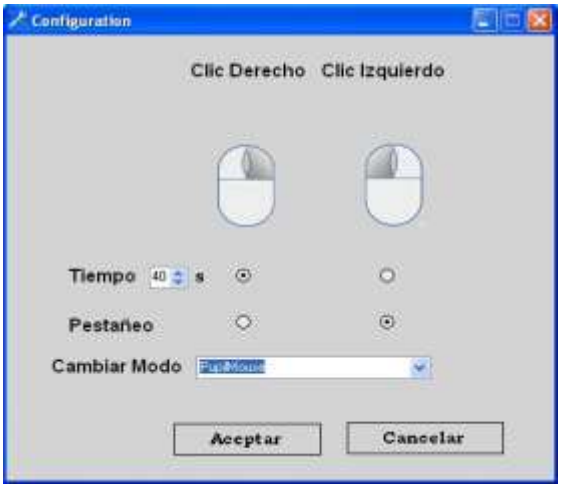


Figura 20: Interfaz de configuración

- Interfaz de calibración en esta interfaz el usuario puede entrenar la mirada siguiendo puntos ya predefinidos, para controlar el desplazamiento del cursor del ratón de la computadora con el movimiento de los ojos.

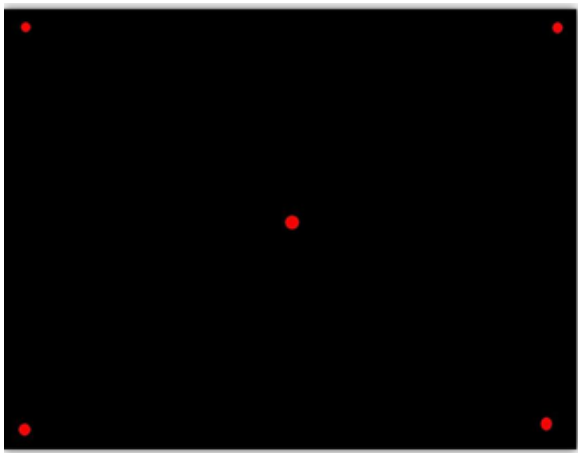


Figura 21: Interfaz de calibración

3. Estándares de codificación

Los estándares de codificación no son más que las pautas que le facilitan a los desarrolladores tener una estructura común en su lenguaje. Es la forma de escribir su código ordenadamente.

En el desarrollo del componente para controlar las funcionalidades del ratón de la computadora haciendo uso de una cámara digital se utilizaron varios estándares de codificación a la hora de escribir el código. A continuación se explican los estándares utilizados.

3.1 Convenciones de nomenclatura

- Pascal: el primer carácter de todas las palabras se escriben con mayúscula y los restantes con minúscula. Ejemplo: *MouseControl*.
- Camell: el primer carácter de todas las palabras, excepto de la primera palabra se escribe en mayúsculas y los otros caracteres en minúsculas. Ejemplo: *mouseControl*.[27]

La convención de nomenclatura usada para definir las clases y los métodos es la de Pascal.

Para la definición de las variables la nomenclatura usada es la Camell, ejemplo: *rightClic*, pero en la mayoría de los casos la variable cuenta con una sola palabra y queda así, ejemplo: *phases*.

En los comentarios el formato a usar será // nunca /****/, se comentarán tantas líneas de código como sean necesarias para su entendimiento.

3.2 Manejo de excepciones

El manejo de excepciones o tratamiento de errores es una parte fundamental de la implementación de cualquier componente. Una excepción es un evento adverso que ocurre durante la ejecución del componente y detiene su flujo normal de ejecución. Para su uso se tuvo en cuenta que: nunca se declarara un bloque *match* vacío, evitar anidar bloques *try/catch* en otro bloque *catch* y siempre utilizar validaciones para evitar excepciones.

Para evitar las molestias que ocasionan los errores se hacen un grupo de validaciones que permiten controlarlos, en el control de los errores se le informa al usuario mediante de un mensaje amigable de que lo está cometiendo y como puede darle solución.

4. Pruebas

Un producto puede ser probado por diferentes criterios, en este caso los usados para probar el Componente para controlar las funcionalidades del ratón de la computadora haciendo uso de una cámara digital, son los de: caja negra y caja blanca. Las pruebas no son usadas para determinar que el componente no tiene errores sino que no cuenta con defectos.

Las pruebas pueden ser llevadas a cabo por los casos de prueba y los procedimientos de prueba. Un caso de prueba no es más que un conjunto de entradas de prueba, condiciones de ejecución y resultados esperados, desarrollados para un objetivo concreto, tal como probar que se cumple un requisito específico o probar un camino concreto a través de un caso de uso. Un procedimiento es la especificación de cómo llevar a cabo la preparación, ejecución y evaluación de un caso de prueba.[28]

4.1 Pruebas de caja negra

Las pruebas de caja negra o pruebas funcionales son las que verifican que las funcionalidades del componente trabajen según lo esperado. En las pruebas de caja negra se validan que las entradas al componente sean correctas para obtener una salida válida. La prueba de caja negra intenta encontrar errores de las siguientes categorías: funciones incorrectas o ausentes, errores de interfaz y errores en estructuras de datos. La técnica usada en las pruebas de caja negra realizadas al componente fue la de particiones equivalentes, implementando como instrumento los casos de pruebas. Los caso de pruebas se le efectuarán a los casos de uso: CUS_Mover_cursor y la CUS_Configurar_acción_clic.

En el caso de uso CUS_Mover_cursor el usuario no tiene que realizar entrada de ningún valor para poder trabajar con el componente, pero si tiene que cumplir con determinadas restricciones y de no ser así el componente le alerta sobre su error:

Los posibles errores a cometer por el usuario al trabajar con el componente son:

- Al controlar el desplazamiento del puntero del cursor de la computadora con la nariz la parte detectada de la misma debe estar dentro de una región en específico, de no ser así, se le avisa al usuario con un mensaje en rojo, donde se le dice que está fuera de la región especificada. A continuación se muestra una imagen de cómo se vería lo antes descrito.

