FACULTAD 5

Trabajo de Diploma para Optar por el Título de
Ingeniero en Ciencias Informáticas

Desarrollo de algoritmo de comparación para biblioteca de autenticación biométrica a partir de reconocimiento de patrones del iris

Autor:

Michel Evaristo Febles Parker

Tutores:

Ing. Alejandro González Abascal.

Ing. Ariel Peláez González

Ciudad de La Habana, Junio de 2012

"Año 54 de la Revolución"

Declaración de Autoría

Declaramos ser autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo. Para que así conste firmamos la presente a los ____ días del mes de junio del año 2012.

Michel Evaristo Febles Parker

Firma del Autor

Ing. Alejandro González Abascal

Firma del Tutor

Ing. Ariel Peláez González

Firma del Tutor

Datos de Contacto

Nombre: Ing. Alejandro González Abascal.
Edad: 25 años.
Ciudadanía: cubano.
Institución: Universidad de Ciencias Informáticas (UCI).
Título: Ingeniero en Ciencias Informáticas.
Categoría Docente:
E-mail: agabascal@uci.cu

Graduado con el título de Ingeniero en Ciencias Informáticas en el 2010 en la Universidad de las Ciencias Informáticas.

Nombre: Ing. Ariel Peláez González.
Edad: 24 años.
Ciudadanía: cubano.
Institución: Universidad de Ciencias Informáticas (UCI).
Título: Ingeniero en Ciencias Informáticas.
Categoría Docente:
E-mail: agabascal@uci.cu

Graduado con el título de Ingeniero en Ciencias Informáticas en el 2011 en la Universidad de las Ciencias Informáticas.

"Entre los grandes placeres que nos brinda la vida, está el superar las dificultades paso a paso, escalando uno a uno los peldaños del éxito, concibiendo nuevos deseos y viéndolos realizados."

Samuel Johnson.

Agradecimientos

Quiero agradecer a mis padres y en especial a mi madre por el apoyo brindado en todo momento. A mi tía por tanto amor y consejo. A todas las personas que de una forma u otra han hecho posible el desarrollo de este trabajo. A mis tutores por la confianza depositada y a Kirenia por todo su apoyo.

Dedicatoria

A mi tía y a mi madre.

Resumen

Los sistemas de seguridad de los diferentes complejos industriales en el mundo actual, se encuentran en un desarrollo creciente producto al gran número de ataques a los que están sometidos de manera constante los sistemas informáticos modernos. Existen varias formas de fortalecer estos sistemas y una de las más eficientes es contar con diversas vías de autenticación, entre las que se destacan las que se llevan a cabo a partir de parámetros biométricos. Teniendo en cuenta lo anterior surge la necesidad de incrementar estas técnicas en nuestro país para evitar posibles vulnerabilidades. Para dar solución a esta cuestión se realiza el siguiente trabajo que tiene como objetivo desarrollar una versión inicial de una biblioteca de reconocimiento de patrones del iris en el Centro de Informática Industrial (CEDIN), de la Universidad de Ciencias Informáticas (UCI), sobre plataforma libre capaz de brindarles confianza a los usuarios durante su autenticación al sistema.

Se realizó una investigación de las principales tecnologías de desarrollo y bibliotecas existentes en el mundo y se hizo una propuesta de las más importantes, todo esto para cumplir el objetivo. Además, se muestra la modelación del sistema a partir de diferentes diagramas y se analiza la implementación del mismo. Por último, se muestran las diferentes pruebas que se le realizaron a la biblioteca para determinar su efectividad.

Palabras clave: Autenticación, biometría, comparación, seguridad, sistema, biblioteca.

Índice

Índice de Tablas

Índice de Figuras

Introducción

La seguridad ha constituido un papel fundamental en la historia de la humanidad. Desde sus inicios, esta ha representado el acto de aplicar ciertas medidas tanto escritas como prácticas para impedir el libre acceso de cualquier persona hacia un recurso determinado (material o intelectual) y de tal forma, poseer total control sobre el mismo.

Una de las vías fundamentales de mantener la seguridad actualmente es el control de acceso lógico sobre los diferentes recursos, constituyendo este las diferentes técnicas que se pueden aplicar para llevar un control sobre el personal que accede hacia cierto medio.

La autenticación constituye un elemento fundamental a la hora de identificar a un individuo, existiendo varios criterios y definiciones referentes a la misma, como los que se referencian a continuación:

- ✓ Acción por parte de un usuario de presentar su identidad a un sistema, usualmente se usa un identificador de usuario; estableciendo además que dicho usuario es responsable de las acciones que lleve a cabo en el sistema. (1)
- ✓ Acción de la verificación de la identidad de un usuario, generalmente cuando se ingresa a un sistema, red de computadoras o se accede a una base de datos. (2)

En el mundo existen varias técnicas que permiten realizar la verificación de la autenticidad de la identidad de los usuarios, las cuales se clasifican de la manera siguiente:

- ➢ Algo que solamente el individuo conoce: por ejemplo una contraseña.
- ➢ Algo que la persona posee: por ejemplo una tarjeta magnética.
- ➢ Algo que el individuo es y que lo identifica unívocamente: por ejemplo los sistemas biométricos, los cuales incluyen reconocimiento de huellas digitales, geometría de la mano, iris, retina, reconocimiento facial y voz.
- ➢ Algo que solo el individuo es capaz de hacer: por ejemplo los patrones de escritura. (1)

Los sistemas biométricos constituyen la técnica de autenticación de usuario más eficiente utilizada en la actualidad, empleando para la autenticación patrones que solo el individuo posee o es, como lo constituyen la geometría facial y de la mano, la voz, el iris y la retina. Los mismos surgen motivados por la idea de proporcionar una identificación segura de las personas que intentan acceder a un determinado recurso. Luego se eliminan las falsas identificaciones que pueden existir empleando otras técnicas como

las basadas en tarjetas, contraseñas, código numéricos o la combinación de cualquiera de estas, que sufren la posibilidad de pérdida o sustracción de los elementos de identificación; en cambio los patrones no pueden ser sustraídos ni utilizados por otro individuo.

El reconocimiento del iris constituye una de las técnicas más confiables en el proceso de reconocimiento de personas. Un sistema de reconocimiento del iris consta de varias fases de procesado de los datos aportados por los patrones del iris, como lo constituyen la captura, la segmentación, la normalización, la codificación y la comparación, siendo esta última el proceso donde se comparan el código obtenido de una imagen de iris humano con un código almacenado en una base de datos anteriormente a fin de determinar la autenticidad de la persona a partir de la mayor existencia de elementos comunes en dichos códigos.

En Cuba, desde el año 1959 con el triunfo de la Revolución, el país se ha dado a la tarea de informatizar los sectores de la sociedad, siendo este un proceso ascendente e ininterrumpido. Vinculado a esto se crean diferentes proyectos encargados de realizar todo este proceso; entre los cuales se encuentra la Universidad de las Ciencias Informáticas (UCI), la cual cuenta con varios centros de desarrollo de software. El centro encargado de realizar aplicaciones para el sector industrial es el Centro de Informática Industrial (CEDIN), el cual debe garantizar que sus productos cuenten con los sistemas de seguridad necesarios para proteger los datos.

Entre los principales productos que desarrolla este centro están los SCADAs, que son sistemas de control y adquisición de datos para gestionar ambientes industriales. Los SCADAs que desarrolla el CEDIN son el SCADA Guardián del alba (GALBA), el SCADA UX y el SCADA Sailux. Estos sistemas cuentan con un módulo de seguridad encargado de garantizar la confidencialidad, integridad y disponibilidad de los datos con los que estos operan. Actualmente este módulo cuenta con un sistema de vigilancia por cámara y con un sistema de control de acceso basado en roles con dos técnicas de autenticación o verificación de la identidad de los usuarios:

1. Por reconocimiento de usuario y contraseña

2. Por reconocimiento de huellas digitales

Estos métodos de autenticación presentan un cierto número de vulnerabilidades que son muy conocidas, las cuales se mencionan a continuación.

Por reconocimiento de usuario y contraseña:

- ✓ Debilidad en las contraseñas.

- ✓ Posible ruptura de contraseña por avanzados algoritmos de descifrado.
- ✓ Permanencia de la contraseña en los logs del ordenador.
- ✓ La existencia de programas como los registradores de teclas (keyloggers).

Por reconocimiento de huellas digitales:

- ✓ Heridas y mutilaciones anatómicas.
- ✓ Suciedad y mal estado de la piel.

Situación problémica:

Los métodos de autenticación mencionados anteriormente, que se encuentran entre los más utilizados a nivel mundial, a causa de sus vulnerabilidades, debilitan el nivel de seguridad de los SCADAs que se desarrollan en el CEDIN, haciéndolos propensos a posibles ataques informáticos. Por lo tanto, las actuales técnicas de autenticación son insuficientes para garantizar la seguridad en dichos SCADAs.

Por lo anteriormente descrito se plantea el siguiente **problema a resolver**: ¿Cómo aumentar la seguridad en los SCADAs que se desarrollan en el CEDIN?

Por lo que se define como **objeto de estudio:** el reconocimiento del iris como técnica biométrica de autenticación de usuario.

Y para darle solución al problema se concibió como **objetivo general:** desarrollar una versión inicial de una biblioteca de autenticación de usuarios por reconocimiento de patrones del iris.

Todo lo anterior precisa como el **campo de acción:** los algoritmos matemáticos para el reconocimiento del iris.

Para dar cumplimiento al objetivo planteado se derivan las siguientes **tareas investigativas**:

- ➢ Realización de un análisis valorativo del estado del arte de las investigaciones realizadas sobre los sistemas de autenticación biométrica existentes, para la elaboración del marco teórico de la investigación.
- ➢ Descripción de las herramientas y tecnologías propuestas para el desarrollo de la biblioteca de autenticación biométrica, para seleccionar las herramientas a emplear.
- ➢ Obtención de casos de uso y realización del diseño correspondiente.

- Implementación del algoritmo de comparación de códigos de iris partiendo de los elementos del diseño.
- Empaquetado y pruebas de la biblioteca de autenticación biométrica.
- Comparación de resultados obtenidos con resultados esperados.

Métodos Teóricos:

- ✓ **Análisis Histórico-Lógico:** El presente método permitirá dentro de la investigación constatar la evolución y desarrollo de los sistemas de autenticación biométrica mediante el análisis de patrones oculares, específicamente mediante reconocimiento de iris, a través de los años.
- ✓ **Analítico-Sintético:** Se aplica con el objetivo de analizar y determinar la información más importante referente a los métodos de autenticación biométrica a partir del reconocimiento de patrones del iris

Métodos Empíricos:

- ✓ **Consulta de fuentes de información:** consulta de todo material bibliográfico para conformar el marco teórico.
- ✓ **Consulta de especialistas:** para crear las habilidades necesarias a aplicar en las diferentes etapas de la investigación.
- ✓ **Realización de pruebas:** para verificar la validez y confiabilidad del resultado.

Método estadístico:

- ✓ **Mediante la estadística descriptiva:** con el empleo de tablas, gráficas para valorar los resultados obtenidos con los resultados esperados a fin de determinar la validez y confiabilidad del resultado.

Resultados esperados al concluir el trabajo de Diploma:

Una versión inicial de una biblioteca de reconocimiento de patrones del iris.

El documento está estructurado por tres capítulos:

CAPÍTULO 1. Fundamentación Teórica.

Contiene los aspectos esenciales referentes a los sistemas de autenticación biométrica por reconocimiento de iris, sus fases, características y se describe el estado del conocimiento en este campo.

Se explican las metodologías, tecnologías y herramientas que se van a utilizar para el desarrollo de la aplicación.

CAPÍTULO 2. **Fundamentación de la propuesta de solución.**

Se describe el desarrollo de la solución propuesta a partir de la modelación de diagramas de casos de uso, diagramas de colaboración, diagramas de secuencia, diagramas de clases y diagrama de despliegue. Se plantea el modelo de datos y se especificarán los diferentes patrones utilizados para enriquecer la arquitectura del sistema. Se describe de forma general el diseño estructural de la biblioteca de reconocimiento de iris.

CAPÍTULO 3. **Desarrollo y pruebas.**

Se describen detalles de la implementación del algoritmo de comparación y los resultados de los procesos de pruebas y validación aplicados.

Capítulo 1. Fundamentación Teórica

1. Introducción del capítulo

En el presente capítulo se expondrán las principales fases con las que cuenta un sistema de autenticación por reconocimiento de iris y se hará un estudio del estado del conocimiento del reconocimiento del iris como técnica biométrica. Además, se analizará el método de comparación de códigos de iris utilizado en el sistema desarrollado junto con las tecnologías y herramientas empleadas en dicho sistema.

1.1. Concepto de biometría

Se define la biometría como la ciencia por la que se puede identificar a una persona basándose en sus características biofísicas o de comportamiento, es decir, algo que el ser humano posee de manera intrínseca. El término biometría comprende un amplio espectro de tecnologías mediante el uso de las cuales se permite verificar la identidad de una persona a partir del análisis de la medida de estas características, confiando en atributos propios de cada individuo en lugar de cosas que conocen o poseen. (3)

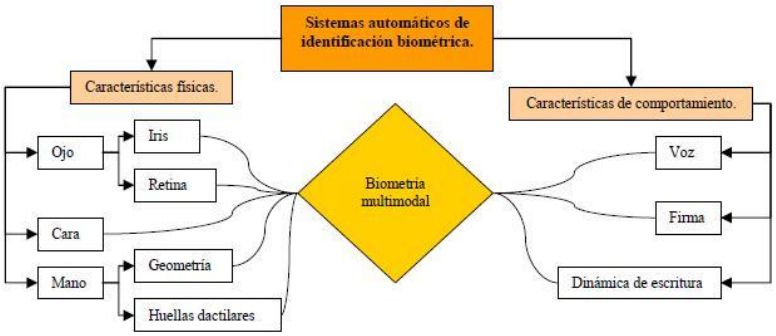

Imagen 1. Sistema automático de identificación biométrica

1.2. Sistemas biométricos

En el uso de tecnología biométrica para la identificación personal, los sistemas llevan a cabo medidas de las características tanto físicas como psicológicas, procediendo de manera directa sobre la parte del cuerpo usada en la persona en cuestión en los casos de medidas físicas, en lo que se denomina biometría estática o fisiológica, y a partir de la obtención de información derivada de diversas acciones comunes en cada persona en los casos de identificación mediante características psicológicas o de comportamiento

que determinan la biometría de tipo dinámico. Las características físicas de un usuario son relativamente estables, sin embargo, una característica relativa al comportamiento puede verse alterada por diversos motivos psicológicos o de situación, lo que implica que en muchos casos sea necesario que los sistemas basados en estas características permitan la actualización de las plantillas de los usuarios.

1.3. Definiciones

El elemento de mayor importancia para el desarrollo de la solución que aborda esta investigación lo constituye la región del iris, donde la primera fase de análisis de un sistema de reconocimiento del iris, implica conocer y comprender la estructura anatómica de esta región del ojo humano.

¨ Disco membranoso del ojo de los vertebrados y cefalópodos, de color vario, en cuyo centro está la pupila. (semeja a un halo). (4)

"… una cortina de fibra muscular pigmentada situada en la parte frontal del ojo, entre la córnea y el cristalino, perforada por una apertura denominada pupila. El iris consta de dos láminas de músculo liso con propiedades contrarias de expansión y contracción. Estos músculos controlan el tamaño de la pupila y de este modo determinan la cantidad de luz que alcanza el tejido sensorial de la retina. El esfínter de la pupila es un músculo circular que estrecha la pupila en condiciones de luz brillante; el músculo dilatador de la pupila expande la abertura de la pupila al contraerse. La cantidad de pigmento contenido en el iris determina el color del ojo. Cuando hay poca cantidad de pigmentación, el ojo es azul, y a medida que aumenta la pigmentación acaba alcanzando niveles cromáticos marrones oscuros o incluso negros."

El iris está compuesto de tejido fibroso llamado estroma, que conecta músculos encargados de contraer y dilatar la pupila en respuesta a los cambios de iluminación. El mismo tiene un diámetro aproximado de 11 milímetros, y en su parte externa está rodeado de una capa blanca denominado la esclera o esclerótica.

A pesar de que los patrones detallados del tejido del iris varían de persona a persona, algunos rasgos anatómicos son comunes. Por ejemplo, en la zona media del iris aparece una región en zigzag, larga y circular, denominada collarete, un reborde circular que marca el sitio del círculo vascular menor del iris. Dispuesto alrededor del radio circular del iris y combinándose con el collarete, se encuentran a menudo dispuestas diversas crestas ligeramente elevadas. Finalmente, encontramos algunas áreas oscuras de estructura oval, y de relativamente profundo relieve, que se denominan criptas de Fuchs, que se forman en la red de colágeno del estroma, que sostiene la estructura global del iris. (3)

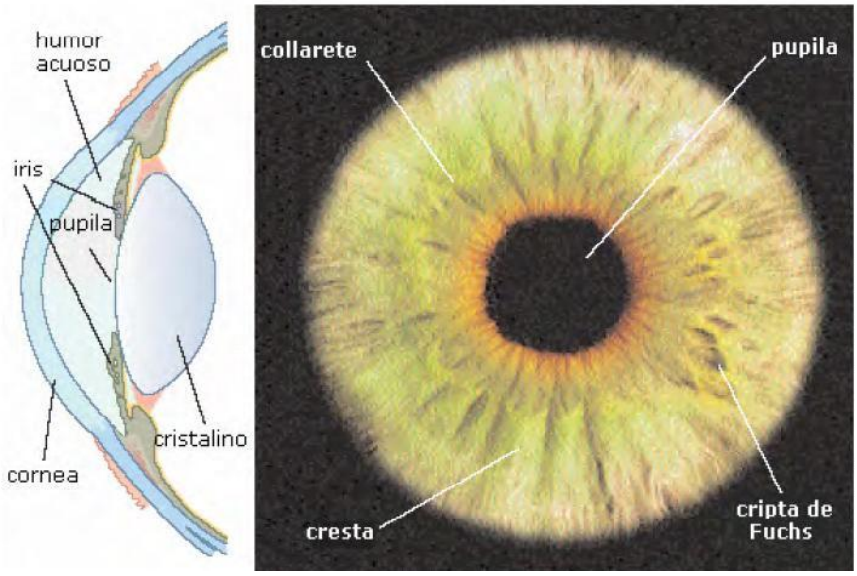

Imagen 2. Anatomía de la zona y detalles de los rasgos anatómicos del iris

1.4. Sistemas de reconocimiento del iris

Actualmente, de todos los sistemas de identidad existentes son muy pocos los que no son de carácter invasivo para el usuario, entre los que se encuentra el reconocimiento mediante patrones del iris. Ya en el mundo actual los sistemas de seguridad que aplican patrones biométricos son cada vez más aceptados, principalmente si su implantación no implica métodos invasivos que afecten la integridad de las personas.

