UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

FACULTAD 1

Título: *Sistema de Identificación mediante Huella Dactilar.*

TRABAJO DE DIPLOMA PARA OPTAR POR EL TÍTULO DE INGENIERO EN CIENCIAS INFORMÁTICAS

Autores: *Osmayda Silverio Leyva*
Ernesto Sabuquet Hurtado

Tutor: *Dr. C. Yusnier Valle Martínez*

Ciudad de la Habana, Junio del 2013

"Si una persona es perseverante, aunque sea dura de entendimiento, se hará inteligente; y aunque sea débil se transformará en fuerte".

Leonardo Da Vinci

DECLARACIÓN JURADA DE AUTORÍA

Declaramos que somos los únicos autores del resultado que exponemos en presente trabajo titulado *Sistema de Identificación mediante Huella Dactilar* y autorizamos al Centro de Identificación y Seguridad Digital de la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

Y para que así conste, firmo la presente declaración jurada de autoría en Ciudad de la Habana a los ____ días del mes de _____ del año _____.

Autor: Osmayda Silverio Leyva

Autor: Ernesto Sabuquet Hurtado

Dr. C. Yusnier Valle Martínez

AGRADECIMIENTOS

Agradecemos a nuestro tutor por la dedicación y entrega que nos has brindado, a todas las personas que han colaborado con nosotros de alguna u otra manera, en especial a los integrantes del departamento de Biometría y del centro en general. A la universidad y los profesores que nos han formados a lo largo de la carrera. A la Revolución.

Ernesto

A mi madre por su amor, su cariño, compresión y su eterna presencia en mi mente, gracias a eso sentí que podría lograr mi sueño de ser un profesional en el campo de la informática y las comunicaciones.

Agradezco a mis hermanas Kenia y Kirenia quienes se encargaron de que años tras años no me faltara nada.

A mis amigos que crecieron juntos a mí y que confiaron en que yo podría graduarme.

A mi compañera de Tesis por ser tan exigente, tan dedicada y por darme la oportunidad de ser su compañero.

Osmayda

A mis padres por darme la vida y todo su amor. Por ser los principales guías y educadores de mi vida. Por enseñarme que en la vida todo se gana con esfuerzo y sacrificio.

A mis abuelos, a mi tío Héctor y mi tía Anolan por formar una parte importante de mi vida, por consentirme todo el tiempo y por la fuerza de voluntad que me han inspirado siempre a ser una mejor persona.

A mi primo-hermano José Javier y mi hermana que son mis mayores tesoros, por sus consejos que me han ayudado mucho.

A mi familia que de una manera u otra han colaborado para que este sueño se haga realidad, y me han apoyado siempre. A mis amigos por estar siempre para mí en los buenos y malos momentos, aguantándome en todo instante.

A mi compañero de tesis por haber sido tan dedicado y comprensivo, y mi compañera en la comunidad Latex Melvis, por darme una mano en el tema.

Resumen

Con el incremento de las tecnologías vigentes en la actualidad y la creciente necesidad de mantener la seguridad en las instalaciones, son múltiples los tipos de sistemas de control de acceso que se despliegan, en dependencia de las necesidades de donde se encuentren instalados. Debido a lo embarazoso que se hace el control de acceso de personal, cuando los bienes tanto físicos como lógicos requieren ser altamente protegidos y existe la posibilidad de suplantación de identidad cuando se realizan los procesos de identificación y autentificación, es preciso el uso de sistemas de identificación mediante técnicas biométricas, específicamente huellas dactilares, las cuales brindan seguridad en cuanto a la autenticidad de las personas y proponen ventajas con respecto a otras tecnologías.

En la Universidad de las Ciencias Informáticas existen varios centros de desarrollo de software, entre los que se encuentra el Centro de Identificación y Seguridad Digital. El departamento Biometría perteneciente a dicho centro, se dedica a desarrollar aplicaciones que emplean técnicas biométricas, el cual a partir de experiencias adquiridas en el área de reconocimiento por huella dactilar, ha identificado la necesidad de desarrollar una aplicación que permita el uso dichas técnicas para el desarrollo de un sistema de identificación.

El presente trabajo describe las diferentes herramientas, tecnologías y artefactos generados en el proceso de desarrollo. Además recoge los resultados de la investigación realizada, describiéndose las principales características de los sistemas analizados, la arquitectura y el diseño de la solución propuesta.

Palabras clave: sistemas de control de acceso, huella dactilar

Abstract.

With the current increasing of technologies and the growing need for security in the institutions, there are multiple types of access control systems that are deployed, depending on the needs of where they are installed. Because how much embarrassing is the access control of personnel when both physical and logical assets need to be highly protected and there is a possibility of identity theft when the processes of identification and authentication are performed, it is necessary to use identification systems using biometrics, specifically fingerprints, which provide security related to the authenticity of people and suggest advantages over other technologies.

At the University of the Informatics Sciences there are various software development centers, among them there is the Center for Identification and Digital Security. Biometrics department which belongs to that center, is dedicated to develop applications using biometrics techniques, this one from the obtained experience in the field of fingerprint recognition, which have identified the need to develop an application that allows the use of these techniques for the development of an identification system.

This research describes different tools, technologies and artifacts generated in the development process. Also includes the results of it, describing the main features of the analyzed systems, the architecture and design for the proposed solution.

Keywords: access control systems, fingerprint

Índice general

Índice de figuras

Índice de tablas

Introducción

Con la evolución en la Informática y las Telecomunicaciones la sociedad está sujeta a estar conectada electrónicamente. Labores tradicionales que eran ejecutadas por seres humanos, gracias a dicha evolución, son ahora llevadas a cabo por sistemas automatizados especializados. Esta evolución ha facilitado que el crecimiento de las instituciones en nuestro país sea directamente proporcional al uso que hacen de las tecnologías para automatizar sus procesos. Dentro de la amplia gama de posibles actividades que pueden automatizarse, se encuentra, aquella relacionada con el control de acceso a diferentes áreas y lugares de una establecida instalación.

Por control de acceso se entiende, la supervisión del flujo de persona a lugares restringidos, ya sea manual o con la utilización de otros mecanismos[1]. Los sistemas de control de acceso son aplicaciones útiles para la seguridad de entidades con abundante personal, puesto que registran el acceso de cada empleado a las diferentes zonas de la instalación, permitiendo controlar si existe una infracción (ya sea robo o invasión de propiedad), así como la fecha y el responsable de la misma. Estos sistemas están compuestos por tres funcionalidades, la primera de ellas es la identificación, en el cual la persona declara su identidad como usuario hábil para acceder a la instalación; la siguiente es la autenticación, proceso en donde la persona ofrece pruebas que permite a la solución verificar su identidad; y por último tiene lugar el proceso de autorización, que consiste en dar a la persona los privilegios que tiene en la instalación[2]. Con la amplia evolución que han experimentado dichos sistemas en los últimos años, ha disminuido en gran medida la preocupación de que personal no autorizado acceda a una instalación determinada.

Los sistemas de control en su mayoría incorporan el uso de diferentes instrumentos o dispositivos como una tarjeta de identificación, teclados, escáneres de huellas digitales y otros tipos de tecnología. En el mundo existe una amplia progresión de estos sistemas, entre los que se pueden mencionar los sistemas biométricos, que posibilitan verificar la identidad de las personas a través de patrones biométricos, entre los que se destacan la lectura de huellas dactilares, de iris y reconocimiento facial.

Los sistemas de control se están imponiendo en temas de seguridad por lo fáciles y prácticos que resultan a la hora de utilizarlos. En dependencia del tamaño de las instalaciones, de los recursos económicos disponibles y del personal vinculado a las mismas, estos sistemas pueden ser o no factibles, por lo que se desarrollan diferentes alternativas para resolver este problema. Existen algunos casos donde la entrada y salida a un área o local es supervisada mediante técnicas biométricas.

El análisis biométrico es una de las técnicas de identificación y autentificación más confiables, debido a que se basa en características biofísicas o de comportamiento[3]. El interés de la sociedad por utilizar patrones biométricos ha permitido que los sistemas que basan su funcionamiento en conceptos de

biometría dactilar para ser específicos, sean más habituales, siendo evidente en pasaportes, ordenadores, teléfonos móviles y sistemas de acceso a instalaciones para citar algunos ejemplos.

La técnica de biometría dactilar representa legalmente una de las más utilizadas como prueba indiscutible de identidad, siendo un sistema que además de ser efectivo, es cómodo de aplicar, y la autenticación es obtenida rápidamente. Además, existen numerosos estudios científicos que avalan la unicidad de la huella de una persona, y lo que es más importante, la estabilidad con el tiempo, la edad, entre otros.

En los últimos años, han aparecido recursos técnicos y analíticos que agilizan y optimizan la labor de identificación por medio de huellas dactilares. La organización de impresiones dactilares en archivos manuales utilizando sistemas decadactilares, está pasando a la historia, por la implementación de los AFIS[1]. Este es un sistema informático compuesto de hardware y software integrados que permite la captura, consulta y comparación automática de huellas dactilares. Este sistema integrado utiliza tecnología digital: la huella se escanea para su búsqueda y el propio ordenador se encarga de cotejar la información existente en la base de datos con la huella incorporada.

Debido al constante avance tecnológico que existe en el mundo, las empresas y corporaciones crecen cada día y necesitan sistemas de acceso que ofrezcan mayor seguridad. En la actualidad, se ha registrado una tendencia al uso de dispositivos biométricos para la identificación, entre los cuales se destacan el reconocimiento por huella dactilar, dando paso a un modelo innovador de control de acceso, que logra procesamiento rápido, autenticación personal y mitigación de riesgos. Dicho modelo constituye la base para un sistema de identificación segura que además de supervisar el acceso, es el responsable de la seguridad, que se necesita para proteger la instalación.

La Universidad de las Ciencias Informáticas (UCI) consta de gran cantidad de personal, por lo que es complejo mantener un control estricto de entrada y salida del personal. Para lograr una mayor seguridad en el control y registro del personal, se requiere de un sistema de control de acceso en sus instalaciones. En la UCI el Centro de Identificación y Seguridad Digital (CISED), está compuesto por varias líneas de investigación y desarrollo, una de las cuales especializa su trabajo en el área de técnicas biométricas, fomentando la creación de soluciones informáticas relacionadas con las mismas. En el departamento Biometría surge la idea de desarrollar una solución que posibilite el control de acceso a instalaciones utilizando dichas técnicas.

Partiendo de lo anteriormente expuesto se manifiesta la siguiente **situación problemática:** El CISED es uno de los centros productivos que cuenta diariamente con un notable número de entradas y salidas, tanto de trabajadores como de estudiantes. En dicho centro, la manera de tener registrado el control

[1] Acrónimo en inglés de **Automated Fingerprint Identification System** (Sistema Automatizado de Identificación por Huellas Dactilares)

de acceso a sus instalaciones es a través de un sistema de identificación mediante el solapín, el cual no cuenta con suficiente confiabilidad pues los solapines pueden perderse y esto aumentaría el riesgo de que exista una suplantación de identidad y por ende que puedan acceder personal no autorizado a sus instalaciones.

Derivada de la situación ante expuesta surge el siguiente **problema de investigación:** El sistema de identificación de personas en el CISED no garantiza confiabilidad en el control de acceso a las instalaciones del centro.

Para darle solución al problema propuesto anteriormente se define como **objeto de estudio:** El proceso de identificación de personas mediante huella dactilar.

El **objetivo general** es: Desarrollar un sistema de identificación de personas mediante huellas dactilares que garantice confiabilidad en el control de acceso a las instalaciones del CISED.

Los **objetivos específicos** para darle cumplimiento al objetivo general son:

- Analizar los referentes teórico-prácticos que preceden la realización del presente trabajo, con relación a los sistemas de identificación mediante huellas dactilares y los procesos de búsqueda de huellas asociados a estos.
- Definir las herramientas a utilizar para el desarrollo del sistema que se propone.
- Realizar análisis y diseño del sistema para su posterior implementación.
- Implementar un sistema que permita identificar personas a partir de sus huellas dactilares.
- Realizar pruebas de software al sistema desarrollado para verificar su correcto funcionamiento.

Como **hipótesis** a defender se tiene que: Si se desarrolla un sistema de identificación de personas mediante huellas dactilares entonces se alcanzará una mayor confiabilidad en el control de acceso a las instalaciones del CISED.

Como **variable independiente** se tiene: El sistema de identificación mediante huella dactilar.

Como **variable dependiente**: La confiabilidad.

Indicadores para la variable independiente

Los sistemas de identificación mediante huella dactilar pueden ser medidos a través de:

- El desempeño: que se refiere a la exactitud, la rapidez y la robustez alcanzada en la identificación, además de los recursos invertidos y el efecto de factores ambientales y/u operacionales. El objetivo de esta restricción es comprobar si el sistema posee una exactitud y rapidez aceptable con un requerimiento de recursos razonable.

- La aceptabilidad: que indica el grado en que la gente está dispuesta a aceptar un sistema biométrico en su vida diaria. Es claro que el sistema no debe representar peligro alguno para los usuarios y debe inspirar "confianza" a los mismos.

Indicadores para la variable dependiente

La confiabilidad de un sistema biométrico por huella dactilar puede ser medible por la exactitud que propone FVC-OnGoing[2].

Para cumplir con los objetivos diseñados se definieron las siguientes **tareas investigativas** a desarrollar:

No	Tareas Investigativas	Responsable
1	Caracterización de los referentes teórico-prácticos en los procesos de búsqueda de huellas en los bancos de datos de un AFIS.	Osmayda Silverio Leyva
2	Definición de la metodología, las tecnologías y herramientas a utilizar para el desarrollo del sistema.	Osmayda Silverio Leyva
3	Discusión y definición de la arquitectura del sistema.	Osmayda y Ernesto
4	Confección del modelo de dominio de la aplicación.	Ernesto Sabuquet Hurtado
5	Análisis y Diseño de los Casos de Uso del sistema.	Osmayda y Ernesto
5.1	Definición de la arquitectura.	Ernesto Sabuquet Hurtado
5.2	Definición de los patrones de diseño.	Osmayda Silverio Leyva
5.3	Confección del modelo de datos.	Ernesto Sabuquet Hurtado
6	Implementación del sistema.	Osmayda y Ernesto
7	Realización de pruebas de software al sistema.	Osmayda y Ernesto

Tabla 1: Tareas Investigativas.

Para el cumplimiento de las tareas planteadas se emplearon como **métodos científicos de investigación: Métodos teóricos:**

- **Analítico-sintético:** posibilita la revisión de fuentes bibliográficas y la fundamentación de los elementos más importantes que tienen relación con el objeto de estudio, el cual es de mucha importancia para el estado del arte.
- **Análisis histórico-lógico:** brinda las maneras y líneas de actuación generales que se deben seguir para acceder a la esencia del problema, además la compresión lógica del objeto de estudio

[2]Sistema basado en la evaluación automatizada de algoritmos de reconocimiento de huellas dactilares. Las pruebas se llevan a cabo con una serie de conjuntos de datos y los resultados se reportan en línea mediante el uso de indicadores y métricas de resultados bien conocidos.

haciendo un análisis riguroso de sus antecedentes y el proceso evolutivo por el cual han transitados todas las tecnologías relacionadas con la biometría dactilar y los sistemas de control de acceso.

Métodos empíricos:

- **Observación:** permite analizar cada fase del proceso y saber si se está realizando cada tarea correctamente y tomando experiencia de la misma para aplicarla a las demás.
- **Entrevista:** sirve para recopilar información, ideas y opiniones acerca de la investigación contribuyendo con el desarrollo de la misma.

Al concluir el presente trabajo se espera como **posible resultado:**

- Una aplicación web que permita identificar el personal mediante el macheo de huellas dactilares para mejorar la confiabilidad del control de acceso a las instalaciones del CISED.

Estructura del documento

El presente Trabajo de Diploma está estructurado con tres capítulos, donde se detalla el proceso investigativo a cumplir:

Capítulo1: Fundamentación teórica: contiene la fundamentación teórica del tema tratado en la investigación. Se describen los principales conceptos abordados, se realiza un estudio del estado del arte del tema tratado a nivel internacional, nacional y en la Universidad. Además se analizan las principales metodologías, lenguajes, tecnologías y herramientas para darle solución al problema planteado.

Capítulo2: Propuesta de solución: se identifican las características del sistema a través de los requisitos funcionales recogidos en las historias de usuario y los requisitos no funcionales, se realiza el plan de iteraciones y plan de entrega, además de proponer la arquitectura y los patrones de diseño para la construcción de la aplicación.

Capítulo3: Implementación y prueba: se da cumplimiento a los planes trazados a través de las fases: Iteraciones a primera liberación y Producción, se modela el diagrama de despliegue y de componente, se codifica la solución diseñada y finalmente se realizan las pruebas a la aplicación.

Capítulo 1: Fundamentación teórica

Introducción

Con el acelerado desarrollo de la informática y las tecnologías, los sistemas de control de acceso se han visto envueltos en una notable evolución. Son muchas las razones por las cuales en la actualidad, numerosas empresas e instituciones requieren de la implementación de un sistema de control de acceso a sus instalaciones, las más comunes son: para controlar y registrar los horarios de entrada y salida del personal a su jornada laboral y para lograr mayor seguridad en dichas instalaciones.

En el presente capítulo se realiza un estudio de los principales sistemas de control de acceso existentes, basado principalmente en reconocimiento biométrico, así como conceptos asociados a estos para ayudar a una mejor comprensión del problema en cuestión. Se describen las principales características de las herramientas y tecnologías que se utilizarán en el desarrollo de la solución.

1.1. Seguridad en los Sistemas de Control de Acceso

Históricamente, las sociedades y el ser humano han tenido la necesidad de controlar el acceso a ciertas áreas y lugares. Con el decursar de los años, la sociedad ha adquirido y modificado la cultura de seguridad de sus bienes, por lo que existe una variada gama de modelos que brindan soluciones efectivas para tener restringido el acceso a diferentes áreas. Para que los sistemas de control de acceso brinden una mejor seguridad deben incorporar alguna tecnología, dependiendo de las necesidades de seguridad y las consideraciones prácticas. A continuación se describen algunas de las tecnologías que pueden ser incorporadas para aumentar la seguridad en dichos sistemas:

1.1.1. Tarjeta de banda magnética

La tarjeta magnética es una tarjeta blanca estándar con una pista magnética en su reverso. La pista magnética se usa para leer y el almacenamiento de una determinada cantidad de información en un control de acceso, en un sistema de fidelización o en tarjetas bancarias[4]. Algunas ventajas de las tarjetas con banda magnética son que, es una tecnología sumamente desarrollada y difundida, los costes necesarios para los equipamientos precisos son relativamente bajos y presentan una alta duración, siempre y cuando sean tratadas con cierto cuidado, evitando que se rayen o sean expuestas a campos magnéticos que la borren, por lo que no es recomendable usarlas en ambientes industriales.

1.1.2. Tarjeta de código de barra

El código de barras es un código basado en la representación mediante un conjunto de líneas paralelas verticales de distinto grosor y espaciado, que en su conjunto contienen una determinada información. Este es un código multifilas, continuo, de longitud variable, que tiene alta capacidad de almacenamiento de datos. El código consiste en un patrón de marcas; estos subjuegos están definidos en términos de valores particulares de una función discriminadora, cada subjuego incluye 929 codewords disponibles y tiene un método de dos pasos para decodificar los datos escaneados. La tarjeta es un archivo portátil de datos que tiene una capacidad de hasta 1800 caracteres numéricos, alfanuméricos y especiales. El código contiene toda la información, no se requiere consultar a un archivo[5].

Algunas de la ventajas que presentan estas tarjetas son que, se imprimen a bajo coste, los equipos de lectura e impresión son flexibles y fáciles de conectar e instalar, por lo que al pasar la tarjeta por el lector no existe el riesgo de rozamiento, permitiendo una vida útil mayor. Como desventaja presenta, que no se pueden rayar porque se alterarían e incluso el código se puede volver ilegible, además de que lo que se identifica es un objeto y no una persona por lo que puede ser manipulado por otro usuario que no sea el propietario.

1.1.3. Sistema de radiofrecuencia

La tecnología RFID[1] es similar a la lectura de código de barras. No solo tiene como ventaja facilitar la creación de sistemas que almacenen mucho más información, sino que también permite identificar un producto u objeto como único. El sistema completo de RFID representa un método para almacenar y recuperar datos remotos a través de proximidad, éste se compone de tres partes o módulos básicos: una tarjeta o etiqueta, un dispositivo lector y un sistema de cómputo que contiene una base de datos. Estas se pueden clasificar en 2: tarjetas pasivas (sin alimentación interna, menor tamaño, menor coste) o tarjetas activas (alimentación interna, mayor almacenamiento)[6].

En la actualidad, los sistemas implementados con tecnología RFID son ampliamente utilizados por ser tan efectivos, prácticos y de bajo coste el mantenimiento, siendo este prácticamente imposible de duplicar. Sin embargo tiene como desventaja lo costoso que pueden llegar a ser, además su instalación y programación requieren de personal altamente calificado.

1.1.4. Sistema biométrico

Son sistemas automatizados que realizan labores biométricas, es decir, el reconocimiento lo establece mediante una característica física de una persona con un patrón conocido. Estos sistemas están com-

[1] Acrónimo en inglés de **Radio Frecuency Identification** (Identificación por Radio Frecuencia)

puestos por dos dispositivos, uno de captura y otro de almacenamiento de datos, donde el de captura escanea una característica personal y crea una imagen digital o análoga y esta imagen es luego almacenada para poder ser verificada con una base de datos interna o externa, mediante conexiones de red.

Actualmente son sistemas utilizados en muchos lugares debido a que la identidad de una persona puede establecerse de manera rápida y confiable. Brindan un alto grado de seguridad, debido a que una contraseña o una tarjeta de identificación puede ser copiada o robada por cualquier persona, mientras que para este sistema se necesita la presencia del individuo para su identificación. Además la utilización de un dispositivo biométrico brinda costos de administración pequeños, ya que solo se le debe dar mantenimiento al lector y una persona es la encargada de mantener la base de dato actualizada.

Después de haber realizado un análisis de las tecnologías que pueden ser incorporadas en un sistema de control de acceso, se decide utilizar los sistemas biométricos, puesto que del resto de las tecnologías, las tarjetas no cuentan con suficiente confiabilidad, el sistema de radiofrecuencia presentan costos muy elevados de implementación, además de los gastos en equipamiento, siendo los sistemas biométricos quienes brindan un alto grado de seguridad dentro de las tecnologías actuales, por lo que es más factible de integrar al sistema a implementar.

