Universidad de las Ciencias Informáticas

Facultad 1

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

Módulo de control de asistencia mediante identificación por huella dactilares.

Autores: Rosana Castro García

Emmanuel Gómez Soler

Tutores: Lic. Yadier Perdomo Cuevas

Ing. Royli Hernández Delgado

La Habana, Junio de 2015

"If you think technology can solve your security problems, then you don't understand the problems and you don't understand the technology."

Bruce Schneier

Declaración de autoría

Declaramos que somos los autores del resultado que exponemos en la presente memoria titulada *Módulo control de asistencia mediante identificación por huellas dactilares*, para optar por el título de Ingeniero en Ciencias Informáticas.

El presente trabajo fue desarrollado en el transcurso de los años 2014-2015.

Autorizamos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo. Finalmente declaramos que todo lo anteriormente expuesto se ajusta a la verdad, y asumimos la responsabilidad moral y jurídica que se derive de este juramento profesional.

Y para que así conste, firmamos la presente declaración jurada de autoría en La Habana a los DD días del mes de MM del año 2015.

Rosana Castro García

Firma del Autor

Emmanuel Gómez Soler

Firma del Autor

Ing. Royli Hernández Delgado

Firma del Tutor

Lic. Yadier Perdomo Cuevas

Firma del Tutor

Datos de contacto

Nombre: Yadier Perdomo Cuevas

Especialidad de graduación: Ciencias de la Computación.

Responsabilidad que ocupa: Director de Redes y Servicios Telemáticos.

Categoría docente: Asistente.

Años de experiencia en el tema: Siete años.

Años de graduado: Diez años.

Nombre: Royli Hernández Delgado

Especialidad de graduación: Ingeniera en Ciencias Informáticas.

Responsabilidad que ocupa: Especialista de la producción en el Departamento de Desarrollo de Componentes del CISED en adiestramiento.

Años de experiencia en el tema: Un año.

Años de graduado: Dos años.

Dedicatoria

A mis padres:

Por ser mis guías en los caminos tan enredados de esta vida, por el gran apoyo que recibí desde pequeña y por empujarme a alcanzar mis metas cuando ya las creía perdidas. Gracias por ser los mejores padres del mundo.

Rosana

A mis padres:

Porque hicieron todo en la vida para que yo pudiera lograr mis sueños, por motivarme y darme la mano, a ustedes por siempre mi corazón y mi agradecimiento.

Emmanuel

Agradecimientos

A mis padres: Por haberme regalado este mundo tan lleno de maravillas, por la confianza que depositaron en mí y por el cariño que me han dado. Por haberme dado las fuerzas para seguir adelante y estar presentes en cada momento feliz y triste de mi vida. Gracias por haberme empujado a cumplir mis sueños y convertirme en una persona de bien. Quiero que estos resultados que obtuve los haga sentirse aún más orgullosos de mí puesto que siempre ha sido mi objetivo ser una fuente de orgullo para ustedes. Los quiero mucho a ambos y deseo con todo mi corazón podar gozarlos por el resto de mi vida. Muchas gracias por ser mis padres.

A mi hermana: Ella es la persona que ha sido y será mi confidente toda la vida, es el ser más importante en toda la faz de la tierra y al que más quiero por encima de todo, a la que le agradezco por cubrirme siempre las fechorías y las travesuras en las que me he envuelto durante todos estos años. Te quiero mucho Hilde y te deseo todo lo mejor en tu camino.

A mi viejita hermosa: Mima muchas gracias por tus consejos, por tus palabras de aliento y por entenderme cuando tenía problemas que otros no entendían. Gracias por aguantarme los desplantes y malcriadeces de muchacho al fin. Por tu dedicación, que aunque pase el tiempo y me ponga grande, no ha cambiado un solo momento sino que crece y se vuelve enorme. Eres la mejor abuela del mundo; Eres mi abuela y Te Amo.

A mi familia: Es tan grande mi familia que ni siquiera puedo pensar en nombrarlos a todos para agradecerles lo que han hecho por mí, así que generalizando, gracias a todas mis tías y tíos que uffff son un montón, a mis primas y primos que son muchos más. Gracias tanto a todos lo que están presentes y a los que ya no forman parte de este mundo pero si a mí corazón, les agradezco desde lo más profundo de mi alma la confianza, el cariño y respeto que me han dedicado y sobre todo por la ayuda que me ha sido brindada siempre que la necesité. También a mis suegros y toda la familia de mi novio aquella que me acogió y me hizo parte de ella, gracias por la ayuda que me han brindado y por aceptarme tal como soy.

A mis amistades: Esta es mi otra gran familia, grandes hermanos y hermanas, de corazón gracias por todos los momentos que hemos compartido y que nunca pero jamás voy a olvidar. A la Peri gran amiga y hermana y a la que le deseo que corra con mucha suerte por la vida, a todos los del grupo 1X01 y digo así porque no solo hablo por los que estamos juntos en este último año si no que por todos los que me han acompañado todos estos años de la carrera y que por distintos motivos no han podido continuar. También le doy gracias a aquellos que conocí por el 2009, los Arieles, Memo, Gago, Yoelmy, Estelita, Yohanis Cecilia, ufff a todos. También a todos aquellos que durante estos seis años me han ayudado, a los profesores que se encargaron de formarme como toda una profesional Nela, Radel, Manuel Rivero principal responsable de mi permanencia en la universidad, sin palabras, a todos aquellos que confiaron en mí y que me solventaron las dudas cuando las presentaba.

A mi novio: Por sus constantes regaños para ponerme nuevamente a trabajar en lograr mis metas, por su empeño en convertirme en la profesional que soy actualmente. Gracias por tu amor y cariño y por aguantar estos cinco años mis ataques. Gracias por los momentos lindos que me has dado y por ayudarme y calmarme en los momentos tristes. Gracias por estar ahí cuando te he necesitado, Te adoro.

A mis tutores: A ti Yadier muchas gracias por defendernos a capa y espada, por hacernos entender tu punto de vista con esas formas tan particulares que tienes. Gracias por tu ayuda y por tu apoyo, te admiro muchísimo. A ti Royli te debo esas madrugadas sin dormir que me dejaban muerta para el día siguiente, también las incontables mañanas y tardes de café, los ataques que cogía cada vez que algo no te gustaba, pero sabes qué...? todo eso, hoy te lo agradezco. No puedo pensar en un mejor tutor.

A mi compañero de tesis: Además de ser mi compañero de tesis también eres uno de esos grandes amigos a los que nunca podré olvidar, gracias por aguantarme esos días en los que me portaba imposible y peleonera. Gracias por los resultados que alcanzamos, te los debo a ti. Te agradezco por esos días completos que lucían interminables, las madrugadas que no cesaban, sin duda no me puedo quejar. Gracias a estos meses juntos me gané un amigo para toda la vida y quiero que sepas que puedes contar conmigo para lo que sea. Te deseo todo el éxito posible en tu vida profesional y espero que todo lo que hemos logrado te sirva para construir un futuro brillante.

Rosana

A mis padres por su amor, trabajo y sacrificios en todos estos años, gracias a ustedes he logrado llegar hasta aquí y convertirme en lo que soy.

A mis abuelas por ser una fuente de apoyo incondicional en toda mi vida.

A Isis por su amor y comprensión todos estos años, por estar ahí cuando más la necesitaba.

A mi hermana, mis tíos y toda mi familia que siempre me han apoyado en todos los momentos.

A mi amigo y tutor Royli por su paciencia y dedicación, por enseñarme a ser mejor siempre.

A mi tutor Yadier por sus consejos y el tiempo dedicado.

A mi compañera de tesis Rosana quien me ha apoyado y acompañado… ¡al fin nos graduamos!

A mis amigos por los grandes momentos vividos, por los recuerdos.

Emmanuel

Resumen

Los Sistemas de control de acceso en la actualidad son instalados para brindar seguridad en muchas de las grandes y pequeñas empresas a nivel mundial. Pero no basta con que se tenga un punto de acceso en determinado lugar sino que se ha convertido en una necesidad conocer quien tiene acceso o no a estas instituciones y con qué frecuencia se hace. Esto es posible si se tiene un control de asistencia dentro de este tipo de sistemas, que brinden datos relacionados con cada actividad que se realice principalmente con las entradas y salidas a estos lugares. Otro elemento importante en estos sistemas es el uso de las técnicas biométricas, principalmente las huellas dactilares en el proceso de identificación, lo que los hace más confiables y seguros.

El Centro de Identificación y Seguridad Digital perteneciente a la Universidad de las Ciencias Informáticas dispone de un Sistema de control de acceso que se encuentra instaurado en el propio centro. Este sistema inicialmente cuenta con un registro de entrada pero los datos que provee son insuficientes y la calidad de estos no está a la altura de los sistemas homólogos analizados en la investigación a nivel mundial. Además el proceso de identificación se hace utilizando credenciales que provocan la suplantación de identidad entre los propios usuarios del centro. Es por ello que se hace necesario desarrollar un Módulo de control de asistencia utilizando las huellas dactilares como técnica biométrica para la identificación de cada usuario que se integre con el Sistema de control de acceso antes mencionado.

En el documento se recogen los resultados de la investigación realizada, haciendo un estudio de las principales características de los sistemas de control de asistencia existentes. Se explica la arquitectura y el diseño del sistema propuesto. Se describen las herramientas y tecnologías utilizadas, así como los artefactos generados en el proceso de desarrollo.

Palabras clave: Control de acceso, identificación, control de asistencia, huellas dactilares.

Índice

Índice de figuras

Índice de tablas

Introducción

En la actualidad las brechas de seguridad han aumentado y se convierten en una amenaza para cualquier empresa o institución. Esto sucede principalmente por los accesos no autorizados que se realizan a determinado lugar, por personas que no tienen permisos de acceso a un área determinada de la institución para la que labora o bien porque no pertenecen a ella. Para cubrir esta necesidad en las organizaciones existen mecanismos de control de acceso sobre las áreas (seguridad física) que se desean proteger (1).

En sus inicios los controles de acceso físico se basaban esencialmente en el trabajo de personas que actuaban como vigilantes, en el mejor de los casos, asistidos por equipos de Circuito Cerrado de Televisión (*CCTV*). Con el incremento de la potencia de cómputo en los últimos años, la incorporación de las aplicaciones sustentadas en la biometría y las ciencias informáticas en general se han desarrollado nuevas tecnologías y sistemas informáticos destinados al control de acceso físico. Entre los más convencionales se encuentra el uso de tarjetas identificativas tales como el Documento de Identificación Nacional (DNI) o credenciales especiales creadas para un ambiente específico (2).

Utilizar la biometría para la identificación de personas es fácil pues no es necesario haber memorizado ni cambiado una característica identificativa, mucho menos se corre el riesgo de perderla. Al no poder ser adivinada o transferida una característica humana, el nivel de seguridad que proporcionan los sistemas de control de acceso que implementan técnicas biométricas para la identificación de personas es más alto que hacer uso de una contraseña u otro método común. La identificación por reconocimiento facial, huellas dactilares, frecuencia de voz, iris, retina son ejemplos de estas técnicas. La huella dactilar es la más antigua de las técnicas biométricas y el rasgo individual más utilizado para el proceso de identificación personal por su facilidad de adquisición, de uso, fiabilidad y gran aceptación por parte de los usuarios (3). Por ello en el mundo existen muchas soluciones informáticas implementadas en las empresas que se auxilian de esta técnica biométrica para identificar a sus trabajadores. No solo ha sido una necesidad controlar quién está autorizado a acceder o no a un establecimiento, sino que también es relevante conocer qué se hace cuando ya se ha identificado y accedido a este.

Al ser los sistemas de control de acceso una de las tecnologías con más demanda en el mercado actual y para cubrir la necesidad antes descrita en las empresas se han migrado sistemas mecánicos *(relojes para control de personal, reloj de control de asistencia)* (4) y con personal especializado, a tener procesos

automatizados de control de asistencia, con diferentes tipos de tecnologías y dispositivos (5). Su objetivo es gestionar el control de entrada/salida, los horarios, calendarios, días festivos entre otras funcionalidades de los usuarios que están registrados en el sistema, generando reportes diarios de cada actividad realizada, así como controlar la circulación de personal ajeno a las instituciones. Son numerosas sus aplicaciones, y tienen en diferentes entornos funcionalidades que son similares, como por ejemplo: en la esfera laboral son implementados para llevar un control de la puntualidad y asistencia de los trabajadores, gestionar de manera automática las incidencias tanto de entradas como salidas así como contabilizar el tiempo de utilización de los recursos. También es válido destacar que al tener controlados los horarios de los trabajadores se pueden establecer estadísticas para el pago por los servicios reales prestados por cada uno de ellos. Por otra parte en entornos educacionales son utilizados para detectar cuando se han manifestado faltas injustificadas, absentismos, tardanzas y otras incidencias por parte de los estudiantes y profesores, con el objetivo de actuar en consecuencia de ellas lo antes posible (5).

El Centro de Identificación y Seguridad Digital *(CISED)* perteneciente a la Universidad de las Ciencias Informáticas (*UCI*) dispone de un sistema de control de acceso que cuenta con un procedimiento de identificación basado en una credencial única para cada usuario, conocida como solapín. La credencial contiene un código de barras y un número, donde una vez que son ingresados al sistema cualquiera de estos *tokens*, queda identificado el usuario en el mismo, y con acceso a toda la red interna, así como también queda registrada su entrada. A pesar de que los *tokens* de seguridad utilizados en este proceso son únicos para cada individuo no son suficientes y se prestan fácilmente a fraudes de identidades. Esto ocurre principalmente debido a que en el proceso de identificación interviene además un técnico, quien puede no percatarse del error, y permitir el acceso a una persona que dice ser quien no es. Esta vulnerabilidad del sistema es aprovechada tanto por usuarios del centro como por ajenos a este trayendo consigo algunos inconvenientes que a continuación se detallan. Una persona ajena al centro podría realizar robos al tener a su disposición los medios físicos, además si posee las credenciales de autenticación del usuario que suplanta puede acceder al repositorio, a servidores de archivos, de documentación, o a las bases de datos disponibles donde fácilmente puede modificar, sustraer o compartir la información utilizando los medios de comunicación por la red *(mensajería instantánea o correo electrónico)*. También podría realizar, desde cualquier computador, ataques que impliquen la caída de servidores y servicios así como introducir programas malignos en las computadoras que dañen la integridad de la información contenida en ellas.

Además de incurrir en lo antes descrito, los usuarios del centro que marcan la asistencia de otros usuarios provocarían que el sistema genere registros no confiables que afecten el control del flujo del personal.

Aunque a través del control de acceso que se encuentra instaurado se registran los ingresos diarios de cada usuario no se puede determinar en qué momento del día estos egresan del centro. Además no cuenta con un procedimiento que permita gestionar los horarios de los trabajadores por lo que no es posible conocer cuánto tiempo han permanecido en su lugar de trabajo y si han estado cumpliendo con las tareas que se les asignaron. Tampoco el sistema genera reportes que permitan obtener datos relacionados con el comportamiento de los usuarios para realizar el control de los mismos ya sean los días no trabajados, las llegadas tardes, porciento de horas trabajadas, entre otros.

A partir de la situación planteada anteriormente y de los factores identificados que ponen a disposición de personal ajeno los recursos del centro, surge el siguiente **problema de investigación** ¿Cómo aumentar la confiabilidad del proceso de identificación del control de asistencia del personal que accede al entorno seguro del CISED?