El proceso biométrico de reconocimiento mediante iris se basa en el análisis del característico anillo de color que circunda la pupila del ojo. Tanto la base fisiológica del iris, de la que depende en gran parte la viabilidad de esta característica corporal como elemento biométrico, como el desarrollo de la tecnología asociada. (3)

La identificación basada en el reconocimiento de iris se viene utilizando para la autenticación de usuarios desde hace ya varios años. Para llevar a cabo el proceso, se hace uso de una cámara de elevada precisión, que permite obtener imágenes de alta resolución en blanco y negro del ojo del usuario, en un

entorno correctamente iluminado. Generalmente esto se hace mirando a través del lente de una cámara fija, la persona simplemente se coloca frente a la cámara y el sistema automáticamente localiza los ojos, los enfoca y captura la imagen del iris. Esta imagen se somete a deformaciones pupilares (el tamaño de la pupila varía enormemente en función de factores externos, como la luz) y de ella se extraen patrones, que a su vez son sometidos a transformaciones matemáticas hasta obtener una cantidad de datos suficiente para los propósitos de autenticación. Esa muestra, denominada iriscode es comparada con otra tomada con anterioridad y almacenada en la base de datos del sistema, de forma que si ambas coinciden el usuario se considera autenticado con éxito; la probabilidad de una falsa aceptación es la menor de todos los modelos biométricos.

El sistema de reconocimiento finalizará con la etapa de comparación, en la que se pueden aplicar diferentes algoritmos, como por ejemplo los basados en la mínima distancia entre el patrón de almacenado y los modelos obtenidos en cada una de las capturas realizadas cuando un individuo utiliza el sistema.

La fase de comparación implica el comparar el código obtenido de tomar una imagen en vivo del iris del usuario con un código específico almacenado en una base de datos. El proceso de cotejar ambas plantillas, se puede realizar mediante una comparación bit a bit de ambas (operador lógico OR Exclusivo), a través de un proceso muy rápido de cálculo de la distancia hamming entre ambos códigos, también mediante otras posibles medidas de distancias similares, o incluso métodos de medida de la correlación entre las dos imágenes del iris. (3)

1.4.1. Fases de los sistemas de reconocimiento de iris

Los sistemas de reconocimiento de iris inician capturando una imagen de ojo de la persona, después se aísla la textura del iris y se descarta el ruido producido por los párpados, las pestañas y cualquier reflejo ambiente. A continuación, esta textura se normaliza y posteriormente es codificada, resultando una representación de la textura del iris que simplificará el proceso de identificación y verificación. Como se explicó anteriormente una vez generado el código de iris, se pueden realizar dos acciones, almacenar el código en una base de datos para referencia futura, o bien realizar una búsqueda del código en una base de datos existente con el objetivo de identificar o verificar la identidad de la persona.

1.4.1.1. Captura

La captura constituye el proceso de captar toda la información que devuelve el hardware del medioambiente, específicamente en este caso se basa en captar la imagen del ojo humano que devolvería el dispositivo de captura.

1.4.1.2. Segmentación

La segmentación constituye el proceso de delimitar la región que comprende al iris. Dicha región se encuentra delimitada por la pupila (límite interno) y la esclera o esclerótica (límite externo).

1.4.1.3. Normalización

Antes de iniciar el proceso de codificación es necesario normalizar la región del iris definida durante la segmentación. La normalización en esencia constituye el hecho de tomar el área circular de la imagen y transformarla en un área rectangular de dimensiones fijas.

1.4.1.4. Codificación

Este proceso tiene como objetivo fundamental la obtención de una matriz binaria a partir de aumentar las intensidades altas y bajas de la imagen rectangular obtenida durante el proceso de normalización. La idea de aumentar las altas y bajas intensidades posibilita obtener una imagen con gran resaltado de los puntos oscuros y claros que en esta se encuentran a partir de diferentes filtros aplicados a dicha imagen.

1.4.1.5. Comparación

La esencia de esta fase es la de poder determinar la existencia o no de un usuario determinado. Se comparan dos códigos de iris, uno es el que se obtiene del usuario y otro es uno ya existente en una base de datos. Es muy poco probable que posterior a realizarse todo el proceso de captura de la imagen, segmentado, normalizado y codificado un mismo iris genere códigos de iris exactamente iguales. Esto viene dado a consecuencia de variaciones en la intensidad de la iluminación del medio ambiente al igual que en los ángulos en los que se captura la imagen del ojo humano; ya sea por la distancia en la que se capta la imagen o por la rotación de la imagen tomada a causa del propio movimiento inherente del ser humano. Todo esto ocasiona corrimiento en los patrones distintivos del iris.

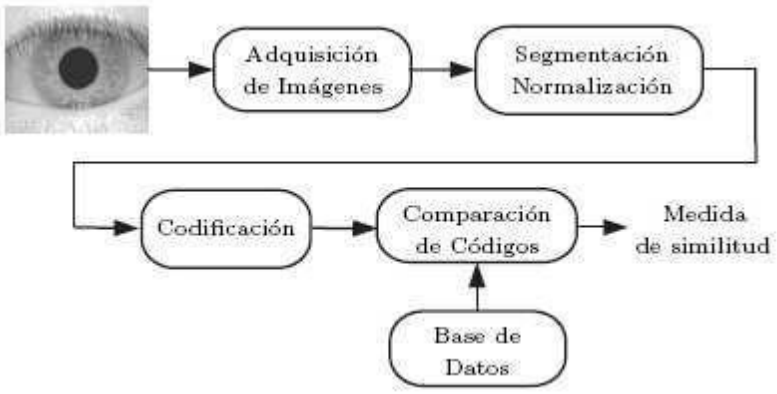

Imagen 3. Diagrama en bloque del sistema de reconocimiento del iris. (5)

1.4.1.6. Distancia de hamming

A fin de poder dar solución a la fase de comparación, es necesario el empleo el método de la distancia de hamming, el cual se describe como:

$$\mathrm{HD} = \frac{\|(codeA \otimes codeB) \bigcap maskA \bigcap maskB\|}{\|maskA \bigcap maskB\|}$$

Imagen 4. Distancia de hamming

Esta fórmula está compuesta por los códigos de iris a comparar, sean estos codeA y codeB respectivamente. A cada código se le asocia una máscara de ruido la cual representa los valores que se desprecian a la hora de analizar la información de un código de iris por ser afectada dicha información por los párpados o las pestañas, correspondiendo maskA y maskB a los códigos A y B respectivamente.

Otra forma de expresar la Distancia de hamming viene determinada por la expresión:

$$d_H = \frac{1}{L}\sum_{i=1}^{L} x_i \oplus t_i$$

Imagen 5. Distancia de hamming (3)

Siendo L la longitud del vector de características, Xi la i-ésima componente del mencionado vector, Ti la componente i-ésima del modelo o patrón, y $\oplus$ representa la operación binaria XOR.

Este constituye un método bien sencillo el cual arroja como resultado un número entre 0 y 1, el cual se corresponde con un porciento de en cuánto se diferencian dos códigos de iris, considerándose que por encima de 0.46 ya se puede descartar la posibilidad de que los códigos pertenezcan a la misma persona.

1.5. Aplicaciones y tendencias actuales de los sistemas de reconocimiento de iris.

La tecnología usada en el reconocimiento de iris, tiene un nivel de madurez adecuado para su uso comercial y sigue habiendo investigaciones en curso, principalmente en Asia, aunque cada vez más en Europa, acerca de métodos alternativos. Los sistemas trabajan con buenos resultados en modo de identificación y requieren en pocas ocasiones repetir el proceso de reclutamiento para un mismo usuario,

en comparación con otras tecnologías, haciendo de este tipo de sistemas biométricos ideales para sistemas de identificación a gran escala. (3)

La primera implementación conocida de un sistema de reconocimiento de iris fue realizada por John Daugman, posterior a que ya desde un siglo antes se planteara la idea de utilizar el iris humano como medio de reconocimiento de personas. Dicho trabajo definió las bases matemáticas y estadísticas que luego se utilizarían en casi todas las implementaciones siguientes hasta la actualidad, donde mayormente los sistemas de reconocimiento de iris disponibles comercialmente, emplean los algoritmos desarrollados por Daugman. También se pueden mencionar los sistemas desarrollados por Wildes, Boles y Boashash, Lim y más recientemente, el sistema desarrollado por Monro. (5)

Los sistemas de escaneo de iris que se encuentran actualmente en el mercado son muy eficientes, relativamente rápidos y flexibles en términos de operatividad. Trabajan en un rango de entre 10 y 20 cm, aunque existen incluso sistemas en proceso que llegan a operar en distancias tales como 5 m. El tiempo de verificación puede ser muy corto, por ejemplo, el tiempo necesario para hacer una búsqueda en una base de datos de 1 millón de Iriscodes en un PC trabajando a 2,2 GHz puede ser de aproximadamente 1,7 segundos. (3)

1.5.1. Ámbito Internacional

El reconocimiento del iris es una técnica que como se mencionaba anteriormente se encuentra en actual expansión ya que al no ser una técnica invasiva para las personas posee buena aceptación. Prácticamente todos los sistemas públicos de reconocimiento de iris que están instalados y en funcionamiento hoy en día incluyen los algoritmos creados y patentados en 1994 por J. Daugman. Entre las compañías que hacen uso de esta tecnología se encuentran LG, Oki, Panasonic, Sagem, IrisGuard, Sarnoff, IRIS, Privium, CHILD Project, CanPass, y Clear. (6)

Ejemplo de ello lo constituye el lector de reconocimiento de iris ICAM4000 de LG con un precio alrededor de los 3 855 dólares (7) y BM-ET200 de Panasonic con un precio alrededor de los 3190 euros, muy útil en aeropuertos, controles de entrada, salida en almacenes, laboratorios, oficinas, hospitales y cárceles. (8). Además, en la telefonía celular ya se encuentra disponible la identificación del iris en lugar de un número PING para acceder al teléfono, lo cual ha provocado una mayoritaria aceptación de la biometría en Europa y también está el empleo de esta técnica en empresas privadas que desean mantener cierto nivel de seguridad en áreas restringidas como en la compañía U.S. Government Solutions (9)

Entre las investigaciones que se han realizado sobre sistemas de reconocimiento de iris se pueden citar las siguientes:

Sistema de reconocimiento de personas mediante su patrón de iris basado en la transformada de Wavelet: (3) En la presente investigación se exponen los fundamentos de la biometría, una de las más novedosas tecnologías, desarrollada con el objetivo de mejorar la seguridad a nivel general de diversos aspectos presentes en la sociedad actual. En particular, se centra en la técnica de reconocimiento biométrico de iris, quizás la que mayor proyección de futuro tiene entre las numerosas existentes dentro de la biometría (huella, geometría de la mano, retina, cara,…). El reconocimiento del iris presenta unas prestaciones muy propicias para formar parte de un sistema de seguridad altamente fiable. Uno de los fundamentos en los que se basa el diseño del sistema que se desarrolló es el uso de la transformada wavelet, que permite el análisis de las señales unidimensionales, obtenidas a partir de las imágenes del iris y que precisan de una explicación exhaustiva para conocer sus fundamentos.

Esta investigación utiliza el método de la distancia de hamming para comparar los códigos de iris y se logró desarrollar un sistema de reconocimiento de iris utilizando herramientas privadas. Además, el costo de la misma accedió a un total de 44 646,66 euros con un nivel de acierto de un 99%.

Reconocimiento del iris: (9) En la presente investigación se propone un sistema de reconocimiento del iris basado en el método de Daugman. Para la comparación de los patrones se utiliza la distancia de hamming, obteniendo buenos resultados.

También, se utilizan herramientas privadas para el desarrollo del sistema que se obtiene. Aunque no se especifica exactamente el costo del componente desarrollado, solamente por el empleo de herramientas privadas se puede deducir que simplemente con el pago de las licencias de software se encárese la solución hallada.

Sistema de reconocimiento de iris: (10) En la presente investigación se describe un sistema de reconocimiento de iris utilizando una cámara digital convencional. Nuevas técnicas son propuestas para los algoritmos de las etapas de segmentación y codificación, utilizando la distancia de hamming para la fase de comparación. Una interfaz gráfica de usuario fue implementada y una base de datos de iris fue compilada para evaluar la operatividad del sistema.

En esta investigación se logró desarrollar un sistema de reconocimiento de iris con una cámara digital convencional a fin de poder abaratar el alto costo de estos sistemas, aunque una cámara digital actualmente alcanza valores por encima de los 150 dólares. Para el desarrollo de las pruebas al sistema se emplearon las bases de datos de iris NURID (National University of Rosario Iris Database) y la CASIA Iris Database.

Implementación de un sistema de identificación de personas en tiempo real por reconocimiento de iris: (11) En la presente investigación se desarrolló un sistema de reconocimiento de iris en tiempo real

mediante el empleo de una cámara de video y se emplea la distancia de hamming para comparar los códigos de iris.

Esta investigación a pesar de constar con el hardware necesario para el desarrollo del sistema, tuvo la limitante de no constar con una base de datos de iris propia para realizar la validación del mismo por lo cual, al ser insuficiente el número de imágenes de iris que se poseían, se emplearon las bases de datos de iris MMU Iris database y la CASIA iris database, ambas en su versión 1.0. Mediante el empleo de estas bases de datos el sistema arrojó un 97% de identificaciones positivas contra un 3% de identificaciones negativas. No obstante, el empleo de herramientas privadas para el desarrollo del sistema y del empleo de una cámara de video incrementa el costo del componente desarrollado.

Reconocimiento de iris: (6) En la presente investigación se desarrolló un sistema de reconocimiento de iris utilizando herramientas libres y se emplearon los métodos de John Daugman.

En esta investigación se desarrolló un sistema de reconocimiento de iris utilizando herramientas libres con bueno resultados, ascendiendo los costos a un total de 7220 euros

1.5.2. Ámbito Nacional

1.5.2.1. Universidad de las Ciencias Informáticas (UCI)

En la UCI se encontraron evidencias de investigaciones sobre sistemas de autenticación biométrica por reconocimiento de patrones del iris:

Algoritmo de segmentación para biblioteca de autenticación biométrica a partir del reconocimiento del iris: En la presente investigación se desarrolló un mecanismo de localización del iris que sienta las bases para la elaboración de una biblioteca de autenticación mediante el análisis de patrones oculares para el proyecto Seguridad del CEDIN. Se define GNU/Linux como Sistema Operativo, UML como lenguaje de modelado, RUP como metodología de desarrollo, C++ como lenguaje de programación, OpenCV como herramienta de desarrollo, Eclipse como IDE, Visual Paradigm como herramienta CASE y Doxygen como generador de documentación. (12)

Algoritmo de codificación para biblioteca de autenticación biométrica a partir del reconocimiento de patrones del iris: En la presente investigación se desarrolló la fase de codificación para la elaboración de una biblioteca de autenticación mediante el análisis de patrones oculares en el proyecto Seguridad del CEDIN, de la UCI, sobre plataforma libre. Se realizó una investigación de las principales tecnologías de desarrollo y bibliotecas existentes en el mundo. Se muestra la modelación del sistema desarrollado a partir

de diferentes diagramas y se analiza la implementación del mismo. Se muestran las diferentes pruebas que se le realizaron al algoritmo para determinar su efectividad. (5)

Las dos investigaciones anteriores permiten determinar que en la UCI se analizaron las ventajas que brinda el reconocimiento de iris como técnica de autenticación, obteniéndose con las mismas, la implementación de componentes que vienen a dar solución a las fases de captura, segmentación, normalización y codificación para una biblioteca de autenticación de usuarios por reconocimiento de patrones del iris utilizando herramientas libres. Además, sientan las bases para el desarrollo futuro de una biblioteca de reconocimiento del iris. Ya con el desarrollo de la fase de comparación, se pueden integrar todos los componentes esenciales desarrollados en las investigaciones realizadas y desarrollar una versión inicial de una biblioteca de reconocimiento de patrones del iris utilizando herramientas libres, lo cual hasta ahora según el estudio realizado aún no existe en nuestro país.

Otro elemento que demuestra el interés en la UCI de utilizar la biometría como técnica de autenticación lo constituye la investigación:

Algoritmo de detección facial para sistemas de autenticación biométrica: (13) El principal aporte de esta investigación, lo constituye es identificar un método de detección de rostro en tiempo real, realizar una descripción detallada de sus etapas y desarrollar una aplicación que permita su validación desarrollada en software libre.