1.2. Técnicas biométricas más comunes

Actualmente las técnicas y tecnologías biométricas desarrolladas han facilitado soluciones efectivas tanto para empresas de menor o mediana escala que precisan de precios económicos y facilidad de uso, como para grandes empresas que requieren máxima seguridad y robustez según sus características. El desarrollo correcto de un software de control de acceso biométrico permitirá emplear un buen control de horarios, ingresos y salidas del personal, garantizando además la rapidez, confiabilidad, seguridad y eficiencia. Primeramente es necesario conocer cuáles son las técnicas biométricas más comunes implementadas o estudiadas que pueden utilizarse, analizando las ventajas y desventajas de estas[7].

1.2.1. Reconocimiento facial

Los sistemas de control de acceso que utilizan como técnica el reconocimiento facial permite la identificación de personas en movimientos y el reconocimiento de personas que no estén dispuestos a ser reconocidos por el sistema. Es un método no invasivo y no requiere de acciones manuales. El rendimiento y eficacia de los sistemas de reconocimiento facial disponibles comercialmente es razonable, pero la naturaleza compleja y mutable de los rasgos faciales impone una serie de restricciones tanto en

la forma y ángulo de obtención de la imagen, como en el fondo de iluminación. Esta técnica presenta algunos inconvenientes, es susceptible a problemas de iluminación y el sistema de captura necesita de una fuente de luz auxiliar. Estos sistemas son vulnerables al reconocimiento de sujetos que se han sometido a operaciones de cirugía plástica y no tiene en cuenta en todos los casos los efectos de la edad, por lo que su fiabilidad, facilidad de uso y aceptación por parte de los usuarios es baja.

1.2.2. Iris

La identificación de personas a través del ojo humano ha dado lugar a dos tipos de sistemas biométricos totalmente diferentes: una basada en las características del iris ocular y otra que utiliza las características distintivas de la retina. "El iris es un músculo dentro del ojo que regula el tamaño de la pupila, controlando la cantidad de luz que entra en el ojo"[8].

Una de las características que hace del iris una aplicación potencial para la identificación biométrica es la protección que ofrece la córnea ante cambios originados por accidentes. Además las pequeñas variaciones en su apertura con cambios de iluminación, proporciona un mecanismo sencillo para detectar si el sujeto que se está identificando está vivo. Otra característica importante es que este tipo de tecnología adquiere los datos necesarios para su funcionamiento de forma no invasiva para el usuario.

Según varios estudios, el patrón del iris contiene más información para identificar unívocamente a una persona que una huella dactilar, asegurando que esta técnica presenta una unicidad extremadamente alta lo que llevaría a unas tasas de falsa aceptación nulas, que garantizan la viabilidad de esta técnica biométrica[9]. Su principal problema es el alto coste de los dispositivos, además la captura en algunos individuos es muy difícil, el iris se le puede ocultar fácilmente con pestañas, párpados, lentes y reflejos de la córnea; la adquisición de una imagen de iris requiere de un mayor entrenamiento y una mayor atención que la mayoría del resto de los sistemas biométricos. Esta presenta baja fiabilidad, facilidad de uso y aceptación por parte de los usuarios.

1.2.3. Voz

La comunicación mediante el habla es la forma más habitual de transmitir información entre personas. El proceso de reconocimiento de voz depende de las características de la estructura física del tracto vocal de un individuo así como también de sus características de comportamiento. Esta es una elección popular de reconocimiento biométrico remoto, dada la disponibilidad de dispositivos para tomar las muestras de voz (red telefónica y micrófonos de las computadoras), sin necesidad de contacto entre el individuo y el lector de voz y su facilidad de integración. A pesar de esto, el momento de ejecutarse el proceso de identificación haciendo uso de sistemas reconocedores de voz presenta variabilidad pues una persona no puede repetir de forma exacta una misma frase o palabra. Algunas desventajas que

estos sistemas también presentan es la susceptibilidad al canal de transmisión y a las variaciones del micrófono y su ruido; además, son susceptibles a ataques por spoofing a través de la utilización de una voz grabada[10].

1.2.4. Geometría de la mano

La técnica es sencilla y relativamente fácil de usar. Por otra parte, la geometría de la mano no es una característica completamente exclusiva de un individuo. En consecuencia, los sistemas de reconocimiento basados en ella no pueden ser ampliados a sistemas que requieran la identificación de un individuo dentro de una población grande. Además, la información geométrica de la mano es variable durante el período de crecimiento, ya sea por causa del uso de joyas (anillos) o ciertas enfermedades (artritis). Esto puede plantear nuevos retos en la correcta extracción de la información. Otro punto a nombrar es el elevado coste de hardware relacionado y el tamaño físico de un sistema basado en este carácter biométrico; se trata de un sistema grande y no puede integrarse en determinados dispositivos portátiles.

1.2.5. Huella dactilar

Es el rasgo biométrico más utilizado para autenticación y presenta la mayor gama de tecnologías de captura con distintas características de funcionamiento. Es una técnica que se ha estudiado y desarrollado mucho, presenta unicidad, estabilidad y rendimiento elevado. Asimismo, tiene como ventajas su alta tasa de precisión y que, habitualmente, los usuarios tienen conocimientos suficientes sobre su utilización. El método utilizado para el análisis de las huellas dactilares comienza a partir de la toma de una imagen de la huella, concluyendo el proceso con la consecución de un registro de esta. Instalar un terminal de reconocimiento de huella digital resulta sencillo y práctico.

La huella dactilar de todas las técnicas biométricas es la más extendida por su madurez, coste, usabilidad y rapidez en identificación[11]. Es la técnica por excelencia de identificación, siendo bastante confiable, fácil de adquirir, fácil de usar y por ende goza de gran aceptación por parte de los usuarios. La comodidad de su uso es total, evitando el uso de tarjetas, llaves o memorizar números identificativos.

1.3. Estado del arte sobre sistemas similares

Cuando se piensa en control de acceso, se podría imaginar a un portero o a alguien que permitía el acceso a una determinada área de una institución. El conocimiento de una contraseña, la utilización de un determinado uniforme, la posesión de una determinada llave, han permitido desde siempre el acceso a lugares restringidos. Con los años, esta identificación se ha estado haciendo a través de tarjetas, muchas de las cuales se usan hoy en día, sin embargo estos objetos pueden ser robados o falsificados

y usados por cualquier persona, por lo que se deben buscar métodos que no dependan de una "llave" determinada, sino que la propia persona sea la llave que le permita autenticarse, y es aquí donde entra la biometría.

La biometría como proceso es un método automático de reconocimiento de individuos, basado en características biológicas y de comportamiento que se pueden medir. La biometría es un componente de la arquitectura de un sistema global y diversos planes de contingencia que varían dependiendo del ambiente o circunstancia. Las tecnologías biométricas están siendo aplicadas en variados lugares para aumentar la seguridad y comodidad de la sociedad.

1.3.1. Soluciones existentes en el mundo

En la actualidad existen diferentes empresas y compañías internacionales con varios años de experiencia en brindar soluciones para el control de acceso, centradas fundamentalmente en el reconocimiento biométrico. Algunas de ellas son: Kimaldi, TSB, Dointech, SCSSA, entre otras. A continuación se describen algunas soluciones de estos sistemas implantadas en distintos sectores.

- La empresa mexicana TSB ha logrado implementar un esquema de autentificación que permite registrar con precisión las entradas y salidas del personal autorizado, identificar con certeza a la persona que desea entrar a las instalaciones, lograr revertir en corto tiempo la cartera vencida de sus clientes, y contar con un mejor control de visitantes. Tiene como elementos para su solución:
  - Control biométrico, con terminales de huella dactilar para garantizar que no exista suplantación de identidad y con tecnología que puede almacenar hasta 50 mil personas con 2 huellas cada una, con capacidad de enlace a su red local, para instalar proyectos con cientos de terminales biométricas, si es necesario.
  - Control con credenciales de radiofrecuencia para facilitar el paso a las personas autorizadas o credenciales inteligentes aprovechando que puede ser utilizada para brindar más servicio al alumnado como pago de copias, de cuotas, de servicios de cafetería, y muchos más.
  - Control mixto que es una solución más segura ya que cuenta con más de 1 elemento de autentificación.
- El producto CAB, desarrollado por la empresa de servicios de tecnologías de la información Colombia Programadores. Es un producto basado en la biometría de impresiones dactilares que ofrece dos versiones, una versión estándar recomendada para hasta 100 trabajadores y una versión empresarial para más de 100 trabajadores. Genera diversos reportes de asistencia, de ingresos y salidas, tiene una velocidad de búsqueda de la foto de la huella almacenada en la base de datos super rápida y está desarrollado en tecnología .NET. El dispositivo de la lectura de la Huella se vende aparte y tiene garantía y soporte por 3 meses.

- SmartCard Systems es una empresa líder en el desarrollo de Sistemas Inteligente de Identificación y Control de Acceso, aplicables a las distintas necesidades de una empresa determinada. Tiene desarrollado un control de acceso y asistencia en universidades, academias y escuelas que permite saber en todo momento que alumnos se encuentran dentro de la institución y otros datos gerenciales como por ejemplo, tiempo promedio de permanencia, asistencias promedio semanales, etc.

1.3.2. Soluciones existentes en Cuba

Cuba no se ha quedado rezagada en el desarrollo de sistemas biométricos. La empresa DATYS, Tecnologías y Sistemas tiene una división destinada al desarrollo de softwares biométricos (Biomesys SUITE), que desde el año 2006 ya implementa el Biomesys AFIS ahorrándole al país mas de 10 millones de dólares. Permite identificar y autenticar a las personas, neutralizar los intentos de suplantación de identidad, identificar cadáveres y discapacitados y fortalece los documentos de certificación oficial de identidad. Además brinda herramientas de visualización, búsqueda automática al introducir impresiones en la base de datos, etc.

1.3.3. Soluciones existentes en la UCI

La Universidad de las Ciencias Informáticas no posee un sistema de control de acceso biométrico, aunque existen otros sistemas implementados para la ayuda del control de acceso en la misma, tales como:

Sistema de control de acceso a la universidad: permite controlar el acceso de cualquier persona a la UCI. Funciona en cada una de las puertas de la escuela y se verifican las personas que tienen acceso mediante control visual y documental.

Sistema de control de acceso a los laboratorios de producción: permite controlar el acceso del personal los laboratorios de un determinado centro. Están situados en puertas estratégicas, verificando que la persona esté en la base de datos correspondiente con el proyecto, mediante el número de solapín o el código de barra del mismo.

Estos sistemas no son muy confiables. El solapín puede ser extraviado y encontrado por personal ajeno a las instalaciones del centro, permitiéndole acceder sin autorización. Es por esto que la UCI requiere de un sistema de control de acceso más confiable, como por ejemplo un sistema de identificación por huella dactilar.

1.3.4. Valoración de los sistemas

A partir del estudio realizado sobre los sistemas de control de acceso biométrico existentes en el ámbito internacional, e incluso en el ámbito nacional queda evidenciado que aquellos con mejores prestaciones son propietarios lo que implica gran inversión económica para nuestro centro. Además se suman otros factores por los que no son factibles de utilizar para dar solución a la situación problemática, entre los que se pueden mencionar la utilización de tarjetas de proximidad y puertas electrónicas; que aunque la integración con varias tecnologías son formas de autenticación más fiable, necesitan de tecnologías con las que no se cuentan para el desarrollo del sistema.

1.4. Metodología, Lenguajes y Tecnologías a utilizar

En este tópico se hace referencia a algunas de las herramientas y tecnologías existentes en el mercado que serán utilizadas para el desarrollo de la aplicación Sistema de Control de Acceso mediante huellas dactilares para el CISED.

1.4.1. Metodología de desarrollo de software

Uno de los temas más conocidos en el mundo de la informática es el de las metodologías de desarrollo de software. Una metodología tiene como objetivo aumentar la calidad del software que se produce en cada una de las fases de desarrollo, por medio de una mayor transparencia y control sobre el proceso; producir lo esperado en el tiempo y coste esperado.

Todo sistema a implementar se rige por una metodología, que no es más que un conjunto de pasos a seguir en pos del resultado final. En la esfera del desarrollo de software, se busca obtener un producto de alta calidad, que satisfaga las necesidades del usuario. Las metodologías de desarrollo están dividas en dos grupos que se conocen como, metodologías tradicionales y metodologías ágiles, siendo las tradicionales pensadas para hacer un uso exhaustivo de documentación durante todo el ciclo del proyecto y recomendada para los proyectos con grandes equipos de desarrollo, mientras que las ágiles hacen énfasis en la capacidad de dar respuesta a los cambios, promoviendo el trabajo en equipo y manteniendo una buena relación con el cliente.

A continuación la tabla 2 obtenida del articulo *Metodologías Ágiles en el Desarrollo de Software.* por los autores **Mª Carmen Penadés; Patricio Letelier y José Hilario Canós.** [12] representa las diferencias entre las metodologías tradicionales y las metodologías ágiles, tomando como referencia no solo el proceso en sí, sino también al contexto de equipo y organización que es más favorable a cada una de estas filosofías de procesos de desarrollo de software.

Metodología Ágil	**Metodología Tradicional**
Basadas en heurísticas provenientes de prácticas de producción de código.	Basadas en normas provenientes de estándares seguidos por el entorno de desarrollo.
Especialmente preparados para cambios durante el proyecto.	Cierta resistencia a los cambios.
Impuesta internamente (por el equipo).	Impuestas externamente.
Proceso menos controlado, con pocos principios.	Proceso mucho más controlado, con numerosas políticas/normas.
No existe un contrato tradicional o al menos es bastante flexible.	Existe un contrato prefijado.
El cliente es parte de equipo de desarrollo.	El cliente interactúan con el equipo de desarrollo mediante reuniones.
Grupos pequeños (menos de 10 integrantes) y trabajando en el mismo sitio.	Grupos grandes y posiblemente distribuidos.
Pocos artefactos.	Más artefactos.
Pocos roles.	Más roles.
Menos énfasis en la arquitectura de software.	La arquitectura de software es esencial y se expresa mediante modelos.

Tabla 2: Diferencia entre metodologías ágiles y no ágiles.

Como resultado del análisis de las diferencias entre las metodologías ágiles y tradicionales, se llega a la conclusión de que se hará uso de una metodología ágil para el desarrollo de nuestro producto, determinando como elementos importantes para su selección, los siguientes:

- El equipo de desarrollo está compuesto por 2 integrantes que trabajan en el mismo sitio y el proyecto es de corta duración.
- Se necesitan pocos roles ya que el equipo de desarrollo es pequeño.
- La arquitectura del producto se va definiendo y mejorando a lo largo del proyecto.
- No existe un contrato firmado con un cliente.
- Se esperan cambios durante el desarrollo del proyecto en los requisitos.
- El cliente es parte del equipo de desarrollo.

Existen varias metodologías ágiles, de las más utilizadas a nivel mundial se realiza una comparación teniendo en cuenta: la vista del sistema como algo cambiante, colaboración entre los miembros del equipo y características propias de la metodología tales como: simplicidad, excelencia técnica, adaptabilidad y prácticas de colaboración. La tabla 3 obtenida del autor **J Highsmith.** [13] representa una

comparación, donde los resultados y valores más altos representan una mayor agilidad.

	ASD[2]	Crystal	DSDM[3]	FDD[4]	LD[5]	Scrum	XP[6]
Sistema como algo cambiante	5	4	3	3	4	5	5
Colaboración	5	5	4	4	4	5	5
Adaptabilidad	5	5	3	3	4	4	3
Simplicidad	4	4	3	5	3	5	5
Excelencia técnica	3	3	4	4	4	3	4
Prácticas de colaboración	5	5	4	3	3	4	5

Tabla 3: Metodologías ágiles más utilizadas.

Después de realizar un estudio de los resultados de las diferentes metodologías ágiles existentes, se decide optar por la metodología XP teniendo en cuenta los siguientes aspectos:

- XP se centra más en la programación o creación del producto que en la administración del proyecto.
- XP recomienda que las tareas a desarrollar sean desarrolladas por 2 personas en vez de una, como en otras metodologías ágiles logrando una mayor calidad del código al ser revisado y discutido mientras se escribe.
- Además de ser dentro de las metodologías ágiles una de las más usadas en la Universidad, por lo que se posee más documentación sobre ella y existen varios especialistas que conocen sobre la misma.

1.4.1.1. Programación Extrema o XP

Es una de las metodologías de desarrollo de software, enmarcada dentro de las metodologías ágiles de desarrollo. Constituye la metodología más utilizada dentro del grupo de las ágiles. Mientras que RUP[7] intenta reducir la complejidad del software por medio de estructura y la preparación de las tareas pendientes en función de los objetivos de la fase y actividad actual, XP, como toda metodología ágil, lo intenta por medio de un trabajo orientado directamente al objetivo, basado en una serie de valores, principios y prácticas aumentando la productividad a la hora de gestionar un software.

Valores que promueve

[2]Acrónimo en inglés de **Adaptive Software Development** (Desarrollo Adaptable de Software)

[3]Acrónimo en inglés de **Dynamic Systems Development Method** (Método de Desarrollo de Sistemas Dinámicos)

[4]Acrónimo en inglés de **Feature Driven Development** (Desarrollo Basado en Funcionalidades)

[5]Acrónimo en inglés de **Lean Development** (Desarrollo Magro)

[6]Acrónimo en inglés de **Extreme Programming** (Programación Extrema)

[7]Metodología de desarrollo de software pesada. Su acrónimo en inglés es **Rational Unified Process** (Proceso Unificado Racional)

En el ciclo de vida de un proyecto los cambios van a aparecer y a veces el equipo de desarrollo no está preparado para enfrentarlos, XP desarrolla los siguientes valores para garantizar el éxito de un proyecto de desarrollo de software [14]:

- Comunicación: Crear software requiere de sistemas comunicados.
- Simplicidad: Empezar con lo necesario y requerido y trabajar desde ahí.
- Retroalimentación: Del sistema, del cliente, y del equipo.
- Valentía: Programa para hoy y no para mañana.
- Respeto: El equipo debe trabajar como uno, sin hacer decisiones repentinas.

Prácticas

La mayoría de estas prácticas no son nuevas, sino que han sido escogidas de diferentes metodologías ya existente reconocidas por la industria como las mejores prácticas durante años[14].

- Retroalimentación a escala fina: Desarrollo guiado por prueba, juego de planificación, cliente presente, programación en pares.
- Proceso continuo en lugar de por lotes: Integración continua, liberación pequeña, refactorizar.
- Entendimiento compartido: Diseño simple, metáfora del sistema, propiedad colectiva del código, convenciones del código.
- Bienestar del programador: Paso sostenible.

Artefactos esenciales de XP

- Historia de usuario
- Tareas de ingeniería
- Pruebas de aceptación
- Pruebas unitarias y de integración
- Plan de entrega
- Código

Ciclo de vida

La figura 1 muestra el ciclo de desarrollo de XP, iterativo e incremental compuesto por seis fases[15]:

- Exploración: En esta fase los usuarios escriben las tarjetas de historia que ellos quieren que sean incluidas en la primera versión. Cada una de las tarjetas de historia describe una funcionalidad que será añadida al programa. El equipo de proyecto durante este tiempo se dedica a familiarizarse con las tecnologías y herramientas que utilizará a lo largo del proyecto, probando las herramientas y construyendo un prototipo simple para probar las posibilidades de la arquitectura.

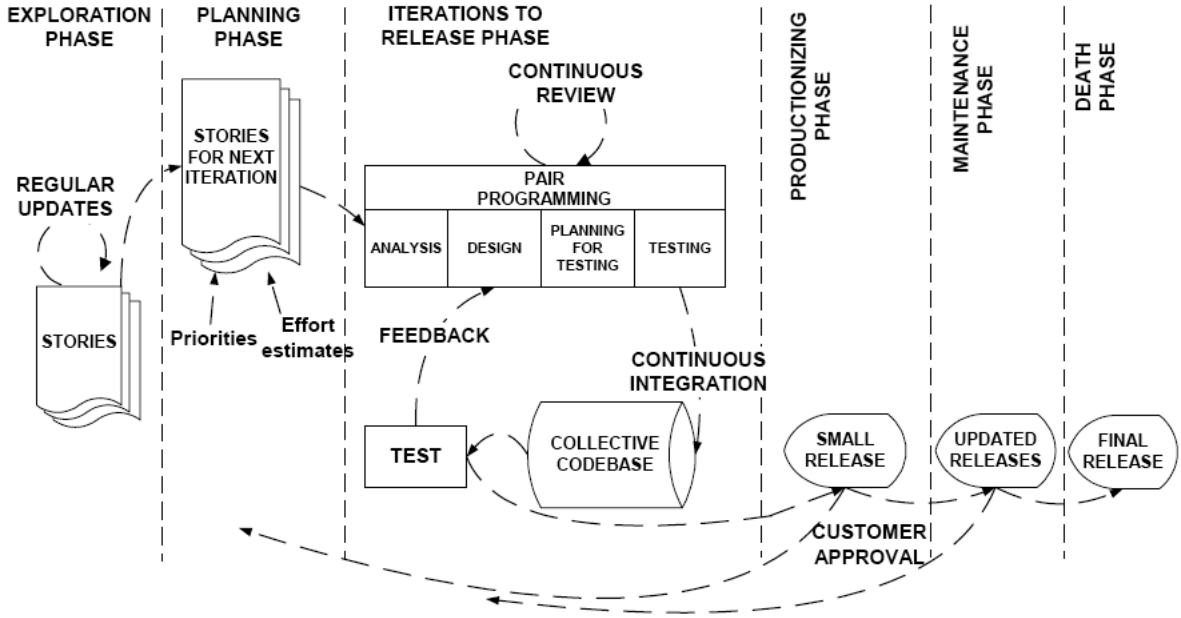

Figura 1: Ciclo de vida de XP.