Para dar solución al problema antes mencionado se define como **objeto de estudio** los Sistemas de Control de Asistencia mediante identificación por huellas dactilares.

Como una alternativa de solución al problema planteado, se define como **objetivo general** desarrollar un módulo de control de asistencia de personal con identificación por huellas dactilares para aumentar la confiabilidad de la identificación de las personas en el sistema de control de acceso al entorno seguro desarrollado en el CISED.

Se definen como **objetivos específicos**:

1. Confeccionar el marco teórico de la investigación para lograr mayor entendimiento sobre el estado del arte en Cuba y en el mundo referente a los sistemas de control de acceso y asistencia mediante identificación por huellas dactilares.
2. Realizar el análisis y diseño del módulo de control de asistencia mediante identificación por huellas dactilares para su posterior implementación.
3. Implementar el módulo de control de asistencia mediante identificación por huellas dactilares.

4. Realizar pruebas al módulo de control de asistencia mediante identificación por huellas dactilares para comprobar su correcto funcionamiento.

Las **tareas de investigación** definidas para darle respuesta al objetivo de la investigación son:

1. Caracterización de los principales procesos de un sistema de control de asistencia mediante la identificación de personas por huellas dactilares. (Rosana Castro García)
2. Análisis de aplicaciones y tendencias actuales existentes en el ámbito nacional y extranjero para la implementación de controles de asistencia mediante huellas dactilares. (Emmanuel Gómez Soler)
3. Caracterización de las herramientas que se utilizarán en el entorno de desarrollo. (Emmanuel Gómez Soler)
4. Análisis de técnicas de ingeniería de requisitos aplicables al dominio del problema. (Rosana Castro García)
5. Definición del estándar de codificación y explicación de las técnicas de programación a utilizar. (Rosana Castro García y Emmanuel Gómez Soler)
6. Descripción de la arquitectura. (Rosana Castro García)
7. Ajuste de la solución según los patrones de diseño más adecuados para la usabilidad del módulo en el CISED. (Emmanuel Gómez Soler)
8. Implementación de los requisitos especificados. (Emmanuel Gómez Soler y Rosana Castro García)
9. Obtención de los artefactos generados según el proceso de desarrollo aplicado para la realización de una propuesta de solución. (Rosana Castro García y Emmanuel Gómez Soler)
10. Definición de las pruebas a realizar al módulo de control de asistencia. (Emmanuel Gómez Soler)
11. Validación de la solución implementada mediante pruebas de funcionalidad. (Rosana Castro García y Emmanuel Gómez Soler).

En la investigación se utilizarán diferentes **métodos científicos** que sirven de ayuda y guía para la recopilación y organización de la información, en este proceso intervienen los métodos teóricos y los métodos empíricos.

Métodos teóricos:

- **Analítico - Sintético**: Se utilizará con el objetivo de analizar los diferentes conceptos asociados al dominio del problema como: sistemas de control de asistencia, huellas dactilares, entre otros.

También para extraer los elementos más importantes relacionados con el tema que ocupa la investigación.

- **Análisis Histórico - Lógico:** Se utilizará con el objetivo de indagar sobre el avance alcanzado por los sistemas de control de asistencia mediante huellas dactilares y los aportes que han brindado a la sociedad.
- **Modelación:** Se utilizará en la creación de los diagramas generados en el diseño del software para un mejor entendimiento del módulo.

Método empírico:

- **Entrevista:** Se utilizará para dialogar con el cliente en los diferentes encuentros que se realicen para definir las funcionalidades, restricciones y las interfaces del módulo.

Justificación de la investigación

Los sistemas de control de acceso basados en la identificación utilizando técnicas biométricas brindan un alto nivel de seguridad e identificación personal en variados escenarios. Estos sistemas además de ofrecer la posibilidad de obtener reportes acerca de los incidentes de seguridad, al contar con controles de asistencia facilitan la gestión de horarios y de visitantes entre otros factores que son importantes para una institución. Sumándole a estos la identificación mediante el uso de las huellas dactilares se acrecienta la seguridad en este tipo de sistemas y por tanto disminuyen los fraudes de identidad.

Adquirir sistemas de esta índole resulta muy costoso para el país, por lo que es necesario que se desarrollen soluciones propias que cuenten con la calidad y requerimientos que demandan este tipo de productos internacionalmente. El aporte práctico de esta investigación va dirigido a desarrollar un módulo de control de asistencia utilizando las huellas dactilares en el proceso de identificación de las personas que acceden diariamente al CISED.

Estructura del Documento

El presente documento se encuentra dividido en tres capítulos estructurados de la siguiente forma:

Capítulo 1: Fundamentación teórica. En este capítulo se describen los principales conceptos relacionados con el dominio del problema de investigación, se realiza un estudio del estado del arte sobre los sistemas

de control de asistencia mediante huellas dactilares y se analizan las principales metodologías, tecnologías y herramientas para darle solución al problema planteado.

Capítulo 2: Análisis y diseño de la propuesta de solución. En este capítulo se realiza una descripción general de la solución propuesta y su funcionamiento, además se detallan los principales aspectos relacionados con el diseño. Se realiza el levantamiento de los requisitos funcionales y no funcionales, así como se define la arquitectura que organice la lógica del módulo de control de asistencia mediante identificación por huellas dactilares. También se especifican los patrones del diseño que se van a aplicar y los artefactos derivados de la metodología de desarrollo de software que se seleccione.

Capítulo 3: Implementación y prueba. En este capítulo se exponen los resultados de la solución propuesta por lo que se muestran aspectos relacionados con la implementación del módulo de control de asistencia mediante identificación por huellas dactilares, como los diagramas de despliegue y componentes. Además se realizan las pruebas pertinentes al módulo de control de asistencia, dirigidas a verificar su correcto funcionamiento y se muestran los resultados obtenidos.

Capítulo 1: Fundamentación teórica

1.1 Introducción

En este capítulo se abordan los principales elementos relacionados con el dominio del problema de investigación, para ello se realiza un estudio del estado del arte sobre los sistemas de control de asistencia que utilizan en el proceso de identificación las huellas dactilares. Por último se analizan varias metodologías, tecnologías y herramientas en aras de seleccionar las más adecuadas para darle solución al problema.

1.2 Conceptos fundamentales

Huella Dactilar: Una huella dactilar es la impresión visible o moldeada que produce el contacto de las crestas papilares de un dedo de la mano (generalmente se usa el dedo pulgar o el dedo índice) sobre una superficie, que como característica individual distingue a todos los seres humanos (6).

Identificación: La identificación personal es un proceso que se puede realizar de distintas maneras ya sea visual (cuando una persona reconoce o es reconocida por otra), por medio de un documento que alegue o acredite que es la persona que dice ser o bien mediante el uso de sistemas automáticos tales como: captor de huella dactilar, reconocimiento facial, reconocimiento voz, control mediante lectura del iris, palma de la mano, lector de venas, entre otros (7).

Control de Asistencia o presencia: Los sistemas de control de asistencia se basan esencialmente en registrar las entradas y salidas de las personas a un local específico en un horario determinado garantizando de esta forma que se pueda conocer en qué momento se ha accedido a ese lugar (5).

1.3 Sistemas de Control de asistencia

Los sistemas de control de asistencia laboral permiten registrar y chequear la asistencia y puntualidad de los trabajadores en sus lugares de trabajo. La importancia de estos sistemas radica en la injerencia que tienen en las actividades de la empresa u organización, pues contribuyen a monitorear la productividad para el logro del cumplimiento de las metas que se tracen. El modo de funcionamiento de estos sistemas es el mismo para todos donde se entrega a cada empleado de la empresa un elemento de identificación para que registre su ingreso y salida que podría ser por proximidad o biometría. A través de ellos se pueden establecer

estadísticas que hacen más eficientes los procesos del departamento de nóminas o recursos humanos principalmente.

A la hora de implementar un control de asistencia en las empresas se tienen en cuenta varios factores que varían en dependencia de las necesidades y los requerimientos de cada una de ellas. Inicialmente el método que se va a utilizar para la identificación del personal es el primer paso a decidir basándose en argumentos de costo y facilidad de uso. Otro factor importante es la cantidad de usuarios que marcan su registro de entrada y salida así como cuántos de ellos lo realizarán en el mismo horario. Por último y no menos importante está el tipo de información o reportes que emite el sistema y el grado de seguridad de ellos.

Otro de los datos con el que se trabaja junto con el registro de entrada y salida es el horario ya que indica en qué momento la persona debe estar en su puesto de trabajo, de modo que cuando se compare el horario con los registros de entradas y salidas se puedan llevar a cabo las tareas de cálculo y control.

1.4 Sistemas homólogos

Los sistemas de control de asistencia en el mundo tienen gran aceptación por parte de las empresas y organizaciones que los implementan para cubrir las necesidades existentes relacionadas con el control de los empleados de cada una de ellas. Estos pueden utilizar o no elementos biométricos para el proceso de identificación por lo que a continuación se describen algunos de ellos y las principales características que presentan.

1.4.1 Sistemas internacionales

Control Biométrico de Acceso y Asistencia de Personal i-Huellas Planígrafo

i-Huellas Planígrafo es un sistema de control biométrico con lector de huellas dactilares. El registro de las entradas y salidas del personal se realiza colocando el dedo en el sensor biométrico. El sistema se encarga de verificar, registrar y comparar la huella dactilar con las almacenadas previamente, identificándola o rechazándola (8). Tiene una interfaz de administración del programa y otra de acceso y verificación de entrada y salida de personal. Cuenta con un simulador de voz, que guía a los usuarios por medio de mensajes de voz de bienvenida personalizados en los momentos de identificación y mensajes de voz de acción. Almacena varios registros de huellas de uno o más dedos para un mismo usuario para en caso de lesión o eventualidad el usuario se pueda identificar. Genera reportes (de acuerdo a cada versión) que

pueden ser exportados a un archivo en Excel, para ver en pantalla y listos para imprimir. Permite seleccionar cualquier rango de fechas, y escoger entre un reporte general agrupado por usuario o un reporte de un usuario específico. Atendiendo a las necesidades del cliente están disponibles varias versiones (i-Huellas básico para control biométrico de asistencia, i-Huellas empresarial para control biométrico de asistencia), diferenciándose en la información almacenada y los reportes generados ya que estos varían de una versión a otra. Este sistema es propietario y requiere versiones de sistemas operativos *Windows XP* en adelante.

Dointech Control de asistencia del personal

El control de acceso desarrollado por Dointech cuenta con un módulo de **registro de control de asistencia** al que se puede acceder desde cualquier lugar del mundo vía internet permitiendo su monitoreo (9). El modulo principal de este sistema tiene la capacidad de realizar la autenticación de cada empleado a través de un lector de huella dactilar, tarjeta de proximidad, clave o la combinación de cualquiera de los tres. Además cuenta con una cámara integrada para capturar una fotografía en el momento del registro y una cámara de video que se integra a un monitor externo, o a un DVR, para ver el video en tiempo real del personal que está realizando el registro. También cuenta con un *software* embebido que se puede configurar y personalizar atendiendo a las necesidades de las empresas que lo utilicen.

Factores como horarios de entrada y salida, días laborales, la cantidad de empleados, áreas permitidas/restringidas, departamentos, alertas, gestión de visitantes son algunas de las características a las cuales se puede adaptar. No es necesario instalar *software* adicional en las computadoras de las empresas ya que es un *software* completamente web.

El módulo de registro de control de asistencia además posee diferentes niveles de usuarios para mantener mayor seguridad. Tiene varios registros de usuario así como también genera reportes personalizados según las necesidades actuales. Puede ser aplicado en fábricas, empresas, hospitales y en muchas otras esferas del sector público.

1.4.2 Sistemas nacionales

En Cuba el uso de sistemas de control de asistencia se ha extendido en los últimos años a partir de la necesidad que existe de controlar los horarios a los trabajadores en sus respectivos locales de trabajo. La implantación de los mismos está a cargo principalmente de entidades dedicadas a proveer servicios de

seguridad y protección, a continuación se muestran algunos de estos que han permitido mayor seguridad y organización en las instituciones.

Sistema de Control de Asistencia Laboral (KONASI).

KONASI es un sistema desarrollado por la Empresa de Desarrollo de Software del Ministerio de Informática y las Comunicaciones *(Desoft)* en Holguín que registra la entrada y salida de los trabajares de una entidad mediante un dispositivo de código de barras o manualmente (10). Permite gestionar las incidencias relativas al cumplimiento de la disciplina laboral. Además de obtener reportes de información referente a la detección de las incidencias relacionadas con la variación en la utilización del fondo de tiempo laboral diario y mensual.

Entre sus funcionalidades el sistema permite identificar un trabajador mediante un código de barras mostrando sus datos y las especificaciones de su marcaje: fecha y hora. Permite consultar el registro de asistencia de los trabajadores, indicando un área específica de la entidad y un periodo o rango de fechas, por lo que es posible calcular el total de horas de ausencia de cada trabajador. También registra los tipos de afectaciones según las claves de incidencias de cada trabajador y los días en que incurre, ya sea: certificado médico, vacaciones, licencias, entre otros. Seleccionar una clave de incidencia en el registro de asistencia de los trabajadores como identificador oficial. Se obtienen reportes de ausentes diarios, estadísticas de presencia de los trabajadores, resumen de ausencias en periodos seleccionados.

Biomesys Control de Asistencia

Biomesys Control de Asistencia es un sistema que aprovecha las bondades de las tecnologías que aplican la biometría, para registrar los eventos de asistencia en una organización por medio de la identificación de los empleados y de la autenticación de su identidad mediante un sensor biométrico de huellas dactilares (11). A partir de la captura de identificaciones biométricas únicas, el sistema se convierte en un generador de datos altamente confiable por su bajo o casi nulo nivel de vulnerabilidad por la suplantación de identidad. De utilidad para los que pretenden apoyarse en un instrumento sencillo con un enfoque flexible y adaptable a la estructura funcional de la organización.

Este sistema tiene la gran ventaja que se puede implementar en cualquier tipo de organización, pudiendo integrarse con medios de autenticación biométricos y otros medios de seguridad (*códigos de barras, tarjetas magnéticas*). Permite realizar el control de entradas y salidas a través de interfaces de trabajo en ambiente web, siendo altamente configurable y permitiendo la obtención de estadísticas sobre incidencias.

Después de analizar los sistemas de control de asistencia existentes en el mundo se puede concluir que:

- ✓ Los sistemas internacionales tienen como principal inconveniente que son *software* privativo, por lo que además del costo de adquisición que implican presentan limitaciones a la hora de modificar o redistribuir el código fuente, por lo que en el caso particular de Cuba sería más factible utilizar un producto propio que cumpla la calidad y precisión adecuada, pero sobre todo que consuma la menor cantidad posible de recursos financieros.
- ✓ Las soluciones existentes en Cuba también utilizan *software* privativo además tienen procesos de control de asistencia de forma manual lo que no provee resultados competentes en correspondencia con los existentes a nivel internacional y por ende presentan dificultad para integrarse con sistemas que utilicen tecnologías libre.

1.5 Metodologías de desarrollo de software

Una metodología de desarrollo de software es un marco de trabajo para estructurar, planificar y controlar el proceso de desarrollo de software (12), guiar a los desarrolladores para lograr obtener un software de calidad que cumpla con los requerimientos que son especificados por el cliente es el objetivo principal de estas, ajustándose a un costo razonable y a los recursos necesarios para el desarrollo de estos productos.