Figura 22: Error al desplazar el puntero con la nariz

- Al controlar el desplazamiento del puntero del cursor de la computadora con la nariz la misma puede no ser detectada por el componente, esto puede ser por el usuario no tener una postura correcta delante de la cámara digital. A lo que el componente responde avisándole al usuario, mediante un mensaje en rojo, que la nariz está pérdida. A continuación se muestra una imagen de cómo se vería lo antes descrito.

Figura 23: Error para detectar la nariz

- Al controlar el desplazamiento del puntero del cursor de la computadora con los ojos los mismos pueden no ser detectados por el componente, esto puede ser por el usuario no tener una adecuada postura delante de la cámara digital. El componente responde avisándole al usuario, mediante un mensaje en rojo, que los ojos no están siendo detectados. A continuación se muestra una imagen de cómo se vería lo antes descrito.

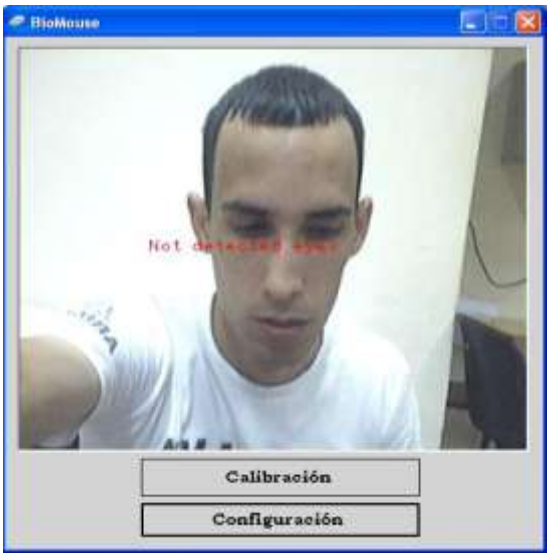


Figura 24: Error para detectar los ojos

En el caso de uso CUS_Configurar_acción_clic el usuario tiene que realizar un grupo de acciones para cambiar la configuración ya predefinida del componente, para realizar el clic. A continuación se recoge en una tabla el diseño del caso de prueba para este caso de uso.

Escenario	Descripción	Variable: Tiempo	Respuesta del sistema	Flujo Central
EC 1.1 Ejecutar el clic.	Ejecutar la acción del clic izquierdo o derecho por tiempo	Tiempo < 60 Válida	Ejecutar el clic en el tiempo definido por el usuario.	1. Seleccionar la opción de configuración. 2. Seleccionar que la ejecución del clic sea por tiempo 3. Seleccionar la cantidad de segundos de espera para ejecutar el clic.

Tabla 7: Descripción del caso de prueba del CUS_Configurar_acción_clic

4.2 Pruebas de caja blanca

Las pruebas de caja blanca o también llamadas estructurales realizan un seguimiento del código fuente según se va ejecutando los casos de prueba, de manera que se determinan de forma concreta las instrucciones y bloques en los que existen errores. Cuando se pasan casos de prueba al programa que se está probando, es conveniente conocer qué porcentaje del programa se ha ejecutado, de manera que se esté próximo a asegurar que todo él es correcto (evidentemente, es imposible alcanzar una certeza del 100%).[28]

Realizar las pruebas de caja blanca manualmente se torna un poco engorroso y lento, por lo que la realización de la misma al componente será de forma automática con el entorno de desarrollo *Visual Studio 2010*. Los resultados arrojados en el proceso de prueba se reflejaran en una tabla para su mejor entendimiento. En la tabla se recogen datos como: el estado de la prueba, descripción y resultado de la misma, además por quien fue ejecutada y controlada.

1.2.1 Pruebas Unitarias en Visual Studio

En programación, una **prueba unitaria** es una forma de probar el correcto funcionamiento de un módulo de código. Esto sirve para asegurar que cada uno de los módulos funcione correctamente por separado. Luego, con las pruebas de integración, se podrá asegurar el correcto funcionamiento del sistema o subsistema en cuestión.

La idea es escribir casos de prueba para cada función no trivial o método en el módulo de forma que cada caso sea independiente del resto[29]. En este caso se le realizaron pruebas a tres métodos, que son los más importantes en el componente, los resultados que arrojaron las pruebas se recogen a continuación.

```
public static float[,] HougeForCircle(float[,] m, float r1, float r2, float threshold)
{
    int w = m.GetLength(1), h = m.GetLength(0);
    float[,] r = new float[h, w];
    if (r1 > r2) { float t = r1; r1 = r2; r2 = t; }

    for (int i = 0; i < h; i++)
    {
        for (int j = 0; j < w; j++)
        {
            if (m[i, j] > threshold)
                AddCircle(r, r1, r2, j, i);
        }
    }
    return r;
}
```

Prueba Unitaria		
Nombre de la Prueba	*HougeForCircleTest*	
Estado	**Tipo**	**Ultima Ejecución**
Satisfactoria	Caja Blanca	05/05/2012
Ejecutado por	**Verificado por**	
Miladys Vázquez Fernández	Miguel Miralles Pantoja	

Descripción	Este método devuelve una matriz de puntos en los cuales se detectan los círculos dentro de la imagen.
Entrada	*Matriz de píxeles.*
Criterio de aceptación	Retorna una matriz con los puntos detectados y después se muestra en la imagen la localización de la pupila.