- Planificación: En esta fase el cliente establece la prioridad de cada historia de usuario y correspondientemente, los programadores realizan una estimación del esfuerzo necesario de cada una de ellas. Se toman acuerdos sobre el contenido de la primera entrega y se determina un cronograma en conjunto con el cliente. Una entrega debería obtenerse en no más de tres meses.
- Iteraciones: Esta fase incluye varias iteraciones sobre el sistema antes de ser entregado. Todo el trabajo de la iteración es expresado en tareas de programación, cada una de ellas es asignada a un programador como responsable, pero llevadas a cabo por parejas de programadores.
- Producción: En esta fase se llevan a cabo un conjunto de pruebas extras, de rendimiento y funcionamiento que son necesarias antes de poder entregar el producto al cliente. Al mismo tiempo, se deben tomar decisiones sobre la inclusión o no de nuevas características a la versión actual del producto.
- Mantenimiento: Se requiere de mantenimiento y soporte al cliente.
- Cierre del proyecto: Es la fase en que los clientes ya no tienen más historias que deban ser implementadas, por lo que deben estar satisfecha las necesidades de los clientes y otros aspectos como fiabilidad, rendimiento, etc.

1.4.2. Lenguaje de modelado

Un lenguaje de modelado es un conjunto estándar de símbolos y de formas que facilita la modelación de un diseño de software. El Lenguaje Unificado de Modelado UML[8], es un lenguaje gráfico para visualizar, especificar y documentar cada una de las partes que comprende el desarrollo de software. UML

[8]Acrónimo en inglés de **Unified Modeling Language**

permite una forma de modelar conceptualmente procesos de negocio y funciones de sistema, además de elementos concretos como son esquemas de base de datos, componentes de software reutilizables y escribir clases en un lenguaje determinado[16].

Divide cada proyecto en un número de diagramas que representan las diferentes vistas del proyecto. Estos diagramas juntos son los que representa la arquitectura del proyecto. Permite modelar, construir y documentar los elementos que forman un sistema software orientado a objetos. Se ha convertido en el estándar de la industria, debido a que ha sido impulsado por los autores de los tres métodos más usados de orientación a objetos: Grady Booch, Ivar Jacobson y Jim Rumbaugh. Tiene como objetivo brindar un material de apoyo que le permita al lector poder definir diagramas propios como también entender diagramas ya existentes[17].

El Lenguaje Unificado de Modelado prescribe un conjunto de notaciones y diagramas estándar para modelar sistemas orientados a objetos, y describe la semántica esencial de lo que estos diagramas y símbolos significan. UML se puede usar para modelar distintos tipos de sistemas: sistemas de software, sistemas de hardware, y organizaciones del mundo real.

1.4.3. Herramienta CASE

Las herramientas CASE[9] son un conjunto de programas y ayudas que dan asistencia a los analistas, ingenieros de software y desarrolladores, durante todos los pasos del Ciclo de Vida de desarrollo de un Software[18].

1.4.3.1. Visual Paradigm for UML

Visual Paradigm for UML es una potente herramienta que permite visualizar y diseñar elementos de software y soporta el ciclo de vida completo del desarrollo de software. Una de sus ventajas sobre las demás herramientas CASE es condición multiplataforma en su edición Community, por lo que tiene la capacidad de ejecutarse sobre diferentes Sistemas Operativos. Es fácil de usar y presenta una agradable interfaz para interactuar con el usuario. Se integra con el Visio para importar imágenes del mismo para realizar los diagramas de despliegue y genera documentación para el proyecto en HTML, MS Word y PDF. Además, importa y exporta diagramas en XML y como imágenes (ya sea con extensiones jpg o png). Otra característica importante es la generación del código C#, VB.NET, Object Definition Language (ODL), Flash ActionScript, Delphi, Perl, y Ruby. Proporciona ingeniería inversa para Java, .NET, .DLL y JDBC.

Soporta UML completo y para gestionar la persistencia y el mapeo de estas clases con la base de datos utiliza Hibernate para Java y NHibernate en el caso de un proyecto .NET. Esta herramienta

[9] Acrónimo en inglés de **Computer Aided Software Engineering** (Ingeniería Asistida por Computadora)

es colaborativa, es decir, soporta múltiples usuarios trabajando sobre el mismo proyecto; genera la documentación del proyecto automáticamente en varios formatos como web o pdf y permite control de versiones.

1.4.3.2. Rational Rose

Rational Rose es la herramienta CASE desarrollada por los creadores de UML (Booch, Rumbaugh y Jacobson), que cubre todo el ciclo de vida de un proyecto: concepción y elaboración del modelo, construcción de los componentes, transición a los usuarios. Es una herramienta privativa, por lo que se debe hacer un previo pago para poder adquirir el producto. Permite especificar, analizar y diseñar el sistema antes de codificarlo. Rational Rose establece una trazabilidad real entre el modelo (análisis y diseño) y el código ejecutable; facilita el desarrollo de un proceso cooperativo en el que todos los agentes tienen sus propias vistas de información (vista de Casos de Uso, vista Lógica, vista de Componentes y vista de Despliegue), pero comparten un mismo modelo a lo largo de todo el ciclo de vida del proyecto.

Rational Rose utiliza un proceso de desarrollo iterativo controlado, donde el desarrollo se lleva a cabo en una secuencia de iteraciones. Permite que haya varias personas trabajando a la vez en el proceso iterativo controlado, para ello posibilita que cada desarrollador opere en un espacio de trabajo privado que contiene el modelo completo y tenga un control exclusivo sobre la propagación de los cambios en ese espacio de trabajo. También es posible descomponer el modelo en unidades controladas e integrarlas con un sistema para realizar el control de proyectos que permite mantener la integridad de dichas unidades.

1.4.3.3. Fundamentación de la selección

Se decide trabajar con la herramienta CASE Visual Paradigm ya que brinda facilidades para el diseño de los diagramas, permite el diseño del producto de forma rápida y con calidad, así como su documentación, puede generar código y realizar ingeniería inversa a diferentes lenguajes de programación. Es también muy robusta, usable y portable. Además, la Universidad cuenta con la licencia para su uso y tiene la característica de ser multiplataforma.

1.4.4. Lenguaje de programación

Un lenguaje de programación es un conjunto de símbolos y reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos y expresiones. Es utilizado para controlar el comportamiento físico y lógico de una máquina[19]. Se decide trabajar con el lenguaje de programación CSharp (C#) debido a que cuenta con algunas características que hacen fácil y fiables su adopción, entre las que se encuentra, sencillez, modernidad, seguridad, orientado a componentes y la más importante orientado a objeto. Permite crear aplicaciones cliente-servidor, aplicaciones de base datos y otras de

forma sólida y segura ejecutadas en .NET Framwork. Además es necesario trabajar con este lenguaje ya que otros componentes que se integran con la aplicación están implementado en C# .NET, por lo que se debe escoger un lenguaje y tecnología compatible con estos componentes.

1.4.4.1. C# o CSharp

C# es un lenguaje de propósito general orientado a objetos diseñado por Microsoft para su plataforma .NET. Toma las mejores características de varios lenguajes persistentes como Visual Basic, Java o C++ y las combinan en uno solo, eliminando aquellos elementos que son innecesarios. Soporta todas las características propias del paradigma de programación orientada a objetos (encapsulación, herencia y polimorfismo). Incluye mecanismos que permiten asegurar que los accesos a tipos de datos siempre se realicen correctamente, lo que permite evitar que se produzcan errores difíciles de detectar por acceso a memoria no perteneciente a ningún objeto y es especialmente necesario en un entorno gestionado por un recolector de basura.

Proporciona una funcionalidad óptima para racionalizar los procesos empresariales, incluidos: compatibilidad con el diseño, el desarrollo y la implementación de servicios web XML con rapidez; diseñadores de formularios y controles visuales para crear aplicaciones basadas en Windows muy completas; herramientas y servicios de diseño para crear poderosas soluciones en Microsoft .NET basadas en servidor.

Es un lenguaje elegante y con seguridad de tipos que permite a los desarrolladores crear una amplia gama de aplicaciones sólidas y seguras que se ejecutan en .NET Framework. Se pueden crear aplicaciones cliente para Windows tradicionales, servicios web XML, componentes distribuidos, aplicaciones cliente-servidor, aplicaciones de base de datos, y muchas tareas más.

1.4.5. Lenguaje del lado del servidor

La versatilidad de un lenguaje está íntimamente relacionada con su complejidad. Es por ello que a la hora de elegir el lenguaje que se quiere utilizar se tiene que saber claramente qué es lo que se quiere hacer y si el lenguaje en cuestión lo permite o no. En el dominio de la red, los lenguajes de lado servidor más ampliamente utilizados para el desarrollo de páginas dinámicas son el ASP, PHP y PERL[20].

1.4.5.1. PHP

PHP[10] es un lenguaje muy potente, mundialmente utilizado en la programación de aplicaciones Web del lado del servidor. Es un lenguaje de programación del lado del servidor gratuito e independiente de plataforma creado originalmente en 1994 por Rasmus Lerdorf, pero como está desarrollado con la política de código abierto, a lo largo de su historia ha tenido muchas contribuciones de otros desarrolla-

[10] Acrónimo recursivo que significa **PHP Hypertext Pre-processor**

dores. Es similar a otros lenguajes como C o C++, rápido, de sintaxis cómoda y su sencillez contribuye a su rápido aprendizaje. Es multiplataforma, se le pueden agregar extensiones fácilmente y dispone de una gran cantidad de librerías. Al ser utilizado como módulo de Apache, lo hace extremadamente veloz. Por estar completamente escrito en C, se ejecuta rápidamente utilizando poca memoria. Para su adquisición no hay que pagar licencia, así que su distribución no es limitada, permitiéndole ampliarse con nuevas funcionalidades si se desea. Posee una vasta gama de funciones que le permiten adaptarse a cualquier entorno y a cualquier sistema operativo, por lo que es más eficiente. Es un lenguaje muy asequible, tanto para aquellas personas que tienen experiencia en la programación de sistemas web dinámicos, como para los que no.

PHP es un poderoso lenguaje e intérprete, incluido como parte de un servidor web en forma de módulo, es capaz de acceder a archivos, ejecutar comandos y abrir conexiones de red en el servidor, pero estas propiedades hacen que cualquier cosa que sea ejecutada en un servidor web sea insegura por naturaleza. Tiene como desventaja que todo el trabajo lo realiza el servidor y no delega al cliente por lo que puede ser más ineficiente a medida que las solicitudes aumenten de número.

1.4.5.2. PERL

PERL[11] es un lenguaje muy utilizado para construir aplicaciones CGI[12] para la web, es un software libre (bajo la licencia GNU y licencia artística), lo que significa que no es necesario pagar para obtenerlo. Además, existen distribuciones y adaptaciones para una cantidad de sistemas operativos como Linux, Unix, Windows, Mac, etc.

Una diferencia fundamental de Perl con respecto a los otros lenguajes es que no limita el tamaño de los datos con los que trabaja, el límite lo pone la memoria que en ese momento se encuentre disponible. Es fácil de usar, aunque es difícil de aprender. Tiene características que soportan una variedad de paradigmas de programación, como la estructural, funcional y la orientada a objetos. La principal desventaja de Perl se encuentra en el tiempo de ejecución de un programa, ya que un programa Perl es compilado cada vez que se ejecuta, por lo que puede resultar más lento que un programa similar escrito en otro lenguaje. Este lenguaje se desarrolló pensando en que el lenguaje fuera práctico (fácil de usar, eficiente, y completo) en lugar de pequeño, elegante y mínimo, además de que está enfocado hacia un desarrollador que posee cierta cantidad de conocimientos sobre el lenguaje y no así hacia un estudiante que está aprendiendo[21].

[11] Acrónimo en inglés de **Practical Extracting and Reporting Languaje**, que indica que se trata de un lenguaje de programación muy práctico para extraer información de archivos de texto y generar informes a partir del contendido de los ficheros

[12] Acrónimo en inglés de **Common Gateway Interface** (Interfaz de pasarela común)

1.4.5.3. ASP

ASP[13] es un lenguaje de programación de servidores para generar páginas Web dinámicamente, liberado por Microsoft en 1996. Una de su principales ventajas es la seguridad que tiene el programador sobre su código, ya que éste se encuentra inicialmente en los archivos del servidor que al ser solicitado a través de la web, es ejecutado, por lo que los usuario no tienen acceso más que a la página resultante en su navegador. Actualmente tiene diversas limitaciones respectos a otros lenguajes como ASP.NET, el cual es sucesor de la tecnología ASP, lanzada al mercado mediante una estrategia de mercado denominada .NET, desarrollado para resolver las limitantes que brindaba su antecesor[22].

ASP.NET es un framework para aplicaciones web desarrollado y comercializado por Microsoft, utilizado para construir sitios web dinámicos, aplicaciones web y servicios web XML. Apareció en enero de 2002 con la versión 1.0 del .NET Framework, siendo un lenguaje totalmente orientado a objetos. Las páginas de ASP.NET, conocidas oficialmente como "web forms"(formularios web), son el principal medio de construcción para el desarrollo de estas aplicaciones web.

Las aplicaciones ASP.NET son alojadas en un servidor web y se tiene acceso a ellas mediante el protocolo sin estado HTTP, que no guarda ninguna información sobre conexiones anteriores. ASP.NET proporciona varias maneras de administrar el estado de las aplicaciones ASP.NET. A diferencia de sus predecesores, ASP.NET puede aprovechar las ventajas del enlace anticipado, la compilación just-in-time, la optimización nativa y los servicios de caché desde el primer momento. La suscripción de ASP.NET administra la autenticación, proporcionando los medios para validar credenciales de usuario y ayudando a los usuarios a administrar sus contraseñas. El servicio de suscripción proporciona interfaces API[14] a las que puede llamar mediante programación para crear nuevos usuarios, validar credenciales y obtener información sobre el usuario. Provee herramientas para compartir datos e información entre distintos sitios, puede mostrar toda una base de datos y realizar rutinas complejas mediante diversos controles, unas pocas líneas y en menos de 5 minutos. Además, el motor de tiempo de ejecución de ASP.NET controla y administra los procesos de cerca, por lo que si uno no se comporta adecuadamente (filtraciones, bloqueos), se puede crear un proceso nuevo en su lugar, lo que ayuda a mantener la aplicación disponible constantemente para controlar solicitudes. ASP.NET permite generar interfaces de usuario, que separan claramente la lógica de aplicación del código de presentación, y controlar eventos en un sencillo modelo de procesamiento de formularios de tipo Visual Basic.

[13] Acrónimo en inglés de **Active Server Pages** (Páginas Activas en el Servidor)
[14] Acrónimo en inglés de **Application Program Interface** (Interfaz de Programación de Aplicaciones)

1.4.5.4. Fundamentación de la selección

La integración nativa .NET Framework con el sistema operativo Windows Server 2003 hace que su ejecución sea más estable y rápida que otros lenguajes de programación. Las páginas creadas con la tecnología ASP.NET funcionan en todo tipo de navegadores, incluyendo Netscape, Safari e Internet Explorer. Las razones antes expuestas, más su gran seguridad, hacen que ASP.NET sea el lenguaje perfecto para el sistema de control de acceso que se desea implementar. Además de ser compatible con los componentes que se integran ha dicho sistema.

1.4.6. Sistema Gestor de Base Datos

Un Sistema Gestor de Base de Datos (SGBD) es una colección de programas cuyo objetivo es servir de interfaz entre la base de datos, el usuario y las aplicaciones. Permite definir y manipular los datos a distintos niveles de abstracción, garantizando la seguridad e integridad de los mismos[23].

Todos los sistemas automáticos emplean base de datos, estos son el soporte para almacenar toda la información que manejan, deben presentar unicidad, consistencia, seguridad e integridad. Existen varios sistemas gestores de base de datos, como son, SQL Server, PostgreSQL, y MySQL. A continuación se muestran características de cada uno de ellos.

1.4.6.1. PostgreSQL

PostgreSQL es un sistema de gestión de bases de datos objeto-relacional, basados en POSTGRES, Versión 4.2, desarrollado en la Universidad de California y distribuido bajo la licencia BSD (Berkeley Software Distribution) de software libre. Es el sistema de gestión de bases de datos de código abierto más potente y robusto del mercado y en sus últimas versiones no tiene nada que envidiarle a otras bases de datos comerciales[24]. Durante todo este tiempo se ha caracterizado por proporcionar estabilidad, potencia, robustez, facilidad de administración e implementación de estándares permitiendole funcionar con grandes cantidades de datos y una alta concurrencia de usuarios accediendo a la vez a el sistema.

PostgreSQL proporciona un gran número de características que normalmente solo se encontraban en las bases de datos comerciales tales como Oracle. Es capaz de manejar complejas rutinas y reglas, realizando consultas SQL declarativas, control de concurrencia multi-versión, soporte multi-usuario, transacciones, optimización de consultas, herencia y valores no atómicos (atributos basados en vectores y conjuntos). Es altamente extensible, soporta operadores funcionales, métodos de acceso y tipos de datos definido por el usuario. Incluye características avanzadas tales como las uniones (joins) SQL92, soporta la especificación SQL99 y soporta integridad referencial, la cual es utilizada para garantizar la validez de los datos de la base de datos. Cuenta con un API flexible que permite a los vendedores

proporcionar soporte al desarrollo fácilmente para el RDBMS[15] PostgreSQL. Estas interfaces incluyen Object Pascal, Python, Perl, PHP, ODBC, Java/JDBC, Ruby, TCL, C/C++, y Pike. Tiene soporte para lenguajes procedurales internos incluyendo un lenguaje nativo denominado PL/pgSQL, que es comparable al PL/SQL de Oracle.

Algunas de las ventajas que presenta PostgreSQL es su habilidad para usar Perl, Python, o TCL como lenguaje procedural embebido. Utiliza la tecnología MVCC o Control de Concurrencia Multi-Versión para evitar bloqueos innecesarios. Emplea una arquitectura proceso-por-usuario, es decir cliente/servidor y utiliza multiprocesos en vez de multi-hilos, garantizando la estabilidad del sistema ante cualquier fallo.

1.4.6.2. SQL Server 2008

Microsoft SQL Server 2008 proporciona una plataforma productiva e inteligente en la cual puede confiar que le permite controlar la demanda crítica de aplicaciones, reduce el tiempo y costos de desarrollo, tanto como el manejo de aplicaciones y posibilidad de tener un foco global de toda la empresa[25]. SQL Server 2008 provee de una motor de base de datos escalable y de alta performance ideal para misiones críticas de aplicaciones. Esto requiere del mejor nivel de disponibilidad y seguridad mientras se reduce el costo total de autoría mediante la mejora del manejo de su empresa. La programabilidad de datos de la plataforma Microsoft provee a desarrolladores con un excelente marco de acceso a la programación, servicios web y tecnología de conectividad de datos, así como a la posibilidad de manejar diferentes paquetes de datos.

1.4.6.3. MySQL

MySQL es un sistema de gestión de bases de datos relacional, licenciado bajo la GPL de la GNU. Fue creado por la empresa sueca MySQL AB, que mantiene el copyright del código fuente del servidor SQL, así como también de la marca. Su diseño multi-hilo le permite soportar una gran carga de forma muy eficiente. Algunas de las principales características de este gestor de bases de datos son:

- Aprovecha la potencia de sistemas multiprocesador, gracias a su implementación multi-hilo.
- Soporta gran cantidad de tipos de datos para las columnas.
- Dispone de APIs en gran cantidad de lenguajes (C, C++, Java, PHP, etc.).
- Gran potabilidad entre sistemas.
- Soporta hasta 32 índices por tabla.
- Gestión de usuarios y contraseña, manteniendo un muy buen nivel de seguridad en los datos.

[15] Acrónimo en inglés de **Management system relational database** (Sistema de gestión de bases de datos relacionales)

MySQL es un gestor de base de datos que fue creado con el objetivo de buscar rapidez de respuesta; por tal razón carecía de muchas de las características que tiene un gestor potente, como es la integridad referencial. Con el tiempo se le han adicionado algunas de las funcionalidades que antes no permitía, pero todavía no han alcanzado la robustez necesaria para que sea confiable.

1.4.6.4. Fundamentación de la selección

Después de haber analizado los gestores de bases de datos más utilizadas, viendo sus características fundamentales, se decidió seleccionar PostgreSQL para la realización de este trabajo. Esto se debe a que es un potente gestor que puede compararse con sus competidores como son Oracle y SQL Server. Es libre y corre sobre varias plataformas, haciéndolo multiplataforma. Permite transacciones, es decir, múltiples operaciones de tabla o registros de manera segura. Al proporcionar la funcionalidad del Control de Concurrencia Multi-Versión, este gestor mantiene la consistencia de los datos. Evitando que alguna transacción vea datos inconsistentes que puedan ser causados por la actualización de otra transacción concurrente en la misma fila de datos. Es un excelente optimizador de consultas, permite escribir funciones (procedimientos almacenados) en varios lenguajes (pl/pgsql, pl/perl, pl/python, pl/r), triggers, vistas, índices funcionales y parciales, tipos de datos extensibles. Es reconocido por su gran estabilidad y confiabilidad.

1.4.7. Entorno de Desarrollo Integrado

Un IDE[16] es un programa que integra varias herramientas que facilitan el desarrollo de software sobre uno o varios lenguajes de programación. Sus características más comunes son: editor de código, herramientas para traceo, consulta a bases de datos, depurador de código y construcción de interfaz gráfica. Entre los IDEs utilizados en nuestra universidad que trabajen con el lenguaje de C# se encuentran: Visual Studio .NET y el Mono Develop, pero este último es para el sistema operativo Linux, por lo que no se ajusta a nuestro trabajo que es desarrollado en Windows.

1.4.7.1. Visual Studio 2010 Ultimate

Visual Studio es un entorno de desarrollo integrado (IDE) para sistemas operativos basados en Windows, con desarrollo multiplataforma. Soporta varios lenguajes de programación tales como Visual C++, Visual C#, Visual J#, ASP.NET y Visual Basic.NET, entre otros. Visual Studio 2010 permite a los desarrolladores crear aplicaciones, sitios y servicios web en cualquier entorno que soporte la plataforma de desarrollo .NET. Así se pueden crear aplicaciones que se intercomuniquen entre estaciones de trabajo, páginas web y dispositivos móviles[26].

Microsoft Visual Studio 2010 Ultimate incluye potentes herramientas que simplifican todo el proceso

[16] Acrónimo en inglés de **Integrated Development Environment** (Entorno de Desarrollo Integrado)

de desarrollo de aplicaciones, de principio a fin. Los equipos pueden observar una mayor productividad y ahorro de costes al utilizar características de colaboración avanzadas, así como herramientas de pruebas y depuración integradas que le ayudarán a crear siempre un código de gran calidad.