Cada software implica condiciones muy diversas, con el fin de facilitar y agilizar su creación existen disímiles metodologías, las cuales pueden ser agrupadas en metodologías tradicionales y en metodologías ágiles.

1.5.1 Metodologías tradicionales

Las metodologías tradicionales se focalizan en documentación, planificación y procesos. Su atención va dirigida a llevar una documentación exhaustiva de todo el proyecto y a cumplir con el plan que se traza para la confección del software, definiendo todo esto en la fase inicial del desarrollo del proyecto (13). Otra característica importante de este enfoque es que se tienen altos costos al implementar un cambio lo que no garantiza una buena solución para proyectos donde el entorno es volátil. Algunos ejemplos de metodologías tradicionales son: RUP, MSF, Iconix, y *Win-Win Spiral Model*.

1.5.2 Metodologías ágiles

En cambio, las metodologías ágiles son sencillas y rápidas en su ejecución, donde la documentación no tiene mucha relevancia y la entrega de resultados al cliente, es continua. Son adoptadas para proyectos cortos y pequeños, la aparición de riesgos se puede mitigar relativamente fácil ya que estas metodologías se usan en proyectos que no tienen procesos complejos (14). En síntesis, con las metodologías ágiles existen pocos artefactos, pequeños equipos de trabajo y por ende un número reducido de roles, y un contrato no tradicional que permite flexibilidad en términos de costos, tiempos y compromisos.

Algo distintivo de las metodologías ágiles es que son adaptativas, porque están pensadas para contextos cambiantes, proyectos con requerimientos inestables, cambios tecnológicos y cambios de personal. Están orientadas a las personas, porque necesitan gran participación y compromiso de todos, incluyendo al cliente, en el desarrollo del software.

Son muchas las características que diferencian a las metodologías tradicionales de las ágiles, y estudiarlas para dar respuesta al problema de investigación, hace más fácil elegir la correcta. Para mayor entendimiento se resume en la siguiente tabla comparativa algunas de estas características.

Metodologías Ágiles	Metodologías Tradicionales
Basadas en heurísticas provenientes de prácticas de producción de código	Basadas en normas provenientes de estándares seguidos por el entorno de desarrollo
Especialmente preparadas para cambios durante el proyecto	Cierta resistencia a los cambios
Impuestas internamente (por el equipo de desarrollo)	Impuestas externamente
Proceso menos controlado, con pocos principios	Proceso mucho más controlado, con numerosas políticas/normas
No existe contrato tradicional o al menos es bastante flexible	Existe un contrato prefijado
El cliente es parte del equipo de desarrollo	El cliente interactúa con el equipo de desarrollo mediante reuniones

Grupos pequeños (<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 del software	La arquitectura del software es esencial y se expresa mediante modelos

Tabla 1. Diferencias entre Metodologías Ágiles y Tradicionales.

Por lo analizado previamente se decide hacer uso de las metodologías ágiles para guiar el proceso de desarrollo del módulo. A continuación se analizan tres de estas metodologías para posteriormente seleccionar la que mejor se ajuste a las condiciones del equipo de desarrollo.

Extreme Programming (XP).

XP es una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en el desarrollo de software, promoviendo el trabajo en equipo y la comunicación fluida entre todos los participantes incluyendo el cliente. Se define como especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes, y donde existe un alto riesgo técnico (14).

El ciclo de desarrollo en XP es iterativo e incremental (Ver Figura 1); consta de seis fases: Exploración, Planificación de la entrega, Iteraciones, Producción, Mantenimiento y Muerte del proyecto.

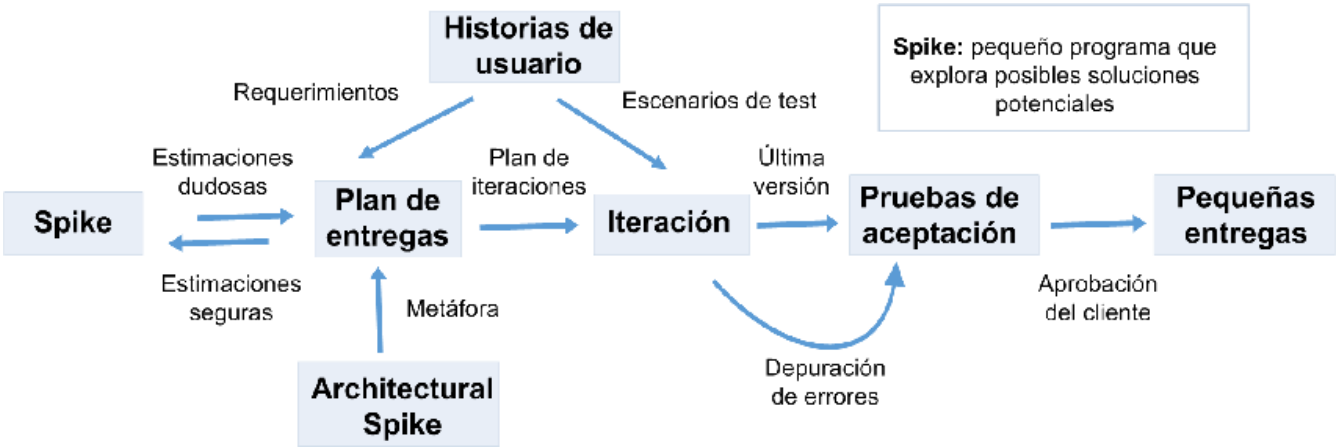

Figura 1. Flujo de Trabajo de XP.

XP define historias de usuario para describir las funciones del sistema, las cuales son escritas por el cliente. Se crea un plan de entrega entre clientes y equipo de desarrollo, tomando como base las historias de usuario y la arquitectura. En cada entrega se discuten entre las partes los objetivos y se definen las iteraciones necesarias para cumplirlos. Los proyectos realizados, guiados por esta metodología, cumplen con lo estrictamente necesario en su funcionalidad en el momento necesario: hacer lo que se necesita cuando se necesita.

Durante todo el desarrollo está presente la retroalimentación entre el cliente y el equipo de desarrollo. El cliente guía el trabajo siempre hacia los elementos de mayor valor en el negocio, o sea, sus prioridades y restricciones, mientras que los programadores estiman el esfuerzo necesario para su codificación. Esta unión es la clave del éxito de XP, pues es el factor que posibilita asumir los cambios sin mayores dificultades (15).

Un proyecto se inicia con una fase de exploración donde se sientan las bases para que sea exitoso su desarrollo. El plan de entrega a seguir es concebido durante la planificación, con la participación del cliente y los desarrolladores. A continuación tienen lugar una serie de iteraciones que no concluyen hasta obtener una primera versión del sistema, o una primera entrega que es lo mismo. Se pone en producción esta versión y comienza el mantenimiento donde se implementan nuevas funcionalidades y se mantiene el sistema funcionando. En la fase de mantenimiento tiene lugar la implementación de nuevas versiones del sistema; cada nueva entrega, debe comenzar por una fase de exploración y es ahí cuando se cierra el ciclo. La única manera de interrumpir el ciclo de XP es que ocurra la muerte del proyecto.

SCRUM

Define un marco para la gestión de proyectos, que se ha utilizado con éxito durante los últimos 10 años. Está especialmente indicada para proyectos con un rápido cambio de requisitos. SCRUM aplica de manera regular un conjunto de mejores prácticas para trabajar en equipo y obtener el mejor resultado posible de un proyecto. Estas prácticas se apoyan unas a otras y su selección tiene origen en un estudio de la manera de trabajar de equipos altamente productivos.

Un proceso de desarrollo SCRUM comienza con la visión general del producto, especificando y dando detalle a las funcionalidades o partes que tienen mayor prioridad de negocio, y que pueden llevarse a cabo en un periodo de tiempo breve. Cada uno de estos periodos de desarrollo es una iteración que finaliza con

la entrega de una parte (incremento) operativa del producto. Estas iteraciones son la base del desarrollo ágil, y SCRUM gestiona su evolución en reuniones breves diarias donde todo el equipo revisa el trabajo realizado el día anterior y el previsto para el siguiente (16) (Ver Figura 2).

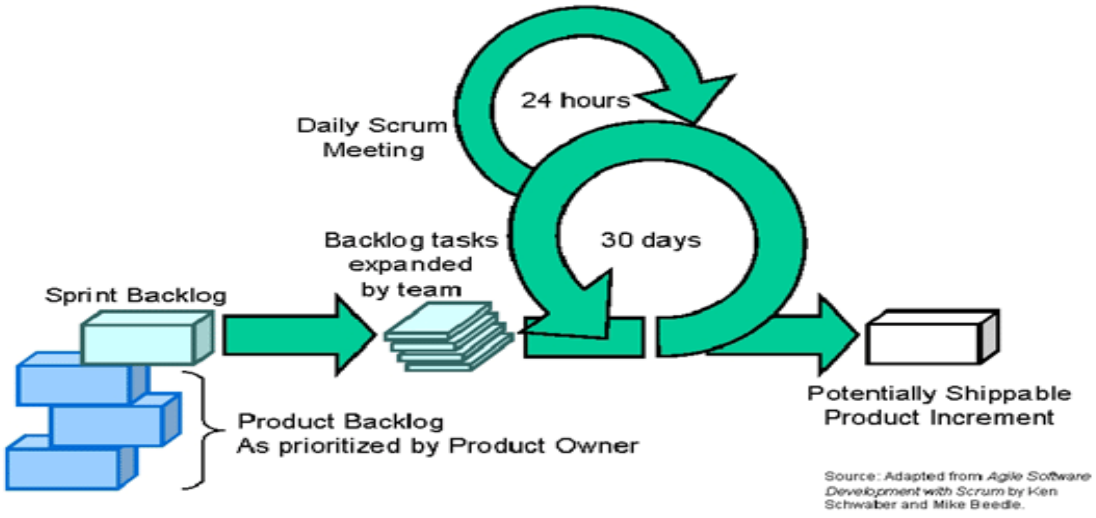

Figura 2. Flujo de trabajo de SCRUM

SCRUM comprende las siguientes fases (17):

1. **Pre-juego.**

   Planificación: Definición de una nueva versión basada en la pila actual, junto con una estimación de coste y agenda. Si se trata de un nuevo sistema, esta fase abarca tanto la visión como el análisis. Si se trata de la mejora de un sistema existente comprende un análisis de alcance más limitado. Arquitectura: Diseño de la implementación de las funcionalidades de la pila. Esta fase incluye la modificación de la arquitectura y diseño generales.

2. **Juego**.

   Desarrollo de *sprints*: Desarrollo de la funcionalidad de la nueva versión con respeto continúo a las variables de tiempo, requisitos, costo y competencia. La interacción con estas variables define el final de esta fase. El sistema va evolucionando a través de múltiples iteraciones de desarrollo o *sprints*.

3. **Post-juego**.

   Preparación para el lanzamiento de la versión, incluyendo la documentación final y pruebas antes del lanzamiento de la versión.

Feature Driven Development (FDD).

La metodología FDD se utiliza en proyectos pequeños de corta duración. Su funcionamiento está basado en iteraciones cortas, produciendo un software que permite monitorizar los procesos en cuanto al avance obtenido. Las iteraciones se deciden a través de las funcionalidades que son pequeñas partes del *software* (18).

Esta metodología consta de 5 fases:

1. Desarrollo de un modelo general.
2. Construcción de una lista de funcionalidades.
3. Plan de entregas en base a las funcionalidades a implementar.
4. Diseñar en base a las funcionalidades.
5. Implementar en base a las funcionalidades.

FDD define métricas para controlar cómo va el desarrollo del proyecto y que se pueda estimar mejor el proyecto en un futuro, además, su código fuente tiene propietario. Los equipos varían en cuanto a la funcionalidad a implementar. El conocimiento de la aplicación se reparte a través del trabajo en equipo y revisiones ya que dicha metodología presenta una jerarquía dentro del equipo de desarrollo.

1.6 Herramientas y tecnologías

En todo proceso de desarrollo de *software* es necesario la utilización de sistemas de soporte que permitan organizar, facilitar, agilizar y automatizar las tareas generadas durante su transcurso. Las herramientas y tecnologías empleadas con este fin son vitales para guiar todo el proceso y a los propios desarrolladores. Con la explotación de las facilidades y ventajas que poseen estas herramientas y tecnologías se podrá diseñar e implementar el módulo cómodamente, así como se logrará una correcta realización de las pruebas, y por consiguiente, la calidad de los resultados obtenidos.

1.6.1 Lenguaje de modelación

El lenguaje de modelado es un conjunto de notaciones que incluye símbolos estandarizados así como las distintas formas de organizarlos y estructurarlos, que facilitan el diseño del *software* orientado a objetos.

Lenguaje Unificado de Modelado (UML)

El Lenguaje Unificado de Modelado es un lenguaje de modelado visual que se usa para especificar, visualizar, construir y documentar artefactos de un sistema de software. Se utiliza además para entender, diseñar, hojear, configurar, mantener y controlar la información sobre los sistemas que se deben construir. Está pensado para usarse con todos los métodos de desarrollo, etapas del ciclo de vida, dominios de aplicación y medios. Pretende unificar la experiencia pasada sobre técnicas de modelado e incorporar las mejores prácticas actuales en un acercamiento estándar (19).

Además, UML incorpora buenas prácticas de diseño tales como la encapsulación, separación de los temas y la captura de la intención del modelo construido de modo que se pueda dar solución a los problemas actuales del desarrollo de *software* como son, el gran tamaño, concurrencia, distribución, patrones y por último y no menos importante el trabajo en equipo.

1.6.2 Herramientas CASE

Las herramientas CASE, cuyas siglas traducidas al español significan Ingeniería de *Software* Asistida por Computadora, incluyen métodos, técnicas, utilidades y documentación orientadas a la automatización del ciclo de vida del *software* (20). Son fundamentales en la disminución del tiempo de desarrollo del *software* y en el aumento de la productividad, dado que permiten enfatizar en el análisis y el diseño para minimizar el esfuerzo de codificación y prueba.

Visual Paradigm

Visual Paradigm es una herramienta CASE multiplataforma que incluye UML y soporta el ciclo de vida completo del desarrollo de *software*. Permite el modelado visual del *software* con el paradigma orientado a objetos. Entre sus características más significativas se encuentran la ingeniería directa e inversa, la modelación de todos los tipos de diagramas de clases y la generación de documentación en varios formatos (21).

Rational Rose Enterprise

Rational Rose es una herramienta CASE desarrollada y mantenida por *Rational Corporation*. Está orientada a objetos sobre la base de UML, facilita el proceso de modelado con un número significativo de estereotipos predefinidos y permite además, la generación de la documentación del software (22). Esta herramienta posee un mecanismo para generar el código de las clases definidas en el diseño UML, aunque no soporta varios lenguajes de programación.

1.6.3 Framework de desarrollo

El término *framework* se refiere a una estructura de *software* compuesta de componentes personalizables e intercambiables para el desarrollo de una aplicación. En otras palabras, un *framework* se puede considerar como una aplicación genérica incompleta y configurable que persigue como objetivos principales acelerar el proceso de desarrollo, reutilizar código ya existente y promover buenas prácticas de desarrollo como el uso de patrones (23).

Django

Django es un *framework* de desarrollo para la web. Su implementación es totalmente sobre Python. Con este marco de trabajo se pueden crear y mantener aplicaciones de alta calidad. Incluye un servidor web ligero que se puede usar mientras se desarrolla. Al mismo tiempo, Django permite trabajar fuera su ámbito según sea necesario (24).