Resultado

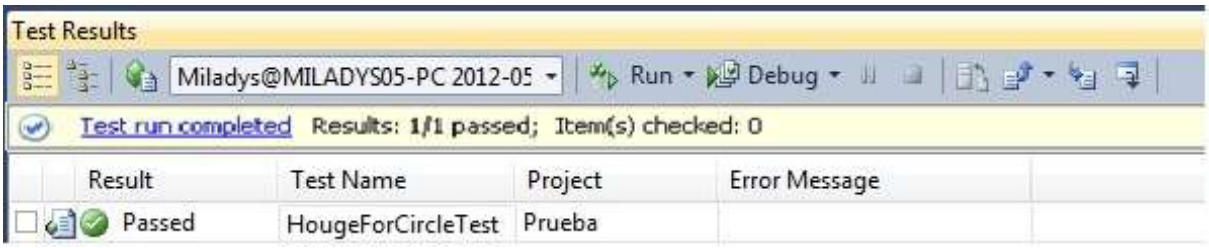


Tabla 8: Prueba unitaria HougeForCircleTest

```
public int isBlink()
{

    int blink = 0;
    if (grayPrevFrame != null)
    {
        subImage = grayframe.Sub(grayPrevFrame);
        subImage = subImage.ThresholdBinary(new Gray(20), new Gray(255));
        subImage.Erode(3);
        StructuringElementEx elemen1 = new StructuringElementEx(3, 3, 1, 1, Emgu.CV.CvEnum.CV_ELEMENT_SHAPE.CV_SHAPE_CROSS);
        subImage = subImage.MorphologyEx(elemen1, Emgu.CV.CvEnum.CV_MORPH_OP.CV_MOP_OPEN, 2);


        subImage.ROI = eyeroi1;
        Contour<Point> contourleft = subImage.FindContours();
        if ((contourleft != null && contourleft.HNext == null && contourleft.Area > (double)10))
        {
            found = true;
            blink++;
        }
        else
        {
            found = false;
            nextFrame.ROI = eyeroi1;
            subImage.ROI = eyeroi1;
            nextFrame.ROI = Rectangle.Empty;
            grayframe.ROI = Rectangle.Empty;
            grayPrevFrame.ROI = Rectangle.Empty;
            subImage.ROI = Rectangle.Empty;

        }
    }
```

Prueba Unitaria		
Nombre de la Prueba	*IsBlinkTest*	
Estado	**Tipo**	**Ultima Ejecución**
Satisfactoria	Caja Blanca	05/05/2012
Ejecutado por	**Verificado por**	
Miladys Vázquez Fernández	Miguel Miralles Pantoja	

Descripción	Para poder ejecutar la prueba se obtienen dos imágenes en escala de grises estas se restan y luego se detectan los cambiaos de intensidad de los píxeles en la imagen para saber si se pestañeo o no.
Entrada	*Image<Gris, byte>*
Criterio de aceptación	Devuelve 1 si el usuario pestañeo y cero si no lo hizo.

Resultado

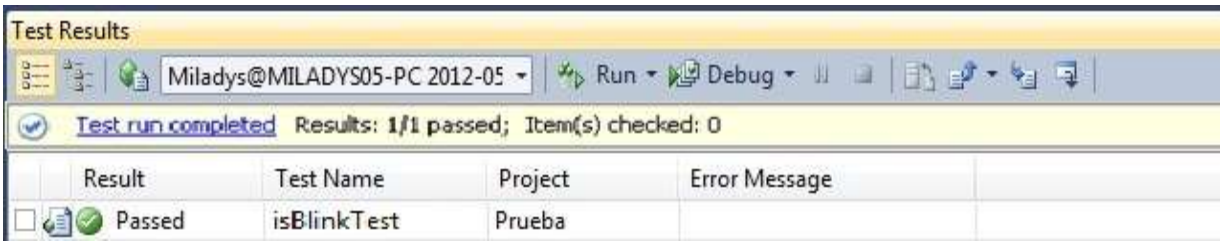


Tabla 9: Prueba unitaria IsBlinkTest

```
public void ProcessImageNose(Rectangle face, Image<Bgr, byte> frame)
{


    if (face.Width != 0)
    {

        Rectangle roinose = new Rectangle(face.Left + face.Width / 4, face.Top + face.Height / 2, face.Width / 2, face.Height / 2);
        //nextFrame.Draw(roinose, new Bgr(255, 244, 46), 2);
        Rectangle imageArea = grayframe.ROI;
        Rectangle mouseStableArea =
        new Rectangle((int)(imageArea.Width * 0.4), (int)(imageArea.Height * 0.4), (int)(imageArea.Width * 0.2), (int)(imageArea.Height * 0.2));
        aux3 = nextFrame.Copy(roinose);
        grayaux3 = aux3.Convert<Gray, byte>();
        //draw the stable area where the face will not trigger a movement;
        nextFrame.Draw(mouseStableArea, new Bgr(255, 0, 0), 1);

        MCvAvgComp[] noses = haarnose.Detect(grayaux3);
        int left = 0, rigth = 5, top = 0, bottom = 5;
        if (noses.Length > 0)
        {   //if there is at least one face


            MCvAvgComp biggestEyes = noses[0];
            for (int i = 1; i < noses.Length; i++)
            {
                if (noses[i].rect.Width * noses[i].rect.Height > biggestEyes.rect.Width * biggestEyes.rect.Height)
                    biggestEyes = noses[i];
            }
            noseroi = new Rectangle(roinose.Left + noses[0].rect.Left - left, roinose.Top + noses[0].rect.Top - top, noses[0].rect.Width + rigth +
            nosecenter = new Point(noseroi.X + noseroi.Width / 2, noseroi.Y + noseroi.Height / 2);
            //draw a yellow rectangle around the face
            //nextFrame.Draw(noseroi, new Bgr(255, 255, 0.0), 1);

            //draw a green cross at the center of the biggest face
            nextFrame.Draw(
                new Cross2DF(nosecenter, noseroi.Width * 0.1f, noseroi.Height * 0.1f),
                new Bgr(0, 255, 0), 1);
```

Prueba Unitaria		
Nombre de la Prueba	*ProcessImageNoseTest.*	
Estado	**Tipo**	**Ultima Ejecución**
Satisfactoria	Caja Blanca	05/05/2012
Ejecutado por	**Verificado por**	

Miladys Vázquez Fernández	Miguel Miralles Pantoja
Descripción	Se detecta el rostro dentro de la imagen y después se pasa a detectar la nariz para obtener su localización la cual será mostrada en la imagen.
Entrada	*Imagen<Gris, byte>*
Criterio de aceptación	Se muestra en la imagen la localización de la nariz con la cual se mueve el ratón.