Visual Studio 2010 Ultimate presenta IntelliTrace, una valiosa característica de depuración que hace que el argumento "no reproducible"sea cosa del pasado. Los evaluadores pueden archivar errores enriquecidos y modificables para que los desarrolladores puedan reproducir siempre el error del que se informe en el estado en el que se encontró. Otras características incluyen análisis de código estático, métricas de código y creación de perfiles. Este IDE incorpora herramientas avanzadas de pruebas para ayudar a garantizar la calidad del código en todo momento, beneficiándonos de las pruebas de IU codificadas, que automatizan la realización de pruebas de la interfaz de usuario en aplicaciones basadas en web y en Windows, así como de pruebas manuales, Test Professional, pruebas de rendimiento de web, pruebas de carga, cobertura de código y otras características completas que no se encuentran en otras ediciones de Visual Studio.

La plataforma de colaboración sobre la que se asienta la solución de administración de ciclo de vida de aplicaciones de Microsoft es Team Foundation Server (TFS) automatizando y simplificando el proceso de entrega de software, y proporciona rastreabilidad completa y la posibilidad de comprobar en tiempo real el estado de los proyectos (para todos los miembros del equipo) con potentes herramientas de elaboración de informes y paneles. Visual Studio 2010 Ultimate permite crea soluciones nuevas así como mejorar las aplicaciones ya existentes en una gran variedad de plataformas, entre las que se incluyen Windows, Windows Server, Web, Cloud, Office y SharePoint, entre otras, todo en un único entorno de desarrollo integrado; ofrece un conjunto completo de características de laboratorio de pruebas, incluido el aprovisionamiento de entornos a partir de plantillas, la configuración y el desmontaje de entornos virtuales y entornos de puntos de comprobación. La creación de aplicaciones de éxito requiere un proceso de ejecución uniforme que beneficie a todos los componentes del equipo. Las herramientas de administración del ciclo de vida de las aplicaciones (ALM) integradas en Visual Studio 2010 Ultimate contribuyen a que las organizaciones colaboren y se comuniquen de forma efectiva en todos los niveles, y a que se hagan una idea precisa del estado real del proyecto, lo que garantiza que se ofrezcan soluciones de gran calidad al tiempo que se reducen los costos. El Explorador de arquitectura de Visual Studio 2010 Ultimate ayuda a entender los activos de código existentes y otras interdependencias. Los diagramas por capas ayudan a garantizar el cumplimiento de la arquitectura y permiten validar artefactos de código con respecto al diagrama. Además, Visual Studio 2010 Ultimate admite los cinco diagramas de UML más comunes que conviven junto con su código.

1.4.8. Plataforma de desarrollo .NET

1.4.8.1. Framewok 4.0

.NET Framework es el entorno de trabajo de la tecnología .NET, englobándola completamente. Es un elemento indispensable dentro de la tecnología .NET, ya que es el marco de trabajo y la ejecución común de dicha tecnología.(Ve Figura 2) Los principales componentes de .NET Framework son los lenguajes de compilación, las bibliotecas de clases que incluye ADO.NET, ASP.NET, formularios Windows Forms y Windows Presentation Foundation (WPF) y CLR[17][27].

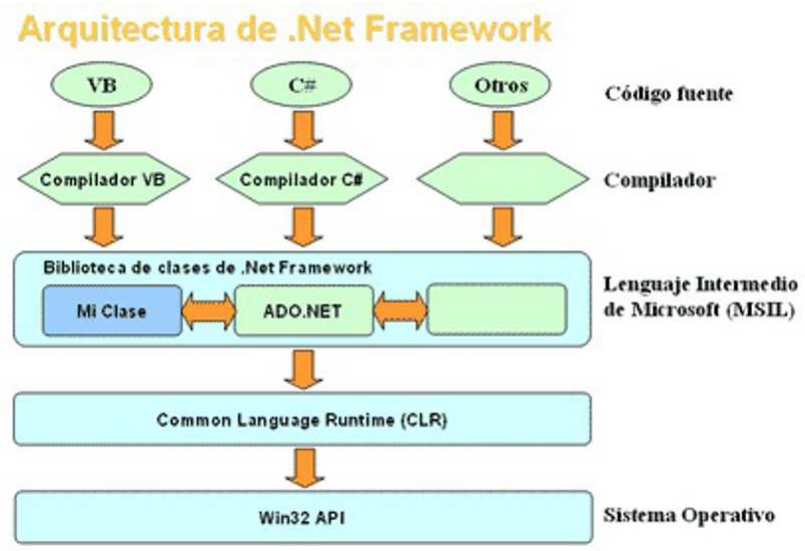

Figura 2: Arquitectura de .NET Framework.

.NET Framework 4.0 incorpora nuevas mejoras en relación con las versiones anteriores. Esta versión tiene mayor compatibilidad con equipos multinúcleos y un mejor rendimiento. En cuanto a ASP.NET tiene más control sobre HTML, identificadores de elemento y hojas CSS personalizadas facilitando la creación de formularios (Web Forms) que admiten optimización del motor de búsqueda y son conformes a los estándares. Contiene nuevos filtros de búsqueda, plantillas de entidad, datos dinámicos, tipos numéricos y archivos asignado en memoria. Framework 4.0 permiten a los desarrolladores aprovechar la eficacia de procesadores multinúcleo, con las nuevas características de programación en paralelo, como la compatibilidad con bucles en paralelo, biblioteca en paralelo de tareas (TPL) y estructuras de datos de coordinación mejorando la eficiencia sin tener que lidiar directamente con las dificultades de la creación y sincronización de hilos, usando además un lenguaje más natural para el programador. Además tiene mejoras en el filtrado de datos, se le incluyó nuevos atributos y plantillas de campo, la compatibilidad con las relaciones de mucho a mucho y con la herencia.

[17]Entorno Común de Ejecución, por sus siglas en inglés, **Common Language Runtime**

1.4.8.2. ASP.NET MVC

El marco de ASP.NET MVC[18] proporciona una alternativa al modelo de formularios Web Forms de ASP.NET para crear aplicaciones web. El marco de ASP.NET MVC es un marco de presentación de poca complejidad y fácil de comprobar que (como las aplicaciones basadas en formularios Web Forms) se integra con las características de ASP.NET existentes, tales como páginas maestras, la autenticación basada en pertenencia y roles, la administración de estado de sesión y perfil y la arquitectura del proveedor. El marco de ASP.NET MVC facilita la administración de la complejidad, al dividir una aplicación en modelo, vista y el controlador. No usa el estado de vista ni formularios basados en servidor haciendo que el marco de MVC sea ideal para los desarrolladores que deseen un control completo sobre el comportamiento de una aplicación. Utiliza un modelo controlador frontal que procesa las solicitudes de la aplicación web a través de un controlador único. Esto permite diseñar una aplicación que admite una infraestructura de enrutamiento avanzada, con un eficaz componente de asignación de direcciones URL que le permite compilar aplicaciones que tienen direcciones URL comprensibles y que admiten búsquedas. Proporciona una mayor compatibilidad con el desarrollo basado en pruebas (TDD). Todos los contratos principales del marco de MVC se basan en interfaz y se pueden probar mediante objetos ficticios, que son objetos simulados que imitan el comportamiento de objetos reales en la aplicación. Puede hacer una prueba unitaria de la aplicación sin tener que ejecutar los controladores en un proceso de ASP.NET, lo cual hace que las pruebas unitarias sean rápidas y flexibles. Funciona bien para las aplicaciones web en las que trabajan equipos grandes de desarrolladores y para los diseñadores web que necesitan un alto grado de control sobre el comportamiento de la aplicación.

1.4.9. Tecnologías relacionadas con la huella dactilar

1.4.9.1. SourceAFIS

El SourceAFIS es una alternativa de código abierto a software propietario existentes en el mercado. El sistema hace uso del él para realizar extracción y comparación de minucias en los procesos de verificación e identificación. Posee compatibilidad con la plataforma .NET y es soportado por el sistema operativo Windows desde XP hasta versiones superiores. Además posee independencia del lector utilizado para la captura del rasgo biométrico. Presenta SDK[19] para desarrollo en lenguajes como C#, Java y VB.NET. Ofrece una API simple e incorpora documentación en C# con referencias para la API y ejemplos en consola. Por todo lo antes expuesto se decide utilizar esta tecnología en el desarrollo del sistema, implicando menos costos y dificultades en su adquisición[28].

[18] Modelo Vista Controlador, por sus siglas en inglés **Model View Controller**

[19] Kit de desarrollo de software, conjunto de herramientas de desarrollo de software que le permite al programador crear aplicaciones para un sistema concreto.

1.4.9.2. DGM

El DGM[20] es un sistema para la interacción y control centralizado de dispositivos en aplicaciones web que facilita la integración de dispositivos de hardware en aplicaciones web del lado del cliente. Este sistema está compuesto por tres subsistemas fundamentales: el Servicio local para el manejo de dispositivos, el framework JavaScript DGMJS (Framework DGMJS) y el subsistema para el control centralizado de dispositivos. Estos subsistemas trabajan en conjunto, permitiendo la interacción y control de los dispositivos de hardware de una forma cómoda y elegante, basada en un desarrollo estandarizado que facilita la extensión de la aplicación sin grandes esfuerzos por parte de los desarrolladores. Incluir el DGM para el desarrollo del sistema permite realizar la captura de la huella para la inserción en la base de datos y el proceso de identificación[29].

1.4.9.3. Lector de huella dactilar

Es un dispositivo de seguridad encargado de detectar los relieves del dedo por medio de luz o por medio de sensores eléctricos, posteriormente genera una imagen digital la cual es enviada a la computadora y almacenada en una base de datos en los que se le asocia con la información de una persona. Cada vez que se coloca el dedo sobre la superficie óptica del lector, este envía la información y la computadora determina a que persona corresponde o si se trata de alguien no identificado.

El lector de huella digital tiene una función de seguridad, ya que por medio de él es posible identificar personas dentro de una empresa y llevar un estricto control como su hora de llegada y salida exactas, también para el acceso a ciertos lugares restringidos, para realizar transacciones bancarias, etc. Los sistemas de control de acceso por huella dactilar brindan seguridad en cuanto a la autenticidad de las personas, tiene una alta fiabilidad, alta aceptación por los usuarios mundialmente y alta facilidad de uso. Además, en el mercado son muchos los lectores que se encuentran por lo que el costo de su adquisición no es elevada.

1.5. Conclusiones Parciales

En la actualidad, en el mundo existen numerosos sistemas de control de acceso, implementados para satisfacer necesidades propias de una determinada institución o empresa, por lo que se decide realizar un sistema de acuerdo con las necesidades específicas de nuestro centro.

Se seleccionó XP como metodología de desarrollo, pues esta promueve valores y prácticas aumentando la productividad a la hora de gestionar un software, además de ser adecuada para equipos pequeños y proyectos de alto riesgo. Como lenguaje de modelado se eligió UML, haciendo uso del Visual Paradigm for UML, para diseñar y visualizar los elementos del software. Entre las herramientas que se

[20] Acrónimo en inglés de **Device Grid Manager** (Administrador de Dispositivos en Red)

decidieron para la aplicación web se encuentra Visual Studio 2010 Ultimate por ser un entorno de desarrollo integrado multiplataforma, que soporta varios lenguajes de programación como son C Sharp y ASP.NET, que son los lenguajes en los que están implementados otros componentes que se integran con la aplicación. Como gestor de base datos se selecciona PostgreSQL, por ser un potente gestor estable y confiable, además de ser libre y correr sobre varias plataformas. Todas estas tecnologías juntas aseguran una codificación robusta y estable.

Capítulo 2: Propuesta solución

Introducción

En este capítulo se propone una solución al problema científico, utilizando la metodología de desarrollo de software XP. De las fases que presenta XP, este capítulo persigue como objetivo mostrar la evolución de la solución durante las fases iniciales: Exploración y Planificación, además de mostrar los diversos artefactos generados por la misma.

2.1. Modelo de dominio

La creación de un modelo del dominio posibilita identificar conceptos asociados con el entorno en que se desarrolla la solución y mostrar las relaciones entre ellos. El modelo de dominio de la figura 3 muestra las relaciones que existen entre los principales conceptos que componen el Sistema de Control de Acceso, destacándose los distintos subsistemas y los aspectos organizativos que deben ser tomados en cuenta para su desarrollo.

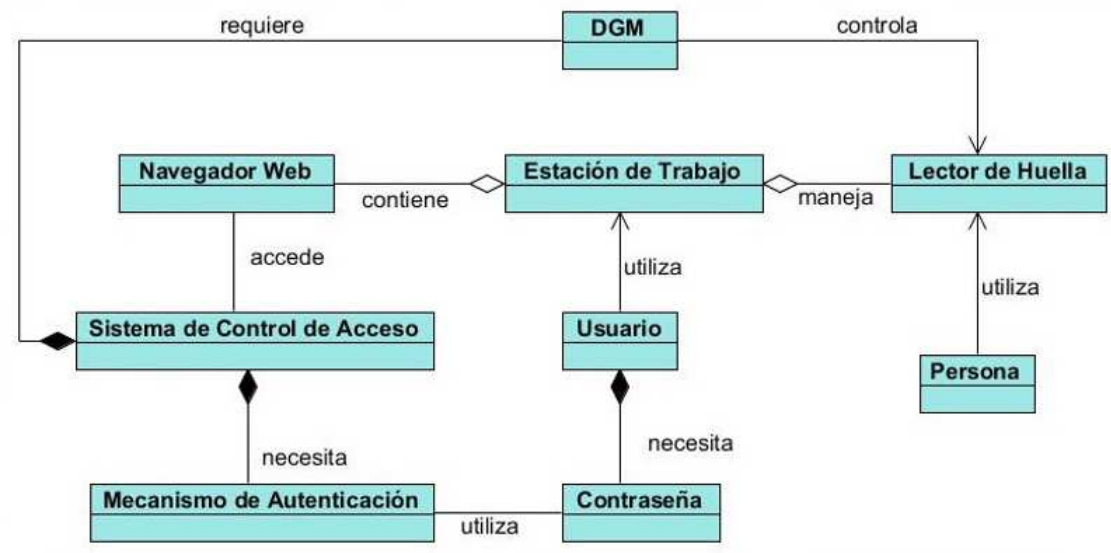

Figura 3: Modelo de dominio.

2.1.1. Conceptos asociados al modelo de dominio

- **Estación de trabajo:** Computadora con la que el usuario va a trabajar.
- **Navegador Web:** Interfaz gráfica que permite visualizar la información que contiene la página web con la que interactúa el usuario.
- **Sistema de Control de Acceso:** Encargado de gestionar todas las acciones.

- **Mecanismo de Autenticación:** Mecanismo que permite comprobar que el usuario esté autorizado a acceder al sistema.
- **Contraseña:** Clave que solo el usuario conoce, para decidir si es quien dice ser.
- **Usuario:** Persona que interactúa con el sistema.
- **Lector de Huella Dactilar:** Dispositivo conectado a la estación de trabajo para la captura de la huella.
- **DGM:** Sistema que trabaja local, con el objetivo principal de manejar los dispositivos (lector de huella) en la web.
- **Persona:** Individuo del centro a identificar.

2.2. Exploración

2.2.1. Historia de usuario

En el entorno de XP las historias de usuario son utilizadas para describir las funcionalidades que serán añadidas al sistema, son asignadas al desarrollador encargado de la programación con un número de horas de desarrollo estimado, descritas por el cliente y el analista en conjunto. Las historias de usuario guían la construcción de los test de aceptación (casos de prueba).

	Historia de Usuario
Número: 1	**Nombre de HU:** Autenticar usuario
Modificación de HU Número: Ninguna	
Usuario:	**Puntos reales:**
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Medio
Puntos estimados: 2	**Iteración asignada:** 1
Programador responsable: Ernesto Sabuquet Hurtado	
Descripción: Consiste en autenticar el usuario en el sistema, introduciendo el usuario y la contraseña, para validar que puede entrar el sistema. Esto está organizado por roles, cada rol puede ejecutar un conjunto de acciones.	
Observaciones: En caso de que la autenticación sea fallida, se notifica enviando un mensaje de error, si es correcta entra al sistema satisfactoriamente.	

Tabla 4: Historia usuario: Autenticar usuario.

Historia de Usuario	
Número: 2	**Nombre de HU:** Realizar conexión con el DGM
Modificación de HU Número: Ninguna	
Usuario:	**Puntos reales:**
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Medio
Puntos estimados: 1	**Iteración asignada:** 1
Programador responsable: Osmayda Silverio Leyva	
Descripción: El sistema intenta establecer conexión con el DGM (controlador de dispositivos).	
Observaciones: En caso de no establecer conexión, se mostrará un mensaje de error.	

Tabla 5: Historia usuario: Realizar conexión con el DGM.

Historia de Usuario	
Número: 3	**Nombre de HU:** Enrolar personal del centro
Modificación de HU Número: Ninguna	
Usuario: Administrador y rol que tenga asignado ese permiso.	**Puntos reales:**
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Medio
Puntos estimados: 2	**Iteración asignada:** 1
Programador responsable: Ernesto Sabuquet Hurtado	
Descripción: Consiste en adicionar tanto la huella izquierda como la derecha, el nombre, los apellidos, solapín y carnet de identidad de una persona del centro a la base de datos.	
Observaciones: En caso de que no sea administrador del sistema no puede añadir a ninguna persona a la base de datos.	

Tabla 6: Historia usuario: Enrolar personal del centro.

Historia de Usuario	
Número: 4	**Nombre de HU:** Buscar persona

Modificación de HU Número: Ninguna	
Usuario: Todos los roles	**Puntos reales:**
Prioridad en negocio: Media	**Riesgo en desarrollo:** Medio
Puntos estimados: 1	**Iteración asignada:** 1
Programador responsable: Osmayda Silverio Leyva	
Descripción: Consiste en buscar los datos de una persona, mediante la introducción del # de solapín.	
Observaciones: En caso de que no sea administrador del sistema, no podrá realizar esta acción y si no es encontrado el # de solapín en la base de datos, mostrará un mensaje de error.	

Tabla 7: Historia usuario: Buscar persona.

	Historia de Usuario
Número: 5	**Nombre de HU:** Identificar personal del centro
Modificación de HU Número: Ninguna	
Usuario: Todos los roles	**Puntos reales:**
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Medio
Puntos estimados: 2	**Iteración asignada:** 2
Programador responsable: Ernesto Sabuquet Hurtado	
Descripción: Se verifica que la huella de la persona que se va a identificar, se encuentre en la base dato, para poder tener acceso al centro.	
Observaciones: Si no se encuentra la huella en la base de datos, mostrará un mensaje de error; y esta operación la puede realizar tanto el técnico como el administrador.	

Tabla 8: Historia usuario: Identificar personal del centro.

	Historia de Usuario
Número: 6	**Nombre de HU:** Crear rol
Modificación de HU Número: Ninguna	
Usuario: Administrador y rol que tenga asignado ese permiso.	**Puntos reales:**
Prioridad en negocio: Media	**Riesgo en desarrollo:** Medio
Puntos estimados: 1	**Iteración asignada:** 2

Programador responsable: Osmayda Silverio Leyva
Descripción: Consiste en adicionar un nuevo rol al sistema.
Observaciones: En caso de que no sea administrador del sistema, no podrá realizar esta acción.

Tabla 9: Historia de usuario: Crear rol.

	Historia de Usuario
Número: 7	**Nombre de HU:** Eliminar rol
Modificación de HU Número: Ninguna	
Usuario: Administrador y rol que tenga asignado ese permiso.	**Puntos reales:**
Prioridad en negocio: Media	**Riesgo en desarrollo:** Medio
Puntos estimados: 1	**Iteración asignada:** 2
Programador responsable: Osmayda Silverio Leyva	
Descripción: Consiste en eliminar algún rol que ya no sea necesario.	
Observaciones: En caso de que no sea administrador del sistema, no podrá realizar esta acción.	

Tabla 10: Historia usuario: Eliminar rol.

	Historia de Usuario
Número: 8	**Nombre de HU:** Crear usuario
Modificación de HU Número: Ninguna	
Usuario: Administrador y rol que tenga asignado ese permiso.	**Puntos reales:**
Prioridad en negocio: Media	**Riesgo en desarrollo:** Medio
Puntos estimados: 1	**Iteración asignada:** 2
Programador responsable: Ernesto Sabuquet Hurtado	
Descripción: Consiste en crear un usuario adicionando los datos y seleccionando el rol que va a desempeñar.	
Observaciones: En caso de que no sea administrador del sistema, no podrá realizar esta acción.	

Tabla 11: Historia de usuario: Crear usuario.

	Historia de Usuario
Número: 9	**Nombre de HU:** Eliminar usuario
Modificación de HU Número: Ninguna	
Usuario: Administrador y rol que tenga asignado ese permiso.	**Puntos reales:**
Prioridad en negocio: Media	**Riesgo en desarrollo:** Medio
Puntos estimados: 1	**Iteración asignada:** 2
Programador responsable: Ernesto Sabuquet Hurtado	
Descripción: Consiste en eliminar el nombre del usuario que ya no va a trabajar con la aplicación.	
Observaciones: En caso de que no sea administrador del sistema, no podrá realizar esta acción.	

Tabla 12: Historia de usuario: Eliminar usuario.

2.2.2. Requisitos no funcionales

Los requerimientos no funcionales, no se refieren directamente a las funciones específicas que proporciona el sistema, sino a las propiedades emergentes de estos como la usabilidad, fiabilidad, entre otros. Estos son de gran importancia ya que el incumplimiento de uno de ellos puede significar la inutilización del sistema entero.

- Interfaz externa o apariencia

RNF1: El diseño de la aplicación será agradable para el usuario.

RNF2: El idioma utilizado será español.

RNF3: Los elementos del sistema contendrán claro y bien estructurados los datos, y al mismo tiempo permitirán la interpretación correcta e inequívoca de la información.

- Software

RNF4: Computadora personal con plataforma de sistema operativo Microsoft Windows XP en adelante, pero de 32 bit.

RNF5: Se requiere que estén instalado en la PC cliente los drivers del lector de huella.

RFN6: La computadora donde esté instalada la aplicación requiere de .NET framework 4.0.

RNF7: Requiere de la instalación del DGM.

- Restricciones del diseño

RNF8: La herramienta de desarrollo será Visual Studio 2010.

RNF9: Se debe desarrollar el sistema en el lenguaje C# con ASP.NET MVC3 y como gestor de base de datos PostgreSQL 9.1.1.

- Fiabilidad

RNF10: El sistema debe ser seguro y que limiten los permisos según el tipo de usuario que acceda a la aplicación.