Entre las múltiples facilidades que ofrece Django se pueden destacar el sistema de plantillas para separar la presentación de un documento de sus datos, la construcción automática de interfaces de administración, las listas genéricas que recogen ciertos estilos y patrones comunes en su desarrollo y los abstraen, de modo que se puede escribir rápidamente vistas comunes de datos sin tener que escribir mucho código. Además tiene un sistema de caché robusto y con un nivel de granularidad ajustable, que permite guardar páginas dinámicas para no tener que recalcularlas cada vez que se pidan. Django también puede integrarse fácilmente con bases de datos y aplicaciones existentes asimismo se pueden construir aplicaciones multilenguaje permitiendo especificar cadenas de traducción de más de cuarenta idiomas.

.Net

El *framework* .NET es una tecnología que admite la compilación y ejecución de la siguiente generación de aplicaciones y servicios Web XML. El diseño de este tiene como objetivos proporcionar un entorno coherente

de programación orientada a objetos en el que el código de estos se pueda almacenar y ejecutar de forma local o remota pero distribuida en Internet, proveer un entorno de ejecución de código que minimice los conflictos en el despliegue y versionado de software, elimine los problemas de rendimiento de los entornos en los que se utilizan scripts o intérpretes de comandos y que promueva la ejecución segura del producto (25).

Basar toda la comunicación en estándares del sector para asegurar que el código de .NET se pueda integrar con otros tipos de código, es otro de los objetivos de este *framework*, así como ofrecer al programador una experiencia coherente entre tipos de aplicaciones muy diferentes, como las basadas en *Windows* o en la Web.

1.6.4 Lenguaje de programación

Un lenguaje de programación es un idioma artificial diseñado para expresar procesos que pueden ser ejecutados por máquinas computadoras. Pueden usarse para crear programas que controlen el comportamiento físico y lógico de una máquina, para expresar algoritmos con precisión, o como modo de comunicación humana. Está formado por un conjunto de símbolos y reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos y expresiones (26).

CSharp (C#)

CSharp es un lenguaje de programación simple pero eficaz diseñado para escribir aplicaciones empresariales. Este lenguaje es una evolución de los lenguajes C y C++, utiliza muchas de las características de C++ en las áreas de instrucciones, expresiones y operadores. Además presenta considerables mejoras e innovaciones en áreas como seguridad de tipos, control de versiones, eventos y recolección de elementos no utilizados (liberación de memoria). También proporciona acceso a los tipos de API más comunes: .NET *Framework*, COM, Automatización y estilo C. Asimismo, admite el modo *unsafe*, en el que se pueden utilizar punteros para manipular memoria que no se encuentra bajo el control del recolector de elementos no utilizados (27).

Python

Python es un lenguaje de propósito general, independiente de plataforma y orientado a objetos, preparado para realizar cualquier tipo de programa, desde aplicaciones *Windows* a servidores de red o incluso, páginas

web. Es un lenguaje interpretado, lo que significa que no se necesita compilar el código fuente para poder ejecutarlo, lo que ofrece ventajas como la rapidez de desarrollo e inconvenientes como una menor velocidad (28).

Dispone de muchas funciones incorporadas en el propio lenguaje, para el tratamiento de *strings*[1], números, archivos, etc. Además, existen muchas librerías que podemos importar en los programas para tratar temas específicos como la programación de ventanas o sistemas en red o cosas tan interesantes como crear archivos comprimidos en .zip (28).

1.6.5 Entorno de desarrollo integrado (IDE)

Un entorno de desarrollo integrado es un programa compuesto por un conjunto de herramientas para un programador. Puede dedicarse en exclusiva a un lenguaje de programación o bien, puede utilizarse para varios. Consta de un editor de código, un compilador, un depurador y un constructor de interfaz gráfica GUI (29).

Visual Studio .Net

Visual Studio 2013 es un entorno de desarrollo solamente para los sistemas operativos *Windows*. Con este IDE es posible crear aplicaciones de escritorio, sitios y aplicaciones web, así como servicios web en cualquier entorno que soporte la plataforma .NET y aplicaciones que se intercomuniquen entre estaciones de trabajo, páginas web y dispositivos móviles. Soporta varios lenguajes de programación tales como C++, C#, J#, ASP.NET y *Visual Basic* .NET, aunque actualmente se han desarrollado las extensiones necesarias para muchos otros (30).

Algunas de las ventajas que este IDE ofrece radican en las mejoras a *Browser Link* permitiendo correr la aplicación creada en muchos navegadores diferentes (IE, Firefox, Chrome, emuladores para apps móviles) en la máquina de desarrollo. La nueva versión se enfoca en algunas mejoras en el editor, tales como Peek y CodeLens, las herramientas de diagnóstico para la interfaz gráfica, así como el desempeño en lo que toca a la respuesta al usuario de la aplicación y el consumo de energía, además actualizaciones mayores a Asp.NET, desarrollo web entre otras.

[1] En programación se refiere a una cadena de caracteres.

Con la incorporación de *plug-in* a Visual Studio 2013 se extienden con capacidades que mejoran y complementan las herramientas existentes. PTVS es un *plug-in* gratuito de código abierto que convierte a *Visual Studio* en un entorno de programación Python. Soporta CPython, IronPython, IPython, Django, y la computación en la nube con bibliotecas de cliente para *Windows*, *Linux* y *MacOS*.

Pycharm

PyCharm es un entorno de desarrollo para los desarrolladores de Python y Django, presenta un editor de códigos inteligente, que entiende los detalles específicos de Python y ofrece extraordinarios mejoradores de la productividad entre ellos, formateo automático de código, finalización de código, refactorizaciones, importación automática, navegación de código con un solo clic (31). Respaldadas por avanzadas rutinas de análisis de código, estas características hacen de PyCharm una poderosa herramienta tanto en las manos de los desarrolladores Python profesionales como en las de quienes recién comienzan a usar la tecnología.

1.6.6 Servidor distribuidor de mensajes

Los servidores distribuidores de mensajes son aquellos que se ocupan de establecer a través de mensajes, las conexiones entre aplicaciones de modo que perfeccione la comunicación entre estas.

RabbitMQ

RabbitMQ es un servidor de mensajería multiplataforma escrito en el lenguaje de programación Erlang, líder en la implementación completa del protocolo AMQP[2], caracterizado por el manejo de mensajes y colas, el enrutamiento, la exactitud y la seguridad (32). Este servidor no solo concentra las potencialidades de AMQP sino también las de OTP, uno de los sistemas más utilizado en el mundo por las empresas de telecomunicaciones. Es por ello que las librerías de OTP Erlang le proporcionan a RabbitMQ una base sólida para la construcción de software fiable. Se puede decir que RabbitMQ combina la robustez y escalabilidad de OTP con la flexibilidad del modelo de mensajería de AMQP.

Una sola cola en este servidor es capaz de almacenar una gran cantidad de mensajes que supera los 25 mil por segundo en correspondencia no solo con las propiedades y tamaño de los mensajes sino también

[2] Por sus siglas en inglés: Advanced Message Queue Protocol. Es un protocolo de estándar abierto que está orientado a resolver problemas de procesamiento de datos mediante la abstracción del problema de productores-consumidores, en donde los productores son los programas que envían mensajes y los consumidores aquellos que los reciben.

con la cantidad y carga de los clientes. Los mensajes se pueden escribir en el disco duro lo que permite que aumente la capacidad de almacenamiento de mensajes y el crecimiento de las colas sin presionar la memoria RAM (33).

Con la creación de un corredor virtual se alcanza una alta escalabilidad y disponibilidad que le permite a RabbitMQ replicar y compartir información con solo clusterizar varios nodos. El monitoreo y control de todos los aspectos del corredor es muy sencillo, ya que cuenta con una interfaz web para ello. Además, es extensible dada la variedad de plugins que ofrecen el equipo de desarrollo y los propios usuarios (34).

1.6.7 Gestores de bases de datos

Un sistema gestor de base de datos es básicamente un sistema computarizado para guardar registros; es decir, es un sistema computarizado cuya finalidad general es almacenar información y permitir a los usuarios recuperar y actualizar esa información con base en peticiones. La información en cuestión puede ser cualquier cosa que sea de importancia para el individuo u organización; en otras palabras, todo lo que sea necesario para auxiliarle en el proceso general de su administración (35).

PostgreSQL

PostgreSQL es un sistema de base de datos objeto-relacional. Cuenta con una arquitectura probada en cuanto a fiabilidad e integridad de la información. Es multiplataforma y totalmente compatible con ACID. Tiene soporte completo para claves foráneas, uniones, vistas, disparadores y procedimientos almacenados (en varios idiomas). Soporta el almacenamiento de grandes objetos binarios, como imágenes, sonidos o videos. Dispone de interfaces de programación nativas para C/C++, Java, .Net, Perl, Python, Ruby, ODBC, entre otros. Es altamente escalable tanto en la enorme cantidad de datos que puede manejar como en el número de usuarios concurrentes que puede soportar (36).

Por otra parte se entiende que la velocidad de respuesta que ofrece este gestor con bases de datos relativamente pequeñas puede parecer un poco deficiente, aunque esta misma velocidad la mantiene al gestionar bases de datos realmente grandes, cosa que resulta loable.

MySQL

MySQL es un sistema de gestión de bases de datos relacional, licenciado bajo la GPL o licencia pública general de la GNU. Su diseño multihilo le permite soportar una gran carga de forma muy eficiente. Este

gestor de bases de datos es, probablemente, el gestor más usado en el mundo del software libre, debido a su gran rapidez y facilidad de uso. Esta gran aceptación es debida, en parte, a que existen infinidad de librerías y otras herramientas que permiten su uso a través de gran cantidad de lenguajes de programación, además de su fácil instalación y configuración.

Es válido destacar que MySQL carece de algunas de las principales características de las bases de datos relacionales puesto que como surgió para dar respuestas rápidas, sus desarrolladores fueron implementando lo que precisaban, intentando hacerlo funcionar de forma óptima. Los *Triggers y Procedures* no están contenidos en la base de datos aunque se pretende incluir el uso de *procedures* almacenados en estas, pero no el de *triggers*, ya que estos reducen de forma significativa el rendimiento de la base de datos, incluso en aquellas consultas que no los activan.

1.7 Selección de la metodología y el marco de trabajo

Luego de analizar las características de distintas metodologías de desarrollo de software así como las herramientas y tecnologías, y atendiendo a las particularidades de la solución a desarrollar se concluye que:

Aunque SCRUM es una buena opción para desarrollar proyectos de poca envergadura y puede adaptarse fácilmente a cambios de requisitos si se hace necesario, su escasa documentación afecta el resultado final del producto y por consiguiente su futuro soporte y mantenimiento.

El principal problema que presenta FDD y que por consiguiente no se ajusta al número de desarrolladores con que cuenta el equipo de investigación, está representado por la necesidad de contar con miembros experimentados que puedan definir los roles y funciones de cada uno y que marquen el camino a seguir desde el principio con la elaboración del modelo global.

XP será la metodología rectora que guiará el desarrollo de la solución propuesta ya que está definida para equipos pequeños de desarrollo, tiene prioridad de satisfacer al cliente mediante un desarrollo iterativo e incremental, abierto a los cambios y caracterizado por un código simple, al mismo tiempo que aplica un conjunto de prácticas que harán la entrega del componente menos complicada y más satisfactoria tanto para los clientes como para el equipo de entrega.

En el caso de las herramientas analizadas para la modelación y el diseño del sistema, no existen muchas diferencias entre *Visual Paradigm* y *Rational Rose* en cuanto a las funcionalidades básicas necesarias, por

lo que se seleccionó *Visual Paradigm* 8 teniendo en cuenta que es la herramienta con la que más experiencia cuenta el equipo de trabajo por concepto de uso.

El IDE que se seleccionó para el desarrollo es Visual Studio 2013 Community pues posibilita la construcción de aplicaciones de forma rápida y sencilla, garantizando que sean seguras, confiables y administrables, características que son deseadas y fundamentales en el módulo a desarrollar. Además se descartó el uso del IDE *Pycharm* pues algunos componentes del sistema es necesario desarrollarlos en Visual Studio para aprovechar las ventajas del *framework* .Net y mantener el mismo ambiente de desarrollo para todos los componentes existentes.

Se utilizará el *framework* .Net en su versión 4.0 ya que ofrece una manera rápida y económica, que a la vez es segura y robusta, a la hora de desarrollar aplicaciones o soluciones como las denomina la misma plataforma.

La implementación se realizará mediante el lenguaje de programación de propósito general C# debido a que el sistema a desarrollar utiliza un lector de huellas dactilares, que hace uso de librerías en .NET para el manejo de las huellas dactilares que están escritas en dicho lenguaje (verifier_300_wrapper.dll, FingerprintImageQualityNew.dll, RabbitmqManagement.dll RabbitMQ.Client.dll). También se utilizará *Python* 2.7 pues el módulo a desarrollar se integrará con un sistema que esta implementado con esta tecnología auxiliado por la potente ayuda que proporciona el *framework* Django 1.6.

En cuanto al gestor de base de datos *PostgreSQL* 9.0 es la opción que más se ajusta a la solución ya que posee una gran escalabilidad, haciéndolo idóneo para ser usado en aplicaciones que realicen varias peticiones al día, de igual forma posee alta concurrencia admitiendo que se puedan realizar varias operaciones al mismo tiempo sobre una misma tabla. Además permite la gestión de usuarios, así como también los permisos asignados a cada uno de ellos.

1.8 Conclusiones parciales

El análisis de las características y funcionamiento de algunos de los controles de asistencia creados a nivel mundial, así como las soluciones existentes en el país, demostraron la necesidad de desarrollar un módulo de control de asistencia mediante identificación por huellas dactilares para el control de acceso del CISED que controle el flujo de personal y disminuya la suplantación de identidad. El análisis de *Visual Paradigm* como herramienta CASE para modelar UML, C# y Python como lenguajes de programación, Visual Studio

como IDE de desarrollo, para apoyar el ciclo de vida de un software guiado por un enfoque ágil con la metodología XP, demostró las potencialidades de todas estas tecnologías, en aras de lograr la creación del módulo propuesto en la presente investigación de acuerdo a sus necesidades específicas.

Capítulo II: Análisis y diseño de la propuesta de solución

2.1 Introducción

En el presente capítulo se detallan las principales características de la solución propuesta. Con el propósito de representar las principales relaciones entre los conceptos a tratar en la propuesta de solución, se define un modelo de dominio que describe el funcionamiento del módulo de control de asistencia. Se realiza el levantamiento de los requisitos funcionales y no funcionales y posteriormente estos son descritos en las historias de usuario. De acuerdo con la etapa de planificación del módulo se materializan el plan de entrega y el de iteraciones y por parte del diseño se realizan los diagramas de clases a partir de las tarjetas CRC y el resto de las tareas ingenieriles. Se describe la arquitectura del módulo, así como de los patrones de diseño utilizados, logrando de esta forma un entendimiento mayor del sistema para su posterior implementación.