Resultado

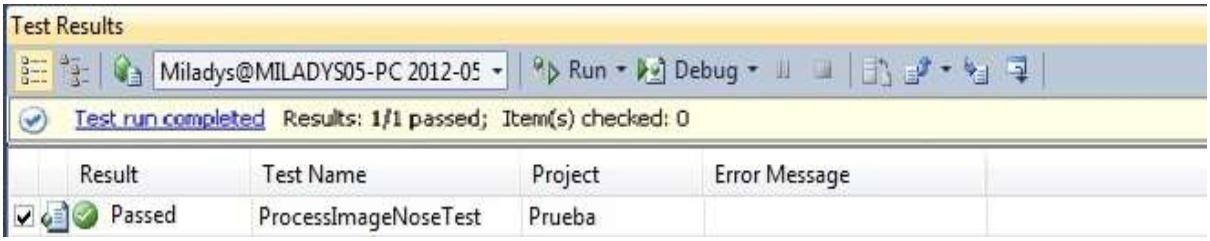


Tabla 10: Prueba unitaria ProcessImageNoseTest

4.3 Resultados de las pruebas

Al componente se le realizó una iteración de pruebas de caja negra y una de caja blanca. En la de caja negra de 4 requisitos funcionales implementados se elaboraron 2 casos de pruebas. En la iteración de prueba de caja negra hecha a los 2 casos de pruebas se encontraron 2 no conformidades, de las cuales se le dieron solución a todas en su totalidad. Además se realizaron las pruebas de carga y stress, demostrando que lo requisitos no funcionales relacionados con el rendimiento y usabilidad del componente fueron cumplidos en su totalidad. En la siguiente gráfica se muestran los valores de los resultados arrojados por las pruebas de caja negra.

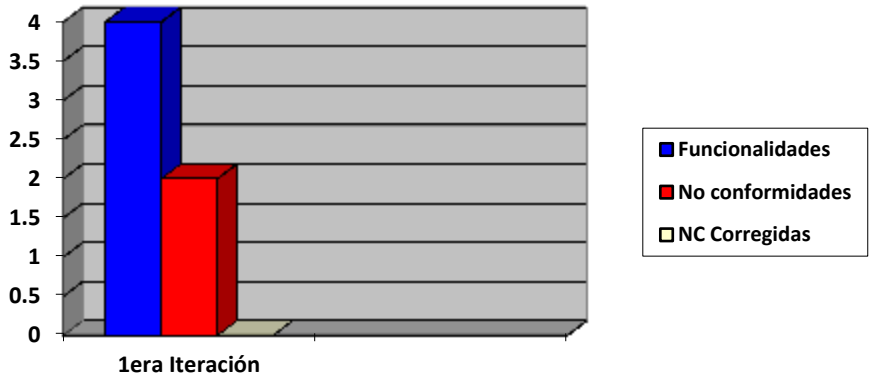


Tabla 11: Gráfica de iteraciones de prueba

5. Conclusiones parciales

Al terminar este capítulo el componente a desarrollar ha sido implementado satisfactoriamente. Los requisitos propuestos para el mismo han sido puestos en práctica y con el funcionamiento esperado. Los resultados que se obtuvieron en las pruebas realizadas al componente fueron buenos, demostrando que el objetivo esperado fue alcanzado. Concluyendo que el contar con este componente en el Centro de Identificación y Seguridad Digital permitirá el interactuar de las personas con los ordenadores de una manera diferente.

Conclusiones Generales

Esta investigación permitió arribar a algunas conclusiones esenciales, entre ellas está que la tendencia al controlar el desplazamiento del puntero del cursor del ratón de la computadora es mediante la cabeza o el rostro en general y que las acciones del clic generalmente son por tiempo o sonido. Partiendo de las conclusiones generales antes planteadas se definieron específicas, las cuales se desglosan a continuación:

- Los componentes que permiten controlar las funcionalidades del ratón de la computadora tienen gran utilidad en la sociedad, por ser una manera diferente para las personas de interactuar con los ordenadores y facilitarle el uso de las tecnologías a las personas con discapacidades físicas.

- Los componentes más factibles según lo investigado son los que permiten controlar las funcionalidades del ratón apoyados en otros dispositivos que no sea solo una cámara digital, estos dispositivos de *hardware* pueden ser sensores ópticos.

- El componente cuenta con una nueva manera de controlar las funcionalidades del ratón de la computadora, haciendo más fácil su uso para las personas que deseen trabajar con el mismo.

Con todo los antes expuesto se concluye que el objetivo general de la investigación se logró, ya que el componente facilita controlar las funcionalidades del ratón de la computadora de maneras que anteriormente no habían sido expuestas. Permite controlar el desplazamiento del puntero del cursor de dos maneras diferentes, permitiendo contar con una mejor estabilidad el puntero en su movimiento y la acción del clic puede ser configurada para ser ejecuta por tiempo o por el pestañeo según el usuario lo desee.

Recomendaciones

Se recomienda seguir con la investigación y desarrollo del presente trabajo para así mejorar y agregar nuevas funcionalidades dentro de las cuales se citan:

- Lograr poder apuntar con el cursor de la computadora sobre un objetivo específico con la mirada.
- Mejorar o implementar un algoritmo para la detección de la pupila más robusto y veloz.
- Crear un clasificador que sea capaz de detectar la pupila ya que así la velocidad de detección de la pupila dentro de una imagen mejoraría grandemente.
- Agregar la opción de poder arrastrar un objeto con el cursor utilizando la mirada o la nariz y ejecutar la acción de doble clic utilizando otra parte del rostro o con algún sonido.
- Utilización de la nueva versión que se está implementando de *Emgu Cv* (2.4) la cual contará con la librería *Blob* que se puede utilizar para detectar la pupila debido a que encuentra manchas dentro de una imagen.