- Usabilidad

RNF11: La aplicación deberá permitir la incorporación de nuevas funcionalidades.

- Requerimiento mínimo de hardware.

RNF12: Lector de huella dactilar incorporado a la PC cliente.

- Portabilidad

RNF13: La aplicación debe ser compatible con cualquiera de los lectores de huella dactilar siguientes: Digital Person, DactyScan (40, 84, 26), MC500, MSC500, VisaScan3, PoliScan2, MultiScan500, MultiScan1000, DactyScan 15.

2.2.3. Metáfora

La metáfora proporciona un contexto al equipo para entender los elementos básicos y sus relaciones, proporcionando integridad conceptual. Es importante que el cliente y los desarrolladores estén de acuerdo y conozcan la metáfora a usar, para así poder discutir y trabajar en los mismos términos de una forma más precisa. La metáfora es una vía fácil y concreta para explicar cómo funciona el sistema teniendo contenido suficiente para guiar la arquitectura del proyecto.

La solución para gestionar el control de acceso a las instalaciones del CISED mediante huellas dactilares puede ser utilizada por cualquier instalación que tenga una infraestructura adecuada y requiera de este sistema. El control de acceso debe ser lo más seguro posible; para ello se va a desarrollar una aplicación web que permita verificar que las personas identificadas estén autorizadas a acceder a las instalaciones del centro.

Está conformado por un servidor AFIS, encargado de realizar la operación de extracción de minucias y macheo de las huellas; un servidor de base de datos para el manejo de permisos en los usuarios que interactúan con la aplicación; el servidor web Internet Information Server que es donde estará montada

la aplicación, accediendo a él por una dirección URL a través del navegador web instalado en la estación cliente; y un componente DGM para el manejo del lector de huella en la web, instalado en la estación cliente también. El sistema confirmará que la persona realmente pertenece al centro a través de la comparación de huella dactilar realizada por el servidor AFIS.

2.3. Arquitectura

2.3.1. Estilo Arquitectónico

La figura 4 muestra el estilo arquitectónico cliente-servidor propuesto para la implementación del sistema de control de acceso. El estilo arquitectónico cliente-servidor genérico tiene dos tipos de nodo en la red: clientes y servidores, por lo que se hace referencia a veces como dos capas. Por lo general las aplicaciones web realizadas en ASP.NET MVC siguen un estilo cliente-servidor pero al mismo tiempo el servidor sigue un estilo Modelo Vista Controlador.

Teniendo en cuenta las características del sistema, cuyo papel fundamental es mediar entre la aplicación a desarrollar y la base de datos, se identifica una capa cliente que contiene el controlador de dispositivo DGM y el navegador instalado para acceder a la aplicación y la capa servidora, donde se encuentra la aplicación y las bases de datos con las que cuenta el sistema.

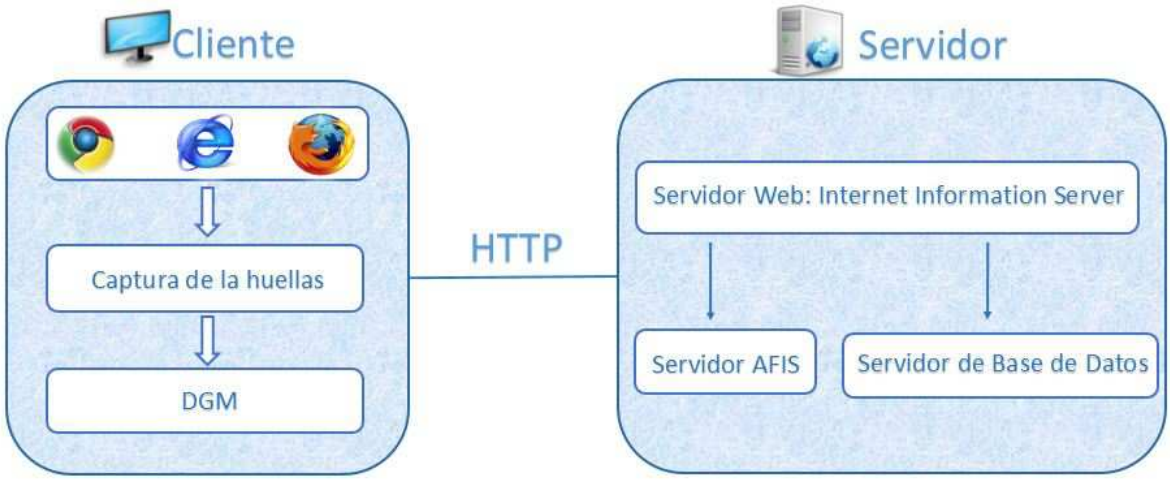

Figura 4: Diagrama de arquitectura.

2.3.2. Patrón arquitectónico

Un patrón arquitectónico de software describe un problema particular y recurrente del diseño, que surge en un contexto específico, y presenta un esquema genérico y probado de su solución.

Para la realización del sistema informático en cuestión, como se muestra en la figura 5 se utiliza el

patrón arquitectónico MVC, el cual es un patrón de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos. Plantea la separación del problema en tres capas: la capa del modelado que representa la realidad; la capa controladora la cual conoce los métodos y atributos del modelo, recibiendo y realizando lo que el usuario quiere hacer; y la capa vista que muestra un aspecto del modelo y es utilizada por la capa anterior para la interacción con el usuario.

La vista es la representación gráfica del modelo, es decir, transforma el modelo en una página web, permitiendo al usuario interactuar con la aplicación. ASP.NET crea una carpeta Views, la cual contendrá por carpetas las vistas relacionadas con el negocio del sistema.

El controlador es la capa encargada de manejar y responder las solicitudes del usuario, procesando la información necesaria y modificando los cambios convenientes en la vista o el modelo.

El modelo representa la información con la cual va a trabajar la aplicación, es decir, la lógica del negocio, que ASP.NET crea la clase AccountModels por defecto, con varias funcionalidades dentro de la carpeta Models, esta clase pude contener todas las validaciones relacionadas con el modelo de dato del sistema o se pueden crear nuevas clases necesarias dentro de dicha carpeta.

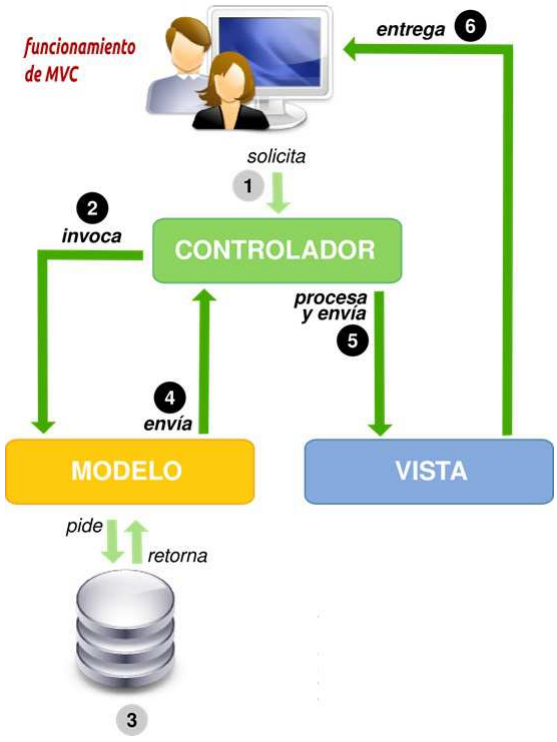

Figura 5: Patrón arquitectónico MVC.

2.4. Planeación

2.4.1. Plan de entrega

Después de confeccionar las Historias de Usuario, se realiza una estimación del esfuerzo necesario para marcar cuánto tiempo se demora la implementación de cada una de ella, incluyendo las fechas de salida de las versiones funcionales del producto como se muestra en la tabla 13.

Entregable	**Iteración**	**Fin de iteración**
Mecanismo-Autenticación	1	Enero 2013
Buscar y Enrolar persona	1	Febrero 2013
Administrar rol	2	Marzo 2013
Administrar usuario	2	Abril 2013

Tabla 13: Plan de Entrega.

2.4.2. Estimación de tiempo

Los programadores estiman el tiempo que necesitan para llevar a cabo cada Historia de usuario. El valor del tiempo se expresa por semana como se muestra en la tabla 14.

Historia usuario	**Estimación(semana)**
Autenticar usuario	2
Realizar conexión con el DGM	1
Enrolar personal del centro	3
Buscar persona	1
Identificar personal del centro	2
Crear rol	2
Eliminar rol	2
Crear usuario	3
Eliminar usuario	2

Tabla 14: Estimación de tiempo.

2.4.3. Plan de iteraciones

Como parte del ciclo de vida de un proyecto usando la metodología XP se crea el plan de duración de cada una de las iteraciones que se han definido, que tiene como objetivo mostrar la duración y el orden en que serán implementadas las historias de usuario dentro de cada iteración. Para la solución se han definido 9 historias de usuario divididas en 2 iteraciones, de acuerdo a los intereses del cliente, para una duración total del proyecto de 18 semanas como se muestra en la tabla 15.

Iteración	**No.HU**	**Historia de usuario**	**Duración estimada**
Iteración 1	HU1	Autenticar usuario	7 semanas
	HU2	Realizar conexión con el DGM	
	HU3	Enrolar personal del centro	
	HU4	Buscar persona	
Iteración 2	HU5	Identificar personal de centro	11 semanas
	HU6	Crear rol	
	HU7	Eliminar rol	
	HU8	Crear usuario	
	HU9	Eliminar usuario	

Tabla 15: Plan de Iteraciones.

2.5. Diseño

2.5.1. Tarjetas CRC

Las tarjetas CRC (Clase-Responsabilidad-Colaboración) son una herramienta para estimar si el conjunto de clases obtenida hasta este punto (o versiones posteriores del mismo) responden bien a las necesidades dinámicas del sistema. Constituyen una primera aproximación a los objetos que luego se van a utilizar, sobre la que es muy fácil plantear criterios básicos de viabilidad. Se crean a partir de la lista de clases básicas. Si resultan evidentes, se incluyen responsabilidades y colaboraciones. A continuación la tablas 16 muestra una de las tarjetas, encontrándose el resto en el anexo I.

AccountController	
Autenticación del usuario para permitir el acceso a los diferentes módulos del sistema.	UsersManager

Tabla 16: Tarjeta CRC: AccountController.

2.5.2. Patrones de diseño

Los patrones de diseño, son el esqueleto de las soluciones a problemas comunes en el desarrollo de software. No requiere una implantación con lenguajes de programación por lo que ofrece, un idioma común entre programadores, brindando una solución ya probada y documentada a problemas de desarrollo de software que están sujetos a contextos similares. Debemos tener presente los siguientes elementos de un patrón: su nombre, el problema (cuando aplicar un patrón), la solución (descripción abstracta del problema) y las consecuencias (costos y beneficios)[30]. Cabe destacar que existen varias familias de patrones de diseño, por lo que se escogió trabajar con dos de los más usados, los de la familia GOF[1] y los de la familia GRASP[2].

Los patrones de diseño **GOF** se clasifican en 3 grandes categorías: creacionales, estructurales y de comportamiento. En el desarrollo de la aplicación se utilizó el patrón Método de fabricación, por sus siglas en inglés **Factory Method** incluido en la categoría de creacionales.

Método de fabricación: El patrón se aplica centralizando la solución en una clase constructora (AfisInstanceFactory), la cual utilizará la creación de objetos de un subtipo, de un tipo determinado, ocultando al usuario la casuística para elegir el subtipo que crear.

Los patrones de la familia **GRASP**, son considerados como "buenas prácticas" de aplicación recomendable en el diseño de software. En el desarrollo de la solución se utilizaron los siguientes patrones de esta familia:

Experto: El GRASP de experto en información es el principio básico de asignación de responsabilidades. Este indica que la responsabilidad de la creación de un objeto o la implementación de un método, debe recaer sobre la clase que conoce toda la información necesaria para crearlo, este caso la clase controladora AdministradorController. El uso de este patrón propicia que los objetos se valen de su propia información para hacer lo que se les pide, lo que favorece al hecho de tener sistemas más robustos y de fácil mantenimiento.

Creador: El patrón creador nos ayuda a identificar quien debe ser el responsable de la creación de una nueva instancia de alguna clase, en este caso, la clase *AccuontContrller*[3], es la encargada de la

[1]Pandilla de los cuatros, por sus siglas en inglés, **Gang of Four**

[2]Patrones generales de software para asignar responsabilidades, por sus siglas en inglés, **General Responsibility Asignment Software Patterns**

[3]Se le hace un cambio de tipo de letra porque es una clase propia de la solución desarrollada en Visual Studio 2010.

lógica del negocio, por lo que en ella se crean un conjunto de objetos relacionados con la conexión y además usa directamente las instancias creadas de algún objeto.

Bajo Acoplamiento: Permite tener menos dependencia entre las clases. De tal forma, que en caso de producirse alguna modificación en alguna de las clases, se tiene la mínima repercusión posible en el resto de ellas, potenciando la reutilización, y disminuyendo la dependencia entre las clases.

Controlador: El patrón controlador sirve como intermediario entre una determinada interfaz (Administrador) y el algoritmo que la implementa, de tal forma que es la que recibe los datos del usuario y la que los envía a las distintas clases (AdministradorController) según el método llamado. El uso de este patrón es aplicable debido a que, la lógica de negocio está separada de la capa de presentación, aumentando la reutilización de código y a la vez teniendo un mayor control.

2.5.3. Diseño de la Base de Datos

Una base de datos es una serie de datos relacionados que forman una estructura lógica, reconocible desde un programa informático, donde dicha estructura no sólo contiene los datos en sí, sino la forma en la que se relacionan. Para la creación de una base de datos se debe realizar primeramente el modelado de la misma, de tal forma que se haga una representación abstracta de la información que se desea almacenar. Haciendo uso de diagrama entidad relación de la herramienta CASE Visual Paradigm for UML se realizó el modelo de datos que se muestra en la figura 6.

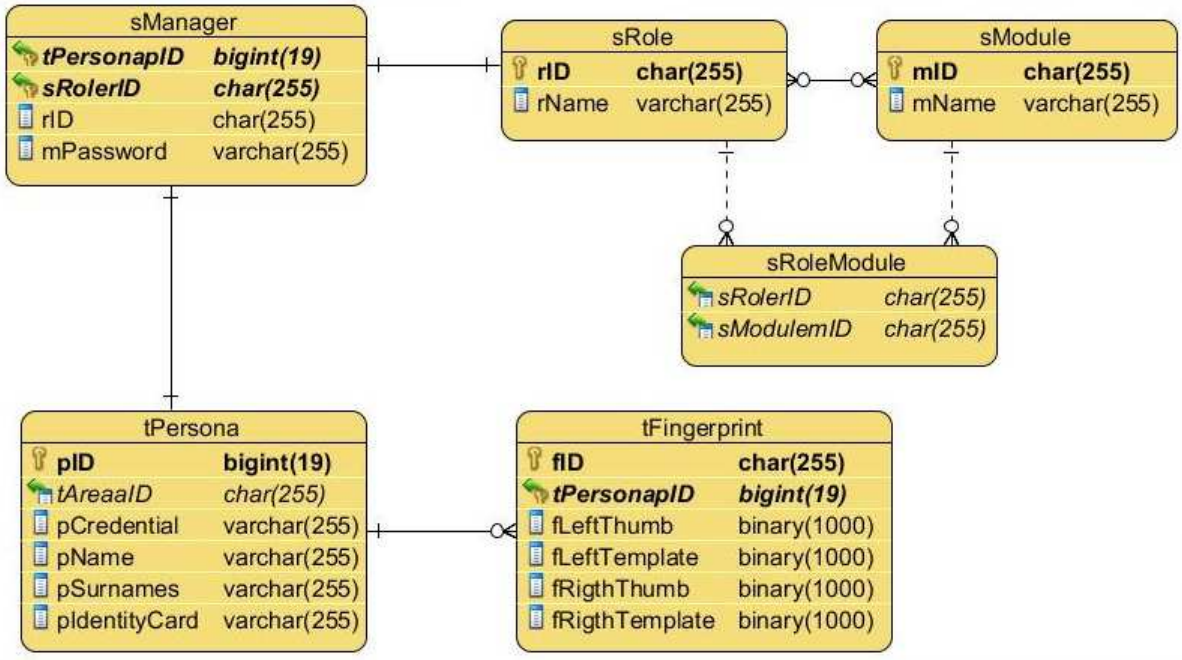

Figura 6: Modelo de datos.

2.6. Conclusiones Parciales

Luego del estudio de las herramientas, tecnologías y elementos tangibles asociados al dominio de la solución, se determinan las bases del porqué la elección de un modelo de dominio, el cual agruparía todos los conceptos asociados de la solución propuesta así como las relaciones entre dichos conceptos. Se definieron las principales funcionalidades, recogidas en uno de los artefactos que propone XP, en este caso, las historias de usuarios; así como los requisitos no funcionales, los cuales brindan las cualidades que se deben tener en cuenta para desarrollar una solución adecuada. Se precisó el plan de iteraciones, el plan de entrega y la estimación de tiempo para lograr que las historias de usuarios se realicen en el tiempo considerado y la aplicación sea entregada en la fecha acordada. Se confeccionó la metáfora del sistema, dando a paso a establecer la arquitectura con la cual se implementará la solución, el patrón arquitectónico y los patrones de diseño a utilizar en el desarrollo de la misma. Además, se elaboraron las tarjetas CRC y se confeccionó el modelo de datos, quedando preparada una representación abstracta de la información que se almacenará.

Capítulo 3: Implementación y Prueba

Introducción

Seguido de la fase de Exploración y Planificación, XP define las fases Iteraciones a primera liberación y Producción. En la Planificación se definieron las iteraciones y en cada iteración se diseñan, prueban y codifican cada una de las historias de usuario. Esto permite aumentar la seguridad de evitar efectos colaterales no deseados a la hora de realizar modificaciones y refactorizaciones dividiendo dichas pruebas en dos grupos: pruebas de aceptación destinadas a evaluar si al final de una iteración se consiguió la funcionalidad requerida, diseñada por el cliente final y pruebas unitarias, encargadas de verificar el código y diseñadas por los programadores. El objetivo que se persigue con la elaboración de este capítulo es mostrar la evolución de la solución durante las fases de Iteraciones a primera liberación y Producción. Además de explicar el diseño de la solución, a través del diagrama de componentes y del diagrama de despliegue, ofreciendo una panorámica del funcionamiento de la solución.

3.1. Diagrama de componentes

Los diagramas de componentes describen los elementos del sistema y sus relaciones, representan todos los tipos de elementos de software que entran en la fabricación de aplicaciones informáticas. Las relaciones de dependencia se utilizan en los diagramas de componentes para indicar que un componente utiliza los servicios ofrecidos por otro componente. Estos diagramas incorporan las dependencias entre componentes de software, incluyendo componentes de código fuente, componentes del código binario, y componentes ejecutables como se muestra en la figura 7.

3.1.1. Descripción del diagrama de componente

El diagrama muestra cómo están separados los datos en tres principales componentes distintos. Como la aplicación está implementada en ASP.NET MVC se dividen los componentes por paquetes.

El paquete Vista incluye todos los componentes de las vistas para cada funcionalidad del sistema. Los componentes Enrolar e Identificar hacen uso del componente DGM para la captura de la huella de la persona.

El paquete Controlador contiene los componentes relacionados con cada componente del paquete Vista. Los componentes de este paquete son los encargados de manejar y responder a las solicitudes enviadas por el usuario.

El paquete Modelo contiene los componentes del negocio. AccountModel, es un componente creado

por defecto por ASP.NET MVC, la cual utiliza la dll.NpgSQL para poder conectarse a la base de datos realizada en Postgre. Todos los componentes de este paquete son conectados a dicha base de datos.

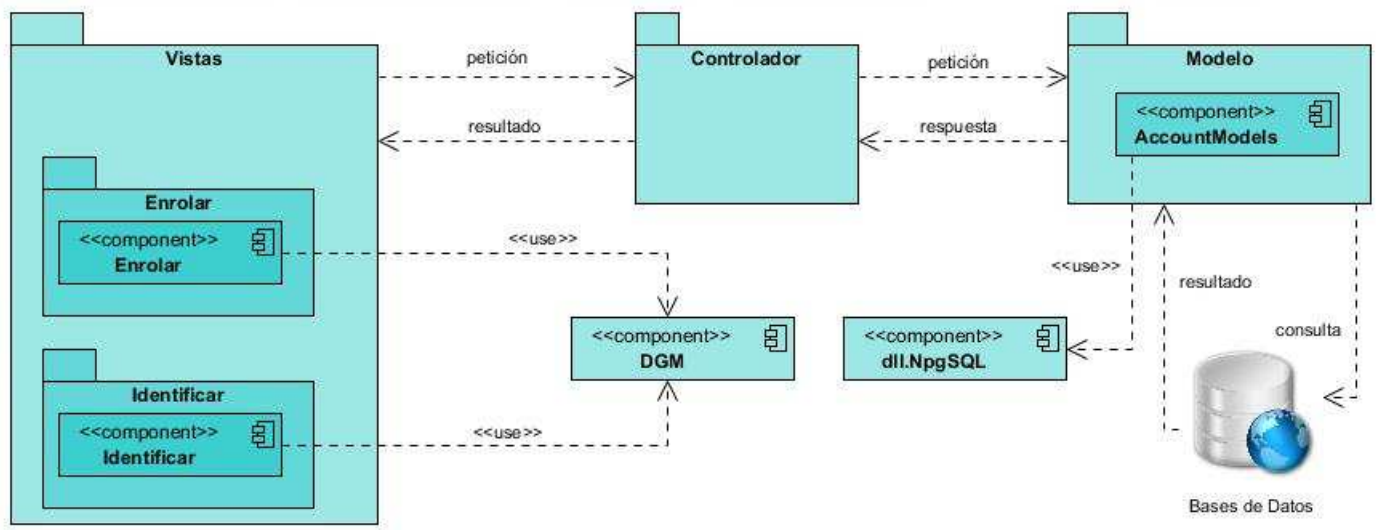

Figura 7: Diagrama de componente.

3.2. Diagrama de despliegue

El Diagrama de Despliegue es utilizado para modelar el conjunto de hardware utilizado en la implementación del sistema y las relaciones entre sus componentes. Estos diagramas son utilizados para modelar entre otros sistemas, los sistemas cliente/servidor. La siguiente figura 8, muestra cómo será desplegado el sistema implementado.