1.5.3. Análisis del estado del arte

El estudio realizado respecto al estado actual de los sistemas de reconocimiento de iris ha permitido determinar la existencia de numerosas investigaciones en este campo. Mayormente por el alto costo que implica un sistema de este tipo solamente lo desarrollan empresas privadas, donde los productos presentan un elevado costo superando los 2000 euros, que llevándolos a dólares, incrementan notablemente el precio. Analizando el estado actual de la economía internacional; incluso en caso de que el euro se encuentre a un cambio de 1.2 dólares por euro, el costo ascendería como mínimo a 3600 dólares; aunque los productos poseen un buen grado de aceptación y de confiabilidad, generalmente por encima de un 95%. Además, mayormente estos sistemas se desarrollan con tecnologías privadas lo cual implica un costo en materia de licencias de software. Es notorio de señalar el poco desarrollo de estos sistemas utilizando herramientas libres, encontrándose solamente reseña de una investigación que emplea este tipo de herramientas aunque los costos del mismo superan los 7000 euros y por consecuencia el producto. También, se encontró reseñas de una investigación que desarrolló un sistema de reconocimiento de iris utilizando una cámara digital convencional, este proyecto logró abaratar los costos en materia de hardware, aunque hay que chequear el nivel de calidad de la captura de la imagen

que garantice un buen procesado de los datos; aunque como en muchos casos, se emplearon herramientas privadas para el desarrollo de la solución.

En nuestro país es muy pobre el empleo del reconocimiento del iris como técnica de autenticación de usuarios. Solamente se han desarrollado investigaciones en la UCI que aún no llegan a conformar una biblioteca de autenticación de usuarios a partir del reconocimiento del iris, aunque logran darle solución a fases esenciales de una biblioteca de este tipo. El principal aporte de estas investigaciones, aparte de ser pioneras en este campo para Cuba, es el empleo de herramientas libre que evitan los costos por licenciamiento de software.

Es necesario mencionar la diferencia que existe entre un sistema de reconocimiento de iris y una biblioteca de autenticación de usuarios por reconocimiento del iris. Un sistema constituye un software que realiza las fases de reconocimiento del iris funcionando junto con el hardware que capta las imágenes del ojo humano, en cambio, una biblioteca constituye un conjunto de clases y métodos que realizan las fases fundamentales para el reconocimiento de iris, que al carecer de hardware e interfaz de usuario, constituye la herramienta a utilizar para crear un sistema con gran facilidad, incorporando solamente la interfaz de usuario y el dispositivo de captura para obtener un sistema.

Según el estudio que se está realizando no existen reseñas del desarrollo de una biblioteca de reconocimiento de iris tanto internacional como nacional. Por lo que el desarrollo de una versión inicial de una biblioteca de este tipo, empleando herramientas libres, permitiría dar un paso de avance en los sistemas de seguridad biométricos en Cuba y posibilitaría ir desarrollando un producto que en un futuro pudiese competir en el mercado evitando los costos de licenciamiento y el bloqueo económico de los Estados Unidos.

1.5.4. Aplicaciones de la biometría (3)

Gran parte de las aplicaciones biométricas están relacionadas con la seguridad y son ampliamente utilizadas para propósitos militares y gubernamentales. Las diferentes aplicaciones que se pueden dar a la biometría podrán ser divididas en las siguientes categorías:

- **Control de inmigración y fronteras:** puestos de control fronterizos, aduanas, emisión de pasaportes y visas, casos de asilo.
- **Fuerzas de seguridad**: investigación criminal, instituciones penitenciarias, arresto domiciliario, armas inteligentes.

- **Permisos de conducción:** el objetivo buscado es que un mismo permiso no pueda ser utilizado por distintos conductores.

- **Seguridad en entornos informáticos**: acceso a ordenadores personales, acceso a redes, comercio electrónico, correo electrónico, encriptación.

- **Servicios financieros:** cajas de seguridad, transacciones bancarias.

- **Servicios sanitarios:** seguridad para mantener la privacidad de las historias médicas.

- **Sistemas de control de acceso:** tanto en edificios institucionales, corporativos y gubernamentales como en edificios residenciales.

- **Sistemas de pago de aseguradoras:** en América, algunos estados han conseguido ahorrar grandes cantidades de dinero empleando procedimientos de verificación biométrica. El número de individuos que ha reclamado indemnizaciones se ha visto reducido significativamente, confirmando la fiabilidad de los sistemas.

- **Sistemas de visita en prisiones:** el visitante deberá someterse a procedimientos de reconocimiento para evitar cambios de identidad con internos durante las visitas, hecho más común de lo que en un principio pueda parecer.

- **Sistemas de votación:** con este tipo de aplicaciones se busca reducir si no eliminar completamente posibles fraudes electorales motivados por ejemplo, por el múltiple ejercicio del derecho al voto de un individuo

- **Telecomunicaciones:** teléfonos móviles, tarjetas telefónicas, teletienda.

1.6. Herramientas y tecnologías usadas

1.6.1. Metodología de desarrollo de software: RUP

De las metodologías que utilizan UML como lenguaje de modelado se decidió usar RUP (Rational Unified Process, Proceso Unificado de Racional) por ser con la que estamos más familiarizados, además de que provee un entorno de proceso de desarrollo configurable, basado en estándares; permite tener claro y accesible el proceso de desarrollo que se sigue; puede ser configurado de acuerdo con las necesidades de la organización y del proyecto; provee a cada participante con la parte del proceso que le compete directamente. RUP se caracteriza por ser:

- Dirigido por casos de uso: los casos de uso son los artefactos primarios para establecer el comportamiento deseado por el sistema.

- Centrado en la Arquitectura: la arquitectura es utilizada para conceptualizar, construir, administrar y evolucionar el sistema en desarrollo.

- Iterativo e incremental: maneja una serie de entregas ejecutables; integra continuamente la arquitectura para producir nuevas versiones mejoradas. (14)

Las principales disciplinas o ciclos de esta metodología son:

Disciplinas	**Descripción**
Modelado del negocio	Describe los procesos de negocio, identificando quiénes participan y las actividades que requieren automatización. Propone comprender los problemas de la organización e identificar posibles mejoras, evaluar el impacto del cambio introducido con la automatización, asegurar que todos los miembros tienen una misma visión de la organización.
Requerimientos	Esta disciplina se encarga de convertir las peticiones de los clientes en un conjunto de requerimientos de software, que definan el alcance y lo que debe hacer el producto a ser construido. Propone mantener un acuerdo a los interesados de lo que el sistema debe hacer, provee a los desarrolladores de una mejor visión de los requerimientos del sistema, define la frontera del sistema, crea las bases para la planificación y estimación.

Análisis y Diseño	Esta disciplina define como transformar artefactos resultantes del flujo de requerimientos de software en especificaciones del diseño del proyecto a desarrollar, en él se hace evolucionar la arquitectura, de modo que se adapte al entorno del usuario final.
Implementación	Define como desarrollar, organizar realizar pruebas de unidad e integración de todos los componentes implementados basado en las especificaciones del sistema.
Pruebas	Este flujo se centra en encontrar y documentar los defectos en la calidad del software, validar y probar todos los supuestos hechos durante el diseño, verificar que el producto se desempeña como fue diseñado y cubre todos los requisitos pactados durante el flujo de “Captura de Requerimientos”.
Instalación	Se encarga de garantizar que el producto está disponible para los usuarios en su ambiente, se realizan actividades como empaque, instalación, asistencia a usuarios.
Administración del proyecto	Centra los aspectos esenciales del desarrollo iterativo como son la planificación, captura de

	riesgos, seguimiento del proceso de desarrollo de software y aplicación de métricas. Las restantes serán utilizadas durante el proceso de administración.
Administración de configuración y cambios	Se encarga de capturar y sincronizar la evolución de todos los productos generados, introduce el uso de herramientas que faciliten el trabajo colaborativo en el equipo.
Ambiente	Contiene actividades que describen los procesos y herramientas que soportarán el equipo de trabajo del proyecto; así como el procedimiento para implementar el proceso en una organización.

RUP divide el proceso de desarrollo en ciclos teniendo un producto final al concluir cada ciclo.

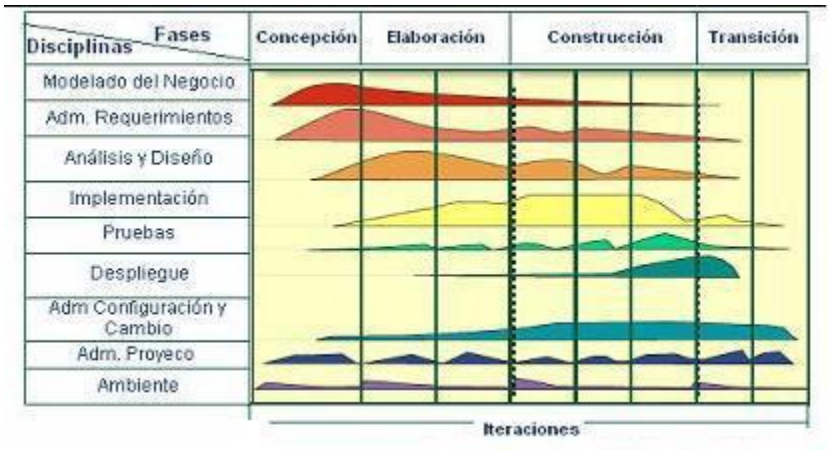

Imagen 6. Disciplinas y fases de RUP

Cada ciclo se divide en fases que finalizan con un hito donde se debe tomar una decisión importante:

- Conceptualización (Concepción o Inicio): Se describe el negocio y se delimita el proyecto describiendo sus alcances con la identificación de los casos de uso del sistema.

- Elaboración: Se define la arquitectura del sistema y se obtiene una aplicación ejecutable que responde a los casos de uso que la comprometen. A pesar de que se desarrolla a profundidad una parte del sistema, las decisiones sobre la arquitectura se hacen sobre la base de la comprensión del sistema completo y los requerimientos (funcionales y no funcionales) identificados de acuerdo al alcance definido.

- Construcción: Se obtiene un producto listo para su utilización que está documentado y tiene un manual de usuario. Se obtiene uno o varios entregables del producto que han pasado las pruebas. Se ponen estos entregables a consideración de un subconjunto de usuarios.

- Transición: El entregable ya está listo para su instalación en las condiciones reales. Puede implicar reparación de errores. (14)

1.6.2. Sistema Operativo: GNU/Linux

Debido a la gran cantidad de cálculo que requieren estos sistemas de autenticación se decidió utilizar GNU/Linux con su distribución de Ubuntu 11.04 como Sistema Operativo (SO) por ser muy eficiente en lo referido al procesamiento de información. Es multitarea, multiusuario, multiplataforma y multiprocesador y en las plataformas Intel se ejecuta en modo protegido; protege la memoria para que un programa no pueda hacer caer el resto del sistema; carga solo las partes de un programa que se usan lo que proporciona una mayor eficiencia; comparte la memoria entre programas aumentando la velocidad y disminuyendo el uso de memoria; utiliza toda la memoria libre para caché; permite usar bibliotecas enlazadas tanto estática como dinámicamente, además de ser un SO libre por el que no se debe pagar una patente y se distribuye con su código fuente lo que permite adaptar el sistema a las necesidades de cada desarrollador.

1.6.3. Lenguaje de Modelado: UML

Se decidió utilizar el Lenguaje Unificado de Modelado (UML) por ser el lenguaje de modelado de sistemas de software más conocido y utilizado tanto en nuestra universidad como mundialmente. Es un lenguaje para visualizar, especificar, construir y documentar los artefactos de un sistema. Está compuesto por elementos gráficos que se combinan para conformar diagramas. El desarrollo de sistemas con UML incluye actividades específicas, cada una de ellas contiene a su vez otras sub-actividades las que sirven como una guía de cómo deben ser las actividades desarrolladas y secuenciadas con el fin de obtener sistemas exitosos. UML no tiene propietario y está basado en el común acuerdo de la comunidad informática.

1.6.4. Lenguaje de Programación: C++

El uso de C++ como lenguaje para el desarrollo viene dado por su robustez y eficiencia a pesar de sus más de 20 años y constituye el lenguaje con el que más nos encontramos familiarizados. Además, su increíble versatilidad que permite programar desde el software más simple a los programas más complicados como incluso sistemas operativos. Tiene la ventaja de ser portable, lo que quiere decir que un programa con el código escrito en C++ se podrá compilar en cualquier sistema operativo sin necesidad de cambiar mucho el código fuente.

C++ es un lenguaje de programación diseñado a mediados de los años 1980 por Bjarne Stroustrup. La intención de su creación fue el extender al exitoso lenguaje de programación C con mecanismos que permitan la manipulación de objetos. En ese sentido, desde el punto de vista de los lenguajes orientados a objetos, el C++ es un lenguaje híbrido.

Posteriormente se añadieron facilidades de programación genérica, que se sumó a los otros dos paradigmas que ya estaban admitidos (programación estructurada y la programación orientada a objetos). Por esto se suele decir que el C++ es un lenguaje de programación multiparadigma.

Una particularidad del C++ es la posibilidad de redefinir los operadores (sobrecarga de operadores), y de poder crear nuevos tipos que se comporten como tipos fundamentales.

1.6.5. IDE: Eclipse

Un entorno de desarrollo integrado (IDE por sus siglas en inglés) es un programa compuesto por una serie de herramientas para un programador. Puede estar dedicada a un lenguaje de programación en específico o bien puede utilizarse para varios. Entre las herramientas más comunes que poseen los IDEs están: un editor de código, un compilador, un intérprete, un depurador y un constructor de interfaces gráficas de usuario.

En la selección del IDE para desarrollar influyeron varios requisitos: se necesitaba que este permitiera la programación en C++, que no fuera un software privado, que contara con herramientas integradas para la gestión de la configuración tanto del código como de la documentación, que presentara un buen completamiento de código y que brindara facilidades en la vinculación de bibliotecas y en la configuración del compilador.

1.6.6. CASE: Visual Paradigm

Las herramientas CASE (Computer Aided Software Engineering, Ingeniería de Software Asistida por Ordenador) constituyen un conjunto de aplicaciones informáticas con herramientas y métodos asociados que proporcionan asistencia automatizada en el proceso de desarrollo del software a lo largo de su ciclo de vida. Están destinadas a aumentar la productividad en el proceso de desarrollo de software en términos de tiempo y de dinero. Entre las herramientas que proporcionan están herramientas para la planificación, estimación y control de proyectos; análisis, diseño, implementación, validación del software y herramientas para mantenimiento del software. (15)

Como herramienta CASE se utilizó Visual Paradigm, las principales características que se necesitan y que en específico cumple esta herramienta son: es multiplataforma y modela todos flujos de trabajo de RUP. Visual Paradigm para UML es una Herramienta Case Cruzado de Ciclo de Vida, lo cual significa que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue, incluye también actividades como la gestión de proyectos y la estimación. El software de modelado UML, ayuda a una más rápida construcción de aplicaciones de mejor calidad y a un menor coste. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. Incluye además importación desde Rational Rose, exportación/importación XMI, generador de informes, editor de figuras, integración con MS Visio, plug-in, integración IDE con Visual Studio, IntelliJ IDEA, Eclipse, NetBeans y otros. Apoya un conjunto de lenguajes tanto en la generación del código como en la Ingeniería Inversa por ejemplo Java, C + +, CORBA IDL, PHP, XML Schema, Ada y Python.

1.6.7. Generador de documentación: Doxygen

El paquete Doxygen contiene un sistema de documentación para C++, C, Java, Objective-C, Corba IDL y, en parte, PHP, C# y D. Es útil para generar documentación HTML y/o un manual de referencia a partir de un grupo de ficheros fuente documentados. También soporta generación de salida en RTF, PostScript, PDF con hiperlinks, HTML comprimido y páginas de manual Unix. La documentación se extrae directamente de las fuentes, lo que hace mucho más fácil mantener consistente la documentación con el código fuente.

1.6.8. Deboostrap

Constituye una herramienta libre que permite realizar una instalación mínima de un sistema Debian denominado sistema base dentro de una distribución de GNU/Linux ya instalada en un ordenador.

1.6.8.1. Sistema base

Un sistema base constituye un entorno de un Sistema GNU/Linux (cualquier distribución) que se puede implantar en cualquier directorio del sistema de ficheros de la distribución de Linux que se esté utilizando. Dicho sistema cuenta con una instalación mínima de los directorios y elementos fundamentales de cualquier Sistema GNU/Linux. En caso de ser necesario un entorno de prueba para desarrollar o compilar una aplicación sin necesidad de modificar los componentes del sistema operativo, se puede optar por la opción de instaurar un sistema base empleando el comando chroot que redirecciona el directorio raíz del sistema operativo hacia el directorio del sistema base. Esto permite además, poder instalar herramientas y bibliotecas para el desarrollo de una aplicación o biblioteca de códigos; incluyendo la posibilidad de poseer un sistema portable con todos los elementos necesarios para continuar el desarrollo.

1.7. Conclusiones

En el presente capítulo se explicaron las principales fases con las que cuenta un sistema de autenticación por reconocimiento de iris. Además, se enfatizó en la distancia de hamming como método de comparación existente para códigos binarios de iris, siendo este el método a utilizar en el desarrollo de la investigación. También, se explicaron las herramientas a utilizar en dicho proceso de desarrollo junto con la metodología de desarrollo a seguir para lograr este objetivo. Se hizo un estudio del estado del conocimiento con respecto a los sistemas biométricos existentes dando como resultado la existencia de un gran número de estos, aunque todos basándose en los procesos y técnicas descritas anteriormente.

Capítulo 2. Fundamentación de la propuesta de solución.

1.1. Introducción al capítulo

En el presente capítulo se desarrollará el proceso de modelación del sistema y se llevará a cabo una descripción de las características del mismo. Por tal motivo se especifican los requisitos funcionales y los no funcionales con los que debe cumplir la aplicación; además, se especifican los actores, se describen los casos de uso y se construyen una serie de diagramas que permitirán una mejor comprensión del sistema.

1.2. Reglas del negocio

Los códigos de iris a utilizar para el proceso de comparación deben ser los que genera la biblioteca en el procesado inicial de las imágenes del ojo humano.