2.2 Propuesta de solución

Se desarrollará un módulo de control de asistencia que permita controlar el flujo de personal que accede al centro. Para aumentar la confiabilidad de este proceso se utiliza una técnica biométrica que permite identificar unívocamente a la persona que ejecuta la acción de identificarse, la huella dactilar. Además se establecerán niveles de usuarios así como también se especificarán horarios a los que pueden acceder los usuarios siempre y cuando estos tengan los permisos necesarios para hacerlo. Se permitirá la gestión de horarios en los cuales cada usuario tendrá accesos. La clasificación de los horarios es variable en dependencia del rango de horarios que sean necesarios en la institución en la que se implante el módulo en desarrollo. También se gestionarán permisos de acceso a los usuarios del centro que se establecerán con el objetivo de que accedan solo aquellos que pertenezcan al centro y a determinada área de este. Por otra parte el acceso a los locales será gestionado para que solo puedan tener acceso aquellos usuarios que tengan permiso de acceso en el horario en el que pretendan hacer uso de un local.

El módulo demanda el uso de dos aplicaciones, la primera el manejador del escáner y la segunda el sistema de control de asistencia. El manejador del escáner se encarga de capturar las huellas del personal que intente acceder al centro, y posteriormente enviarla al sistema de control de asistencia. Este último cuando recibe una huella la envía al Sistema de Identificación Biométrica (SAIBIO) para su identificación, es

importante destacar que el SAIBIO es un sistema externo del Sistema de control de acceso al que se integrará el Módulo de control de asistencia. La comunicación entre los tres componentes antes mencionados se establecerá mediante el uso de un servidor de aplicaciones también externo en el que estará ejecutándose el RabbitMQ. Una vez identificada la identidad de la persona a quien pertenece la huella se comprueba si es usuario del centro y si tiene permiso para acceder al local en ese momento registrándose la entrada al mismo. En caso de que ya se haya registrado la entrada se le da la posibilidad al usuario de registrar su salida o si ha cometido un error al volver a registrar su entrada. Estos registros se almacenan en el sistema ya sean asistencias, como en los casos anteriores, o anomalías. Todas las anomalías que ocurran en el sistema dígase llegadas tardes, accesos denegados, que la persona no registre su salida, además de los errores en las dobles entradas, se almacenan como incidencias.

A partir de los registros de entrada y salida y de las incidencias se generan reportes que permiten llevar el control de asistencia sobre todas las personas que acceden al centro. De los registros generados se podrán obtener datos como los días trabajados, horarios incumplidos, accesos denegados, entre otros.

2.3 Modelo de dominio

Con el objetivo de lograr una mejor comprensión y entendimiento de lo planteado en la propuesta de solución quedan plasmadas en el modelo del dominio las relaciones existentes entre los conceptos a tratar en ella (Ver figura 3) así como la descripción de cada uno.

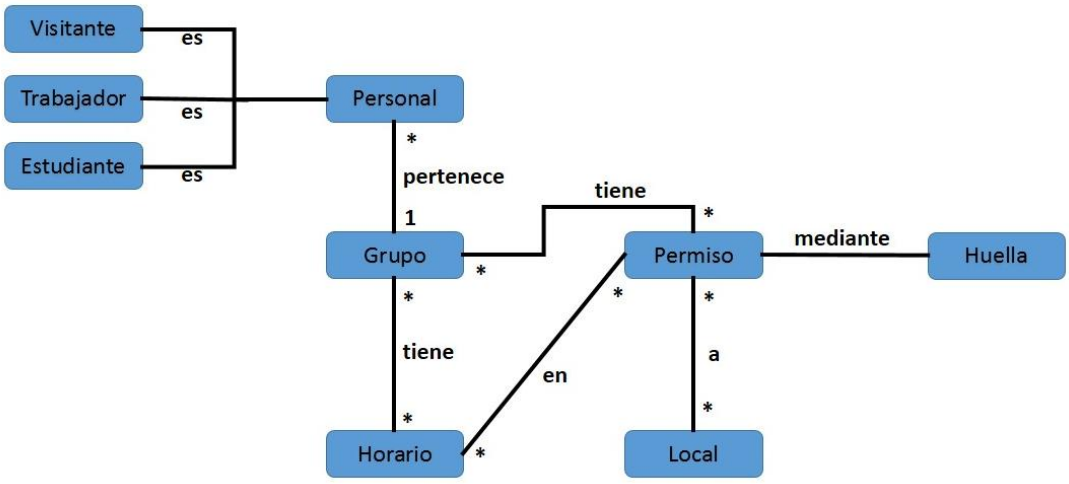

Figura 3. Modelo de dominio.

2.3.1 Descripción de los conceptos representados en el modelo de dominio

Personal: Son los usuarios que acceden al centro y se clasifican en visitante, trabajador y estudiante.

Grupo: Se encarga de gestionar los permisos de acceso a un local en un horario determinado para cada persona.

Permiso: Determina mediante la huella dactilar el acceso al centro.

Horario: Establece en qué momento del día se puede acceder al centro.

Local: Son los diferentes puntos de acceso que existen en el centro.

2.4 Requisitos del sistema

El levantamiento de los requisitos del sistema es uno de los procedimientos más importantes en el proceso de desarrollo de cualquier *software* ya que es donde se identifica lo que el cliente realmente necesita. Este procedimiento es conveniente para el equipo de desarrollo ya que le permite trabajar en base a lo acordado para obtener un producto con la calidad requerida sin perder tiempo en funcionalidades no necesarias.

2.4.1 Requisitos funcionales

Los requisitos funcionales (RF) son aquellas características con las que el sistema debe cumplir, a continuación se listan las necesarias para la implementación del módulo.

RF 1. Capturar imagen de huella dactilar con lector de huellas.

RF 2. Mostrar imagen de huella dactilar.

RF 3. Identificar huella dactilar.

RF 4. Mostrar resultado de la identificación de huella dactilar.

RF 5. Permitir o denegar acceso al centro.

RF 6. Generar *logs* para control de asistencia.

RF 7. Generar reportes.

- **RF 7.1.** Reporte de usuarios que han accedido al centro por días.
- **RF 7.2.** Reporte de usuarios que han incumplido con su horario laboral.
- **RF 7.3.** Reporte de horas trabajadas para cada usuario.
- **RF 7.4.** Reporte de días trabajados por cada usuario.
- **RF 7.5.** Reporte de días trabajados por todas las personas en un rango de fechas.
- **RF 7.6.** Reporte de días trabajados por una persona en un rango de fechas.

RF 7.7. Reporte de días trabajados por todas las personas en un horario específico.

RF 7.8. Reporte de días trabajados por una persona en un horario específico.

RF 8. Exportar los reportes generados a Excel.

RF 9. Gestionar Grupos.

RF 9.1. Insertar Grupo.

RF 9.2. Modificar Grupo.

RF 9.3. Eliminar Grupo.

RF 9.4. Listar Grupo.

RF 10. Gestionar Horarios.

RF 10.1. Insertar Horario.

RF 10.2. Modificar Horario.

RF 10.3. Eliminar Horario.

RF 10.4. Listar Horario.

RF 11. Gestionar Locales.

RF 11.1. Insertar Local.

RF 11.2. Modificar Local.

RF 11.3. Eliminar Local.

RF 11.4. Listar Local.

RF 12. Gestionar Permisos.

RF 12.1. Insertar Permiso.

RF 12.2. Modificar Permiso.

RF 12.3. Eliminar Permiso.

RF 12.4. Listar Permiso.

RF 13. Gestionar la conexión al sistema distribuidor de mensajes.

RF 13.1. Insertar la conexión al sistema.

RF 13.2. Modificar la conexión.

RF 13.3. Eliminar la conexión.

RF 13.4. Listar conexión.

2.4.2 Requisitos no funcionales

Los requisitos no funcionales (RNF) son aquellas propiedades que de una forma u otra permiten que el sistema sea atractivo, fácil de usar, confiable. A continuación se describen los requisitos no funcionales con los que debe contar el módulo a desarrollar.

Software:

PC Cliente:

- ✓ El módulo será usado bajo el sistema operativo Windows.
- ✓ Framework de desarrollo: .NET 4.5.
- ✓ Librería para el manejador del escáner: verifier_300_wrapper.

PC Servidor

- ✓ El módulo será usado bajo los sistemas operativos Windows y GNU/Linux.
- ✓ Servidor Apache 2.22.
- ✓ Framework de desarrollo: Django 1.6.

Hardware:

La PC Cliente y el Servidor deben contar mínimo con las siguientes características:

- ✓ Procesador Intel Pentium 4 o superior.
- ✓ CPU 2.1 3GHZ o superior.
- ✓ 512MB de Memoria RAM o superior.
- ✓ En la PC Cliente se debe tener un escáner Verifier 300 USB para la captura de la huella dactilar.

Seguridad:

- ✓ Solo podrán acceder al módulo biométrico de control de asistencia los administradores y técnicos del sistema.
- ✓ Se deben definir distintos niveles de usuarios de acuerdo a los permisos que estos tengan en el sistema.
- ✓ La comunicación entre la PC Cliente y el Servidor de aplicaciones web se realiza de forma segura mediante HTTPS.

Restricciones en el Diseño y en la Implementación

- ✓ La implementación del módulo debe ser desarrollada en los lenguajes C# y Python, puesto que el sistema al que se debe integrar está codificado en dicho lenguaje, además de que las librerías existentes para el manejador del escáner están escritas en C#.
- ✓ La solución debe ser un módulo desarrollado en el *framework* Django 1.6 para que este pueda aprovechar las funcionalidades del sistema de control de acceso a entornos seguros desarrollado por el CISED.

2.5 Historias de usuario

Las historias de usuario (HU) es uno de los artefactos más importantes generados por la metodología XP donde el cliente describe las características que debe presentar el producto final. Estas características están asociadas a los RF y a los RNF y deben ser independientes unas de otras así como verificables y estimables entre otras condiciones.

A continuación se describen tres historias de usuario que representan gran valor en la solución propuesta.

Historia de usuario	
Número: HU-03	**Usuario:** Rosana Castro García.
Nombre: Identificar huella dactilar.	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Alto
Puntos estimados: 2	**Iteración asignada:** 1
Programador responsable: Emmanuel Gómez Soler.	
Descripción: El módulo debe recibir la imagen de la huella dactilar que le envía el escáner y enviarla al sistema de identificación biométrico para que realice la identificación de la misma.	
Observaciones:	

Tabla 2. Historia de usuario: HU-03. Identificar huella dactilar.

Historia de usuario	
Número: HU-05	**Usuario:** Emmanuel Gómez Soler.
Nombre: Permitir o denegar acceso al centro.	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Alto

Puntos estimados: 2	**Iteración asignada:** 1
Programador responsable: Rosana Castro García.	
Descripción: El módulo debe comprobar si la persona identificada tiene acceso al centro y al local en el momento en que se está identificando.	
Observaciones:	

Tabla 3. Historia de usuario: HU-05. Permitir o denegar acceso al centro.

Historia de usuario	
Número: HU-08	**Usuario:** Emmanuel Gómez Soler.
Nombre: Exportar los reportes generados a Excel.	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Medio
Puntos estimados: 0.5	**Iteración asignada:** 2
Programador responsable: Rosana Castro García.	
Descripción: El módulo debe exportar los reportes generados a tablas en formato Excel.	
Observaciones:	

Tabla 4. Historia de usuario: HU-08. Exportar los reportes generados a Excel.

2.6 Planificación

En esta etapa el cliente y el equipo de desarrollo especifican en conjunto el plan de entrega y el de iteraciones, puntualizando en ellos la prioridad de las HU definidas y el esfuerzo que requiere la implementación de cada una de ellas entre otros factores, de modo que el proceso quede debidamente organizado.

2.6.1 Plan de entrega

Una vez establecidas por el cliente las prioridades de las HU se obtiene un cronograma con las versiones resultantes en cada iteración. En la solución propuesta se codificarán 20 HU en un estimado de 2 iteraciones de acuerdo con el plan de entrega para una duración total del proyecto de 16 semanas. En la primera iteración se entregarán los resultados obtenidos de la implementación de las principales funcionalidades y más complejas de la propuesta de solución que se encuentran recogidas en los RF del RF 1 al RF 7.5. En

la segunda iteración se entregarán los resultados obtenidos de la implementación de los RF del RF 7.6 al RF 13.

Entregable	Iteración	Fin de iteración
RF 1 – RF 7.5	1	Febrero 2015
RF 7.6 – RF 13	2	Abril 2015

Tabla 5. Plan de entrega.

2.6.2 Plan de iteraciones

En el plan de iteraciones queda reflejado el esfuerzo en semanas que se debe hacer por cada HU definida, y una vez realizado los desarrolladores pueden trabajarlas de manera organizada agilizando todo el proceso. A continuación se muestra la estimación en semanas del esfuerzo realizado por cada HU de la propuesta de solución.

Iteración	No. HU	Historia de usuario	Estimación (Semanas)
1	HU-01	Capturar imagen de huella dactilar con lector de huellas.	2
	HU-02	Mostrar imagen de huella dactilar.	0.5
	HU-03	Identificar huella dactilar.	2
	HU-04	Mostrar resultado de la identificación de huella dactilar.	0.5
	HU-05	Permitir o denegar acceso al centro.	2
	HU-06	Generar *logs* para control de asistencia.	2
	HU-7.1	Reporte de usuarios que han accedido al centro por días.	0.5
	HU-7.2	Reporte de usuarios que han incumplido con su horario laboral.	0.5
	HU-7.3	Reporte de horas trabajadas para cada usuario.	0.5
	HU-7.4	Reporte de días trabajados por cada usuario.	0.5

	HU-7.5	Reporte de días trabajados por todas las personas en un rango de fechas.	0.5
2	HU-7.6	Reporte de días trabajados por una persona en un rango de fechas.	0.5
	HU-7.7	Reporte de días trabajados por todas las personas en un horario específico.	0.5
	HU-7.8	Reporte de días trabajados por una persona en un horario específico.	0.5
	HU-08	Exportar los reportes generados a Excel.	0.5
	HU-9	Gestionar Grupos.	0.5
	HU-10	Gestionar Horarios.	0.5
	HU-11	Gestionar Locales.	0.5
	HU-12	Gestionar Permisos.	0.5
	HU-13	Gestionar la conexión al sistema distribuidor de mensajes.	0.5

Tabla 6. Plan de iteraciones.

2.7 Diseño

En el diseño se adquiere la comprensión necesaria del funcionamiento del módulo con la definición de una estructura lo más sencilla posible que responda a la satisfacción de los RF y RNF. Según XP, la metodología rectora de esta investigación, se define la arquitectura, interfaces, y otras características del módulo que se deben tener en cuenta en la implementación.

2.7.1 Tarjetas CRC

Las tarjetas CRC son una técnica utilizada en XP para diseñar la solución informática según el paradigma orientado a objetos. Sus siglas se refieren a Clases, Responsabilidades y Colaboradores que son precisamente los elementos fundamentales de estas tarjetas. Una clase es cualquier persona, evento, concepto, pantalla o reporte y las responsabilidades son las funcionalidades que realiza y sus atributos, los colaboradores son las demás clases con las que interactúa con el fin de cumplir sus responsabilidades.

A continuación se muestra la tarjeta CRC de la clase PyRabbitmq con sus correspondientes funcionalidades.

Clase: PyRabbitmq	
Responsabilidades.	**Colaboradores**.
conexion	
channel	
externalRequestsQueue	
externalReplyQueue	
host	
vhost	
usser	
password	
Conect	• pika
Disconect	• pika
SendRequestIdentifyRequest	• pika
ReciveFingerprintSync	• pika
ReciveScoreReplySync	• pika • IdentifyScoreReplay

Tabla 7. Tarjeta CRC PyRabbitmq.