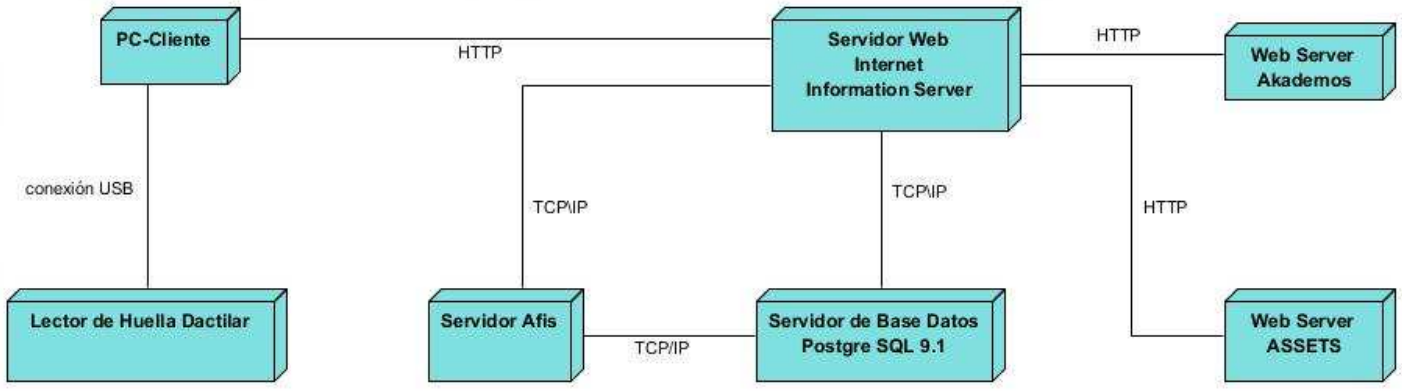

Figura 8: Diagrama de despliegue.

3.3. Iteraciones

En esta fase ocurren varias iteraciones sobre el sistema antes de ser entregado. Al principio de cada iteración se realizan las tareas necesarias de análisis y al final de la última iteración el sistema estará listo para entrar en producción. En esta fase es donde se da cumplimiento al Plan de Iteración, teniendo en cuenta para la confección de este artefacto los siguientes elementos: historias de usuario no abordadas, velocidad del proyecto, pruebas de aceptación no superadas en la iteración anterior y tareas no terminadas en la iteración anterior. Todo el trabajo de la iteración es expresado en tareas de programación, cada una de ellas es asignada a un programador como responsable, pero llevadas a cabo por parejas de programadores. Al finalizar esta fase el cliente estará listo para realizar las Pruebas de aceptación.

3.3.1. Tareas de ingeniería

Las tareas de ingeniería es donde se expresa el trabajo de la iteraciones, las cuales se realizan para especificar las acciones llevadas a cabo por los programadores en cada historia de usuario, ya que éstas no ofrecen el nivel de detalle requerido para saber qué implementar. Según el Plan de iteraciones las historias de usuario se agruparon en dos iteraciones. A continuación la tabla 17 muestra las tareas de ingeniería derivadas de cada historia de usuario por iteración.

Iteración	**Historia de usuario**	**Tareas**
Iteración 1	Autenticar usuario	1. Crear el diseño de prototipo. 2. Crear la interfaz de autenticación. 3. Verificar la autenticación del usuario por contraseña. 4. Garantizar que dado el rol del usuario se muestren solamente los módulos que tiene asignado.
	Realizar conexión con el DGM	1. Instalar el componente DGM. 2. Establecer conexión.

	Enrolar personal del centro	1. Crear el diseño de prototipo. 2. Crear la interfaz de enrolar. 3. Validar que los datos introducidos sean los correctos. 4. Guardar los datos introducidos. 5. Crear campos dinámicos para insertar nuevos datos al pulsar Enrolar.
	Buscar persona	1. Crear el diseño de prototipo. 2. Crear la interfaz de buscar persona.
Iteración 2	Identificar personal del centro	1. Crear el diseño de prototipo. 2. Crear la interfaz de identificar al personal del centro.
	Crear rol	1. Crear el diseño de prototipo. 2. Crear la interfaz de crear rol. 3. Guardar los datos introducidos. 4. Crear campos dinámicos para insertar nuevos datos al pulsar Crear. 5. Garantizar que se actualicen las listas desplegables Roles con los roles existentes hasta el momento.
	Eliminar rol	1. Crear el diseño de prototipo. 2. Crear la interfaz de eliminar rol. 3. Guardar los datos introducidos. 4. Garantizar que se actualicen las listas desplegables Roles con los roles existentes hasta el momento.

	Crear usuario	1. Crear el diseño de prototipo. 2. Crear la interfaz de crear usuario. 3. Guardar los datos introducidos. 4. Crear campos dinámicos para insertar nuevos datos al pulsar Crear. 5. Garantizar que se actualicen las listas desplegables Usuarios, con los usuarios existentes hasta el momento.
	Eliminar usuario	1. Crear el diseño de prototipo. 2. Crear la interfaz de eliminar usuario. 3. Crear campos dinámicos para insertar nuevos datos al pulsar Eliminar. 4. Garantizar que se actualicen las listas desplegables Usuarios, con los usuarios existentes hasta el momento.

Tabla 17: Tareas de ingeniería.

3.3.2. Tareas detalladas

La tabla 18 muestra una tarea de ingeniería detallada y el resto se encuentra en el anexo II.

Tarea de Ingeniería	
Número: 1	**Nombre de HU:** Autenticar usuario
Nombre de la tarea: Crear el diseño de prototipo.	
Tipo de tarea: Desarrollo	**Puntos estimados (días):** 3
Fecha inicio: 14/01/2013	**Fecha fin:** 15/01/2013
Programador responsable: Osmayda Silverio Leyva	
Descripción: Esta tarea da una visión exacta de cómo será representada la interfaz de autenticar usuario, facilitando la comprensión a la pareja desarrolladora.	

Tabla 18: Tarea detallada 1 de la HU1.

3.4. Estándares de codificación

Los estándares de codificación son utilizados para institucionalizar buenas prácticas, haciendo recomendaciones de diseño para lograr mayores niveles de calidad en la elaboración del producto.

3.4.1. Pautas generales

1. Evitar nombres totalmente en MAYÚSCULAS o en minúsculas a menos que estos sean de una sola palabra.
2. Nunca usar nombres que comiencen con caracteres numéricos.
3. Todos los nombres a utilizar deben ser específicos y deben tener la extensión necesario para ser entendidos.
4. Evitar el uso de abreviaturas a menos que el nombre completo sea excesivo. En los casos necesarios, las abreviaturas deben ser totalmente conocidas y aprobadas por todos los integrantes.
5. Nunca usar palabras reservadas para nombres.

3.4.2. Formato

Colocar la sentencia "using" en la parte superior del archivo. El grupo de los espacios de nombres de .NET colocarle por encima de los espacios de nombres particulares y todos ordenados alfabéticamente. Ejemplo:

```
using System;

using System.Collections

using System.Data;

using System.Data.SqlClient;

using System.Xml;

using Gustozzi.BE;

using Acceso_a_Datos;
```

3.4.3. Comentarios

1. No utilizar marcos de asteriscos. Ejemplo:
   //**********

// Cuadro de comentarios
//**********

2. No utilizar comentarios para explicar código obvio.

3.4.4. Uso del lenguaje

1. Nunca omitir los modificadores de acceso. Declarar explícitamente todos los métodos con su apropiado modificador de acceso en lugar de dejarlo con el de por defecto. Ejemplo:
   // Mal! void Escribir_Evento(string mensaje)
   {...}
   // Bien! private void Escribir_Evento(string mensaje)
   {...}
2. Tratar de inicializar las variables cuando se declare. Ejemplo:
   string strVariable System.String

3.4.5. Control de excepciones

1. No usar los bloques try/catch para control de flujos.
2. Solo capturar las excepciones cuando van a ser controladas.
3. Nunca declarar un bloque match vacío.
4. Evitar anidar bloques try/catch en otro bloque catch.
5. Ordenar los filtros de excepciones de más específica a más genérica.
6. Evitar el relanzamiento de excepciones.

3.5. Interfaz gráfica

Una interfaz gráfica es cualquier medio por el cual uno puede interactuar con una computadora a través de algún tipo de software gráfico. Comúnmente, esto se consigue a través del control mediante el teclado y el mouse de cursores, menús, ventanas, íconos y cajas de diálogo, pero puede tomar cualquier forma imaginable. A continuación la figura 9 muestra la interfaz principal para acceder al resto de la aplicación dependiendo del rol que desempeñe el usuario autenticado. El resto de las principales interfaces se encuentran en el anexo III.

Figura 9: Interfaz gráfica principal.

3.6. Pruebas

XP anima a probar constantemente tanto como sea posible. Esto permite aumentar la calidad de los sistemas reduciendo el número de errores no detectados y disminuyendo el tiempo transcurrido entre la aparición de un error y su detección. Las pruebas que XP propone se dividen en dos grupos: pruebas unitarias, encargadas de verificar el código y diseñada por los programadores; pruebas de aceptación, destinadas a evaluar si al final de una iteración se consiguió la funcionalidad requerida diseñadas por el cliente final.

3.6.1. Pruebas de aceptación

Las pruebas de aceptación también conocidas como pruebas del cliente, son utilizadas para probar que las historias de usuario han sido implementadas de forma correcta al final de cada iteración y por lo tanto comprueban que las funcionalidades del sistema se encuentran en relación con las historias de usuario definidas. A continuación la tabla IV muestra un caso de prueba de aceptación y el resto se encuentra en el anexo IV.

	Caso de prueba de aceptación
Código: HU1CP1	**Historia de usuario:** Autenticar usuario.
Responsable de la prueba: Ernesto Sabuquet Hurtado	

Descripción: Prueba para verificar que se autentica un usuario en el sistema.
Condiciones de ejecución: El usuario debe llenar los campos de usuario y contraseña correctamente.
Entrada/Pasos de ejecución: 1. Se accede a la interfaz principal. 2. Se introducen los datos (usuario y contraseña). 3. Se acciona el botón aceptar.
Resultado esperado: El sistema pasa a la interfaz correspondiente, de acuerdo al rol que tenga asignado el usuario en cuestión.
Evaluación de la prueba: Prueba satisfactoria.

Tabla 19: Prueba de aceptación para la HU1.

3.6.2. Pruebas de unidad

Las pruebas unitarias son una forma de probar el correcto funcionamiento de un módulo determinado dentro del sistema. El objetivo fundamental de estas pruebas es asegurar el correcto funcionamiento de las interfases, o flujo de datos entre los componentes. Los test para la metodología XP es considerada una actividad fundamental, al ser ejecutadas constantemente ante cada modificación del sistema, ayudando al programador a observar la refactorización. Una funcionalidad está terminada cuando pasa todas sus pruebas de unidad.

Para realizar las pruebas de unidad a la aplicación se utilizó la herramienta Visual Studio Test Professional, que propone el Microsoft Visual Studio 2010. Esta herramienta proporciona una interfaz para la ejecución de pruebas de varios tipos. Con él se pueden crear planes de pruebas, conjuntos de pruebas y casos de pruebas con capacidades de anidamiento. Durante toda la etapa de desarrollo del software se estuvieron desarrollando pruebas de unidad reiteradamente para comprobar que el sistema estuviese funcionando correctamente. La figura 10 muestra un ejemplo de una de las pruebas realizada a los métodos controladores Administrador y EnrolarAdmin.

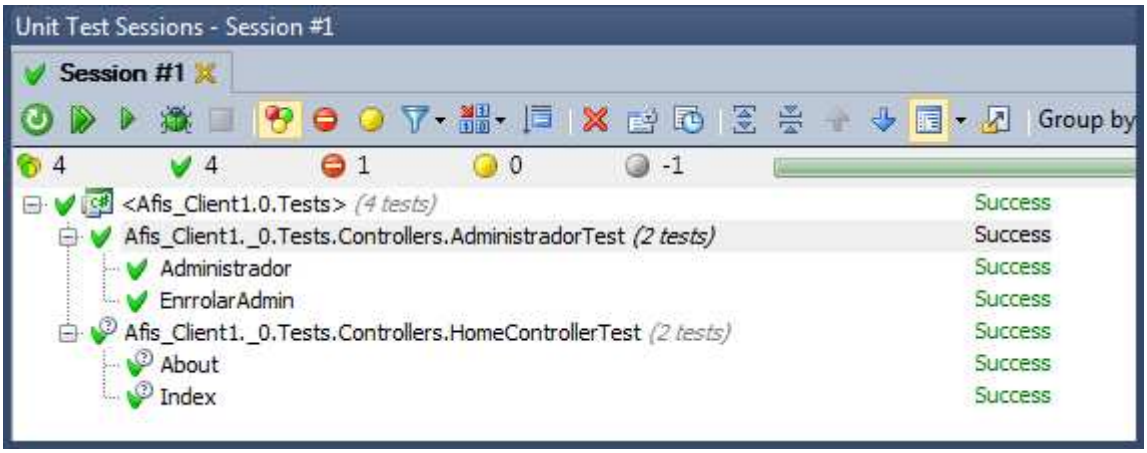

Figura 10: Prueba de unidad.

3.6.2.1. Resultado de las pruebas de aceptación

La tabla 20 muestra el resultado de las 4 iteraciones de prueba realizada a las 2 iteraciones de desarrollo del software:

Iteración	Funcionalidades		
	Funcionalidades con errores	Funcionalidades correctas	Funcionalidades sin implementar
1	0	2	2
1.1	1	2	1
1.2	1	3	0
1.3	0	4	0
2	2	1	2
2.1	1	3	1
2.2	1	4	0
2.3	0	5	0

Tabla 20: Resultado de las pruebas.

La figura 11 representa los resultados anteriormente graficados.

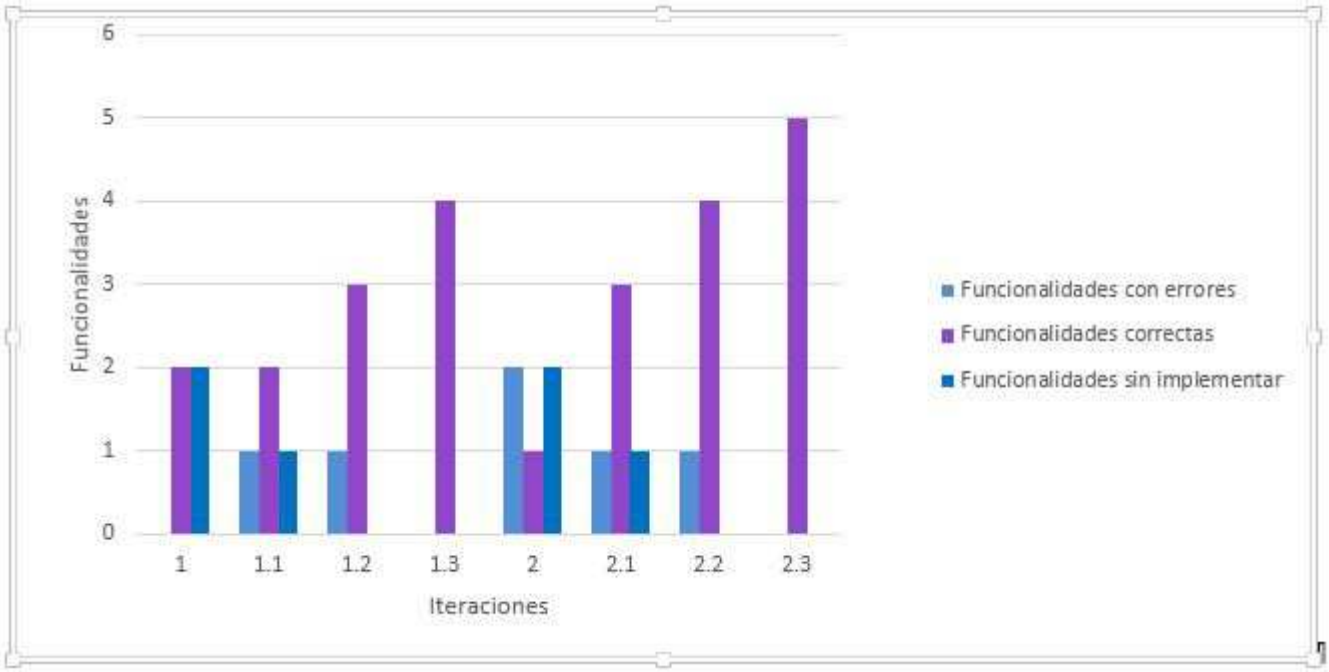

Figura 11: Representación gráfica del resultado.

3.7. Análisis de resultados

Los algoritmos de reconocimiento de huellas dactilares, como el algoritmo basado en extracción de minucias utilizado por el SourceAFIS, son evaluados automáticamente por diferentes sistemas como por ejemplo FVC-OnGoing, luego de pasar por este proceso se obtuvieron los siguientes resultados:

- 3.6 % EER[1], 10.9 % FRR[2] @ 0.01 % FAR[3] en verificación de 1:1.
- 1.17 % EER, 2.5 % FRR @ 0.01 % FAR en pruebas ISO.

A partir de la introducción en la práctica del sistema propuesto se mejorará la confiabilidad en el control de acceso a las instalaciones del CISED, así como la reducción de los tiempos de respuesta ante las peticiones.

[1]Acrónimo de Equal Error Rate, estadística utilizada para mostrar el rendimiento biométrico, es la ubicación en una curva ROC (Característica de funcionamiento del receptor) o DET (Compensación por error de detección) donde la tasa de falsa aceptación y la tasa de falso rechazo son iguales.

[2]Falso Rechazo, error que se produce si el sistema de verificación devuelve como salida que el usuario en la entrada no se corresponde con la plantilla almacenada, y realmente sí es la misma persona.

[3]Falsa Aceptación, error que se produce cuando el sistema indica que la información adquirida del usuario en la entrada sí se corresponde con la plantilla almacenada, cuando realmente se trata de otra persona.

3.8. Conclusiones Parciales

Al concluir las fases de Iteraciones a primera liberación y Producción, queda demostrado:

- Mediante la representación de los diagramas de componentes y despliegue, se visualizaron los principales componentes del sistema divido en paquetes y como quedaría la aplicación desplegada.
- Realizar el desglose de las historias de usuario en tareas de la ingeniería posibilitó que los programadores realizaran las funcionalidades específicas a implementar.
- A partir de las pruebas realizadas al software en el presente capítulo se logró verificar el correcto funcionamiento del mismo y la conformidad con las historias de usuario definidas inicialmente para su desarrollo. Las técnicas utilizadas permitieron no solo validar y verificar el funcionamiento interno del software, sino externo también, ya que se realizaron pruebas unitarias y pruebas de aceptación. Los resultados obtenidos a partir de la ejecución de ambas pruebas demuestran que el producto final cuenta con la calidad deseada y esperada por los clientes.

Conclusiones generales

Con la culminación de este trabajo se ha llegado a las siguientes conclusiones:

- El estudio de los sistemas de identificación mediante huellas dactilares ha permitido demostrar que es una de las técnicas biométricas más confiables desarrolladas hasta el momento, presentado tres de las características de un indicador biométrico, que son, universalidad, unicidad y permanencia. La identificación mediante huella dactilar tiene ciertas desventajas, pero sin duda es una opción bastante económica y con buenos resultados, siempre y cuando se realice el estudio y diseño adecuado para la realización del sistema particular que se quiera implementar.
- Las pruebas realizadas permitieron comprobar que el sistema cuenta con un alto nivel de exactitud en el reconocimiento de la huella y que las funcionalidades implementadas se encuentran en óptimas condiciones.
- El sistema desarrollado constituye una solución del Centro de Identificación y Seguridad Digital que elimina la necesidad de adquisición de software costoso para la seguridad en el control de acceso a sus instalaciones, además de que se le pueden añadir otras funcionalidades para la mejora del funcionamiento del mismo.

Recomendaciones

Se exponen como recomendaciones para las siguientes versiones de la solución:

- Perfeccionar la seguridad del sistema relacionada a la gestión de usuario.
- Integrar el sistema de identificación mediante huella dactilar con el sistema distribuido del proceso de búsqueda de huellas dactilares en el banco de datos de un AFIS.
- Comercializar la aplicación en un futuro, como uno de los productos del Centro CISED de la Universidad de las Ciencias Informáticas.

Referencias bibliográficas

[1] *Sistemas de Control de Acceso y Software de Control de Accesos.* [En línea] [Citado el: 18 de enero de 2013.] http://www.scssa.com.ar/control-de-acceso.htm

[2] **Alonso,Néstor** *S.V. Control Acceso.* [En línea] 2008 [Citado el: 18 de enero de 2013.]

[3] **Sánchez Reillo, Raúl.** *Tesis Doctoral: Mecanismo de autenticación biométrica mediante Tarjeta Inteligente.* Madrid: Universidad Politécnica de Madrid. Departamento de Tecnología Fotónica, 2000.

[4] *Tarjetas magneticas.* A3M. [En línea] Tarjetas magneticas. [Citado el: 16 de enero de 2013.] http://www.a3m.eu/es/tarjetas-plasticas/tarjetas-plasticas-blancas/tarjetas-magneticas

[5] *Tarjeta de código de barra.* Tarjeta con código de barras - Sistema para control de asistencia y accesos LARCON-SIA. [En línea] [Citado el: 16 de enero de 2013.] http://www.larconsia.com/Tarjetas%20de%20codigo%20de%20barras.asp

[6] **Herrera Lozada, Juan Carlos, Peréz Romero, Patricia y Marciano Melchor, Magdalena.** *Tecnología RFID Aplicada al Control de Acceso.* [En línea] [Citado el: 16 de enero de 2013.] http://polibits.gelbukh.com/2009_40/40_08.pdf

[7] **Biometría.** *Preguntas frecuentes.* [En línea] [Citado el: 18 de enero de 2013.] http://www.biometria.gov.ar/acerca-de-la-biometria/preguntas-frecuentes.aspx

[8] **Biometría.** *Reconocimiento del Iris.* [En línea] [Citado el: 18 de enero de 2013.] http://www.biometria.gov.ar/metodos-biometricos/iris.aspx

[9] **UNAM-Facultad de Ingeniería Biometría Informática.** *Reconocimiento del iris ocular.* [En línea] [Citado el: 18 de enero de 2013.] http://redyseguridad.fi-p.unam.mx/proyectos/biometria/clasificacionsistemas/recoiris.html

[10] **Biometría.** *Reconocimiento de Voz.* [En línea] [Citado el: 18 de enero de 2013.] http://www.biometria.gov.ar/metodos-biometricos/voz.aspx

[11] **UNAM-Facultad de Ingeniería Biometría Informática.** *Huella dactilar.* [En línea] [Citado el: 21 de enero de 2013.] http://redyseguridad.fi-p.unam.mx/proyectos/biometria/clasificacionsistemas/recohuella.html

[12] **Penadés, Mª Carmen; Letelier, Patricio y Canós, José Hilario.** *Metodologías Ágiles en el Desarrollo de Software.* Metodologías Ágiles en el Desarrollo de Software. Alicante : s.n., 12 noviembre 2003.