1.3. Modelo de Dominio

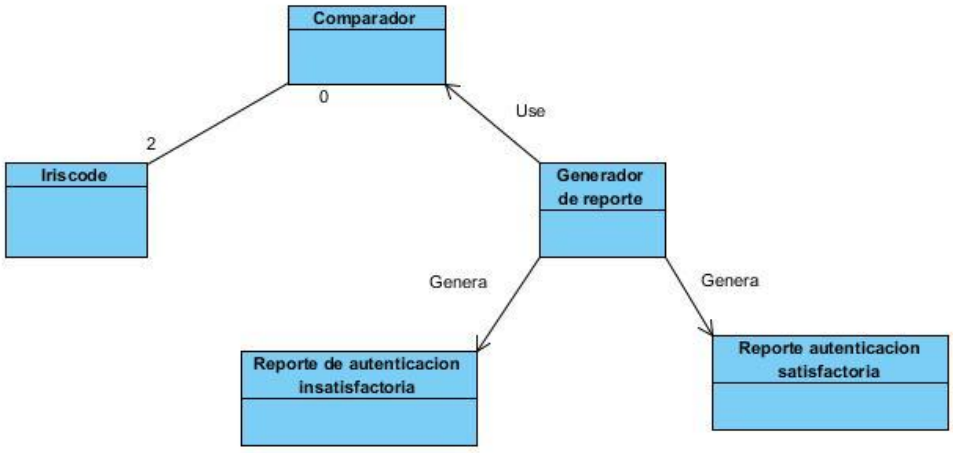

Imagen 7. Modelo de dominio

1.3.1. Glosario de Términos del Dominio

Iriscode: Arreglo binario que contiene la información referente a los patrones visuales de una imagen del iris.

Comparador: Algoritmo que determina el nivel de diferencia entre dos Iriscodes.

1.4. Especificación de los requisitos de software

1.4.1. Requisitos funcionales del sistema.

- RF1. Implementar el algoritmo que calcule la distancia de hamming entre dos códigos de iris.
- RF2. Generar un reporte con los resultados de autenticación.

1.4.2. Requisitos no funcionales del sistema.

Software

- Por políticas del centro productivo de Informática Industrial se usó GNU/Linux.

Requerimientos de hardware

- Se recomienda usar procesadores INTEL ya que la biblioteca de códigos OpenCV (que utilizará la biblioteca a desarrollar, en el procesado de las imágenes del ojo humano en las etapas anteriores al proceso de comparación) está optimizada para este tipo de procesador.

Requerimientos de soporte

- Crear un *.deb de la biblioteca de reconocimiento de iris para facilitar su instalación.

1.5. Definición de los casos de uso del sistema.

Requerimientos en el diseño y la implementación.

- La solución cuenta con un conjunto de patrones que permite la reutilización del código así como facilidad en la actualización del mismo.
- El código cumple con los estándares de codificación establecidos por el CEDIN.

Requerimientos de Soporte.

- Se ofrece servicios de mantenimiento y actualización.

Requerimientos de Usabilidad.

- La biblioteca podrá ser usada por cualquier persona que posea conocimientos básicos en el manejo de la computadora, programación y el sistema operativo GNU/Linux.

Requerimiento asociado al Licenciamiento.

- Se debe garantizar que el sistema se desarrolle bajo los principios del software libre y por tanto cualquier componente de software que se utilice también lo debe ser.

1.6. Análisis y diseño del sistema.

1.6.1. Actores

Un actor representa a una persona, otros sistemas o hardware externo que interactúa con un sistema determinado, el cual asume un conjunto de papeles coherentes durante dicha interacción. Un actor se comunica con un sistema mediante la recepción y el envío de mensajes desde y hacia el sistema a medida que se van ejecutando los casos de uso. Un usuario físico puede actuar como uno o varios actores, desempeñando el papel de los mismos durante su interacción con el sistema; al igual que varios usuarios concretos pueden actuar como diferentes ocurrencias de un mismo actor. (14)

1.6.2. Casos de uso

Los casos de uso están diseñados para cumplir los deseos de un usuario cuando este utiliza un sistema determinado, siendo estas las funcionalidades con las que debe contar dicho sistema para que sea de utilidad para el usuario que lo solicitó, denominadas requisitos funcionales. (14)

1.6.3. Diagrama de casos de uso del sistema

Un diagrama de casos de uso, como se modela a continuación, constituye la representación gráfica de los requisitos funcionales del sistema encapsulados en casos de uso al igual que las relaciones existentes entre estos casos de uso y los actores existentes en el sistema de manera estática. (14)

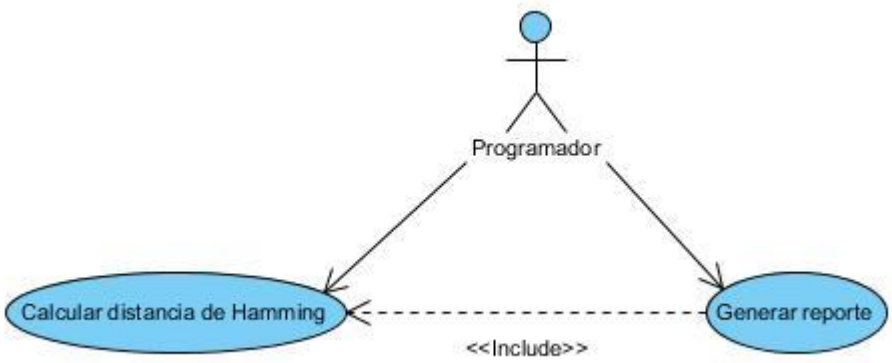

Imagen 8. Diagrama de casos de uso del sistema

1.6.4. Descripción de casos de uso del sistema.

1.6.4.1. CU1 Calcular distancia de haming.

Caso de Uso:	Calcular distancia de hamming
Actores:	Programador.
Resumen:	El caso de uso se inicia cuando el programador solicita que a dos códigos de iris se le calcule la menor distancia de hamming existente entre ellos a partir de poder obtener el nivel de disimilitud existente entre ellos.
Objetivo:	Obtener la mínima distancia de hamming entre dos códigos de iris.
Referencias:	R1
Precondiciones:	
Complejidad:	Alta.
Prioridad:	Crítico.

Flujo Normal de Eventos	
Programador	Biblioteca
1. Solicita calcular la distancia de hamming (código de iris1, código de iris2).	2. Toma los códigos de iris
	1. Calcula y almacena el nivel de diferencia entre los códigos
	2. Va rotando el segundo código a la izquierda. Calcula y almacena el nivel de diferencia entre los códigos por cada rotación
	3. Va rotando el segundo código a la derecha. Calcula y almacena el nivel de diferencia entre los códigos por cada rotación
	4. Selecciona el menor nivel de diferencia entre los códigos calculado.
	5. Devuelve el menor nivel de diferencia entre los códigos
Flujos Alternos	

Programador	Biblioteca
Prototipo de Interfaz	
Poscondiciones	

Tabla 1. Descripción del caso de uso Calcular distancia de haming

1.6.4.2. CU2 .Generar reporte

Caso de Uso:	Generar reporte
Actores:	Programador.
Resumen:	El caso de uso se inicia cuando el programador solicita que se cree un reporte.
Objetivo:	Crear un reporte de la autenticación del usuario.
Referencias:	R1, CU1*(include)*
Precondiciones:	
Complejidad:	Alta.
Prioridad:	Crítico.
Flujo Normal de Eventos	
Programador	Biblioteca
1. Solicita la creación de un reporte (identificador de usuario, identificador de reporte).	✓ Valida el identificador de usuario y el identificador de reporte. ➢ 2.1. Si identificador de usuario menor que 1 e identificador de reporte menor que 0. Alterno 1 ➢ 2.2. Si identificador de usuario menor que 1 e identificador de reporte menor que 0. Flujo normal de eventos.
	✓ Calcula la distancia de hamming.

	✓ Genera un reporte general.
	✓ Compara la distancia de hamming con un valor de decisión. ✓ 5.1. En caso de que la distancia de hamming sea mayor o igual que el umbral de decisión.Alterno2. ✓ 5.2. En caso de que la distancia de hamming sea menor que el umbral de decisión. Flujo normal de eventos
	✓ Genera un reporte de autenticación satisfactoria.

Flujos Alternos	
Alterno 1	
Programador	Biblioteca
	2.1. Identificador de usuario menor que 1 e identificador de reporte menor que 0.
	2. 2. No realiza ninguna otra acción

Alterno 2	
Programador	Biblioteca
	5.1. Distancia de hamming mayor o igual que el umbral de decisión
	5.2. Genera un reporte de autenticación fallida.

Prototipo de Interfaz	
Poscondiciones	

Tabla 2. Descripción del caso de uso Generar reporte

1.6.5. Diseño

1.6.5.1. Diagramas de interacción

Los diagramas de interacción representan el flujo de mensajes que existen entre las diferentes interfaces de un sistema. Se dividen en dos tipos: diagramas de secuencia, que muestran los objetos que participan en la interacción mediante sus líneas de vida y mediante los mensajes que intercambian, organizados en forma de una secuencia temporal y los diagramas de colaboración donde la secuencia de mensajes se indica con los números secuenciales que preceden a las descripciones del mismo y se muestran como flechas, ligadas a las líneas de la relación, que conectan a los roles. (14)

1.6.5.1.1. CU1 Calcular distancia de haming

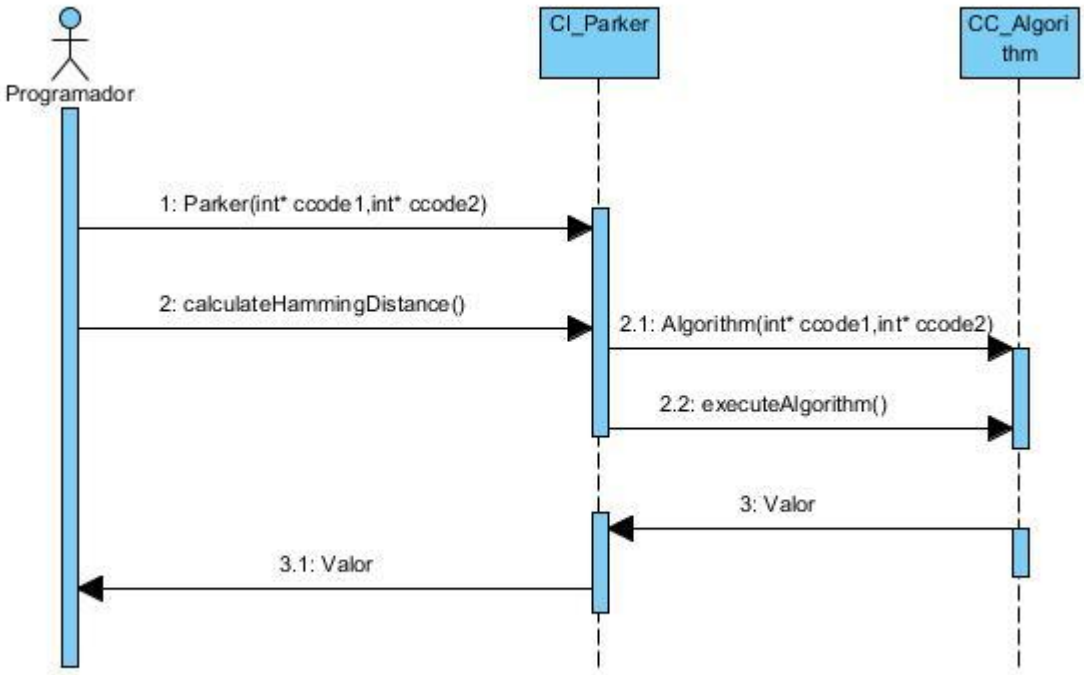

Imagen 9. Diagrama de Secuencia (Calcular distancia de hamming)

1.6.5.1.2. CU2 .Generar reporte

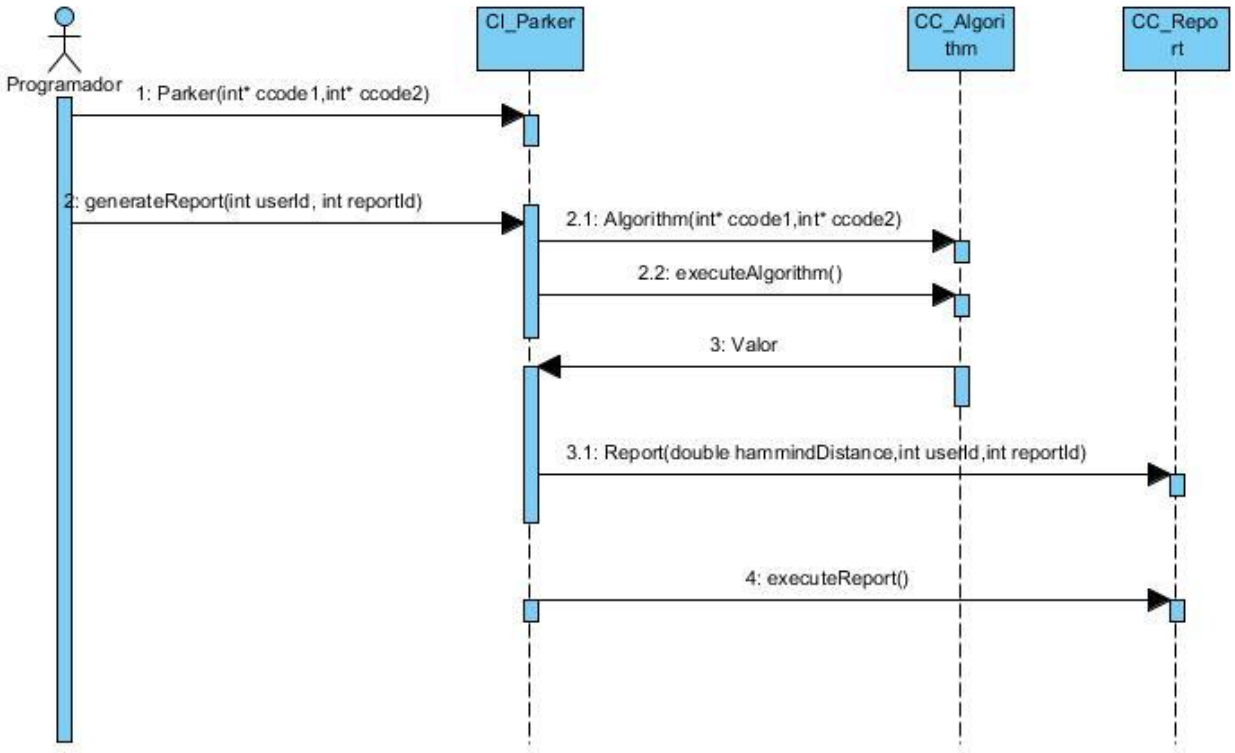

Imagen 10. Diagrama de Secuencia (Generar reporte)

1.6.5.2. Diagramas de clases del diseño

Los diagramas de clases del diseño muestran las clases fundamentales con las que cuenta el sistema, además de los atributos y métodos que las componen así como las relaciones que se establecen entre ellas. (14)

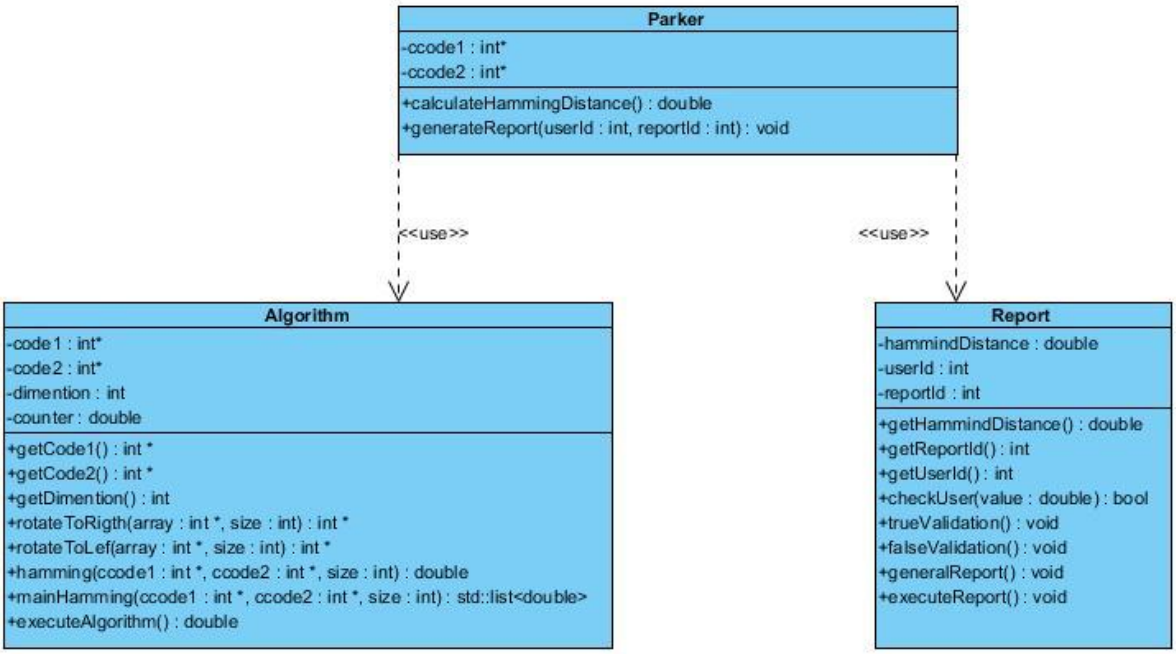

Imagen 11. Diagrama de clases del diseño

1.6.5.3. Diagrama de Componentes

Los diagramas de componentes describen de manera estática, como los elementos del diseño se implementan como componentes, ya sea código fuente, ejecutables, etc. Además, describen cómo se organizan dichos componentes de acuerdo con los mecanismos de estructuración y modulación en el entorno de implementación y en el lenguaje o lenguajes de programación utilizados y las dependencias entre los componentes. (14)

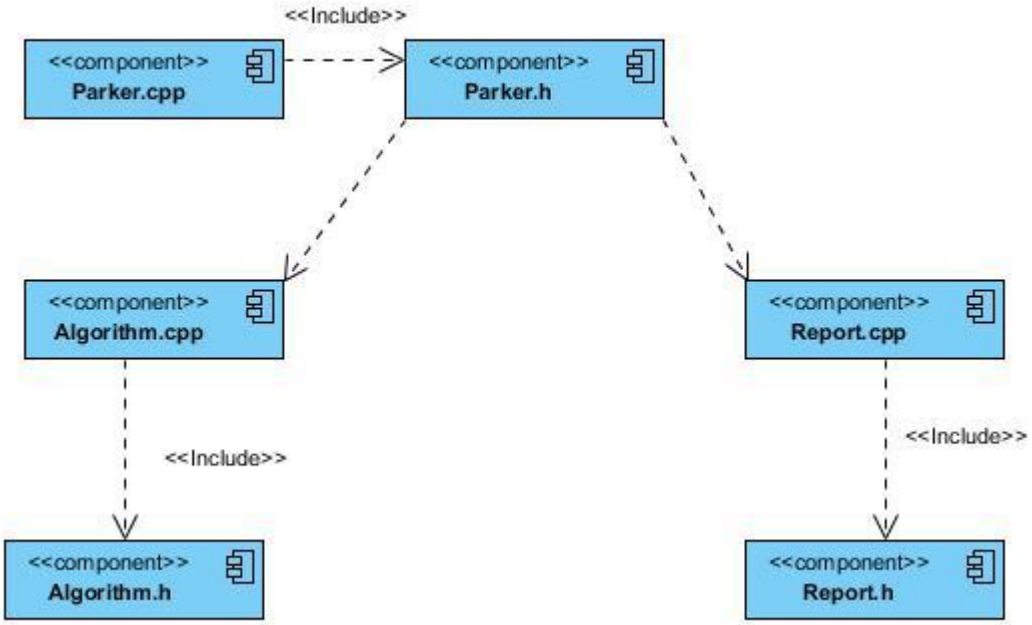

Imagen 12. Diagrama de componentes

1.6.5.4. Diagrama de Despliegue.

Los diagramas de despliegue describen cómo quedarían distribuidos o desplegados los componentes de un sistema en nodos de procesamiento. (14)

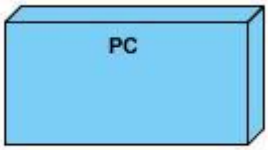

Imagen 13. Diagrama de despliegue

1.6.5.5. Descripción de las clases principales.

A continuación se realizará la descripción de las principales clases, detallando sus funcionalidades y atributos que las componen.