Referencias Bibliográficas

1. **S.A, F.I**. *Que es la Biometría?* [cited; Available from: http://www.biometricos.cl/equipos_biometria/que_es_la_biometria_por_huella_digital.php.
2. **eshistoria.net**. *http://www.eshistoria.net*. [cited; Available from: http://www.eshistoria.net/2009/10/historia-del-video.html.
3. **EcuRed**. *Imagen*. [cited; Available from: http://www.ecured.cu/index.php/Imagen.
4. **Kioskea.net**. *Introducción al video digital*. [cited; Available from: http://es.kioskea.net/contents/video/video.php3.
5. **Marcos Guglielmetti**, et al. *Definición de Camara Digital*. [cited; Available from: http://www.mastermagazine.info/termino/4156.php.
6. **Iborra, M.J.A.**, *Análisis comparativo de métodos basados en subespacios aplicados al reconocimiento de caras*, Universidad de Valencia. p. 68.
7. **Gary Bradski and A. Kaehler**, *Learning OpenCV*. 576.
8. **Lleida, U.d.**, *La Universitat de Lleida desarrolla un mouse virtual para personas con discapacidad.* 2008.
9. **Gips, P.J.**, *Camera Mouse 2011 User Manual*. 2011, Boston. 25.
10. **Joaquim Fonoll and C. Mauri**, *AYUDAS TÉCNICAS DE ACCESO AL ORDENADOR BASADAS EN CÁMARAS WEB*.
11. **Lizama, M.A.**, *Evaluación de Programa: Enable Viacam (eViacam)*. 6.
12. **Ribes, M.T.**, *Aplicación de sensores de flujo óptico para el desarrollo de nuevos sistemas de medida de bajo coste*, Universidad de Lleida. p. 111.
13. **Emmanuel Ospina Piedrahíta and J.P.U. Duque**, *Implementacíon de la Transformada de Hough para la deteccíon de líneas para un sistema de visíon de bajo nivel.*, Universidad nacional de Colombia: Colombia. p. 40.
14. **Urbano, M.G.**, *Reconocimiento de iris*, Universidad de Barcelona. p. 50.
15. **Betke, M.C.a.M.**, *Real Time Eye Tracking and Blink Detection with USB Cameras.* p. 10.
16. **Alejandro Martínez and R. Martínez**, *Guía a Rational Unified Process*.
17. **Roberth G. Figueroa, Camilo J. Solís, and A.A. Cabrera**, *METODOLOGÍAS TRADICIONALES VS. METODOLOGÍAS ÁGILES*.
18. **Cuervo;, M.C. and O.Y.B.** Moreno *Herramientas libres para modelar software*. **Volume**,
19. **Rumbaugh;, J., Ivar Jacobson, and G. Booch**, *El lenguaje unificado de modelado.Manual de referencia*. 528.
20. **intercambiosvirtuales.org**. *Microsoft Visual Studio 2010 Ultimate*. 2010 [cited; Available from: http://www.intercambiosvirtuales.org/software/microsoft-visual-studio-2010-ultimate-espanol.
21. **Raúl Igual, C.M.**, *Tutorial de OpenCV*. 2008.
22. **Sergio De La Llana Alamar, Samuel Molina Vinci, and S.S. Martinez**. *Introducción a la librería OpenCV*. [cited; Available from: http://web-sisop.disca.upv.es/imd/cursosAnteriors/2k3-2k4/copiaTreballs/serdelal/trabajoIMD.xml.
23. **Ivar Jacobson, Grady Booch, and J. Rumbaugh**, *El proceso unificado de desarrollo de software*. 458.
24. **Marcello Visconti and H. Astudillo**, *Fundamentos de Ingeniería de Software.* p. 24.
25. **S.A., S.S.A.-S**. *Modelo Físico*. [cited; Available from: http://www.sparxsystems.com.ar/resources/tutorial/physical_models.html.
26. **Nina., D.M.**, et al., *Artefacto: Diagrama de componentes*. 2009, UNIVERSIDAD SALESIANA DE BOLIVIA: Bolivia. p. 13.

27. **Josep Ribal, Alberto Fernández**, and R. Mendieta, *Estándares de Codificación en C# y Buenas Prácticas de Programación.* p. 49.
28. **Usaola, D.M.P.**, *Mantenimiento Avanzado de Sistemas de Información Pruebas del Software*. p. 38.
29. **Aprendizage(SENA), S.N.d**. *Tipos de Prueba*. [cited; Available from: http://ciclodevidasoftware.wikispaces.com/Tipos+de+Pruebas.

Bibliografía Consultada

1. **Claudia Lora Pomar** and Y.D. Mesa, *Algoritmo de detección facial para sistemas de autenticación biométrica*. 2010, Universidad de las Ciencias Informáticas (UCI): La Habana. p. 100.
2. **Gary Bradski** and A. Kaehler, *Learning OpenCV*. 576.
3. **David Mora, Andrés Páez**, and J.Q. Sepúlveda, *Detección de Objetos Móviles en una Escena Utilizando Flujo Óptico*. p. 5.
4. **Machine Vision Group**, I.O., *An Iterative Algorithm for Fast Iris Detection*, University of Oulu: Finland. p.8.
5. **Ayudhya**, C.D.N. and M.W. School, *A Method for Real-Time Eye Blink Detection and Its Application*, Kasetsart University Jatujak, Bangkok 10900, Thailand. p. 6.
6. **Yoshinobu Ebisawa**, et al., *PupilMouse: Cursor Control by Head Rotation Using Pupil Detection Technique*, Shizuoka University: Hamamatsu, 432-8561, Japan. p. 6.
7. **Michael Chau and M. Betke**, *Real Time Eye Tracking and Blink Detection with USB Cameras*, Boston University: Boston.
8. **Julian Quiroga and C.A.R. Romero**, *Algoritmo Robusto para la Detección Automática del Parpadeo Humano y Seguimiento del Rostro Mediante Patrones Espacio-Temporales con Aplicación al Control sin Contacto*, Universidad de los Andes. p. 5.
9. **S.A, F.I**. *Que es la Biometría?* [cited; Available from: http://www.biometricos.cl/equipos_biometria/que_es_la_biometria_por_huella_digital.php.
10. **eshistoria.net**. *http://www.eshistoria.net*. [cited; Available from: http://www.eshistoria.net/2009/10/historia-del-video.html.
11. **EcuRed**. *Imagen*. [cited; Available from: http://www.ecured.cu/index.php/Imagen.
12. **Kioskea.net**. *Introducción al video digital*. [cited; Available from: http://es.kioskea.net/contents/video/video.php3.
13. **Marcos Guglielmetti**, et al. *Definición de Camara Digital*. [cited; Available from: http://www.mastermagazine.info/termino/4156.php.
14. **Iborra, M.J.A.**, *Análisis comparativo de métodos basados en subespacios aplicados al reconocimiento de caras*, Universidad de Valencia. p. 68.
15. **Gary Bradski and A. Kaehler**, *Learning OpenCV*. 576.
16. **Lleida, U.d.**, *La Universitat de Lleida desarrolla un mouse virtual para personas con discapacidad.* 2008.
17. **Gips, P.J.**, *Camera Mouse 2011 User Manual*. 2011, Boston. 25.
18. **Joaquim Fonoll and C. Mauri**, *AYUDAS TÉCNICAS DE ACCESO AL ORDENADOR BASADAS EN CÁMARAS WEB*.
19. **Lizama, M.A.**, *Evaluación de Programa: Enable Viacam (eViacam)*. 6.