[13] **Highsmith, J.** *Agile Software Development Ecosystems.* (2002). Addison-Wesley Professional.

[14] **Priolo, Ing. Sebastián.** *Programación Extrema.* [En línea] [Citado el: 7 de febrero de 2013.] http://www.fcad.uner.edu.ar/jai/6JAI/XP_6JAI.pdf

[15] **Carvajal Riola, Jose Carlos.** *Tesis final de Máster. METODOLOGÍAS ÁGILES: HERRAMIENTAS Y MODELO DE DESARROLLO PARA APLICACIONES JAVA EE COMO METODOLOGÍA EMPRESARIAL.* Máster en Tecnologías de la Información - UPC - Barcelona.2008

[16] **Flores Cueto, Juan José.** *Método de las 6'D. UML - Pseudocódigo - Java. (Enfoque algorítmico).* [En línea][Citado: 7 de febrero de 2013] http://books.google.com.cu/books?id=lZkdLjZrcq4C&pg=PA39&dq=lenguaje+de+modelado+UML&hl=es&sa=X&ei=lUSIUZOBEsTc4AObvYHgCA&ved=0CGQQ6AEwCQ#v=onepage&q=lenguaje%20de%20modelado%20UML&f=false.

[17] *Introducción a UML.* Programación en Castellano. [En línea] [Citado el: 7 de febrero de 2013.] http://www.programacion.com/articulo/introduccion_a_uml_181

[18] **INEI.** Instituto Nacional de Estadistícas Informaticas. *Herramienta CASE.* [En línea] [Citado el: 7 de febrero de 2013.] http://www.inei.gob.pe/biblioineipub/bancopub/Inf/Lib5103/Libro.pdf

[19] **Lenguajes de programación.** *Lenguajes de programación.* [En línea] [Citado el: 9 de febrero de 2013.] http://www.lenguajes-de-programacion.com/lenguajes-de-programacion.shtml

[20] **Pacheco Velasco, M.C. José Evaristo.** Instituto Tecnológico de Veracruz. *Programación del lado del servidor.* [En línea] [Citado el: 9 de febrero de 2013.] http://www.programaweb.com.mx/pweb/0203ladoServidor.html

[21] *Programación perl: CARACTERISTICAS* [En línea] [Citado el: 9 de febrero de 2013.] http://trabajodeprogramacionperl.blogspot.com/p/caracteristicas.html

[22] **Villaverde Masa, Pablo.** *ASP.Recursos y características.* [En línea] [Citado el: 9 de febrero de 2013.] http://tgp0607.awardspace.com/Recursos_ASP.pdf

[23] **Avila, Katty.** *¿Qué es un Sistema Gestor de Bases de Datos o SGBD?.* CAVSI. [En línea] [Citado el: 11 de febrero de 2013.] http://www.cavsi.com/preguntasrespuestas/que-es-un-sistema-gestor-de-bases-de-datos-o-sgbd/

[24] **Martinez, Rafael** *Sobre PostgreSQL.* PostgreSQL-es. [En línea] 2 de octubre de 2010 [Citado el: 11 de febrero de 2013.] http://www.postgresql.org.es/sobre_postgresql

[25] *Microsoft SQL Server 2008.* Microsoft. [En línea] [Citado el: 11 de febrero de 2013.] http://www.microsoft.com/latam/sql/2008/default.mspx

[26] *Microsoft Visual Studio 2010 Ultimate Español.* IntercambiosVirtuales. [En línea] 2 de junio de 2010. [Citado el: 5 de febrero de 2013.] http://www.intercambiosvirtuales.org/software/microsoft-visual-studio-2010-ultimate-espanol

[27] *.Net Framework.* DesarrolloWeb. [En línea] [Citado el: 6 de marzo de 2013.] http://www.desarrolloweb.com/articulos/1328.php

[28] **Važan, Robert.** *SourceAFIS.* [En línea] [Cited: 1 16, 2013.] http://www.sourceafis.org/blog/datasheet/

[29] **Campos Rodriguez, Sandy y Rodríguez Sánchez, Hector Luis** *Sistema para la interacción y control centralizado de dispositivos en aplicaciones web v2.0.* Universidad de las Ciencias Informáticas, Junio 2009.

[30] **Tedeschi, Nicolás.** *¿Qué es un Patrón de Diseño?* MSDN. [En línea] [Citado el: 28 de febrero de 2013.] http://msdn.microsoft.com/es-es/library/bb972240.aspx

Bibliografía consultada

Sistemas biométricos. [En línea] 9 de diciembre de 2010. [Citado el: 16 de enero de 2013.] http://sistemasbiometricos.co/blog/caracteristicas-principales-de-los-sistemas-biometricos

Seguridad Física-Sistemas Biométricos. [En línea] [Citado el: 16 de enero de 2013.] http://www.segu-info.com.ar/fisica/biometricos.htm

Važan, Robert. *SourceAFIS.* [En línea] [Citado el: 18 de enero de 2013.] http://www.sourceafis.org/blog/datasheet/

Artículos Informativos. *Sistema de Control de Acceso.* [En línea] [Citado el: 18 de enero de 2013.] http://www.articulosinformativos.com/Sistemas_de_Control_de_Acceso-a862383.html

Visual Paradigm para UML. [En línea] [Citado el: 7 de febrero de 2013.] http://www.software.com.ar/visual-paradigm-para-uml.html

Nobrega, Maria De. *Herramientas CASE: Rational Rose.* Curso Sistemas de Información II. [En línea] [Citado el: 7 de febrero de 2013.] http://curso_sin2.blogia.com/2005/060401-herramientas-case-rational-rose.-por-maria-de-nobrega.php

Rational Rose Data Modeler. [En línea] [Citado el: 7 de febrero de 2013.] http://www.rational.com.ar/herramientas/rosedatamodeler.html

Microsoft. *Introducción al lenguaje C# y .NET Framework.* [En línea] [Citado el: 8 de febrero de 2013.] http://msdn.microsoft.com/library/vstudio/z1zx9t92

Alvarez, Rubén. *Introducción a la programación en PHP.* Desarrollo Web. [En línea] 1 de enero de 2001. [Citado el: 9 de febrero de 2013.] http://www.desarrolloweb.com/articulos/303.php

Alvarez, Miguel Angel. *Documento introductorio al lenguaje Perl.* Desarrollo Web. [En línea] 29 de septiembre de 2001. [Citado el: 9 de febrero de 2013.] http://www.desarrolloweb.com/articulos/541.php

ASP.NET. Artículos Grupo Danysoft. [En línea] [Citado el: 9 de febrero de 2013.] http://www.danysoft.com/free/aspnet.pdf

Microsoft Visual Studio - Descargar. [En línea] [Citado el: 5 de febrero de 2013.] http://microsoft-visual-studio.softonic.com/

Paquetes de características de Visual Studio. [En línea] [Citado el: 5 de febrero de 2013.] http://msdn.microsoft.com/es-es/library/vstudio/ff636699(v=vs.100).aspx

Información general sobre ASP.NET MVC. Microsoft. [En línea] [Citado el: 5 de febrero de 2013.] http://msdn.microsoft.com/es-es/library/dd381412(v=vs.108).aspx

NET Framework 4 (Full). Identi. [En línea] [Citado el: 11 de febrero de 2013.] http://www.identi.li/index.php?topic=31059

Lo nuevo en .NET Framework 4. Microsoft. [En línea] [Citado el: 11 de febrero de 2013.] http://msdn.microsoft.com/es-es/library/vstudio/ms171868%28v=vs.100%29.aspx

PostgreSQL. PostgreSQL Global Development Group PostgreSQL. [En línea][Citado el: 11 de febrero de 2013.] http://www.postgresql.org

Espinoza, Humberto. *PostgreSQL. Una Alternativa de DBMS Open Source.* [En línea][Citado el: 11 de febrero de 2013.] http://www.lgs.com.ve/pres/PresentacionES_PSQL.pdf

Casillas Santillán, Luis Alberto; Gibert Ginestá, Marc y Pérez Mora, Óscar *Bases de datos en MySQL.* [En línea][Citado el: 11 de febrero de 2013.] http://ocw.uoc.edu/computer-science-technology-and-multimedia/bases-de-datos/bases-de-datos/P06_M2109_02151.pdf

Pineda Oro, Jeandy Bryan. Trabajo de Diploma para optar por el Título de Ingeniero en Ciencias Informáticas: *Sistema para la interacción y control centralizado de Dispositivos en aplicaciones web.* Junio 2012.

Avances en Informática y Sistema Computacionales Tomo I (CONAIS 2006). [En línea][Citado el: 27 de febrero de 2013.] http://books.google.com.cu/books?id=kn1TFc0EvPUC&pg=PA136&dq=estilo+arquitectonico+MVC&hl=es&sa=X&ei=pAOIUercAtes4AOZhoGAAg&ved=0CDQQ6AEwAQ#v=onepage&q=estilo%20arquitectonico%20MVC&f=false

Visconti, Marcello y Astudillo, Hernán. *Patrones de Diseño.* Fundamentos de Ingeniería de Software. [En línea][Citado el: 27 de febrero de 2013.]

Glosario de términos

AFIS: Sistema Automatizado de Identificación de Huellas dactilares.

Framework: Es una estructura de soporte definida en la cual un proyecto de software puede ser organizado y desarrollado.

GPL: Acrónimo en inglés de General Public License (Licencia Publica General). Regula los derechos de autor de los programas de software libre (free software) promovido por el Free Software Foundation (FSF) en el marco de la iniciativa GNU. Permite la distribución de copias de programas (e incluso cobrar por ello), así como modificar el código fuente de los mismos o utilizarlo en otros programas.

GNU: El Proyecto GNU fue creado en 1984 con el fin de desarrollar un sistema operativo tipo Unix según la filosofía del "software libre".

API: Acrónimo en inglés de Application Program Interface (Interfaz de Programación de Aplicaciones). Un juego de rutinas usados por una aplicación para gestionar generalmente servicios de bajo nivel, realizados por el sistema operativo de la computadora. Uno de los principales propósitos de un API consiste en proporcionar un conjunto de funciones de uso general, de esta forma los programadores se benefician de las ventajas del API, ahorrándose el trabajo de programar todo de nuevo.

HTTP: Protocolo de transferencia de hipertexto.

HTTPS: Protocolo seguro de transferencia de hipertexto a servidores web.

TCP/IP: El nombre TCP/IP proviene de dos protocolos importantes. TCP es acrónimo en inglés de Transmission Control Protocol (Protocolo de Control de Transmisión) e IP que es acrónimo de Internet Protocol (Protocolo de Internet). Es una forma de comunicación básica que usa Internet, la cual hace posible que cualquier tipo de información (mensajes, gráficos o audio) viaje en forma de paquetes sin que estos se pierdan y siguiendo cualquier ruta posible.

DBMS: Acrónimo en inglés de Database Management System (Sistema de Administración de Bases de Datos o Sistema gestor de bases de datos). Es el software encargado de administrar y producir bases de datos.

Anexo I: Tarjetas CRC

IdentificarController	
Dado una huella identifica a la persona.	Operaciones

Tabla 21: Tarjeta CRC: IdentificarController.

EnrolarController	
Adiciona todos los datos de una persona del centro a la base de datos.	Operaciones

Tabla 22: Tarjeta CRC: EnrolarController.

AccountModel	
Almacena todos los datos del modelo.	

Tabla 23: Tarjeta CRC: AccountModel.

UsersManager	
Adiciona, inserta y elimina los datos de un usuario del sistema a la base de datos. Identifica el rol el usuario del sistema.	sManager sManagerQuery tPerson tPersonQuery RolesManager RoleModuleManager

Tabla 24: Tarjeta CRC: UsersManager.

RoleManager	
Crea y elimina un rol. Devuelve el determinado rol del usuario.	sRole sRoleQuery

Tabla 25: Tarjeta CRC: RoleManager.

RoleModuleManager	
Devuelve los módulos a los que tienen acceso los usuarios del sistema.	sRoleModuleQuery

Tabla 26: Tarjeta CRC: RoleModuleManager.

AdministradorController	
Crea, actualiza y elimina los usuarios del sistema.	UsersManager
Crea, actualiza y elimina los roles del sistema.	UsersRoleManager

Tabla 27: Tarjeta CRC: AdministradorController.

Anexo II: Tareas detalladas

	Tarea de Ingeniería
Número: 2	**Nombre de HU:** Autenticar usuario
Nombre de la tarea: Crear la interfaz de autenticación.	
Tipo de tarea: Desarrollo	**Puntos estimados (días):** 1
Fecha inicio: 14/01/2013	**Fecha fin:** 15/01/2013
Programador responsable: Ernesto Sabuquet Hurtado	
Descripción: Se plasma mediante el código los elementos necesarios para que la interfaz pueda asociarse al resto de la aplicación.	

Tabla 28: Tarea detallada 2 de la HU1.

	Tarea de Ingeniería
Número: 3	**Nombre de HU:** Autenticar usuario
Nombre de la tarea: Verificar la autenticación del usuario por contraseña.	
Tipo de tarea: Desarrollo	**Puntos estimados (días):** 4
Fecha inicio: 16/01/2013	**Fecha fin:** 19/01/2013
Programador responsable: Ernesto Sabuquet Hurtado	
Descripción: Se comprueba que los datos de usuario estén correctos.	

Tabla 29: Tarea detallada 3 de la HU1.

	Tarea de Ingeniería
Número: 4	**Nombre de HU:** Autenticar usuario
Nombre de la tarea: Garantizar que dado un usuario se muestren solamente las interfaces que tiene asignada.	
Tipo de tarea: Desarrollo	**Puntos estimados (días):** 4
Fecha inicio: 21/01/2013	**Fecha fin:** 24/01/2013
Programador responsable: Ernesto Sabuquet Hurtado	
Descripción: Una vez que el usuario introduce los datos aparecerá una interfaz con las acciones que puede realizar.	

Tabla 30: Tarea detallada 4 de la HU1.

	Tarea de Ingeniería
Número: 1	**Nombre de HU:** Realizar conexión con el DGM
Nombre de la tarea: Instalar el componente DGM.	
Tipo de tarea: Desarrollo	**Puntos estimados (días):** 3
Fecha inicio: 28/01/2013	**Fecha fin:** 28/01/2013
Programador responsable: Osmayda Silverio Leyva	
Descripción: Se instala el DGM en la pc que realiza la identificación del personal.	

Tabla 31: Tarea detallada 1 de la HU2.

	Tarea de Ingeniería
Número: 2	**Nombre de HU:** Realizar conexión con el DGM
Nombre de la tarea: Establecer conexión.	
Tipo de tarea: Desarrollo	**Puntos estimados (días):** 3
Fecha inicio: 29/01/2013	**Fecha fin:** 1/02/2013
Programador responsable: Osmayda Silverio Leyva	
Descripción: Se instala el DGM en la pc que realiza la identificación del personal.	

Tabla 32: Tarea detallada 2 de la HU2.

	Tarea de Ingeniería
Número: 1	**Nombre de HU:** Enrolar personal del centro.
Nombre de la tarea: Crear el diseño de prototipo.	
Tipo de tarea: Desarrollo	**Puntos estimados (días):** 1
Fecha inicio: 14/01/2013	**Fecha fin:** 15/01/2013
Programador responsable: Osmayda Silverio Leyva	
Descripción: Esta tarea da una visión exacta de cómo será representada la interfaz de enrolar, facilitando la comprensión a la pareja desarrolladora.	

Tabla 33: Tarea detallada 1 de la HU3.

	Tarea de Ingeniería
Número: 2	**Nombre de HU:** Enrolar personal del centro
Nombre de la tarea: Crear la interfaz de enrolar.	

Tipo de tarea: Desarrollo	**Puntos estimados (días):** 2
Fecha inicio: 4/02/2013	**Fecha fin:** 5/02/2013
Programador responsable: Ernesto Sabuquet Hurtado	
Descripción: Se plasma mediante el código los elementos necesarios para que se realice el enrolar por el usuario del sistema.	

Tabla 34: Tarea detallada 2 de la HU3.

	Tarea de Ingeniería
Número: 3	**Nombre de HU:** Enrolar personal del centro
Nombre de la tarea: Validar que los datos introducidos sean los correctos.	
Tipo de tarea: Desarrollo	**Puntos estimados (días):** 4
Fecha inicio: 6/02/2013	**Fecha fin:** 9/02/2013
Programador responsable: Ernesto Sabuquet Hurtado	
Descripción: Se validan mediante códigos las especificaciones de los datos a introducir en los diferentes campos .	

Tabla 35: Tarea detallada 3 de la HU3.

	Tarea de Ingeniería
Número: 4	**Nombre de HU:** Enrolar personal del centro
Nombre de la tarea: Guardar los datos introducidos.	
Tipo de tarea: Desarrollo	**Puntos estimados (días):** 5
Fecha inicio: 11/02/2013	**Fecha fin:** 15/02/2013
Programador responsable: Ernesto Sabuquet Hurtado	
Descripción: Los datos de la persona son guardados por el sistema.	

Tabla 36: Tarea detallada 4 de la HU3.

	Tarea de Ingeniería
Número: 5	**Nombre de HU:** Enrolar personal del centro
Nombre de la tarea: Crear campos dinámicos para insertar nuevos datos al pulsar Enrolar.	

Tipo de tarea: Desarrollo	**Puntos estimados (días):** 5
Fecha inicio: 18/02/2013	**Fecha fin:** 22/02/2013
Programador responsable: Ernesto Sabuquet Hurtado	
Descripción: Se valida mediante código para que los campos aparezcan vacíos y listos para enrolar otra persona.	

Tabla 37: Tarea detallada 5 de la HU3.

	Tarea de Ingeniería
Número: 1	**Nombre de HU:** Buscar persona.
Nombre de la tarea: Crear el diseño de prototipo.	
Tipo de tarea: Desarrollo	**Puntos estimados (días):** 1
Fecha inicio: 14/01/2013	**Fecha fin:** 15/01/2013
Programador responsable: Osmayda Silverio Leyva	
Descripción: Esta tarea da una visión exacta de cómo será representada la interfaz de buscar persona, facilitando la comprensión a la pareja desarrolladora.	

Tabla 38: Tarea detallada 1 de la HU4.

	Tarea de Ingeniería
Número: 2	**Nombre de HU:** Buscar persona
Nombre de la tarea: Crear la interfaz de buscar persona.	
Tipo de tarea: Desarrollo	**Puntos estimados (días):** 2
Fecha inicio: 25/02/2013	**Fecha fin:** 26/02/2013
Programador responsable: Osmayda Silverio Leyva	
Descripción: Se plasma mediante el código los elementos necesarios para que se realice el buscar personal por el usuario del sistema.	

Tabla 39: Tarea detallada 2 de la HU4.

	Tarea de Ingeniería
Número: 1	**Nombre de HU:** Identificar personal del centro.
Nombre de la tarea: Crear el diseño de prototipo.	
Tipo de tarea: Desarrollo	**Puntos estimados (días):** 3

Fecha inicio: 4/03/2013	**Fecha fin:** 6/03/2013
Programador responsable: Osmayda Silverio Leyva	
Descripción: Esta tarea da una visión exacta de cómo será representada la interfaz de identificar, facilitando la comprensión a la pareja desarrolladora.	

Tabla 40: Tarea detallada 1 de la HU5.

	Tarea de Ingeniería
Número: 2	**Nombre de HU:**Identificar personal del centro
Nombre de la tarea: Crear la interfaz de identificar una persona.	
Tipo de tarea: Desarrollo	**Puntos estimados (días):** 2
Fecha inicio: 11/03/2013	**Fecha fin:** 13/03/2013
Programador responsable: Osmayda Silverio Leyva	
Descripción: Se plasma mediante el código los elementos necesarios para que se realice el proceso de identificar una persona.	

Tabla 41: Tarea detallada 2 de la HU6.

	Tarea de Ingeniería
Número: 1	**Nombre de HU:** Crear rol.
Nombre de la tarea: Crear el diseño de prototipo.	
Tipo de tarea: Desarrollo	**Puntos estimados (días):** 3
Fecha inicio: 4/03/2013	**Fecha fin:** 6/03/2013
Programador responsable: Osmayda Silverio Leyva	
Descripción: Se crea la interfaz para especificar los métodos y propiedades que debe cumplir para asociarse al resto de la aplicación.	

Tabla 42: Tarea detallada 1 de la HU6.

	Tarea de Ingeniería
Número: 2	**Nombre de HU:** Crear rol
Nombre de la tarea: Crear la interfaz de crear rol.	
Tipo de tarea: Desarrollo	**Puntos estimados (días):** 2
Fecha inicio: 18/03/2013	**Fecha fin:** 19/03/2013

Programador responsable: Osmayda Silverio Leyva
Descripción: Se plasma mediante el código los elementos necesarios para que se realice el crear rol por el usuario del sistema.

Tabla 43: Tarea detallada 2 de la HU6.

	Tarea de Ingeniería
Número: 3	**Nombre de HU:** Crear rol
Nombre de la tarea: Guardar los datos introducidos.	
Tipo de tarea: Desarrollo	**Puntos estimados (días):** 4
Fecha inicio: 20/02/2013	**Fecha fin:** 23/02/2013
Programador responsable: Osmayda Silverio Leyva	
Descripción: Los datos de la persona son guardados por el sistema.	

Tabla 44: Tarea detallada 3 de la HU6.

	Tarea de Ingeniería
Número: 4	**Nombre de HU:** Crear rol
Nombre de la tarea: Crear campos dinámicos para insertar nuevos datos al pulsar Crear rol.	
Tipo de tarea: Desarrollo	**Puntos estimados (días):** 3
Fecha inicio: 25/03/2013	**Fecha fin:** 27/03/2013
Programador responsable: Osmayda Silverio Leyva	
Descripción: Se valida mediante código para que los campos aparezcan vacíos y listos para crear otro rol.	