Nombre: Parker	
Tipo de clase: Interfaz	
Atributo	**Tipo**

ccode1	int*
ccode2	int*
Para cada responsabilidad:	
Nombre:	`Parker(int* pccode1, int* pccode2)`
Descripción:	Constructor por defecto de la clase
Nombre:	`~Parker()`
Descripción:	Destructor de la clase
Nombre:	`calculateHammingDistance()`
Descripción:	Calcular la distancia de hamming
Nombre:	generateReport(`int usserId, int reportId`)
Descripción:	Genera los reportes de la autenticación

Tabla 3. Descripción de la clase Parker

Nombre: Algorithm	
Tipo de clase: Control	
Atributo	**Tipo**
`code1`	int*
`Code2`	int*
`dimention`	int
`counter`	double
Para cada responsabilidad:	
Nombre:	`Algorithm(int* ccode1,int* ccode2)`
Descripción:	Constructor por defecto de la clase con los dos códigos de iris a los que se les va a calcular la distancia de hamming.
Nombre:	`~Algorithm()`
Descripción:	Destructor de la clase.
Nombre:	`getCode1()`
Descripción:	Devuelve un arreglo que contiene el primer código de iris.
Nombre:	`getCode2()`
Descripción:	Devuelve un arreglo que contiene el segundo código de iris.
Nombre:	`getDimention()`
Descripción:	Devuelve la dimensión que tienen los códigos de iris.
Nombre:	`rotateToRigth(int *array, int size)`
Descripción:	Devuelve el arreglo que se le pasa por parámetro rotado hacia la derecha.
Nombre:	`rotateToLef(int *array,int size)`
Descripción:	Devuelve el arreglo que se le pasa por parámetro rotado hacia la izquierda.
Nombre:	`hamming(int* ccode1,int* ccode2,int size)`
Descripción:	Devuelve el cálculo de la distancia de hamming entre dos códigos de iris

Nombre:	`mainHamming(int* ccode1,int* ccode2,int size)`
Descripción:	Devuelve una lista de todas las distancias de haming obtenidas a partir de haber realizado todas las comparaciones entre todas las rotaciones a la izquierda y derecha entre los códigos de iris
Nombre:	`executeAlgorithm()`
Descripción:	Ejecuta todo el algoritmo y devuelve la menor de todas las distancias de hamming obtenidas en la ejecución del algoritmo

Tabla 4. Descripción de la clase Algorithm

Nombre: Report	
Tipo de clase: Entidad	
Atributo	**Tipo**
`hammindDistance`	double
`reportId`	int
`userId`	int
Para cada responsabilidad:	
Nombre:	`Report(double hammindDistance,int userId,int reportId)`
Descripción:	Constructor por defecto de la clase con la distancia de hamming calculada, el identificador del usuario y el identificador del reporte.
Nombre:	`~Report()`
Descripción:	Destructor de la clase.
Nombre:	`gethammindDistance()`
Descripción:	Devuelve el valor de la distancia de hamming.
Nombre:	`getReportId()`
Descripción:	Devuelve el identificador del reporte.
Nombre:	`getUserId()`
Descripción:	Devuelve el identificador del usuario.
Nombre:	`executeReport()`
Descripción:	Ejecuta el proceso completo de generar un reporte.
Nombre:	`checkUser(double value)`
Descripción:	Chequea si el valor que se le pasa por parámetro menor que 0.46.
Nombre:	`convertInt(int number)`
Descripción:	Convierte y devuelve en valor que se le pasa por parámetro en un tipo de dato string.
Nombre:	`trueValidaton()`
Descripción:	Genera un reporte fichero de texto con la información de una autenticación satisfactoria.
Nombre:	`falseValidaton()`
Descripción:	Genera un reporte fichero de texto con la información de una autenticación insatisfactoria.

Nombre:	`generalReport()`
Descripción:	Genera un reporte fichero de texto con toda la información de la autenticación

Tabla 5. Descripción de la clase Report.

1.7. Patrones de Diseño

Un patrón de diseño es un mecanismo de la ingeniería de software que brinda robustez y flexibilidad a un conjunto de clases; este nombra, abstrae, e identifica los aspectos esenciales de una estructura común del diseño que la hacen útil para la creación de un diseño reusable orientado a objeto. Constituye una descripción de relaciones entre objetos y clases hechos a la medida para lograr la solución de un problema general de diseño en un contexto determinado.

Para el desarrollo del presente trabajo se hace empleo del patrón Facade o Fachada. Este patrón pertenece al grupo de los patrones estructurales, los cuales están centrados en cómo las clases y los objetos se agrupan para formar estructuras más complejas. El patrón Facade provee una interfaz única a un conjunto de fachadas de un subsistema, al definir esta la interfaz única como la de más alto nivel, permite un acceso más fácil a dicho subsistema y posibilita que cualquier aplicación use las funciones que dicha interfaz es capaz de brindarle. La clase Parker, actúa como una interfaz y provee las funcionalidades de las clases Algorithm y Report.

El sistema de clases es flexible; diseñado para crecer sin afectar a los componentes existentes. Las clases comparten sus funcionalidades con la interfaz para que estas puedan ser exportadas. Este funcionamiento será el mismo para los nuevos paquetes que se decidan agregar.

1.8. Arquitectura del sistema

La Arquitectura de Software es una vista del sistema que incluye los componentes principales del mismo, la conducta de esos componentes según se la percibe desde el resto del sistema y las formas en que los componentes interactúan y se coordinan para alcanzar la misión del sistema. (16)

Los patrones arquitectónicos constituyen los patrones de más alto nivel en un sistema dado que especifican la estructura fundamental de dicho sistema. Cada actividad de desarrollo es gobernada por esta estructura, como por ejemplo el diseño detallado de subsistemas, la comunicación y la colaboración entre diferentes partes del sistema, y su posterior extensión.

Para el desarrollo del presente trabajo se aplicó una arquitectura en capas. Este sistema cuenta con una capa de presentación donde se encuentra la clase Parker y una capa de lógica donde se encuentran las clases Algorithm y Report, que manejan el desarrollo del algoritmo.

Capítulo 3. Desarrollo y pruebas.

3.1. Introducción

Este capítulo trata sobre el estándar de codificación y documentación y la implementación del algoritmo de comparación, así como de la distancia de hamming utilizados para la codificación. También se explicarán diferentes pruebas de sistemas realizadas al software.

3.2. Ventajas del empleo de Doxygen

Como se explicaba en capítulos anteriores, Doxygen es un programa que analiza el código en busca de directivas en forma de comentarios y las transforma en documentación, bien HTML. La principal ventaja de su empleo, es sin duda que permite simplemente documentar el código de la aplicación y liberar al programador de la tarea de crear una documentación aparte describiendo todo el código desarrollado.

3.3. Implementación del algoritmo de comparación

Como se explica en capítulos anteriores la comparación es la fase donde se trata de determinar si un usuario constituye un usuario del sistema a partir de hallar el nivel de diferencia existente entre dos códigos de iris correspondientes de una misma persona.

Para darle solución a esta etapa, en este trabajo se hace empleo de un algoritmo basado en la distancia de hamming a fin de obtener un valor entre 0 y 1 como medida de la diferencia entre los dos códigos de iris que permitirá comparar este valor con un umbral de decisión que se definirá posteriormente.

A fin de hallar este valor, inicialmente se realiza una operación XOR en cada posición de los códigos de iris, esto consiste en almacenar un valor igual a cero en caso de que la información sea igual o un valor igual a 1 en caso de que la información sea diferente, finalmente se suman todos los valores iguales a 1, se divide entre la longitud del código de iris y se almacena este valor cómo se muestra en la imagen siguiente.

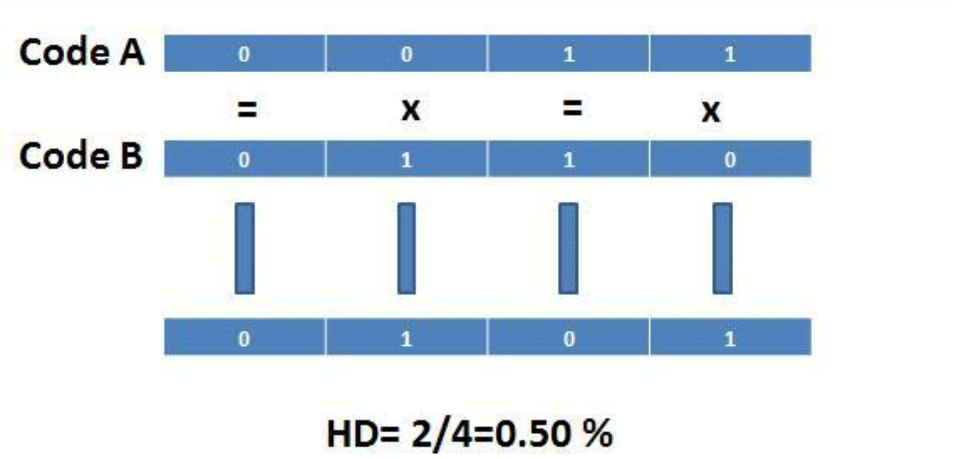

Imagen 14. Cálculo de Distancia de hamming

Es muy difícil que dos códigos de iris sean exactamente iguales incluso proviniendo de la misma persona, esto se debe a variaciones angulares que pueden ocurrir a la hora de capturar la imagen del ojo, movimiento inherente del cuerpo humano y a variación de la intensidad de la luz ambiental. A fin de simular todas estas posibles variaciones, posterior a este primer cálculo se hace una serie de rotaciones a la izquierda al código procedente del usuario iguales a n-1, siendo n la dimensión de los códigos de iris y por cada rotación se realiza el proceso descrito en el párrafo anterior y se van almacenando los valores obtenidos. Posteriormente se realiza el mismo proceso rotando el código hacia la derecha. Al haber obtenido todas las distancias de hamming posibles, se hace una búsqueda para encontrar el menor valor y este se toma para compararlo con el umbral de decisión. Un ejemplo de una parte del proceso antes descrito se observa en la siguiente imagen.

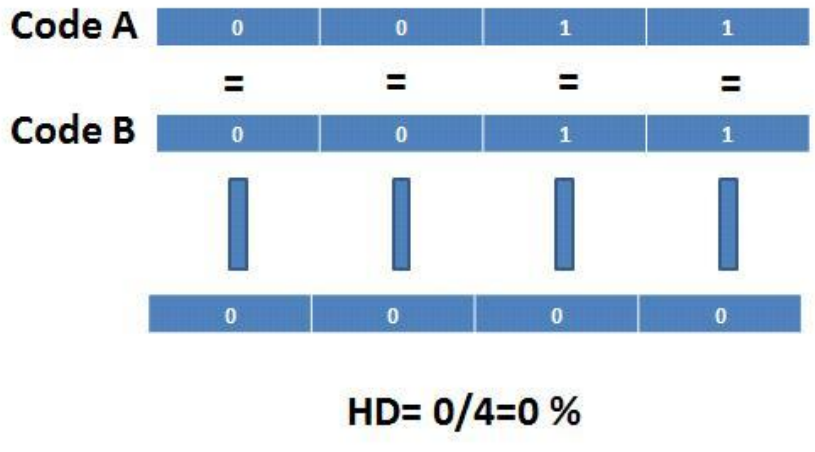

Imagen 15. Cálculo de Distancia de hamming rotando el código B una unidad hacia la derecha

Finalmente, el algoritmo se encarga de generar un fichero de texto con la información del usuario que se autenticó, el tipo de autenticación y el valor calculado.

3.3.1. Definición del umbral de decisión

Un elemento importante durante el proceso de identificación lo constituye el hecho de la toma de decisiones, a fin de poder definir exactamente si un usuario es quien dice ser, para lo cual se debe encontrar un umbral o patrón de decisión que sirva como base de este proceso.

Inicialmente para encontrar dicho patrón se debe realizar un número X de comparaciones entre códigos de iris pertenecientes a imágenes diferentes y proyectar las distancias de hamming resultantes en una gráfica de valores de hamming por el eje X por probabilidad de ocurrencia del valor por el eje Y. Finalmente mediante la gráfica se puede determinar el menor y mayor valor calculado que ha ocurrido para un X número de comparaciones (análisis de valores extremos) al igual que la probabilidad de ocurrencia de los valores más comunes para códigos diferentes. Este proceso también se puede aplicar para comparaciones entre códigos de iris correspondientes a la misma persona. Ahora en dependencia del sistema es que se determina el patrón de reconocimiento, en algunos sistemas puede ser de 0.32, en otros es de 0.33 como se puede ver en la siguiente gráfica:

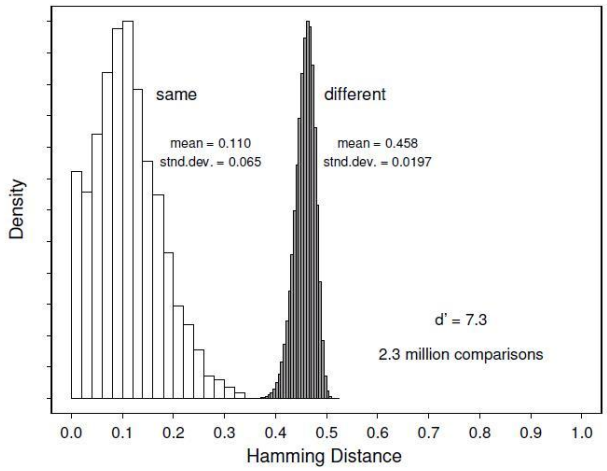

Imagen 16. Ambiente de decisión para reconocimiento de iris. (17)

Partiendo del resultado de 32 comparaciones entre códigos de iris correspondientes a iguales personas y a personas diferentes se realiza el mismo proceso de proyección de valores en una gráfica como se observa en la imagen a continuación.

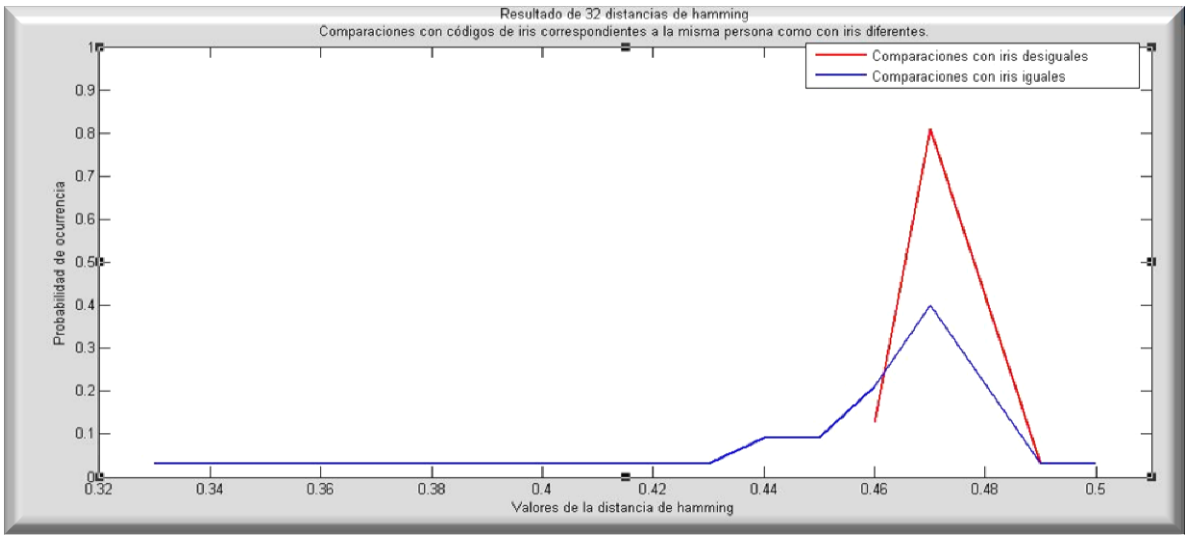

Imagen 17. Distribución de HDs a partir de 32 comparaciones entre diferentes pares de irises de una base de datos

En la imagen anterior la línea azul representa las comparaciones entres iris iguales y la línea roja entre iris diferentes. Como se observa, el menor valor de hamming alcanzado entre códigos diferentes es de 0.46 estableciéndose una zona de duda entre las distancias correspondientes entre 0.46-0.50.