20. **Ribes, M.T.**, *Aplicación de sensores de flujo óptico para el desarrollo de nuevos sistemas de medida de bajo coste*, Universidad de Lleida. p. 111.
21. **Emmanuel Ospina Piedrahíta and J.P.U. Duque**, *Implementacíon de la Transformada de Hough para la deteccíon de líneas para un sistema de visíon de bajo nivel.*, Universidad nacional de Colombia: Colombia. p. 40.
22. **Urbano, M.G.**, *Reconocimiento de iris*, Universidad de Barcelona. p. 50.
23. **Betke, M.C.a.M.**, *Real Time Eye Tracking and Blink Detection with USB Cameras.* p. 10.
24. **Alejandro Martínez and R. Martínez**, *Guía a Rational Unified Process*.
25. **Roberth G. Figueroa, Camilo J. Solís, and A.A. Cabrera**, *METODOLOGÍAS TRADICIONALES VS. METODOLOGÍAS ÁGILES*.
26. **Cuervo;, M.C. and O.Y.B.** Moreno *Herramientas libres para modelar software*. **Volume**,
27. **Rumbaugh;, J., Ivar Jacobson, and G. Booch**, *El lenguaje unificado de modelado.Manual de referencia*. 528.
28. **intercambiosvirtuales.org**. *Microsoft Visual Studio 2010 Ultimate*. 2010 [cited; Available from: http://www.intercambiosvirtuales.org/software/microsoft-visual-studio-2010-ultimate-espanol.
29. **Raúl Igual, C.M.**, *Tutorial de OpenCV*. 2008.
30. **Sergio De La Llana Alamar, Samuel Molina Vinci, and S.S. Martinez**. *Introducción a la librería OpenCV*. [cited; Available from: http://web-sisop.disca.upv.es/imd/cursosAnteriors/2k3-2k4/copiaTreballs/serdelal/trabajoIMD.xml.
31. **Ivar Jacobson, Grady Booch, and J. Rumbaugh**, *El proceso unificado de desarrollo de software*. 458.
32. **Marcello Visconti and H. Astudillo**, *Fundamentos de Ingeniería de Software.* p. 24.
33. **S.A., S.S.A.-S**. *Modelo Físico*. [cited; Available from: http://www.sparxsystems.com.ar/resources/tutorial/physical_models.html.
34. **Nina., D.M.**, et al., *Artefacto: Diagrama de componentes*. 2009, UNIVERSIDAD SALESIANA DE BOLIVIA: Bolivia. p. 13.
35. **Josep Ribal, Alberto Fernández**, and R. Mendieta, *Estándares de Codificación en C# y Buenas Prácticas de Programación.* p. 49.
36. **Usaola, D.M.P.**, *Mantenimiento Avanzado de Sistemas de Información Pruebas del Software*. p. 38.
37. **Aprendizage(SENA), S.N.d**. *Tipos de Prueba*. [cited; Available from: http://ciclodevidasoftware.wikispaces.com/Tipos+de+Pruebas.

Anexos

Anexo 1

Nombre del caso de uso: CUS_Mover_cursor	
Actores	Usuario
Propósito	Controlar el desplazamiento del puntero del cursor del ratón de la computadora con la parte del rostro seleccionada por el usuario.
Resumen	El caso de uso inicia cuando el usuario inicia el componente.
Referencia	RF-3.
Precondiciones	
Flujo normal de eventos	
Acción del actor	**Respuesta del sistema**
1. El usuario inicia el componente.	
	2. El sistema captura la imagen del usuario mediante la cámara digital.
	3. El sistema usa las librerías necesarias para mostrar en la interfaz principal del componente la imagen del usuario.
	4. El sistema usa los clasificadores requerido para detectar el rostro.
	5. El sistema le permite al usuario configurar con que parte del rostro desea controlar el desplazamiento del puntero del cursor de la computadora. Esa configuración se realiza en la

	ventana de configuración y las opciones que se tiene para controlar el desplazamiento del puntero son: - La nariz. - Los ojos.
6. El usuario selecciona la opción con que desea controlar el desplazamiento del puntero del cursor.	
7. El usuario escoge la opción de los ojos para controlar el desplazamiento del puntero del cursor.	
	8. El sistema usa el algoritmo requerido para detectar la pupila.
	9. El sistema señala con un círculo la localización de la pupila detectada.
10. El usuario mueve los ojos para comprobar que se está moviendo correctamente el cursor del ratón de la computadora.	
11. El usuario escoge la opción de la nariz para controlar el desplazamiento del puntero del cursor.	
	12. El sistema detecta la nariz usando los clasificadores.
	13. El sistema señala la nariz con un punto de color.
14. El usuario mueve el rostro para comprobar que se está moviendo el puntero del cursor	

del ratón de la computadora según el punto de referencia.	
Flujo Alterno	
Pos condiciones	

Descripción del CUS_Mover_cursor

Anexo 2

Nombre del caso de uso: CUS_Calibrar_mirada	
Actores	Usuario
Propósito	Entrenar la mirada del usuario.
Resumen	El caso de uso inicia cuando el usuario selecciona la opción de calibración, para entrenar la miranda.
Referencia	RF-4.
Precondiciones	
Flujo normal de eventos	
Acción del actor	**Respuesta del sistema**
1. El usuario inicia el componente.	
2. El usuario selecciona la opción de calibración.	
3. El usuario comienza a entrenar la mirada, lo hace siguiendo los puntos señalados en la interfaz.	
	4. El sistema calcula la distancia entre la pupila y un punto de referencia que tiene ya señalado en el rostro del usuario.