2.7.2 Modelo de datos

Para el modelado de los datos se presenta el modelo de entidad-relación en el que se exponen las interrelaciones entre las tablas de la base de datos y para cada una de ellas sus atributos.

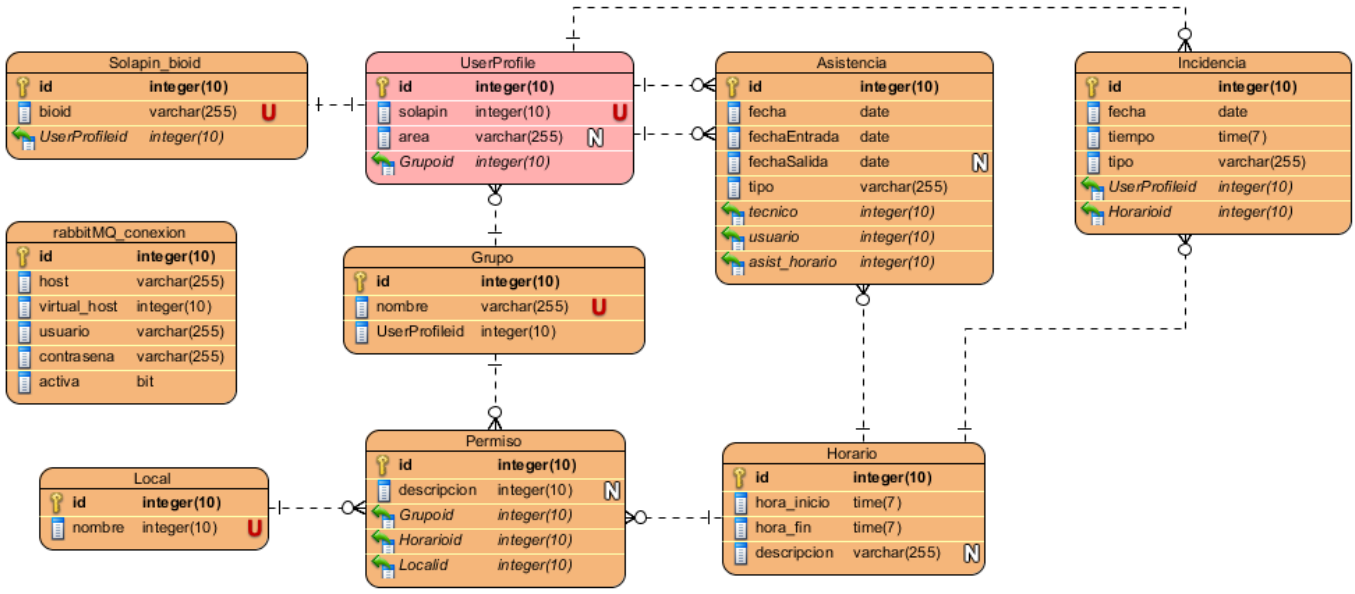

Figura 4. Modelo de datos.

Descripción de las entidades identificadas en el modelo de datos.

Entidades	Descripción
solapin_bioid	Almacena la información de la huella dactilar de las personas a identificar.
UserProfile	Almacena los datos de las personas que tendrán acceso al centro.
Asistencia	Contiene los registros de entrada y salida de cada persona para el control de asistencia.
Incidencia	Contiene los eventos inusuales de cada persona, dígase llegadas tardes, salidas anticipadas, errores en la salida.
Grupo	Clasifica a las personas que pueden acceder al centro.
Permisos	Define a que local puede acceder una persona en un horario determinado.

Local	Guarda la relación de los locales existentes en el centro.
Horario	Guarda la relación de los horarios establecidos en el control de asistencia.
rabbitMQ_conexion	Guarda los parámetros de conexión al servidor distribuidor de mensajes RabbitMQ.

Tabla 8. Descripción de las entidades del modelo de datos.

2.7.3 Patrones de diseño

Los patrones GRASP (General Responsibility Assignment Software Patterns) describen los principios fundamentales de la asignación de responsabilidades a objetos (37). Los patrones de este tipo usados son:

- ✓ **Patrón Controlador:** Es un intermediario entre la capa de presentación y el núcleo de las clases donde reside la lógica del sistema, que coordina la actividad de otros objetos. Ejemplo de este patrón en el módulo se implementó en la clase controladora **views** que procesa todas las peticiones de los usuarios y genera su correspondiente plantilla.

- ✓ **Patrón Experto:** Se asigna una responsabilidad a la clase que tiene la información necesaria para cumplirla, alentando con ello definiciones de clases “sencillas” y más cohesivas que son más fáciles de comprender y de mantener. Así se brinda soporte a una alta cohesión. Se conserva el encapsulamiento, ya que los objetos se valen de su propia información para hacer lo que se les pide. Esto soporta también un bajo acoplamiento, lo que favorece al hecho de tener sistemas más robustos y de fácil mantenimiento. Este patrón se utiliza en las clases IdentifyScoreReplay y trazas.

- ✓ **Patrón Bajo Acoplamiento:** El acoplamiento es una medida de la fuerza con que una clase está conectada a otras clases, con que las conoce y con que recurre a ellas. Una clase con bajo o débil acoplamiento no depende de muchas otras. El empleo del patrón Experto favorece al bajo acoplamiento entre las clases del módulo.

- ✓ **Patrón Alta Cohesión:** En la perspectiva del diseño orientado a objetos, la cohesión (o más exactamente, la cohesión funcional) es una medida de cuán relacionadas y enfocadas están las

responsabilidades de una clase. Una alta cohesión caracteriza a las clases con responsabilidades estrechamente relacionadas que no realicen un trabajo enorme.

Los patrones GOF (Gang of Four) describen las formas en las que pueden ser organizados los objetos para trabajar unos con otros, formando estructuras de mayor complejidad. Los patrones de este tipo usados son:

- ✓ **Observador**: Es un patrón de comportamiento a nivel de objetos que favorece el bajo acoplamiento. Define una dependencia de uno a muchos entre objetos, para de esta forma permitir que el cambio de estado de un objeto sea notificado y actualizado automáticamente en todos los objetos que dependen de él. Ejemplo de este se evidencia en la funcionalidad ReciveFingerprint de la clase PyRabbitmq que se encarga de chequear periódicamente si hay una nueva huella para identificar.

2.7.4 Arquitectura

La definición oficial de Arquitectura de Software, también denominada como Arquitectura Lógica es la que brinda el documento de IEEE Std 1471 -2000, adoptada también por Microsoft, la misma plantea que: “La Arquitectura de Software es la organización fundamental de un sistema encarnada en sus componentes, las relaciones entre ellos y el ambiente y los principios que orientan su diseño y evolución” (38).

La arquitectura de un sistema software puede basarse en un modelo o estilo arquitectónico particular. Un estilo arquitectónico es un patrón de organización de un sistema tal como una organización cliente servidor o una arquitectura por capas. Sin embargo las arquitecturas de la mayoría de los sistemas grandes no utilizan un único estilo sino combinaciones de estos. Pueden diseñarse diferentes partes del sistema utilizando distintos estilos arquitectónicos. Para descomponer las unidades del sistema estructural en módulos hay que decidir la estrategia para descomponer subsistemas o módulos.

Para lograr la comunicación entre todos los componentes a desarrollar se utilizará el estilo arquitectónico cliente-servidor (Ver Figura 5). Este modelo de sistema se organiza como un conjunto de servicios y servidores asociados más unos clientes que acceden y usan los servicios (12). Al ser una arquitectura distribuida se puede hacer un uso efectivo de los sistemas en red con muchos procesadores distribuidos. Es fácil añadir un nuevo servidor e integrarlo con el resto del sistema o actualizar los servidores de forma transparente sin afectar el resto del sistema.

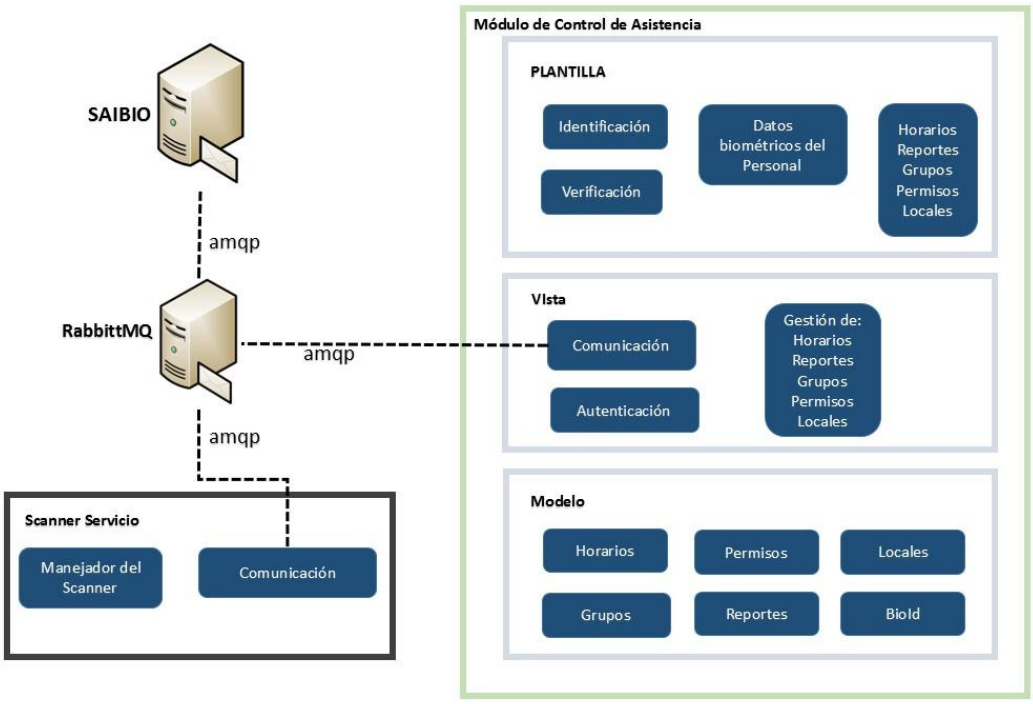

Figura 5. Arquitectura del módulo.

2.7.5 Patrón arquitectónico

Para desarrollar el módulo control de asistencia se utilizará el patrón Modelo-Vista-Plantilla (MVT) propuesto en el *framework* de desarrollo web Django.

En el patrón MVT:

- ✓ *M* significa "Model" (Modelo), la capa de acceso a la base de datos. Esta capa contiene toda la información sobre los datos: cómo acceder a estos, cómo validarlos, cuál es el comportamiento que tiene, y las relaciones entre los datos.
- ✓ *T* significa "Template" (Plantilla), la capa de presentación. Esta capa contiene las decisiones relacionadas a la presentación: como algunas cosas son mostradas sobre una página web u otro tipo de documento.

- ✓ *V* significa "View" (Vista), la capa de la lógica de negocios. Esta capa contiene la lógica que accede al modelo y la delega a la plantilla apropiada: se puede pensar en esto como un puente entre el modelo y las plantillas.

2.8 Conclusiones parciales

La definición del modelo de dominio y la explicación detallada de la propuesta de solución, permitieron identificar con mayor facilidad los requisitos funcionales que debe satisfacer el módulo de control de asistencia, propiciando así la comprensión de su funcionamiento. Las historias de usuario especificadas para cada uno de estos requisitos, detallando entre otros elementos el tiempo que requieren para su codificación, favorecieron la estimación del plan de entregas del producto delimitando el propósito y duración de cada iteración. Con el uso de la arquitectura cliente-servidor especificada, el estilo arquitectónico Modelo-Vista-Plantilla, y algunos de los patrones GRASP y GOF, se logró organizar la vista lógica de la solución, de manera que las clases identificadas y sus relaciones sean las bases para su implementación.

Capítulo III: Implementación y prueba de la propuesta de solución

3.1 Introducción

En el presente capítulo se describe la implementación del módulo de control de asistencia y se cumple con los requisitos obtenidos al inicio de la investigación. Para lograr esto se generan los diagramas de componentes y de despliegue, donde se observan las dependencias lógicas entre los elementos del módulo y los nodos necesarios para el despliegue del mismo respectivamente. Además, se muestran las interfaces de usuario y se da una pequeña descripción de algunas de ellas para un mayor entendimiento a la hora de navegar por esta. Finalmente se muestran las pruebas realizadas para validar el correcto funcionamiento del módulo de control de asistencia.

3.2 Tareas ingenieriles

Las tareas de ingeniería son otro artefacto de la metodología XP, estas se establecieron indicando por cada una de las iteraciones la forma en que se desarrollaría la aplicación, en este caso con mayor nivel de detalle que en las HU. Por cada una de ellas se establece un programador responsable pero estas se implementan en parejas de programadores. A continuación, se observan las tareas de ingeniería correspondientes a la iteración número 1.

Iteración 1	
Historia de usuario	**Tareas**
Capturar imagen de huella dactilar con lector de huellas.	▪ Encender lector de huellas dactilares. ▪ Capturar imagen de huella dactilar. ▪ Enviar imagen de huella dactilar.
Mostrar imagen de huella dactilar.	▪ Recibir y guardar imagen de huella dactilar. ▪ Mostrar imagen.
Identificar huella dactilar.	▪ Enviar la huella dactilar al sistema SAIBIO para su identificación. ▪ Esperar respuesta del sistema.
Mostrar resultado de la identificación de huella dactilar.	▪ Recibir respuesta de la identificación del sistema.

	▪ Mostrar los resultados de la identificación.
Permitir o denegar acceso al centro.	▪ Obtener grupo al que pertenece el usuario. ▪ Verificar que el grupo obtenido tiene permiso de acceso al local al que pretende entrar la persona en el horario especificado.
Generar *logs* para control de asistencia.	▪ Registrar la entrada o salida de la persona identificada. ▪ Registrar la incidencia en caso de que la persona incumpla con su horario de trabajo.
Generar reportes.	▪ Procesar los registros de asistencia e incidencias para obtener información sobre las personas.

Tabla 9. Tareas de ingeniería. Iteración 1.

3.3 Estándares de codificación

XP enfatiza la comunicación de los programadores a través del código, con lo cual es indispensable que se sigan ciertos estándares de programación con el objetivo de lograr un código legible y fácil de cambiar en caso necesario. Estos pueden ser definidos por el equipo de desarrollo, por la organización o simplemente usando otros estándares reconocidos para los lenguajes de programación utilizados. Para la implementación del módulo se definió el estándar de codificación que se describe a continuación.

- ✓ **Pascal:** La primera letra en el identificador y la primera letra de cada subsiguiente palabra concatenada se capitalizan. Por ejemplo: SendMessage.
- ✓ **Camel:** La primera letra en el identificador se pone en minúscula y la primera letra de cada subsiguiente palabra concatenada en mayúscula. Por ejemplo: requestMessage.

La convención Pascal se empleó en los identificadores de las clases, métodos y nombres de ficheros. Mientras que la convención Camel es el estilo de los identificadores de las variables, atributos y parámetros.

3.4 Diagrama de componentes

Durante la codificación de la solución propuesta se pueden distinguir diferentes elementos de implementación que analizados como una sola unidad, constituyen el módulo de control de asistencia mediante identificación por huellas dactilares. Un componente es la parte modular de un sistema, desplegable y reemplazable que encapsula la implementación, un conjunto de interfaces y proporciona la realización de los mismos, típicamente contiene clases y puede ser implementado por uno o más artefactos (ficheros ejecutables, binarios, etc.). Son las piezas reutilizables de alto nivel a partir de las cuales se pueden construir los sistemas.