Siguiendo este estudio se puede establecer un patrón de decisión que sea menor que 0.46, significando esto que entre comparaciones de códigos correspondientes a una misma persona que arrojen una distancia menor a 0.46 se puede considerar que son de la misma persona.

3.4. Pruebas

3.4.1. Técnicas de evaluación dinámica o prueba del software

La Prueba de software se puede definir como una actividad en la cual un sistema o uno de sus componentes se ejecuta en circunstancias previamente especificadas (configuración de la prueba), registrándose los resultados obtenidos. Seguidamente se realiza un proceso de evaluación en el que los resultados obtenidos se comparan con los resultados esperados para localizar fallos en el software. Estos fallos conducen a un proceso de depuración en el que es necesario identificar la falta asociada con cada fallo y corregirla, pudiendo dar lugar a una nueva prueba. Como resultado final se puede obtener una determinada Predicción de Fiabilidad, tal como se indicó anteriormente, o un cierto nivel de confianza en el software probado. (18)

Entre las técnicas de Prueba que existen se encuentran las pruebas de caja negra o pruebas funcionales, que realizan pruebas sobre la interfaz del programa a probar no siendo necesario conocer la lógica del programa, únicamente la funcionalidad que debe realizar.

La Imagen 18 representa gráficamente la filosofía de las pruebas de caja blanca y caja negra. Como se puede observar las pruebas de caja blanca necesitan conocer los detalles procedimentales del código,

mientras que las de caja negra únicamente necesitan saber el objetivo o funcionalidad que el código ha de proporcionar. (18)

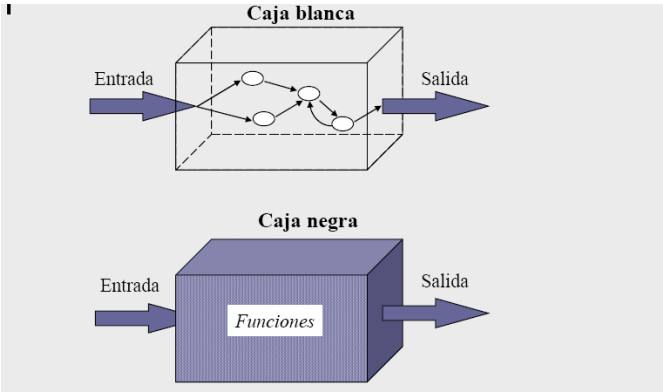

Imagen 18. Representación de pruebas de Caja Blanca y Caja Negra

Existen diferentes criterios de prueba de Caja Negra, siendo ellos:

- Particiones de Equivalencia.
- Análisis de Valores Límite.
- Métodos Basados en Grafos.
- Pruebas de Comparación.
- Análisis Causa-Efecto.

Entre las estrategias de prueba están las Pruebas del Sistema, que tienen como propósito ejecutar completamente el sistema para verificar la correcta integración de los diferentes componentes del mismo, sean estos de hardware o de software y que realizan las funciones adecuadas. Chequeando concretamente que:

- Se cumplen los requisitos funcionales establecidos.

- El funcionamiento y rendimiento de las interfaces hardware, software y de usuario.

- La adecuación de la documentación de usuario.

- Rendimiento y respuesta en condiciones límite y de sobrecarga.

Para la generación de casos de prueba de sistema se utilizan técnicas de caja negra. Este tipo de pruebas se suelen hacer inicialmente en el entorno del desarrollador, denominadas Pruebas Alfa, y seguidamente en el entorno del cliente denominadas Pruebas Beta. (18)

En el nivel de prueba del sistema, los detalles de las conexiones entre clases no afectan. El software debe integrarse con los componentes de hardware correspondientes y se ha de comprobar el funcionamiento del sistema completo acorde a los requisitos. Se centra en las acciones visibles del usuario y las salidas del sistema reconocibles por este. Para asistir en la determinación de casos de prueba de sistema, el ejecutor de la prueba debe basarse en los casos de uso que forman parte del modelo de análisis. El caso de uso brinda un escenario que posee una alta probabilidad con errores encubiertos en los requisitos de interacción del cliente. (18)

3.4.2. Descripción de los casos de prueba

A continuación se detalla el proceso de prueba definido a partir de la descripción de los casos de uso de las funcionalidades desarrolladas previamente sean estas Segmentar imagen, Codificar imagen y Normalizar imagen, junto con las incorporadas en el actual proceso de desarrollo; a fin de chequear el correcto funcionamiento del sistema bajo las diferentes condiciones a las que puede someterse y de eliminar errores ocurridos durante la fase de implementación. Este proceso arrojó una serie de 4 no conformidades, una de aplicación y tres de documentación, las cuales se documentaron y corrigieron siguiendo los estándares definidos por el CEDIN.

3.4.2.1. CUS Segmentar imagen

Escenario	Descripción	Respuesta del sistema	Flujo central
EC [Segmentar imagen Correctamente]	1. Crear un nuevo proyecto de C++ en eclipse indigo. 1.2 Incluir en las cabeceras del proyecto #include<iris/Parker>. 1.3 Copiar dentro del código del proyecto el código siguiente: FireFly * f; f=new FireFly("resources/Entrada/image-0",0); f->exec();. 1.5 Ejecutar el programa para ejecutar el método exec() de la clase Firefly. 1.12.Buscar el subdirectorio resources dentro de la carpeta de la aplicación. 1.13. Buscar el subdirectorio Segmentacion. 1.14. Chequear que dentro de la carpeta existe una fichero de imagen nombrado “Segmentada.jpg”	1.4 Crea un objeto del tipo Firefly y toma la dirección de donde cargar la imagen del a procesar 1.6. Obtiene una imagen del ojo del usuario. 1.7. Pedir la localización de la pupila. 1.8. Localizar la pupila. 1.9. Pedir la localización del iris. 1.10. Localizar el iris. 1.11. Notifica la segmentación del iris.	Debe existir algún sistema de captura o almacén de imágenes que proporciones la imagen a segmentar. Copiar dentro de la carpeta del proyecto creado la carpeta resources que se provee. Incluir la biblioteca en el eclipse. Seleccionar de resources-> Entrada-> image-0

Tabla 6. Caso de prueba CU Segmentar imagen

3.4.2.2. CUS Normalizar imagen

Escenario	Descripción	Respuesta del sistema	Flujo central
EC 1 [Normalizar imagen Correctamente]	1. Crear un nuevo proyecto de C++ en eclipse indigo. 1.2 Incluir en las cabeceras del proyecto #include<iris/Firefly>. 1.3 Copiar dentro del código del proyecto el código siguiente: FireFly * f; f=new FireFly("resources/Entrada/image-0",0); f->exec();. 1.5 Ejecutar el programa para ejecutar el método exec() de la clase Firefly. 1.11. Buscar el subdirectorio Resources dentro de la carpeta de la aplicación. 1.12. Buscar el subdirectorio Normalizacion. 1.13 Chequear que dentro de la carpeta existe una fichero de imagen nombrado “Normalizada.jpg”	1.4 Crea un objeto del tipo Firefly y toma la dirección de donde cargar la imagen del a procesar. 1.6. Pide la segmentación de la imagen de un ojo del usuario. 1.7. Carga una imagen segmentada. 1.8. Pide la conversión del anillo del iris obtenido a una imagen rectangular. 1.9. Convierte la región del iris en una imagen rectangular. 1.10. Notifica la normalización del iris.	Copiar dentro de la carpeta del proyecto creado la carpeta resources que se provee. Incluir la biblioteca en el eclipse. Debe existir una imagen denominada “Segmentada.jpg” dentro del subdirectorio resources->Segmentación. Seleccionar de resources-> Entrada-> image-0

Tabla 7. Caso de prueba para CU Normalizar imagen

3.4.2.3. CUS Codificar imagen

Escenario	Descripción	Respuesta del sistema	Flujo central
EC 1 [Codificar imagen Correctamente]	1. Crear un nuevo proyecto de C++ en eclipse indigo. 1.2 Incluir en las cabeceras del proyecto #include<iris/Firefly> . 1.3 Copiar	1.4 Crea un objeto del tipo Firefly y toma la dirección de donde cargar la	Copiar dentro de la carpeta del proyecto creado la carpeta resources que se provee.

	dentro del código del proyecto el código siguiente: FireFly * f; f=new FireFly("resources/Entrada/image-0",0); f->exec();. 1.5 Ejecutar el programa para ejecutar el método exec() de la clase Firefly. 1.10. Buscar el subdirectorio Resources dentro de la carpeta de la aplicación. 1.11. Buscar la carpeta Codificación. 1.12. Chequear que dentro de la carpeta existe un fichero de imagen nombrado “Codificada.jpg”	imagen del a procesar. 1.6 Normaliza una imagen segmentada. 1.7 Almacena una imagen normalizada. 1.8 Pide el iriscode asociado a la imagen. 1.9 Notifica la codificación del iris.	Incluir la biblioteca en el eclipse. Debe existir una imagen denominada “Normalizada.jpg” dentro del subdirectorio resources->Normalización. Seleccionar de resources-> Entrada-> image-0

Tabla 8. Caso de prueba para CU Codificar imagen

3.4.2.4. CUS Calcular distancia de hamming

Escenario	Descripción	Respuesta del sistema	Flujo central
EC 1Calcular distancia de hamming	1. Crear un nuevo proyecto de C++ en eclipse indigo. 1.2 Incluir en las cabeceras del proyecto #include<iris/Parker> y	1.5 Crea un objeto de tipo Parker y carga los códigos de iris.1.7. Calcula y	Copiar dentro de la carpeta del proyecto creado la carpeta resources que se provee. Incluir la biblioteca en

	#include<fstream>. 1.3 Copiar en el código, el algoritmo de carga de códigos de iris que se provee. 1.4Copiar dentro del código del proyecto el código siguiente: Parker * p; p=new Parker(ccode1,ccode2); .1.6 Crear una variable double para almacenar el valor que devuelve el método calculateHammingDistance() de la interfaz Parker e invocarlo.1.12 Mostrar por consola el valor de la variable.1.13 Chequear que se muestra en la consola un valor entre 0 y 1	almacena el nivel de diferencia entre los códigos. 1.8 Va rotando el segundo código a la izquierda. Calcula y almacena el nivel de diferencia entre los códigos por cada rotación. 1.9 Va rotando el segundo código a la derecha. Calcula y almacena el nivel de diferencia entre los códigos por cada rotación. 1.10Selecciona el menor nivel de diferencia entre los códigos calculado. 1.11. Devuelve el menor nivel de diferencia entre los códigos	el eclipse. Acceder adentro de la carpeta de la aplicación, al subdirectorio “resources”->subdirectorio “Iriscode”. Chequear que dentro de la carpeta Iriscode estén los ficheros image-0 e image-1 que son códigos de iris generados por la biblioteca.

Tabla 9. Caso de prueba para CU Calcular distancia de hammig

3.4.2.5. CUS Generar reporte

Escenario	Descripción	Variable 1	Variable 2	Respuesta del sistema	Flujo central
EC 1 Generar reporte de autenticación satisfactoria correctamente	1. Crear un nuevo proyecto de C++ en eclipse indigo. 1.2 Incluir en las cabeceras del proyecto #include<iris/Parker>. 1.3 Copiar en el código el	V	V	1.5 Crea un objeto de tipo Parker y carga los códigos de iris. 1.7	Copiar dentro de la carpeta del proyecto creado la carpeta
		12	23		

	algoritmo de carga de códigos de iris que se provee. 1.4 Crear un objeto del tipo Parker e inicializarlo con los códigos de iris que se obtienen.1.6. Invocar al método generateReport pasándole por parámetro (identificador del usuario, identificador del reporte. 1.11 Chequear que en el subdirectorio “resources”->subdirectorio “Reports”-> Subdirectorio “SuccefullAutentication” se crea un fichero de texto con la siguiente estructura de título: “user-id del usuario-id del reporte.txt”. 1.12. Chequear que en el subdirectorio “resources”->subdirectorio “Reports”-> Subdirectorio “GeneralReport” se crea un fichero de texto con la siguiente estructura de título: “user-id del usuario-id del reporte.txt”.			Valida los datos: correctos. 1.7. Calcula la distancia de hamming. 1.8 Genera un reporte general. 1.9 Compara la distancia de hamming con un valor de decisión (distancia de hamming menor que el umbral de decisión). 1.10 Genera un reporte de autenticación satisfactoria.	resources que se provee. Incluir la biblioteca en el eclipse. Acceder adentro de la carpeta de la aplicación, al subdirectorio “resources”->subdirectorio “Iriscode”. Chequear que dentro de la carpeta Iriscode estén los ficheros image-0 e image-1 que son códigos de iris generados por la biblioteca.
EC 2 Generar reporte de autenticación insatisfactoria correctamente	2. Crear un nuevo proyecto de C++ en eclipse indigo. 2.2 Incluir en las cabeceras del proyecto #include<iris/Parker>. 2.3 Copiar en el código, el algoritmo de carga de códigos de iris que se provee. 2.4 Cambiar en el algoritmo de carga las líneas de código iguales a: ifstream code1("resources/Iriscode/image-0"); por ifstream code1("resources/Iriscode/image-2"); y ifstream code1("resources/Iriscode/im	V	V	2.6 Crea un objeto de tipo Parker y carga los códigos de iris. 2.8 Valida los datos: correctos. 2.9. Calcula la distancia de hamming. 2.10 Genera un reporte general.	Copiar dentro de la carpeta del proyecto creado la carpeta resources que se provee. Incluir la biblioteca en el eclipse. Acceder adentro de la carpeta de la

	age-1"); por ifstream code1("resources/Iriscode/image-3"); 2.5 Crear un objeto del tipo Parker e inicializarlo con los códigos de iris que se obtienen.2.7. Invocar al método generateReport pasándole por parámetro (identificador del usuario, identificador del reporte. 2.13 Chequear que en el subdirectorio “resources”->subdirectorio “Reports”-> Subdirectorio “InvalidAutentication” se crea un fichero de texto con la siguiente estructura de título: “user-id del usuario-id del reporte.txt”. 2.14. Chequear que en el subdirectorio “resources”->subdirectorio “Reports”-> Subdirectorio “GeneralReport” se crea un fichero de texto con la siguiente estructura de título: “user-id del usuario-id del reporte.txt”.	14	26	2.11 Compara la distancia de hamming con un valor de decisión (distancia de hamming mayor o igual que el umbral de decisión). 2.12 Genera un reporte de autenticación insatisfactoria.	aplicación, al subdirectorio “resources”->subdirectorio “Iriscode”. Chequear que dentro de la carpeta Iriscode estén los ficheros image-0 e image-1 que son códigos de iris generados por la biblioteca.
EC 3 Generar reporte incorrectamente (identificador de usuario incorrecto)	3. Crear un nuevo proyecto de C++ en eclipse indigo.3.2 Incluir en las cabeceras del proyecto #include<iris/Parker>. 3.3 Copiar en el código, el algoritmo de carga de códigos de iris que se provee. 3.4 Crear un objeto del tipo Parker e inicializarlo con los códigos de iris que se obtienen.3.6. Invocar al método generateReport pasándole por parámetro (identificador del usuario, identificador del reporte.3.9 Chequear que en el subdirectorio “resources”->subdirectorio “Reports”->No se crea ningún archivo de reporte en los subdirectorios GeneralReport, InvalidAutentication, SuccefulAutentication	I 0	V 25	3.5 Crea un objeto de tipo Parker y carga los códigos de iris. 3.7 Valida los datos: incorrectos. 3.8.No se realiza más ninguna acción	Copiar dentro de la carpeta del proyecto creado la carpeta resources que se provee. Incluir la biblioteca en el eclipse. Acceder adentro de la carpeta de la aplicación, al subdirectorio “resources”->subdirectorio “Iriscode”. Chequear que dentro de la carpeta

					Iriscode estén los ficheros image-0 e image-1 que son códigos de iris generados por la biblioteca.
EC 4 Generar reporte incorrectamente(identificador de reporte incorrecto)	4. Crear un nuevo proyecto de C++ en eclipse indigo. 4.2 Incluir en las cabeceras del proyecto #include<iris/Parker>. 4.3 Copiar en el código, el algoritmo de carga de códigos de iris que se provee. 4.4 Crear un objeto del tipo Parker e inicializarlo con los códigos de iris que se obtienen.1.6. Invocar al método generateReport pasándole por parámetro (identificador del usuario, identificador del reporte. 4.9 Chequear que en el subdirectorio “resources”->subdirectorio “Reports”->No se crea ningún archivo de reporte en los subdirectorios GeneralReport, InvalidAutentication, SuccefulAutentication	V 25	I -1	4.5 Crea un objeto de tipo Parker y carga los códigos de iris. 4.7 Valida los datos: incorrectos. 4.8.No se realiza más ninguna acción	Copiar dentro de la carpeta del proyecto creado la carpeta resources que se provee. Incluir la biblioteca en el eclipse. Acceder adentro de la carpeta de la aplicación, al subdirectorio “resources”->subdirectorio “Iriscode”. Chequear que dentro de la carpeta Iriscode estén los ficheros image-0 e image-1 que son códigos de iris generados por la biblioteca.