Flujo Alterno	
Pos condiciones	

Descripción del CUS_Calibrar_mirada

Anexo 3

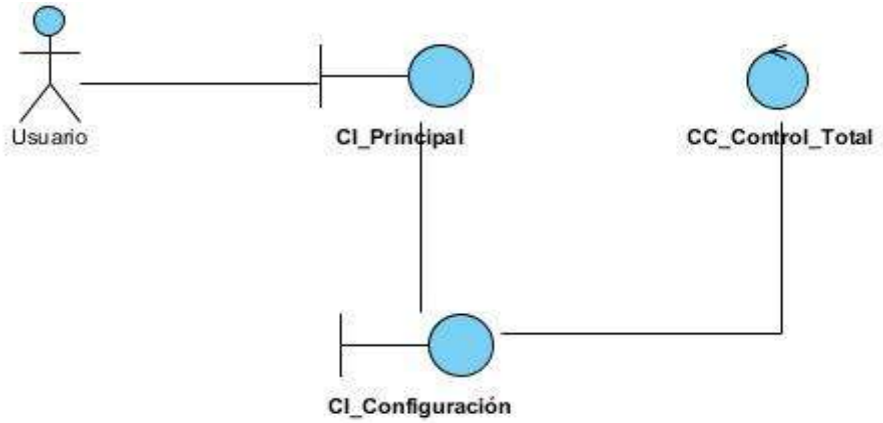


Diagrama de clases del análisis del CUS_Mover_cursor

Anexo 4

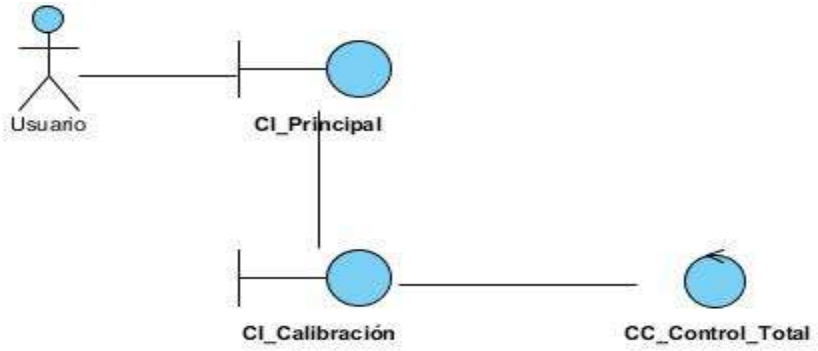


Diagrama de clases del análisis del CUS_Calibración_mirada

Anexo 5

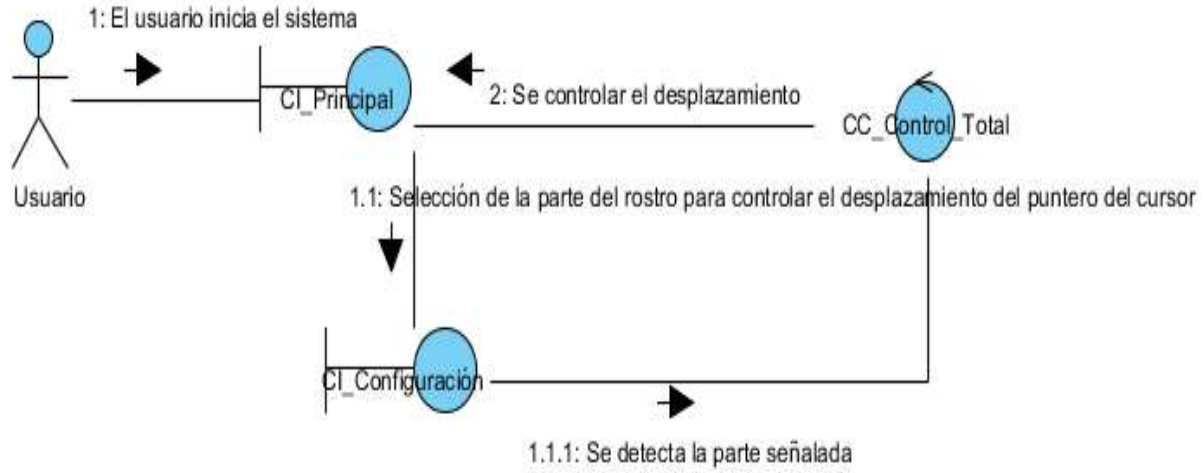