Los diagramas de componentes modelan la vista estática del software, se representan como un grafo de componentes unidos por medio de relaciones de dependencia, pudiendo mostrarse las interfaces que estos soporten. A continuación se expone el diagrama definido para la solución propuesta (Figura 6).

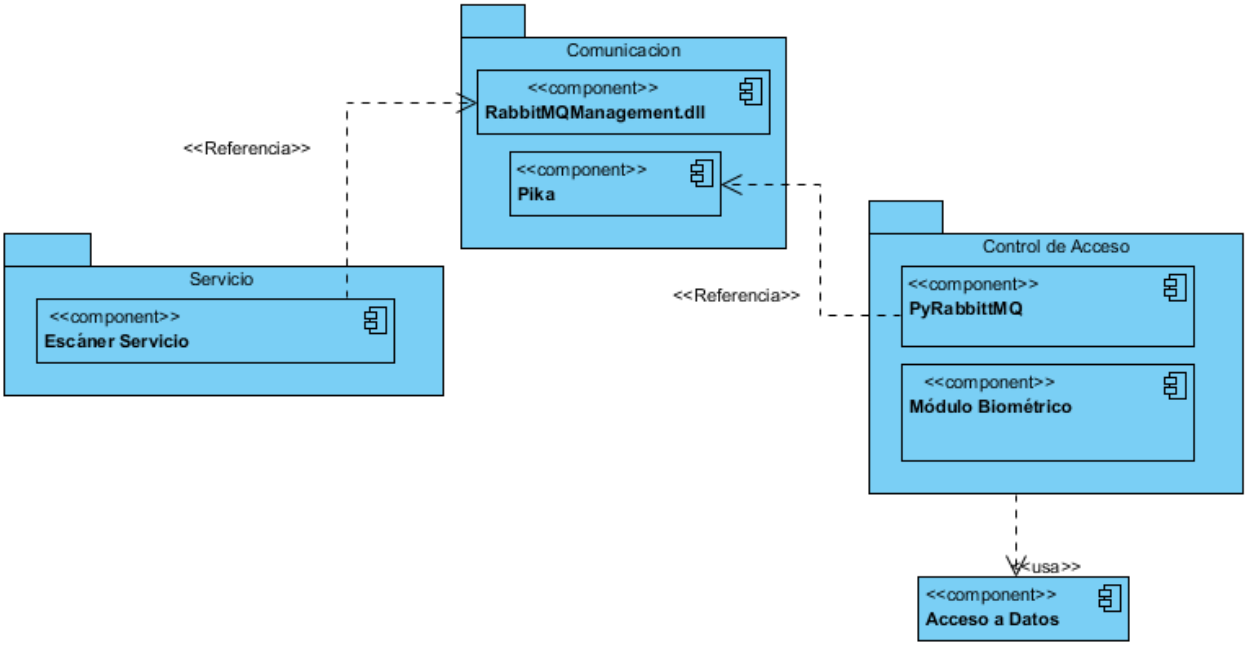

Figura 6. Diagrama de componente.

3.4.1 Descripción del diagrama de componente

La implementación del módulo de control de asistencia requiere de dos componentes importantes, el componente que se encarga de manejar el escáner (Escáner Servicio) y el componente que se ocupa de la gestión e identificación biométrica (Módulo Biométrico).

Se establece la comunicación entre ambos componentes mediante el uso del servidor RabbitMQ haciendo referencia a la librería RabbitMQManagement.dll. El componente de acceso a datos contiene los datos del sistema, y es a través de este que se puede acceder a ellos y modificarlos.

3.5 Diagrama de despliegue

El modelo de despliegue es uno de los artefactos obtenidos durante el proceso de desarrollo del *software*, con el objetivo de capturar los elementos de configuración del procesamiento, las conexiones entre esos elementos y visualizar su distribución en los nodos físicos. De manera general este tipo de modelo está compuesto por nodos o elementos de procesamiento con al menos un procesador, memoria, y posiblemente otros dispositivos; estos últimos caracterizados por ser nodos estereotipados sin capacidad de procesamiento en el nivel de abstracción que se modela y finalmente por conectores que expresan el tipo de conector o protocolo utilizado entre el resto de los elementos del modelo. A continuación se describe el diagrama de despliegue del módulo (Ver Figura 7).

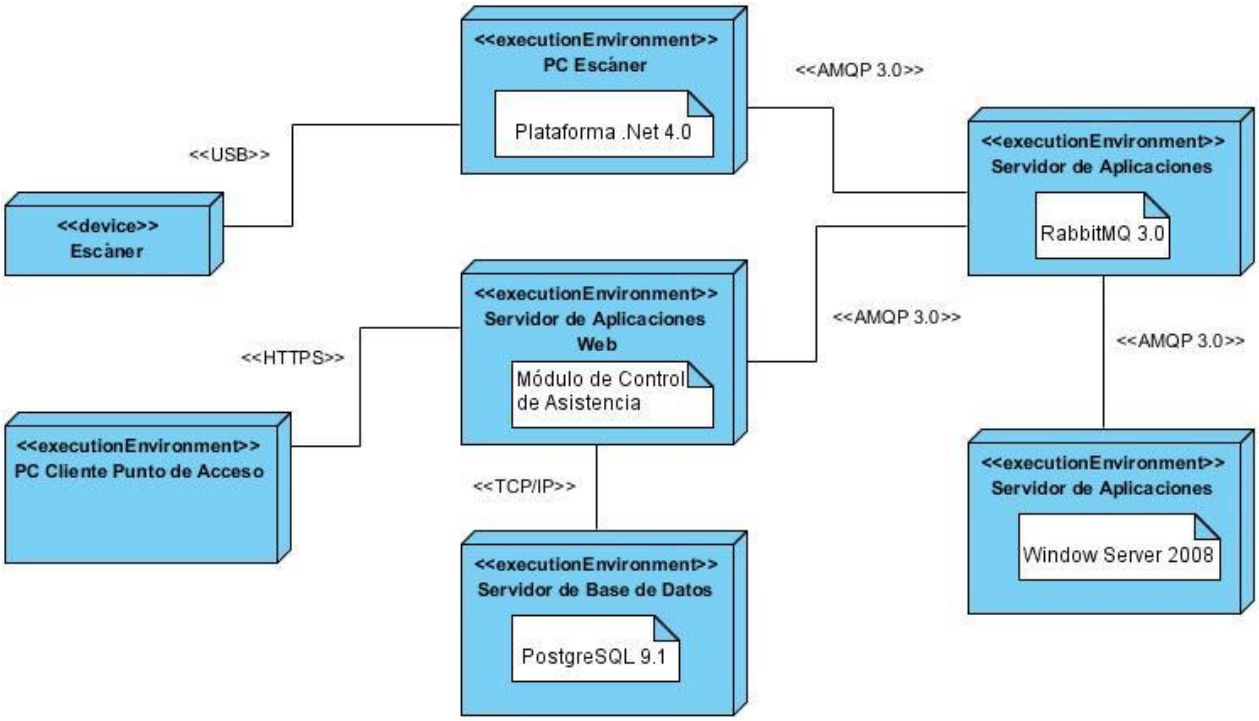

Figura 7. Diagrama de despliegue.

3.5.1 Descripción del diagrama de despliegue

Para el despliegue del módulo se requerirán dos PC de tipo Cliente, una de ellas controlará el servicio del escáner y estará situada en cada uno de los puntos de acceso de la instalación asociándosele el lector de huellas dactilares a través de una conexión por USB. La otra se encargará de manejar la administración del sistema y para la interacción con el módulo se utilizará el protocolo HTTPS garantizando una comunicación segura entre el cliente y el servidor web mediante el uso de SSL cifrándose la información compartida entre ambos. Es importante destacar que estas labores pueden ser delegadas en una misma PC Cliente.

La comunicación entre la PC-Escáner y el servidor de aplicaciones web donde se encuentra el módulo se realizará través del servidor de aplicaciones RabbitMQ cuyas peticiones se gestionarán mediante el uso del protocolo AMQP 3.0. También se utiliza este protocolo para obtener respuestas de identificación desde el servidor de aplicaciones donde se encuentra ejecutándose el SAIBIO.

Para gestionar los datos de todas las personas en el módulo de control de asistencia que corre en el servidor de aplicaciones web se hace necesario consultar, mediante el protocolo TCP/IP, la base de datos del sistema.

3.6 Interfaces de usuario

El módulo en desarrollo debe integrarse con el sistema de control de acceso desarrollado por el CISED por lo que las interfaces de este deben ajustarse a las del propio sistema. Dicho módulo cuenta con dos interfaces principales; la primera hace referencia a la identificación de los usuarios y la segunda a la gestión y configuración del módulo. La figura 8 muestra la interfaz principal de identificación donde una vez seleccionado el local al cual se va a acceder se tiene la posibilidad de identificar al usuario.

Figura 8. Interfaz principal Módulo de control de asistencia.

También se puede observar en la figura 9 los datos que se muestran luego de realizar la identificación de una persona, en este caso de un usuario del sistema que tiene permiso de acceso pero ha llegado tarde.

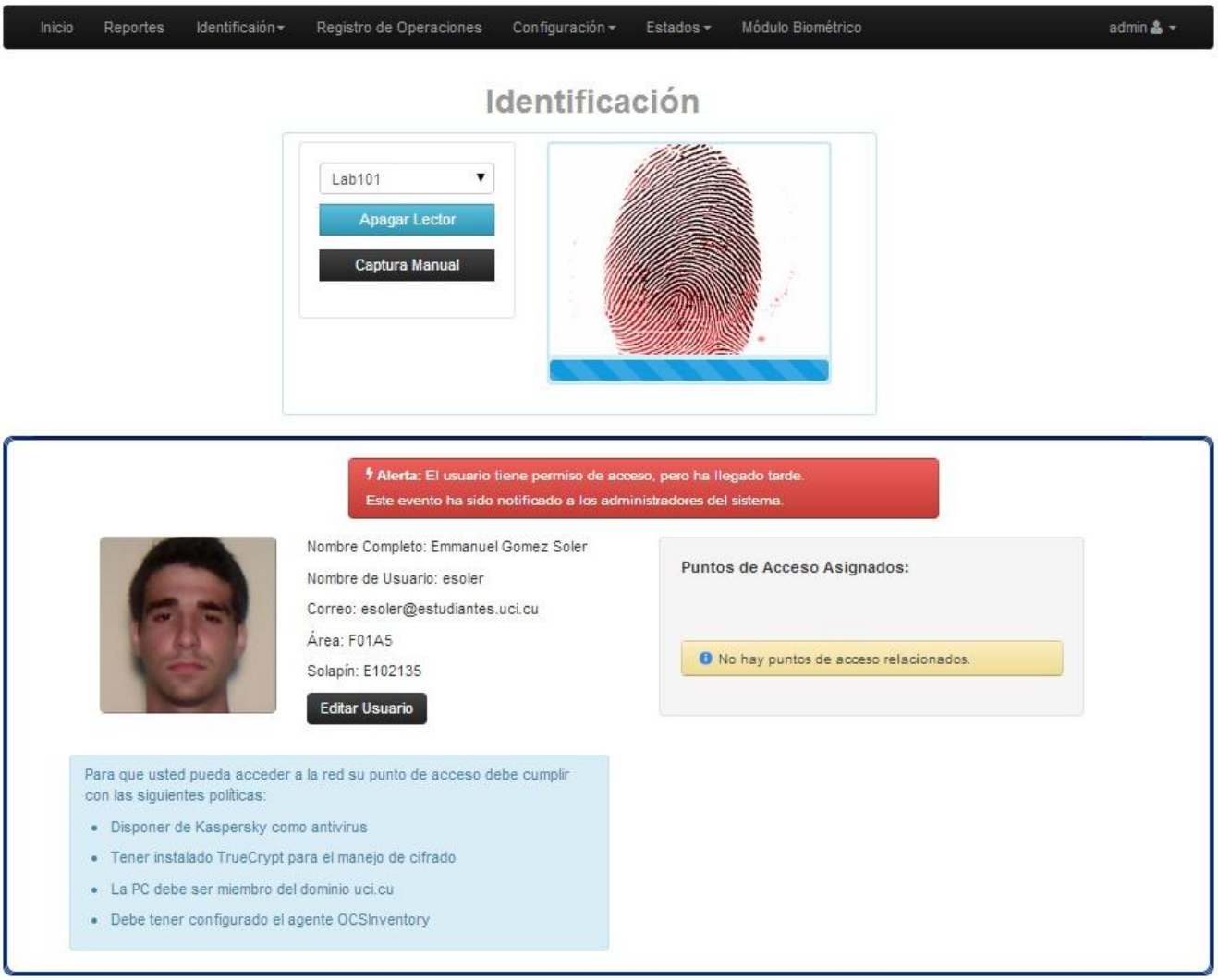

Figura 9. Usuario identificado.

En la imagen 10 se muestra la interfaz de configuración donde son manejados los datos referentes a los reportes generados así como otros aspectos a gestionar.

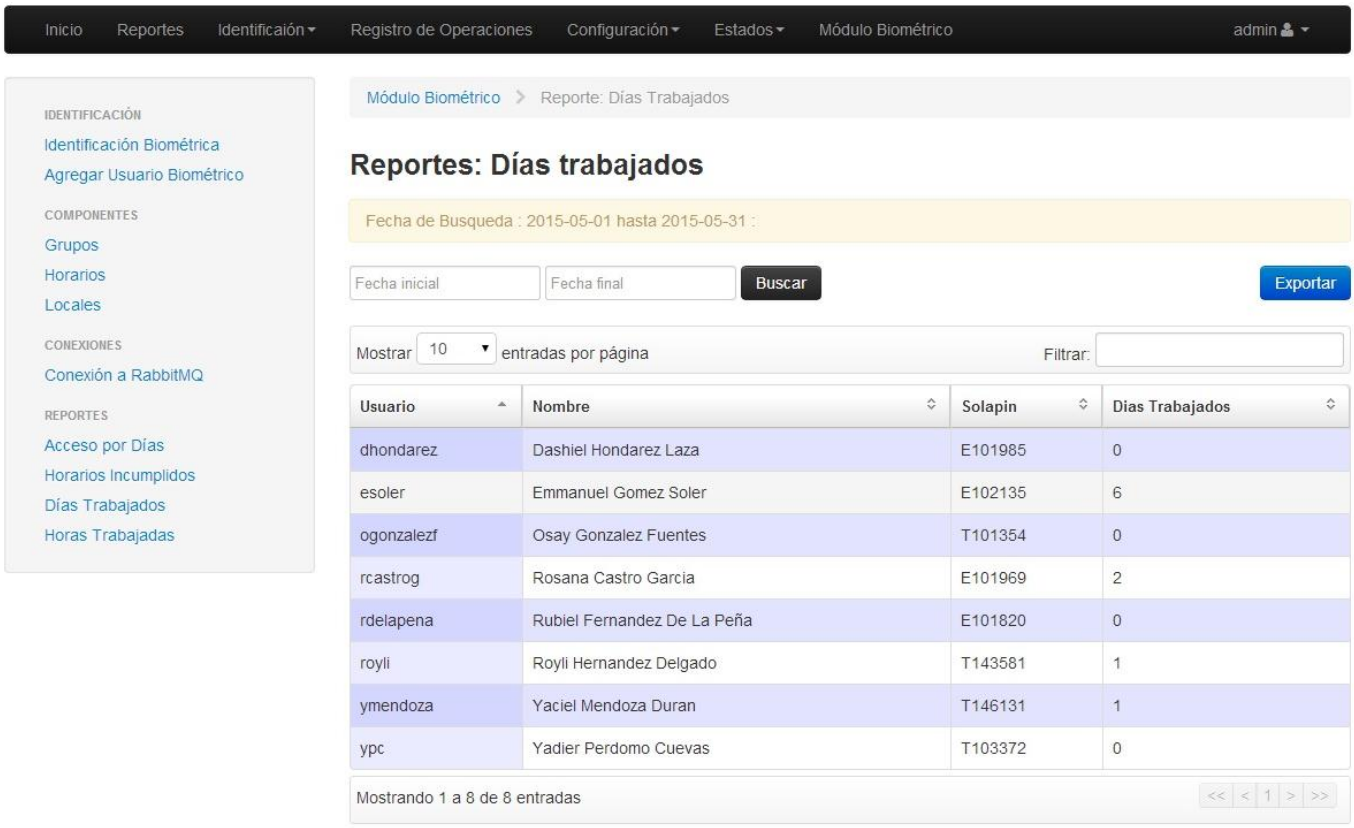

Figura 10. Interfaz de administración.