Tabla 10. Caso de prueba para CU Generar reporte

3.5. Empaquetado

El empaquetado es un proceso que tiene como objetivo obtener un fichero que agrupe todos componentes funcionales de una aplicación, biblioteca de códigos y código fuente, listos para ser instalados en cualquier ordenador. En los sistemas operativos basados en la distribución de GNU/Linux Debian, los ficheros con la extensión .deb constituyen los paquetes que contienen la instalación de aplicaciones, bibliotecas y código fuente como se mencionaba anteriormente. En estos sistemas operativos, con el comando dpkg-deb –b [directorio] [directorio o nombre que se desea que tenga el archivo resultante seguido de la expresión ".deb"] se obtiene un archivo de instalación.

Para poder realizar el compactado primeramente se debe crear una carpeta o directorio que va a contener el software. Dentro de esta carpeta, que se puede titular hipotéticamente Empaquetar, hay que crear la estructura de fichero que va a tener el software desarrollado cuando se instale por completo en el sistema, en este caso, se crea la carpeta la carpeta usr, dentro de esta carpeta se crean las carpetas lib e include. La carpeta lib contiene todos los archivos de ejecución de la biblioteca, dígase los archivos libiris.a, libiris.la, libiris.so y libiris -1.0.0.so; también dentro de esta carpeta se encuentra otro subdirectorio titulado pkgconfig que va a contener al fichero libiris.pc. La carpeta include va a contener a la carpeta iris, la que a su vez constituye el directorio donde la biblioteca va a tener todos los archivos cabecera de la biblioteca.

Junto a la carpeta usr hay que crear una carpeta titulada DEBIAN. Esta carpeta va a contener a un archivo titulado control. El archivo control, constituye el archivo que contiene las reglas por las cuales se va a guiar el comando dpkg-deb –b para realizar el proceso de empaquetado del software. Dentro del mismo se tiene que especificar:

Package: nombre del paquete.

Version: versión.

Architecture: arquitectura.

Maintainer: nombre del que crea el paquete <dirección de correo>.

Depends: bibliotecas de las que depende y las versiones.

Section: sección del sistema a la que pertenece.

Priority: prioridad.

Description: Breve descripción de la función del paquete que se va a crear.

Posterior a realizar este proceso, se procede a acceder por la terminar de comandos del sistema operativo al directorio donde se encuentra la carpeta del empaquetado y se invocan los comandos del empaquetado, obteniéndose finalmente el paquete libiris1.0.deb que contiene la versión inicial de la biblioteca desarrollada.

3.6. Validación de la solución

Actualmente los sistemas de reconocimiento de iris que existen en el mercado desarrollan el hardware con el cual va a trabajar su software, limitación que afecta a este sistema y que impide concluir que la biblioteca desarrollada es de un cien por ciento segura.

Mediante el empleo de bases de datos de iris libres, se realizaron una serie de comparaciones con códigos de iris pertenecientes a una misma persona en cada consulta, a fin de poder validar la fiabilidad del componente desarrollado. Como se puede observar en la imagen 19, a partir de 25 consultas realizadas con la base de iris MMU Iris Database se obtuvieron 16 autenticaciones válidas con solamente 9 autenticaciones inválidas, lo cual define un 64 % de casos positivos contra un 36 % de casos negativos.

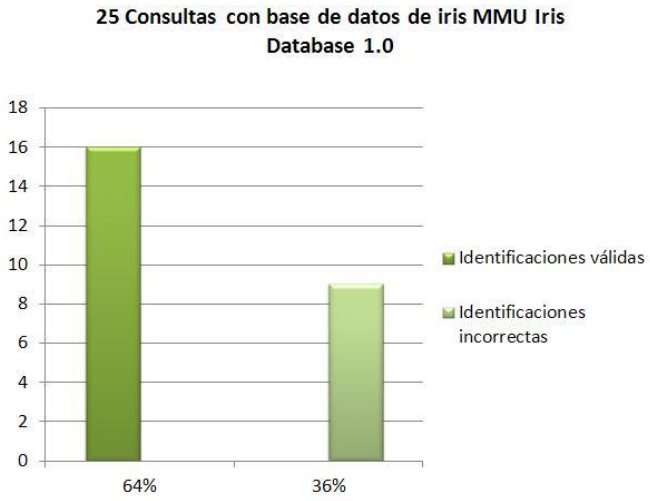

Imagen 19. Nivel de exactitud de la autenticación empleando la base de datos MMU Iris Database 1.0

Como se puede observar en la imagen 20, a partir de 63 consultas realizadas con la base de iris CASIA Iris Database se obtuvieron 43 autenticaciones válidas con solamente 20 autenticaciones inválidas, lo cual define un 68% de casos positivos contra un 32% de casos negativos.

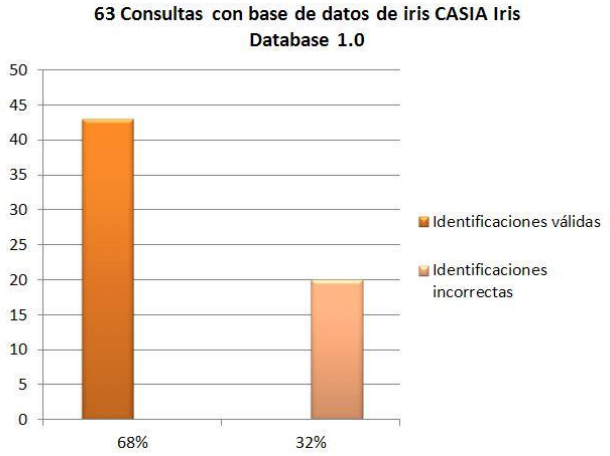

Imagen 20. Nivel de exactitud de la autenticación empleando la base de datos CASIA Iris Database 1.0

Finalmente para un total de 88 consultas realizadas entre las bases de datos de iris se pudo verificar un total de 59 autenticaciones satisfactorias por 29 de autenticaciones inválidas para un 67% de valores esperados contra solamente un 33% de casos insatisfactorios, lo cual nos demuestra la validez de la solución desarrollada como se observa en la imagen siguiente.

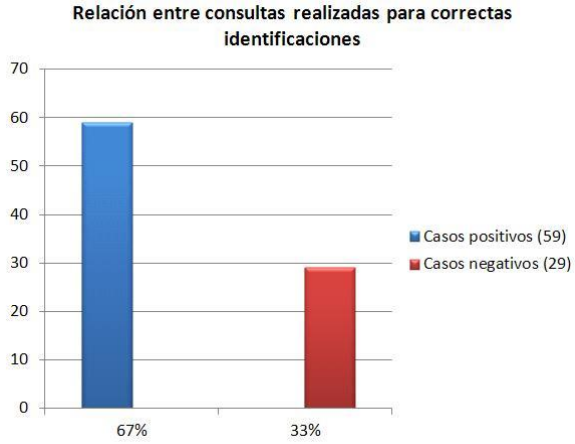

Imagen 21. Relación entre identificaciones correctas e incorrectas para un total de 88 consultas realizadas entre las bases de datos de iris MMU y CASIA

En la investigación titulada Implementación de un Sistema de identificación de personas en tiempo real por reconocimiento de iris (11) se emplean al igual que en la presente investigación la distancia de hamming como método de comparación de códigos de iris y las bases de datos CASIA y MMU obteniéndose entre ambas bases de datos un 97% de casos positivos contra solamente un 3 % de casos negativos. Comparando los resultados obtenidos por cada investigación, la biblioteca desarrollada es menos óptima en materia de resultados positivos, siempre analizado que se compara una biblioteca contra un sistema de reconocimiento de iris, cuya diferencia se explicaba en capítulos anteriores y se compara

con un software que ya compite en el mercado. Aparentemente estos resultados pueden ser muy pobres, pero analizando el caso de que esta constituye la versión inicial de una biblioteca de reconocimiento de iris en un ambiente donde nunca se había realizado un componente de este tipo y no se cuenta con una versión anterior donde se puedan comparar resultados anteriores con los presentes, los resultados alcanzados se consideran satisfactorios para esta etapa alcanzada. Además, permiten tener un resultado estadístico que se puede utilizar para análisis contra resultados arrojados en nuevas versiones de la propia biblioteca donde se deben ir obteniendo resultados más alentadores.

3.7. Conclusiones.

Los estilos de codificación y documentación explicados en este capítulo constituyen los estándares aplicados en el CEDIN lo cual justifica el empleo de los mismos en la biblioteca que se desarrolló. Además, se expresan los resultados obtenidos a partir de la aplicación del proceso de prueba descrito donde de las 4 no conformidades detectadas se pudo corregir la totalidad de las mismas. También se describe el conjunto de pruebas realizadas a la biblioteca a partir de poder determinar su fiabilidad entre diferentes base de datos de iris, demostrándose así la validez del proceso de cálculo de la distancia de hamming y la determinación del umbral de decisión para lograr una correcta autenticación; lo cual permite obtener finalmente una biblioteca portable y que pueda ser comprendida por usuarios comunes sin necesidad de tener un amplio conocimiento respecto al tema.

Conclusiones generales

Con el objetivo de obtener un producto que satisfaga los requerimientos de una biblioteca de reconocimiento de patrones del iris, se logró dar cumplimiento a todas las tareas investigativas propuestas, por lo que al culminar este proceso investigativo y de desarrollo, se puede concluir lo siguiente:

- Se logró desarrollar una versión inicial para una biblioteca de autenticación biométrica por reconocimiento de patrones del iris.

Recomendaciones

A continuación se recomiendan posibles aspectos a tener en cuenta para continuar con la mejora y perfeccionamiento de la biblioteca:

- Realización de un proceso de prueba de caja blanca en cada módulo por separado a fin de obtener un mejor resultado en cada etapa de la biblioteca.

- Realización de pruebas de campo a fin de determinar el comportamiento de la biblioteca en un ambiente real.

Referencias Bibliográficas

1. informática, Colectivo de profesores principales de la asignatura: Seguridad. **Introducción a la seguridad informatica. Conferencia #1. Universidad de las Ciencias Informáticas. Habana.Cuba : s.n., 2011.**

2. **Mayans, Leonardo Sena.** *Seguridad informática.* junio 2000. pág. http://www.ccee.edu.uy/ensenian/catcomp/material/Inform_%20II/SEGURIDAD2.pdf.

3. **Belmonte, Rafael Coomonte.** *Sistema de reconocimiento de personas mediante su patrón de iris basado en la transformada wavelet.* 26 de Mayo de 2006 .

4. **Gabaudan, Francisco Cortés.** *Diccionario médico etimológico.* 28/02/2000.

5. **González, Ariel Peláez.** *Algoritmo de codificación para biblioteca de autenticación biométrica a partir del reconocimiento de patrones del iris.* Habana,Cuba : s.n., 2011.

6. **Urbano, Marcos González.** Reconocimiento de iris. [En línea]

7. **services, New York state. Office of general.** New York state. Office of general services. [En línea] 26 de 1 de 2011. [Citado el: 28 de 5 de 2012.] http://www.ogs.state.ny.us/purchase/prices/7720120191PL_Adirondack.pdf.

8. **Martín, Raquel.** Tecnocosas. [En línea] [Citado el: 24 de 5 de 2012.] http://www.tecnocosas.es/lector-de-reconocimiento-de-iris-de-panasonic/.

9. **Laura Florian Cruz, Fredy Carranza Athó.** *RECONOCIMIENTO DEL IRIS.* Trujillo. Perú : Tópicos especiales en procesamiento gráfico. Escuela académico profesional de informática Universidad Nacional de Trujillo, 2006.

10. *Sistema de reconocimiento de iris.* **Lucas D. Terissi, Lucas Cipollone y Patricio Baldino.** 2, Rosario. Argentina : Revista argentina de trabajos estudiantiles, Marzo 2006, Vol. I. TRATE06-001.

11. **Mottalli, Marcelo Luis.** *Implementación de un sistema de identificación de personas en tiempo real por reconocimiento de iris.* Buenos Aires : s.n., Octubre de 2008.

12. **Abascal, Alejandro González.** Algoritmo de segmentación para biblioteca de autenticación biométrica a partir del reconocimiento del iris. Habana. Cuba : s.n., 2010.

13. **Heidy Alina Nuevo Leon, Alejandro Manuel Rubinos Carvajal, Enrique Reyes Bermudez,Claudia Maria Lora Pomar, Yisel Delgado Mesa.** Algoritmo de detección facial para sistemas de autenticación biométrica. Habana. Cuba : s.n., Septiembre 2010.

14. **Ivar Jacobson, Grady Booch, James Rumbaugh.** El proceso unificado de desarrollo de software. *Entorno virtual de aprendizaje.* [En línea] 2000. [Citado el: 9 de 5 de 2012.] http://eva.uci.cu/mod/resource/view.php?id=8500&subdir=/El_Proceso_Unificado_de_Desarrollo. 84-7829-036-2.

15. **Quesada, Juan Antonio López.** Web Personal: Juan Antonio López Quesada. I.E.S. San Juan Bosco (Lorca-Murcia). Departamento de Informática. Universidad de Murcia. [En línea] [Citado el: 14 de 5 de 2012.] http://dis.um.es/~lopezquesada/documentos/Tema%2013.ppt.

16. **autores., C. d. (2011, Enero). Tycon Software Ingineering. Recuperado el 10 de Marzo de 2011, from Tycon Software Ingineering:.** [En línea] http://www.tycon.com.ar/Tecnolog%C3%ADa/tabid/58/Default.aspx.

17. **Works, How Iris Recognition.** Computer Laboratory University of Cambridge. [En línea] [Citado el: 2012 de 4 de 3.] http://www.CL.cam.ac.uk/users/jgd1000/.

18. **Natalia Juristo, Ana M. Moreno, Sira Vegas (Técnicas de evaluación de software).** Entonrno Virtual de Aprendizaje. [En línea] 17 de Octubre de 2005. [Citado el: 15 de Marzo de 2012.] http://eva.uci.cu/mod/resource/view.php?id=8649&subdir=/Comun.

Bibliografía

1. **informática, Colectivo de profesores principales de la asignatura: Seguridad.** Introducción a la seguridad informatica. Conferencia #1. Universidad de las Ciencias Informáticas. Habana.Cuba : s.n., 2011.

2. **Mayans, Leonardo Sena.** *Seguridad informática.* junio 2000. pág. http://www.ccee.edu.uy/ensenian/catcomp/material/Inform_%20II/SEGURIDAD2.pdf.

3. **Belmonte, Rafael Coomonte.** *Sistema de reconocimiento de personas mediante su patrón de iris basado en la transformada wavelet.* 26 de Mayo de 2006 .

4. **Gabaudan, Francisco Cortés.** *Diccionario médico etimológico.* 28/02/2000.

5. **González, Ariel Peláez.** *Algoritmo de codificación para biblioteca de autenticación biométrica a partir del reconocimiento de patrones del iris.* Habana,Cuba : s.n., 2011.

6. **Urbano, Marcos González.** Reconocimiento de iris. [En línea]

7. **services, New York state. Office of general.** New York state. Office of general services. [En línea] 26 de 1 de 2011. [Citado el: 28 de 5 de 2012.] http://www.ogs.state.ny.us/purchase/prices/7720120191PL_Adirondack.pdf.

8. **Martín, Raquel.** Tecnocosas. [En línea] [Citado el: 24 de 5 de 2012.] http://www.tecnocosas.es/lector-de-reconocimiento-de-iris-de-panasonic/.

9. **Laura Florian Cruz, Fredy Carranza Athó.** *RECONOCIMIENTO DEL IRIS.* Trujillo. Perú : Tópicos especiales en procesamiento gráfico. Escuela académico profesional de informática Universidad Nacional de Trujillo, 2006.

10. *Sistema de reconocimiento de iris.* **Lucas D. Terissi, Lucas Cipollone y Patricio Baldino.** 2, Rosario. Argentina : Revista argentina de trabajos estudiantiles, Marzo 2006, Vol. I. TRATE06-001.

11. **Mottalli, Marcelo Luis.** *Implementación de un sistema de identificación de personas en tiempo real por reconocimiento de iris.* Buenos Aires : s.n., Octubre de 2008.

12. **Abascal, Alejandro González.** Algoritmo de segmentación para biblioteca de autenticación biométrica a partir del reconocimiento del iris. Habana. Cuba : s.n., 2010.

13. **Heidy Alina Nuevo Leon, Alejandro Manuel Rubinos Carvajal, Enrique Reyes Bermudez,Claudia Maria Lora Pomar, Yisel Delgado Mesa.** Algoritmo de detección facial para sistemas de autenticación biométrica. Habana. Cuba : s.n., Septiembre 2010.

14. **Ivar Jacobson, Grady Booch, James Rumbaugh.** El proceso unificado de desarrollo de software. *Entorno virtual de aprendizaje.* [En línea] 2000. [Citado el: 9 de 5 de 2012.] http://eva.uci.cu/mod/resource/view.php?id=8500&subdir=/El_Proceso_Unificado_de_Desarrollo. 84-7829-036-2.

15. **Quesada, Juan Antonio López.** Web Personal: Juan Antonio López Quesada. I.E.S. San Juan Bosco (Lorca-Murcia). Departamento de Informática. Universidad de Murcia. [En línea] [Citado el: 14 de 5 de 2012.] http://dis.um.es/~lopezquesada/documentos/Tema%2013.ppt.

16. **autores., C. d. (2011, Enero). Tycon Software Ingineering. Recuperado el 10 de Marzo de 2011, from Tycon Software Ingineering:.** [En línea] http://www.tycon.com.ar/Tecnolog%C3%ADa/tabid/58/Default.aspx.

17. **Works, How Iris Recognition.** Computer Laboratory University of Cambridge. [En línea] [Citado el: 2012 de 4 de 3.] http://www.CL.cam.ac.uk/users/jgd1000/.

18. **Natalia Juristo, Ana M. Moreno, Sira Vegas (Técnicas de evaluación de software).** Entonrno Virtual de Aprendizaje. [En línea] 17 de Octubre de 2005. [Citado el: 15 de Marzo de 2012.] http://eva.uci.cu/mod/resource/view.php?id=8649&subdir=/Comun.

19. **homepage, Iris recongnition.** Iris recongnition homepage. [En línea] [Citado el: 14 de 3 de 2012.] http://pesona.mmu.edu.my/~ccteo/MMU%20Iris%20Database.rar.