Diagrama de colaboración del CUS_Mover_cursor

Anexo 6

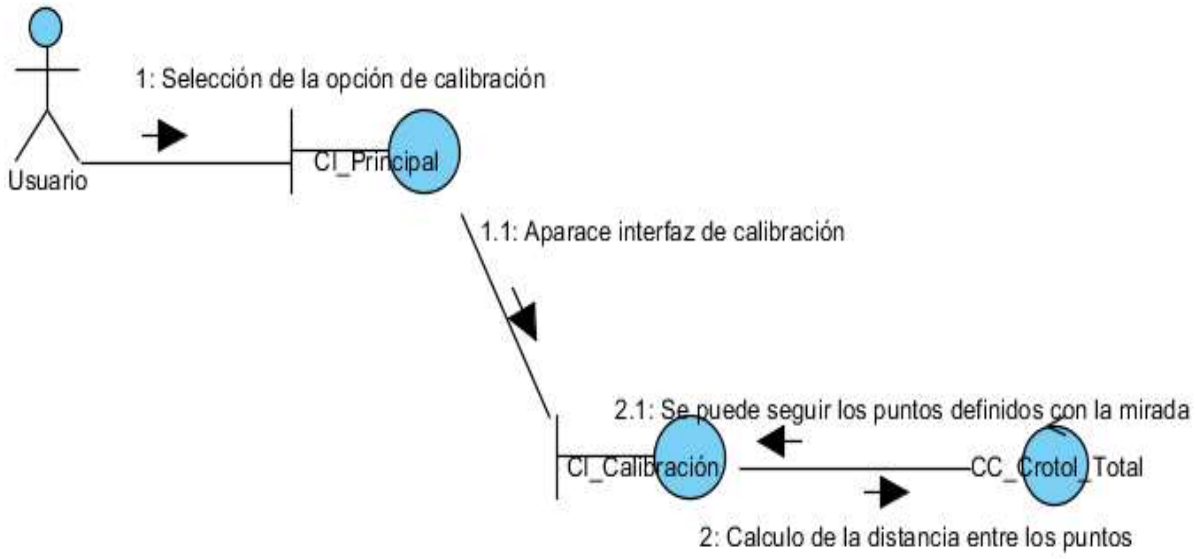


Diagrama de Colaboración del CUS_Calibración_mirada

Anexo 7

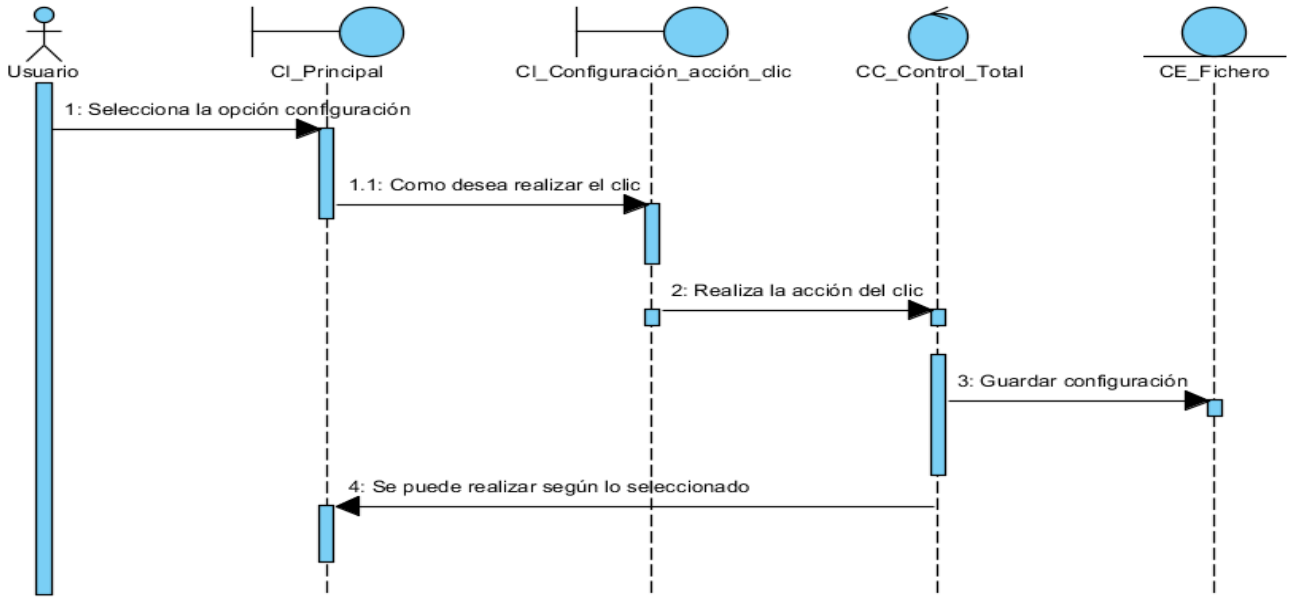


Diagrama de secuencia del CUS_Mover_cursor

Anexo 8

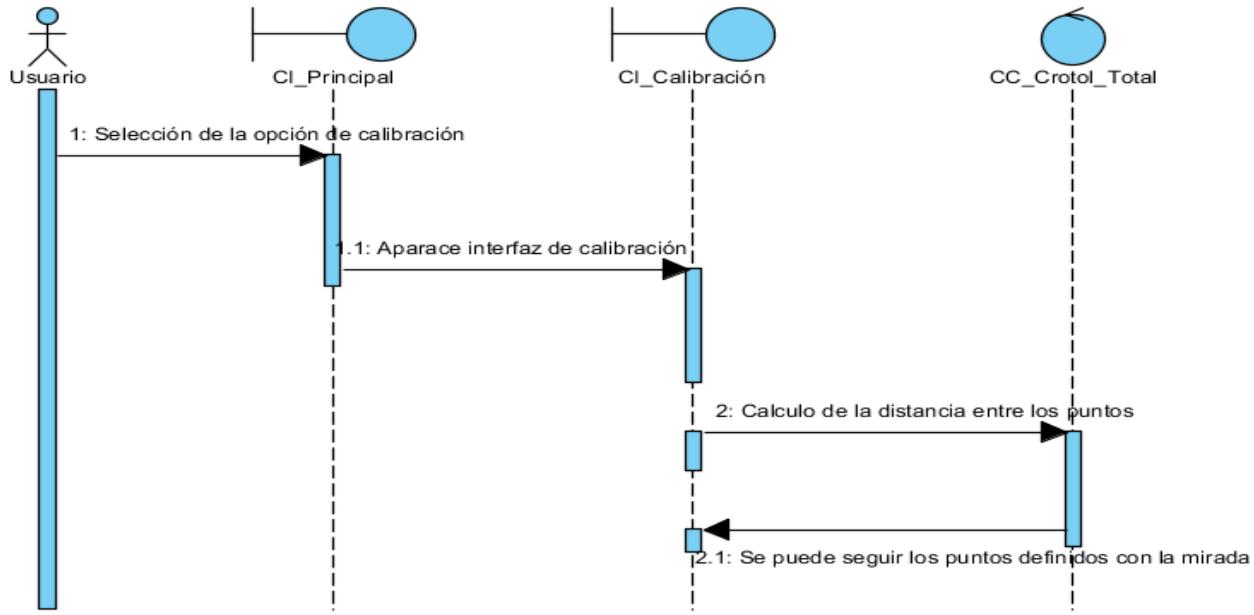


Diagrama de secuencia del CUS_Calibrar_mirada