Tabla 45: Tarea detallada 4 de la HU6.

	Tarea de Ingeniería
Número: 5	**Nombre de HU:** Crear rol
Nombre de la tarea: Garantizar que se actualicen las listas desplegables Roles con los roles existentes hasta el momento.	
Tipo de tarea: Desarrollo	**Puntos estimados (días):** 3
Fecha inicio: 28/03/2013	**Fecha fin:** 30/03/2013

Programador responsable: Osmayda Silverio Leyva
Descripción: Al crearse y guardarse un rol en la base de datos, las listas desplegables se actualizan mediante código.

Tabla 46: Tarea detallada 5 de la HU6.

	Tarea de Ingeniería
Número: 1	**Nombre de HU:** Eliminar rol.
Nombre de la tarea: Crear el diseño de prototipo.	
Tipo de tarea: Desarrollo	**Puntos estimados (días):** 3
Fecha inicio: 4/03/2013	**Fecha fin:** 6/03/2013
Programador responsable: Osmayda Silverio Leyva	
Descripción: Se crea la interfaz para especificar los métodos y propiedades que debe cumplir para asociarse al resto de la aplicación.	

Tabla 47: Tarea detallada 1 de la HU7.

	Tarea de Ingeniería
Número: 2	**Nombre de HU:** Eliminar rol
Nombre de la tarea: Crear la interfaz de eliminar rol.	
Tipo de tarea: Desarrollo	**Puntos estimados (días):** 2
Fecha inicio: 1/04/2013	**Fecha fin:** 2/03/2013
Programador responsable: Osmayda Silverio Leyva	
Descripción: Se plasma mediante el código los elementos necesarios para que se realice el eliminar rol por el usuario del sistema.	

Tabla 48: Tarea detallada 2 de la HU7.

	Tarea de Ingeniería
Número: 3	**Nombre de HU:** Eliminar rol
Nombre de la tarea: Guardar los datos introducidos.	
Tipo de tarea: Desarrollo	**Puntos estimados (días):** 4
Fecha inicio: 3/04/2013	**Fecha fin:** 6/04/2013
Programador responsable: Osmayda Silverio Leyva	

Descripción: Los datos de la persona son guardados por el sistema.

Tabla 49: Tarea detallada 3 de la HU7.

	Tarea de Ingeniería
Número: 4	**Nombre de HU:** Eliminar rol
Nombre de la tarea: Garantizar que se actualicen las listas desplegables Roles con los roles existentes hasta el momento.	
Tipo de tarea: Desarrollo	**Puntos estimados (días):** 5
Fecha inicio: 8/04/2013	**Fecha fin:** 12/04/2013
Programador responsable: Osmayda Silverio Leyva	
Descripción: Al eliminarse un rol las listas desplegables se actualizan mediante código.	

Tabla 50: Tarea detallada 4 de la HU7.

	Tarea de Ingeniería
Número: 1	**Nombre de HU:** Crear usuario.
Nombre de la tarea: Crear el diseño de prototipo.	
Tipo de tarea: Desarrollo	**Puntos estimados (días):** 3
Fecha inicio: 4/03/2013	**Fecha fin:** 6/03/2013
Programador responsable: Osmayda Silverio Leyva	
Descripción: Se crea la interfaz para especificar los métodos y propiedades que debe cumplir para asociarse al resto de la aplicación.	

Tabla 51: Tarea detallada 1 de la HU8.

	Tarea de Ingeniería
Número: 2	**Nombre de HU:** Crear usuario
Nombre de la tarea: Crear la interfaz de crear usuario.	
Tipo de tarea: Desarrollo	**Puntos estimados (días):** 2
Fecha inicio: 15/04/2013	**Fecha fin:** 16/04/2013
Programador responsable: Ernesto Sabuquet Hurtado	
Descripción: Se plasma mediante el código los elementos necesarios para que se realice el crear usuario por el usuario del sistema.	

Tabla 52: Tarea detallada 2 de la HU8.

	Tarea de Ingeniería
Número: 3	**Nombre de HU:** Crear usuario
Nombre de la tarea: Guardar los datos introducidos.	
Tipo de tarea: Desarrollo	**Puntos estimados (días):** 4
Fecha inicio: 17/04/2013	**Fecha fin:** 20/04/2013
Programador responsable: Ernesto Sabuquet Hurtado	
Descripción: Los datos de la persona son guardados por el sistema.	

Tabla 53: Tarea detallada 3 de la HU8.

	Tarea de Ingeniería
Número: 4	**Nombre de HU:** Crear usuario
Nombre de la tarea: Crear campos dinámicos para insertar nuevos datos al pulsar Crear usuario.	
Tipo de tarea: Desarrollo	**Puntos estimados (días):** 5
Fecha inicio: 22/04/2013	**Fecha fin:** 26/04/2013
Programador responsable: Ernesto Sabuquet Hurtado	
Descripción: Se valida mediante código para que los campos aparescan vacíos y listos para crear otro usuario.	

Tabla 54: Tarea detallada 4 de la HU8.

	Tarea de Ingeniería
Número: 5	**Nombre de HU:** Crear usuario
Nombre de la tarea: Garantizar que se actualicen las listas desplegables Usuarios con los usuarios existentes hasta el momento.	
Tipo de tarea: Desarrollo	**Puntos estimados (días):** 5
Fecha inicio: 29/04/2013	**Fecha fin:** 3/05/2013
Programador responsable: Ernesto Sabuquet Hurtado	
Descripción: Al crearse y guardarse un usuario en la base de datos, las listas desplegables se actualizan mediante código.	

Tabla 55: Tarea detallada 5 de la HU8.

	Tarea de Ingeniería
Número: 1	**Nombre de HU:** Eliminar usuario.
Nombre de la tarea: Crear el diseño de prototipo.	
Tipo de tarea: Desarrollo	**Puntos estimados (días):** 3
Fecha inicio: 4/03/2013	**Fecha fin:** 6/03/2013
Programador responsable: Osmayda Silverio Leyva	
Descripción: Se crea la interfaz para especificar los métodos y propiedades que debe cumplir para asociarse al resto de la aplicación.	

Tabla 56: Tarea detallada 1 de la HU9.

	Tarea de Ingeniería
Número: 2	**Nombre de HU:** Eliminar usuario.
Nombre de la tarea: Crear la interfaz de eliminar usuario.	
Tipo de tarea: Desarrollo	**Puntos estimados (días):** 2
Fecha inicio: 6/05/2013	**Fecha fin:** 7/05/2013
Programador responsable: Ernesto Sabuquet Hurtado	
Descripción: Se plasma mediante el código los elementos necesarios para que se realice el eliminar usuario.	

Tabla 57: Tarea detallada 2 de la HU9.

	Tarea de Ingeniería
Número: 3	**Nombre de HU:** Eliminar usuario.
Nombre de la tarea: Guardar los datos introducidos.	
Tipo de tarea: Desarrollo	**Puntos estimados (días):** 4
Fecha inicio: 8/05/2013	**Fecha fin:** 11/05/2013
Programador responsable: Ernesto Sabuquet Hurtado	
Descripción: Los datos de la persona son guardados por el sistema.	

Tabla 58: Tarea detallada 3 de la HU9.

	Tarea de Ingeniería
Número: 4	**Nombre de HU:** Eliminar usuario.
Nombre de la tarea: Garantizar que se actualicen las listas desplegables Usuario.	
Tipo de tarea: Desarrollo	**Puntos estimados (días):** 5
Fecha inicio: 13/05/2013	**Fecha fin:** 17/05/2013
Programador responsable: Ernesto Sabuquet Hurtado	
Descripción: Al eliminarse un usuario las listas desplegables se actualizan mediante código.	

Tabla 59: Tarea detallada 4 de la HU9.

Anexo III: Interfaces gráficas

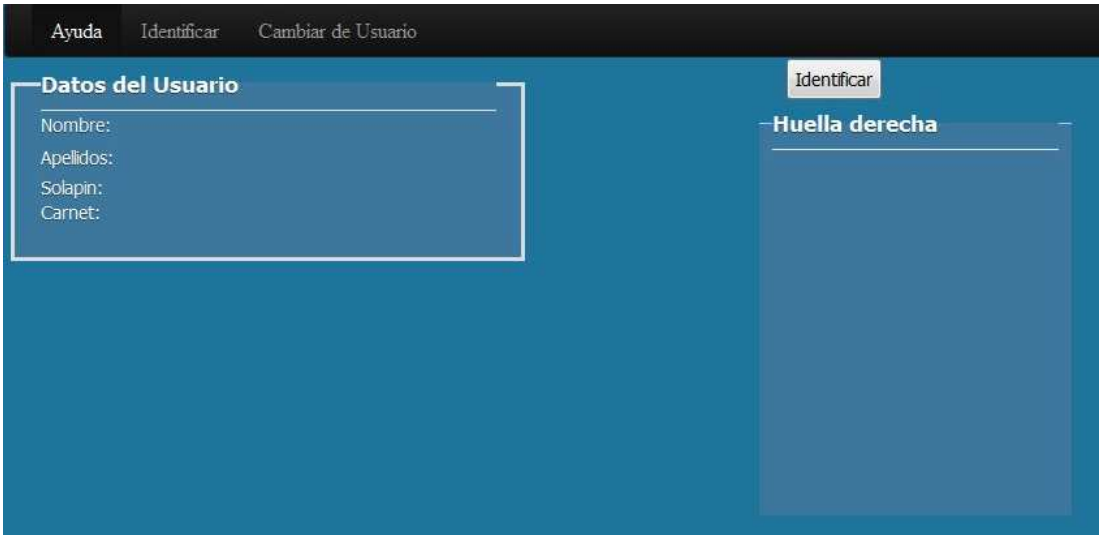

Figura 12: Interfaz gráfica Identificar.

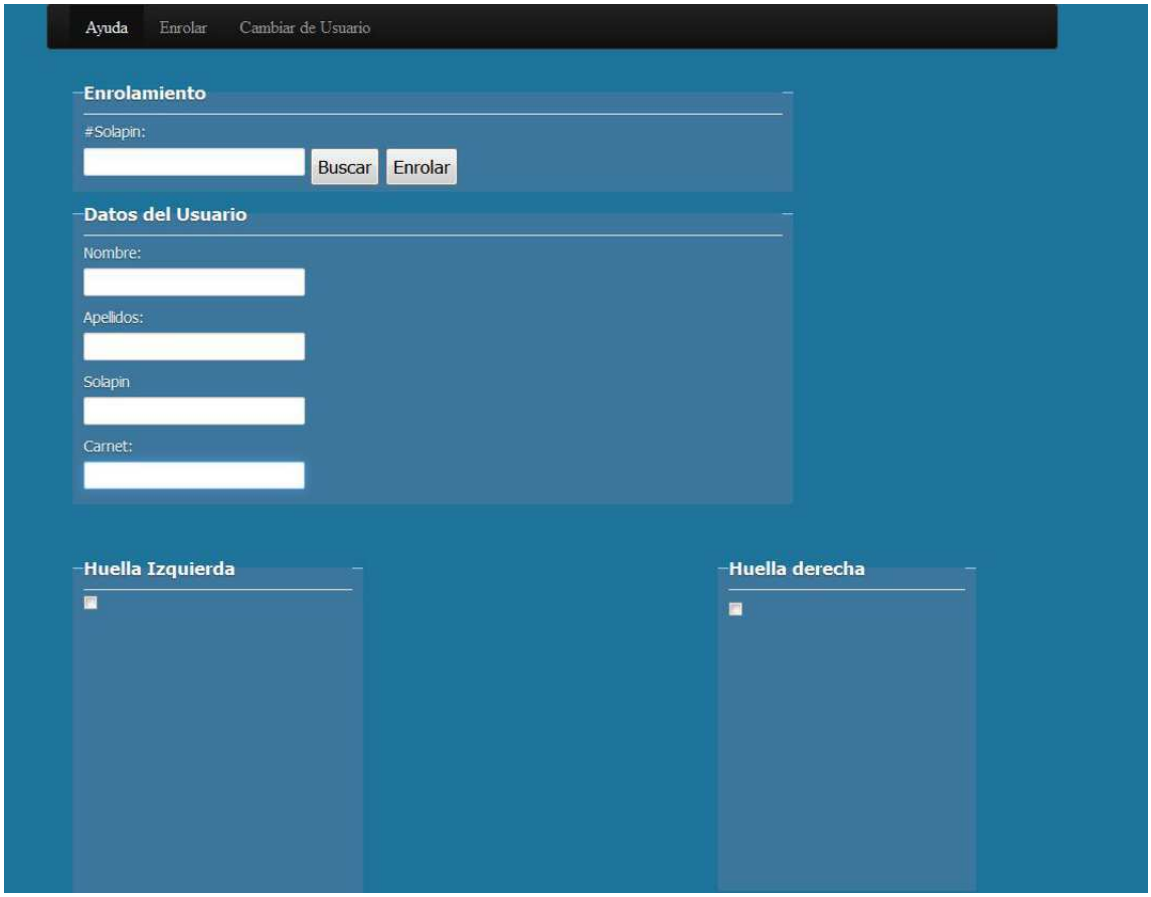

Figura 13: Interfaz gráfica Enrolar.

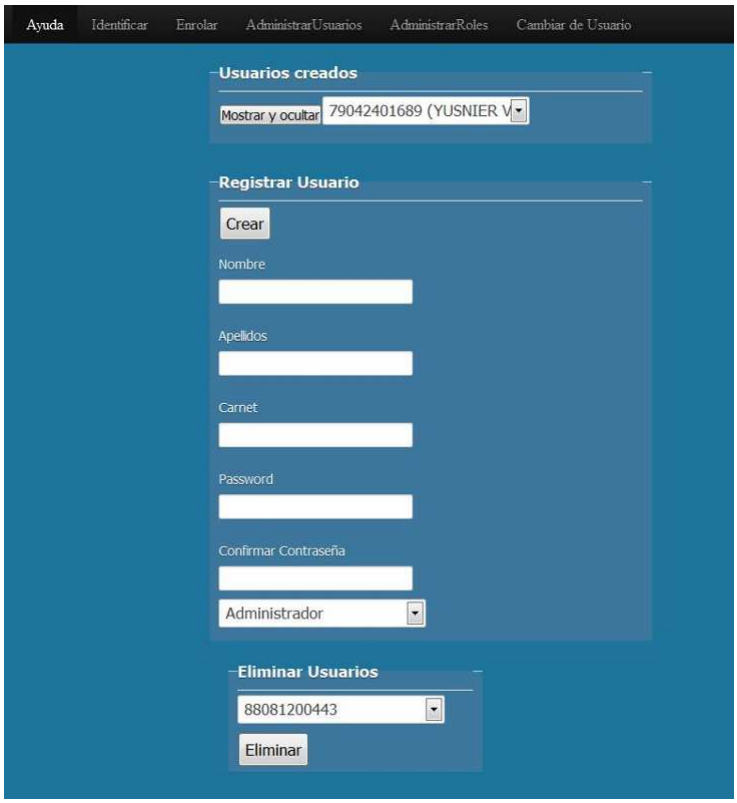

Figura 14: Interfaz gráfica Administrar Usuario.

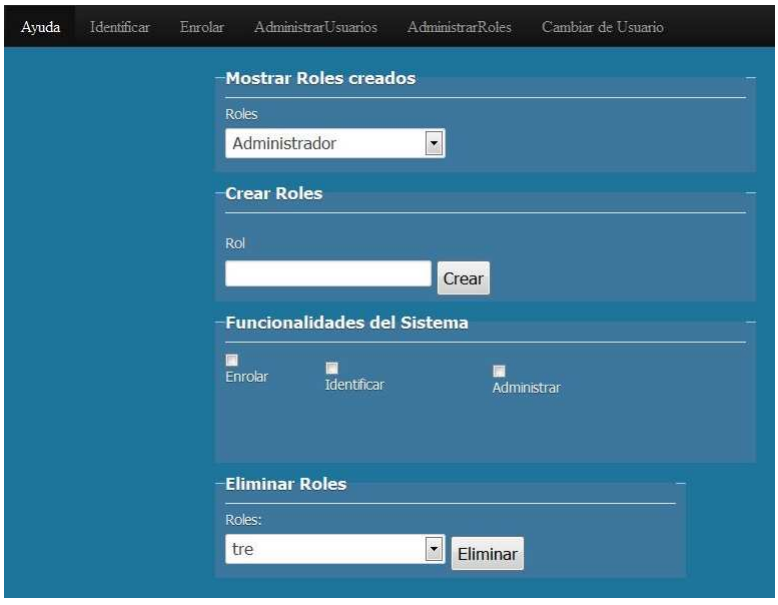

Figura 15: Interfaz gráfica Administrar Rol.

Anexo IV: Pruebas de aceptación

Caso de prueba de aceptación	
Código: HU2CP1	**Historia de usuario:** Realizar conexión con el DGM.
Responsable de la prueba: Osmayda Silverio Leyva	
Descripción: Prueba para verificar que el DGM realiza el proceso de captura de huella.	
Condiciones de ejecución: Debe estar instalado el DGM.	
Entrada/Pasos de ejecución: 1. Se verifica que esté configurado para el escanner de huella a utilizar. 2. Se instalan los drivers del escanner. 3. Se incluyen los archivos DGMJSCore y DGMJSServicesMap en las vistas Enrolar e Identificar para los procesos de dichas vistas. 4. Se incluyen las funciones javascript necesarias para el proceso de conexión y captura de huella .	
Resultado esperado: Mostrar las imágenes de las huellas en las interfaces Enrolar e Identificar.	
Evaluación de la prueba: Prueba satisfactoria.	

Tabla 60: Prueba de aceptación para la HU2.

Caso de prueba de aceptación	
Código: HU3CP1	**Historia de usuario:** Enrolar personal.
Responsable de la prueba: Ernesto Sabuquet Hurtado	
Descripción: Prueba para verificar que dado los datos encontrados a través del # de solapín se enrolan los datos de la persona.	
Condiciones de ejecución: El lector de huella debe estar conectado a la aplicación.	
Entrada/Pasos de ejecución: 1. Se busca los datos de la persona a través del # de solapín. 2. Se capturan las huella del pulgar derecho y pulgar izquierdo. 3. Se acciona el botón enrolar.	
Resultado esperado: Mostrar la interfaz con los campos vacíos, para poder realizar otro enrolamiento.	

Evaluación de la prueba: Prueba satisfactoria.

Tabla 61: Prueba de aceptación para la HU3.

	Caso de prueba de aceptación
Código: HU4CP1	**Historia de usuario:** Buscar persona.
Responsable de la prueba: Osmayda Silverio Leyva	
Descripción: Prueba para verificar que dado un # de solapín salen los datos de esa persona .	
Condiciones de ejecución: Debe estar funcionando el Web Server Akademos y el Web Server ASSETS.	
Entrada/Pasos de ejecución: 1. Se introduce número de solapín de la persona. 2. Se acciona el botón buscar.	
Resultado esperado: Mostrar los datos de la persona (foto, nombre, apellidos y CI).	
Evaluación de la prueba: Prueba satisfactoria.	

Tabla 62: Prueba de aceptación para la HU4.

	Caso de prueba de aceptación
Código: HU5CP1	**Historia de usuario:** Identificar personal del centro.
Responsable de la prueba: Ernesto Sabuquet Hurtado	
Descripción: Prueba para que se encuentre la persona en la BD a través de la huella.	
Condiciones de ejecución: El lector de huella debe estar conectado a la aplicación.	
Entrada/Pasos de ejecución: 1. Se capturan la huella del pulgar derecho o pulgar izquierdo. 2. Se acciona el botón identificar.	
Resultado esperado: Mostrar en los campos los datos de la persona identificada.	
Evaluación de la prueba: Prueba satisfactoria.	

Tabla 63: Prueba de aceptación para la HU5.

Caso de prueba de aceptación

Código: HU6CP1	**Historia de usuario:** Crear rol.
Responsable de la prueba: Osmayda Silverio Leyva	
Descripción: Prueba para verificar que se ha creado un rol con los permisos dado.	
Condiciones de ejecución:	
Entrada/Pasos de ejecución: 1. Se introduce el nombre del rol que se desea crear. 2. Se seleccionan los permisos que tendrá. 3. Se acciona el botón crear.	
Resultado esperado: Se actualiza las listas desplegables Roles.	
Evaluación de la prueba: Prueba satisfactoria.	

Tabla 64: Prueba de aceptación para la HU6.

	Caso de prueba de aceptación
Código: HU7CP1	**Historia de usuario:** Eliminar rol.
Responsable de la prueba: Osmayda Silverio Leyva	
Descripción: Prueba para verificar que se ha eliminado un rol de la base de datos.	
Condiciones de ejecución:	
Entrada/Pasos de ejecución: 1. Se seleccionan el rol ha eliminar de la lista desplegable. 2. Se acciona el botón eliminar.	
Resultado esperado: Se actualizan las listas desplegables Roles.	
Evaluación de la prueba: Prueba satisfactoria.	

Tabla 65: Prueba de aceptación para la HU7.

	Caso de prueba de aceptación
Código: HU8CP1	**Historia de usuario:** Crear usuario.
Responsable de la prueba: Ernesto Sabuquet Hurtado	
Descripción: Prueba para verificar que se ha creado un usuario para la utilización de la aplicación.	
Condiciones de ejecución: El usuario debe estar presente para poner la contraseña.	

Entrada/Pasos de ejecución: 1. Se introducen todos los datos del usuario (Nombre, Apellidos, CI, contraseña) y se selecciona el rol que tendrá. 2. Se acciona el botón crear.
Resultado esperado: Se actualizan las listas desplegables Usuarios.
Evaluación de la prueba: Prueba satisfactoria.

Tabla 66: Prueba de aceptación para la HU8.

	Caso de prueba de aceptación
Código: HU9CP1	**Historia de usuario:** Eliminar usuario.
Responsable de la prueba: Ernesto Sabuquet Hurtado	
Descripción: Prueba para verificar que se elimina el usuario de la base de datos.	
Condiciones de ejecución: .	
Entrada/Pasos de ejecución: 1. Se selecciona de la lista desplegables el usuario que se desea eliminar. 2. Se acciona el botón eliminar.	
Resultado esperado: Se actualizan las listas desplegables Usuario.	
Evaluación de la prueba: Prueba satisfactoria.	

Tabla 67: Prueba de aceptación para la HU9.