20. **software, Web store.** Biometría y Seguridad. [En línea] 2011. [Citado el: 30 de 9 de 2011.] http://www.biometriayseguridad.com/index.php?dispatch=categories.catalog.

21. *DCT-Based Iris Recognition.* **Donald M. Monro, Soumyadip Rakshit, Dexin Zhang.** 4, s.l. : IEEE Transactions on pattern analysis and machine intelligence, APRIL 2007, Vol. 29. 586.

22. **Mayans, Leonardo Sena.** *Seguridad informática.* julio 2000.

23. **informática, Confeccionado por colectivo de profesores principales de la asignatura: Seguridad.** *ACTIVIDAD #17: Control de acceso. Identificación, autenticación y autorización.* La Habana,Cuba : s.n., mayo de 2010.

24. **César González Ferreras, Valentín Cardeñoso Payo y Carlos Vivaracho Pascual.** *IV Jornada de reconocimiento biométrico de personas.* E.T.S. de Ingeniería informática Universidad de Valladolid : Mata Digital, S.L., Septiembre de 2008. ISBN: 978-84-691-5008-5.

Glosario de Términos

A

- Actor:

- Aplicación: Terminología técnica para referirse a un programa de software.

- Arquitectura: Indica la estructura, funcionamiento e interacción entre las partes del software.

- Autenticación: Verificación de la identidad de una persona o de un proceso para acceder a un recurso o poder realizar determinada actividad.

C

- CEDIN: Siglas que identifican el Centro de Informática Industrial de la facultad 5 de la Universidad de las Ciencias Informáticas.

- Cliente: Un sistema o proceso que solicita a otro sistema o proceso que le preste un servicio.

- Contraseña: Información confidencial constituida por una cadena de caracteres, usada para la autenticación de un usuario o en el acceso a un recurso.

- Control de acceso: Mecanismo que en función de la identificación ya autenticada permite acceder a datos o recursos.

H

- Hardware: Componente físico de una computadora o de una red, en contraposición con los programas o elementos lógicos que lo hacen funcionar.

I

- IDE: Siglas en inglés que identifican un Entorno de Desarrollo Integrado (*Integrated Development Environment)*. Es un entorno de programación que ha sido empaquetado como un programa de aplicación, es decir, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica.

- Identificación del usuario: Procedimiento de reconocimiento de la identidad de un usuario.

K

- Keyloggers: Programas registradores de teclas capaces de reproducir las teclas oprimidas por un usuario en una computadora.

P

- Programa: Conjunto de instrucciones que una vez ejecutadas realizan una o varias tareas en una computadora.

R

- Recurso: Cualquier parte componente de un sistema de información.

S

- Software: Conjunto de programas, documentos, procesamientos y rutinas asociadas con la operación de un sistema de computadoras, es decir, la parte intangible o lógica de una computadora.

U

- UCI: Universidad de las Ciencias Informáticas.

- Usuario: Sujeto o proceso autorizado para acceder a datos o recursos.

Anexos

Anexo 1: Imágenes de los resultados arrojados por la biblioteca

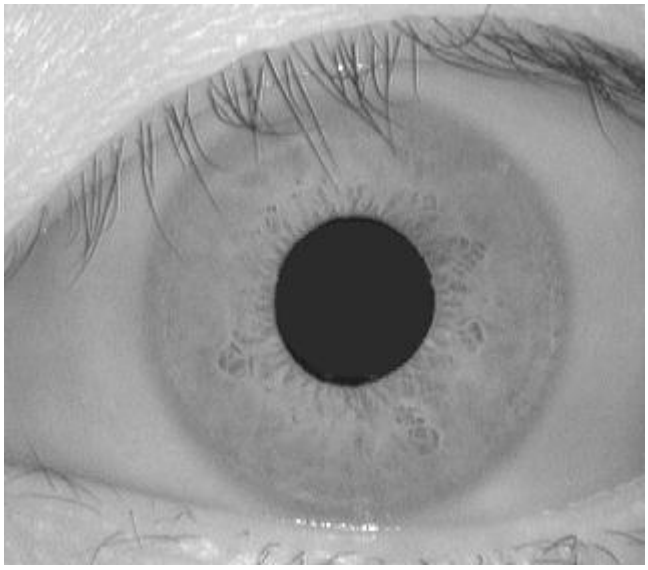

Imagen 22. Imagen de entrada.

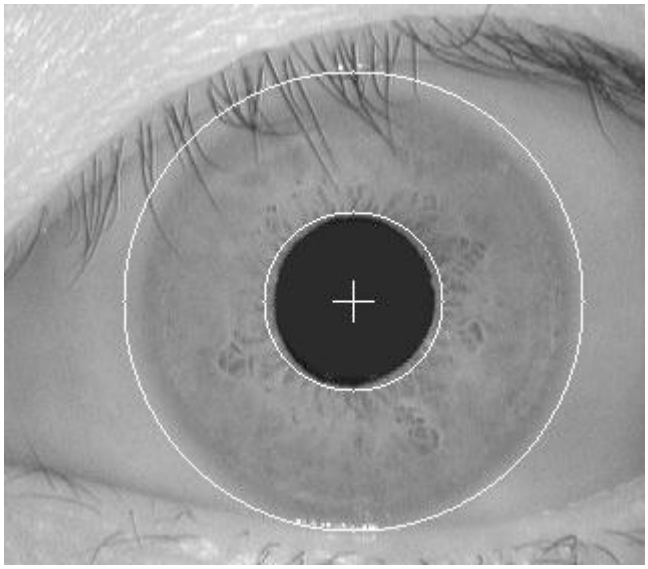

Imagen 23. Imagen segmentada.

Imagen 24. Imagen normalizada.

Imagen 25. Imagen codificada.

Imagen 26. Códigos de iris.

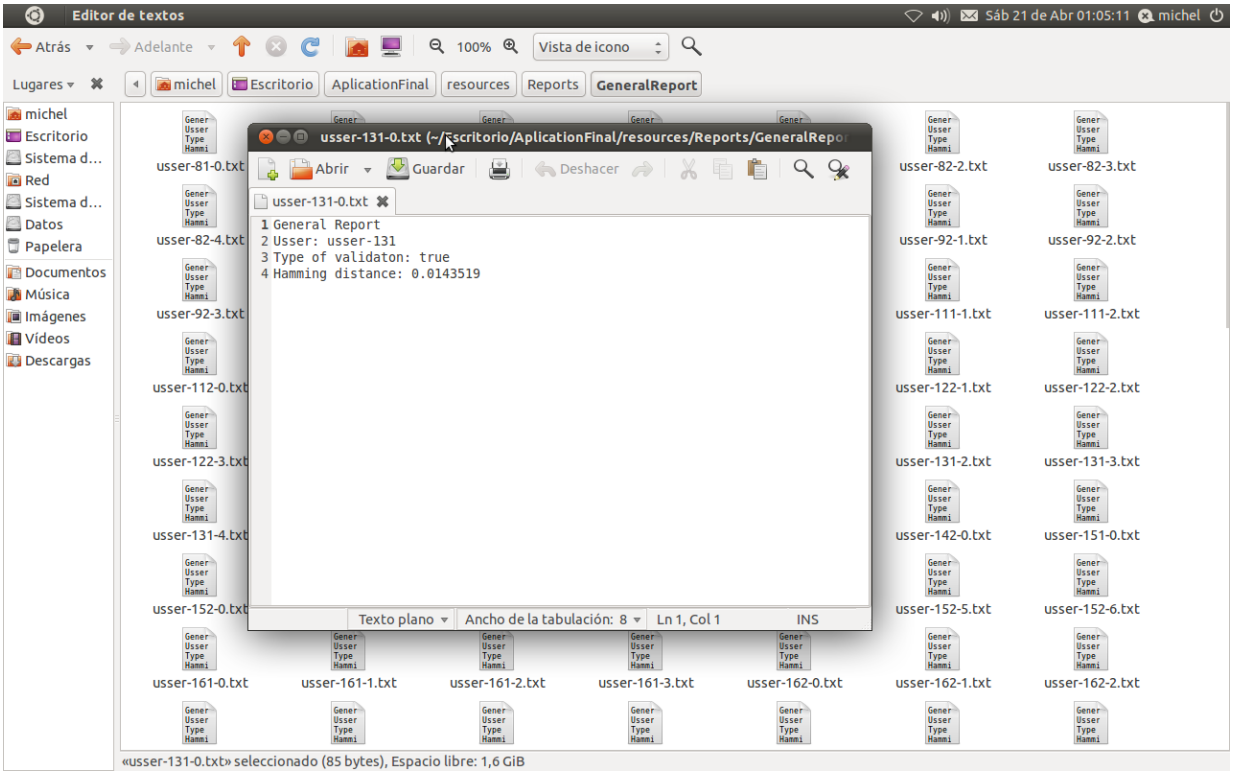

Imagen 27. Reporte general

Anexo2. Vistas de la aplicación

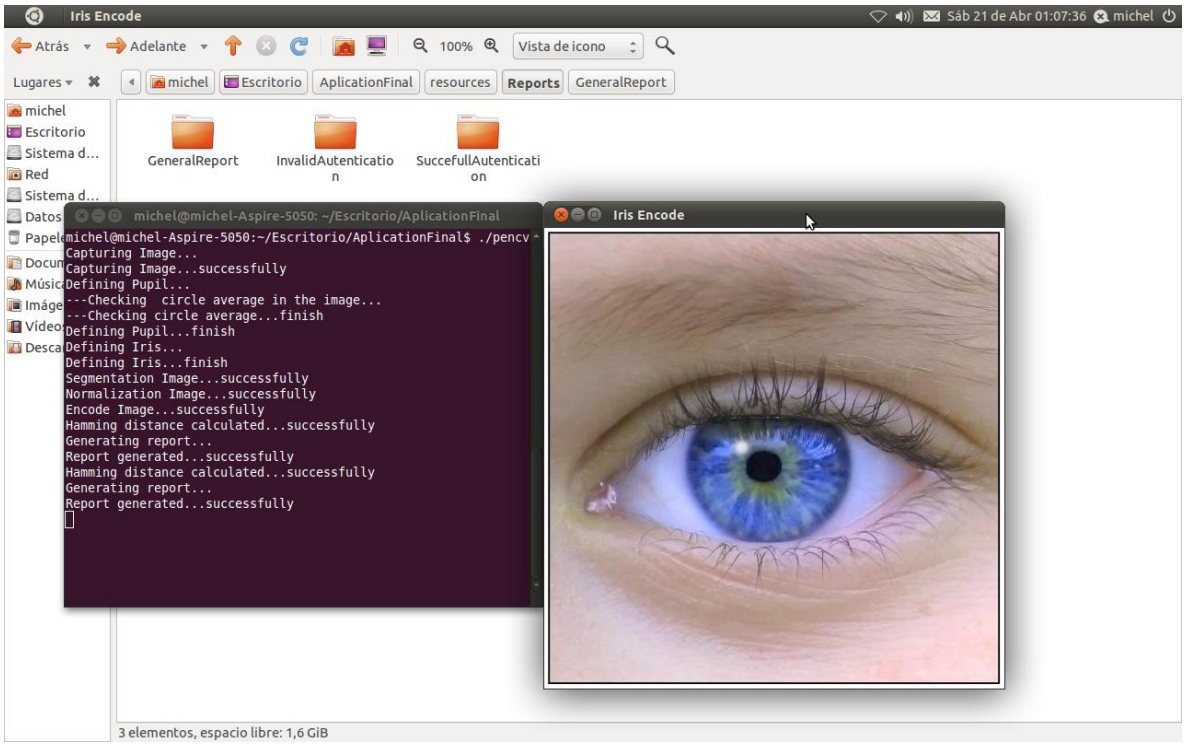

Imagen 28. Vista general de la biblioteca con la interfaz de usuario

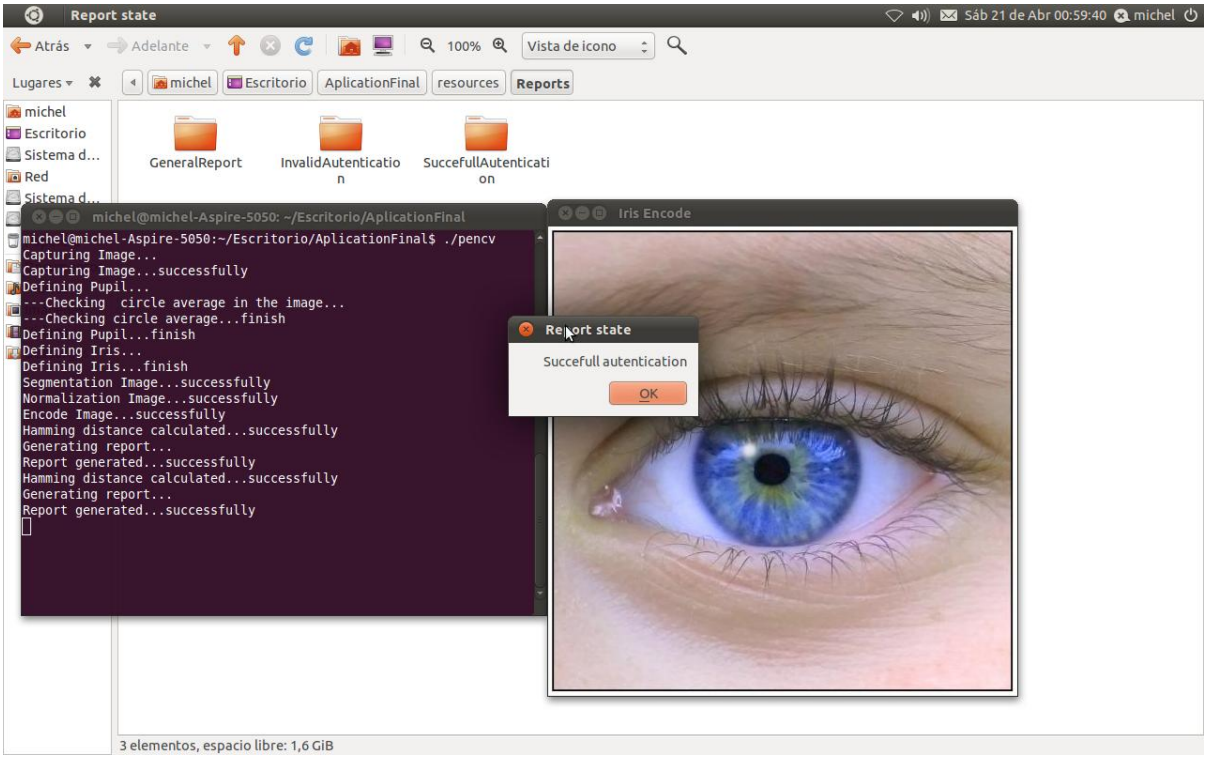

Imagen 29. Autenticación satisfactoria

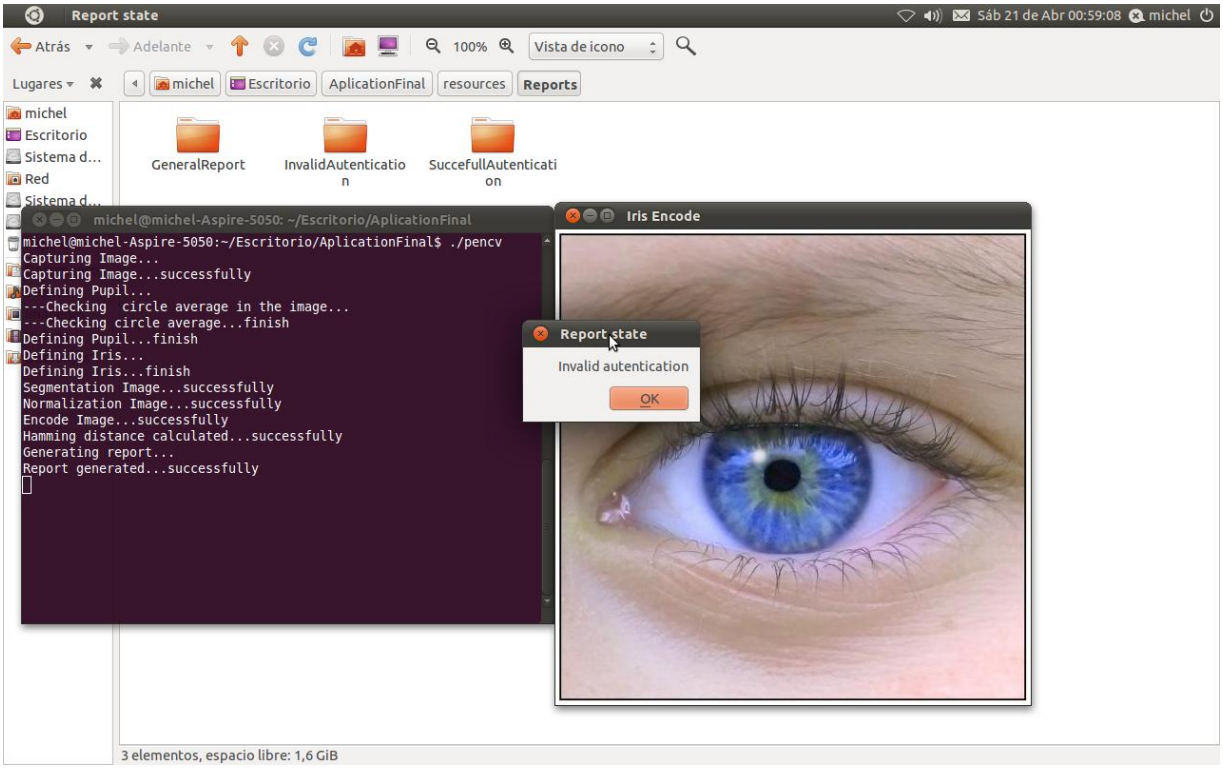

Imagen 30. Autenticación insatisfactoria