3.7 Pruebas

Las pruebas son una actividad en la cual un sistema o componente es ejecutado bajo unas condiciones o requerimientos especificados, los resultados son observados y registrados, y es hecha una evaluación de algún aspecto determinado. La prueba del software es un elemento crítico para la garantía de su calidad y representa una revisión final de las especificaciones del diseño y de la codificación.

3.7.1 Pruebas unitarias

La creación de pruebas unitarias antes de que comience la codificación es un elemento clave del enfoque de XP. Las pruebas unitarias que se crean deben implementarse con el uso de una estructura que permita automatizarlas, de modo que puedan ejecutarse en reiteradas ocasiones y con facilidad. Esto estimula una estrategia de pruebas de regresión, siempre que se modifique el código. El programador es el encargado de escribir las pruebas unitarias y producir el código del sistema. Las pruebas de unidad son las que van

enfocadas a los elementos más pequeños del *software*. Son aplicables a funcionalidades para verificar que los flujos de control y de datos están cubiertos y funcionando como se espera.

Resultados de las pruebas unitarias.

Para la realización de estas pruebas se utilizó un *Add-in* de Visual Studio llamado ReScharper (R#) que permite detectar errores y brinda soluciones instantáneas para corregirlos. El código de prueba a las funcionalidades ReciveFingerprintSync y ReciveScoreReplySync donde la primera lee la huella capturada y la envía a identificar y la segunda recibe la respuesta enviada de la identificación, se muestra en las figuras 11 y 12 respectivamente.

```
6    def test_ReciveFingerprintSync(self):
7
8        getFingerprint=PyRabbittmq()
9        getFingerprint.Conect()
10
11       requestId=getFingerprint.ReciveFingerprintSync('Lab101'+'__FpImg')
12       if requestId=='Esperando por la huella':
13               return True
14       elif requestId=='Error de conexion':
15               return True
16       elif isinstance(requestId,str):
17               return True
18       else:
19           self.fail("Error")
```

Figura 11. Unit Test 1: Funcionalidad ReciveFingerprintSync.

```
21     def test_ReciveScoreReplySync(self):
22         getFingerprint=PyRabbittmq()
23         getFingerprint.Conect()
24         resp=getFingerprint.ReciveScoreReplySync('782395')
25         if resp=='Esperando por la huella':
26                 return True
27         elif resp=='Error de conexion':
28                 return True
29         elif isinstance(resp,IdentifyScoreReplay):
30                 return True
31         else:
32             self.fail("Error")
```

Figura 12. Unit Test 2: Funcionalidad ReciveScoreReplySync.

A continuación se muestran los resultados obtenidos de aplicar las pruebas (Ver Figura 13) con esta herramienta a las funcionalidades antes mencionadas. Como resultado de la aplicación de estas pruebas se obtuvieron 12 no conformidades que fueron corregidas en el momento en que se detectaron.

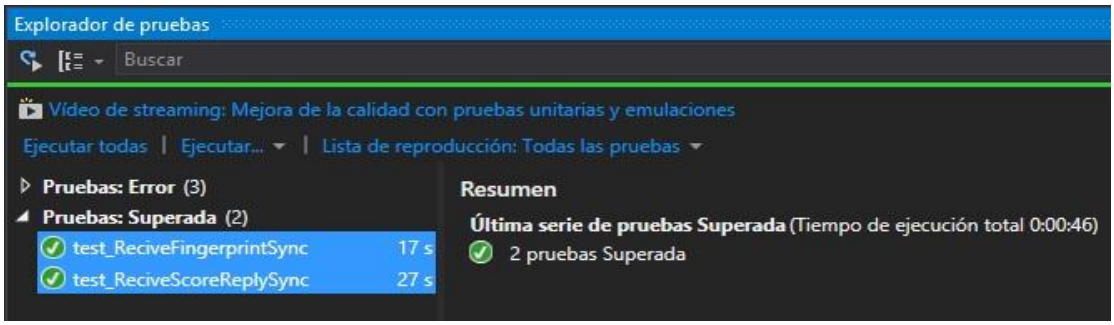

Figura 13. Resultados de Unit test 1 y 2.

3.7.2 Pruebas de integración

Las pruebas de integración se ocupan principalmente de encontrar defectos en el sistema y una vez encontrados el equipo de pruebas tiene acceso al código fuente para intentar encontrar la fuente del problema e identificar los componentes que tienen que ser depurados. A través de ellas se comprueba que los componentes que se integran funcionan juntos, son llamados correctamente y transfieren los datos en el tiempo preciso por medio de sus interfaces.

La integración del sistema implica identificar grupos de componentes que proporcionan alguna funcionalidad del sistema e integrar estos añadiendo código para hacer que funcionen conjuntamente. La integración puede realizarse de manera descendente o ascendente; la primera se ejecuta desarrollando el esqueleto del sistema en su totalidad para luego añadir los componentes y la segunda se ejecuta integrando primeramente los componentes de infraestructura que proporcionan servicios comunes dígase: acceso a bases de datos y redes y a continuación se añaden los componentes funcionales. En la práctica la estrategia de integración es una mezcla de ellas, añadiendo en incrementos, componentes de infraestructura y componentes funcionales (12).

De acuerdo con la metodología seleccionada para realizar este tipo de pruebas el cliente debe implicarse en el proceso y decidir que funcionalidad debe incluirse en cada incremento del sistema por lo que la integración está dirigida por las prioridades de este.

Resultados de la prueba de integración.

Partiendo del hecho que el Módulo de control de asistencia con identificación mediante huellas dactilares debe integrarse al Sistema de control de acceso a entornos seguros desarrollado por el CISED, se decide realizar la integración de forma descendente. Para ello se probó que el módulo es capaz de trabajar con los usuarios registrados en el sistema, además se comprobó que el módulo es capaz de abrir una sesión de acceso a la red en el propio sistema.

La prueba de integración se realizó a la funcionalidad Permitir o denegar acceso al centro donde para su ejecución se necesitó utilizar la tabla UserProfile de la base de datos del Sistema de control de acceso a entornos seguros del CISED para obtener los usuarios de este y comprobar si tienen definidos permisos de acceso en el Módulo de control de asistencia. También se comprobó que luego de que el usuario fuese identificado se registrara en la tabla IdentificationHistory este evento y se le abriera una sesión de acceso a la red. Los resultados obtenidos fueron satisfactorios y por tanto se puede afirmar que el Módulo de control de asistencia puede integrarse al sistema de control de acceso desarrollado por el CISED.

3.7.3 Pruebas de aceptación

La prueba de aceptación del usuario es la prueba final antes del despliegue del producto obtenido. Su objetivo es verificar que el software esté listo y que puede ser usado por los usuarios finales para ejecutar aquellas funciones y tareas para las cuales fue construido. En el caso de XP las pruebas de aceptación son creadas en base a las historias de usuario, en cada iteración del ciclo de desarrollo. El cliente debe especificar uno o diversos escenarios para comprobar que una historia de usuario ha sido correctamente implementada. A continuación se describen tres casos de prueba de aceptación.

Inicialmente el caso de prueba de la HU-03 Identificar huella dactilar permitirá probar que el módulo identifique la huella que se capturó en el lector correctamente.

Caso de Prueba de Aceptación	
Código: CP_HU-03	**HU-03:** Identificar huella dactilar.
Responsable: Emmanuel Gómez Soler	

Descripción: Probar que se identifica a la persona a la que pertenece la huella obtenida del lector de huellas.
Condiciones de ejecución: La aplicación web debe estar conectada al servidor RabbitMQ y al sistema SAIBIO.
Entrada / Pasos de ejecución: 1. Poner el dedo en el lector de huellas.
Resultado esperado: Que el sistema SAIBIO envíe el resultado de la identificación.
Evaluación de la prueba: Satisfactoria.

Tabla 10. CP_HU-03: Identificar huella dactilar.

En el caso de prueba de aceptación correspondiente a la HU-05 Permitir o denegar acceso al centro, se probará que la persona a la cual se le capturó la huella pueda acceder al centro.

Caso de Prueba de Aceptación	
Código: CP_HU-05	**HU-05:** Permitir o denegar acceso al centro.
Responsable: Emmanuel Gómez Soler	
Descripción: Probar que la persona que se identifica pueda acceder a la institución.	
Condiciones de ejecución: Que se halla identificado una persona que pertenece al centro.	
Entrada / Pasos de ejecución: 1. Pulsar “Modulo Biométrico” en la barra de navegación. 2. Escoger en la barra lateral la opción “Identificación Biométrica”. 3. Poner el dedo en el lector de huellas. 4. Ejecutar la acción “Identificar Usuario”.	
Resultado esperado: Que la persona que se identificó tenga permiso de acceso al centro.	
Evaluación de la prueba: Satisfactoria.	

Tabla 11. CP_HU-05: Permitir o denegar acceso al centro.

Para el caso de las pruebas de aceptación realizadas a la HU-08 Exportar los reportes generados a Excel, se probará que dado un reporte ya generado este se pueda exportar a Excel.

Caso de Prueba de Aceptación	
Código: CP_HU-08	**HU-08:** Exportar los reportes generados a Excel.
Responsable: Rosana Castro García	
Descripción: Probar que el sistema es capaz de exportar los reportes generados.	
Condiciones de ejecución: Debe haberse generado el reporte que se desea exportar a Excel.	
Entrada / Pasos de ejecución: 1. Pulsar “Modulo Biométrico” en la barra de navegación. 2. Escoger en la barra lateral cualquier opción para generar reportes. 3. Llenar los campos necesarios. 4. Ejecutar la acción “Buscar”. 5. Ejecutar la opción “Exportar”.	
Resultado esperado: Excel del reporte seleccionado para exportar.	
Evaluación de la prueba: Satisfactoria.	

Tabla 12. CP_HU-08: Exportar los reportes generados a Excel.

Resultados de las pruebas de aceptación.

Se diseñaron 20 casos de prueba para probar las funcionalidades del sistema, en este caso se tuvo en cuenta la importancia que representa para el funcionamiento del módulo de control de asistencia cada una de ellas. Por cada una de las dos iteraciones de implementación se realizaron tres iteraciones de pruebas de aceptación. Resumiendo, en la primera iteración de pruebas de aceptación se detectaron 19 no conformidades, de ellas 14 fueron solucionadas. En la segunda se detectaron 7 no conformidades, donde quedaron resueltas 6 de ellas incluyendo las cinco no resueltas de la primera iteración. Por último, en la tercera iteración se detectaron 2 no conformidades, quedando resueltas ambas además de la que quedó pendiente de la segunda iteración. Finalmente se concluye con la solución de todas las no conformidades detectadas en la etapa de pruebas de aceptación (Ver Figura 14).

Figura 14. Resultados de las pruebas de aceptación.

3.8 Conclusiones parciales

Al finalizar este capítulo se puede constatar que para la implementación y prueba del módulo se desarrollaron nuevos artefactos que permitieron legibilidad y mejor entendimiento de la propuesta de solución. La definición de las tareas de ingeniería y el uso de la codificación definida por Camel y Pascal permitieron la especificación de las funcionalidades a implementar por cada HU y la organización del código respectivamente facilitándole la programación al equipo de desarrollo. Se construyeron los diagramas de componente y de despliegue y se diseñaron las interfaces de prueba para ilustrar la estructura y el funcionamiento del módulo. Se constató que el desarrollo guiado por pruebas asegura la ejecución correcta de la solución en todo el periodo de implementación. Con la realización de las pruebas de aceptación se obtuvieron varias no conformidades que fueron corregidas en su momento pero demostraron que el módulo de control de asistencia cumple con las funcionalidades requeridas por el cliente.

Conclusiones

A raíz de la investigación que se realizó acerca de los sistemas de control de asistencia a nivel nacional e internacional, además del estudio de las diversas características de este tipo de sistemas se pudieron confeccionar las bases para desarrollar el ***Módulo de Control de Asistencia mediante Identificación por Huellas Dactilares***. De forma general, como resultado de la investigación, se arriba a las siguientes conclusiones.

- ✓ La elaboración del plan de iteraciones permitió establecer al equipo de desarrollo un límite de tiempo estimado en semanas para implementar cada funcionalidad y no exceder el tiempo establecido para la entrega del producto final.
- ✓ El uso del estilo arquitectónico cliente-servidor y el patrón arquitectónico MVT permitieron establecer la comunicación entre los componentes del módulo de control de asistencia.
- ✓ La aplicación del control de asistencia que forma parte del módulo final, tiene un alto nivel de independencia, por tanto puede ser usada por otros sistemas de control de acceso.
- ✓ Se demostró que el uso y escritura de pruebas unitarias sobre partes y funciones sensibles del Módulo de control de asistencia asegura la escritura de código sin errores y aumenta la consistencia de la implementación.
- ✓ Las pruebas de aceptación realizadas permitieron satisfacer las expectativas del cliente.
- ✓ Las pruebas de integración realizadas al Módulo de control de asistencia demostraron que este tiene condiciones para integrarse al Sistema de control de acceso del CISED.

Recomendaciones

Una vez concluida la investigación y vencido el objetivo general de esta se recomienda:

1. Migrar el Sistema de control de acceso al que se integra el módulo desarrollado del *framework* de diseño web Bootstrap 2 a su versión 3.
2. Agregar nuevos reportes al control de asistencia para obtener información más detallada sobre cómo se comporta el flujo de personal en el centro.
3. Realizar la identificación a través de otras técnicas biométricas dígase rostro, iris, voz, para cualquier eventualidad que pueda ocurrir con la técnica utilizada.
4. Analizar la posibilidad de desarrollar un *driver* portable con tecnologías libres para manejar el escáner puesto que el actual solo se puede utilizar con *Windows*.

Referencias bibliográficas

1. **Pfleeger, Charles P.***Security in computing.* 2006. 978-0-13-239077-4.

2. **Machin Armas, Melvis y Hernández Delgado, Royli.** *Sistema para la distribución del proceso de búsqueda de huellas dactilares en el banco de datos de un AFIS.* 2013.

3. **Tapiador Mateos, Marino.** *Tecnologías Biométricas aplicadas a la seguridad.* Madrid : Alfaomega, 2005.

4. **Universidad Técnica de Cotopaxi.** *Repositorio Digital.* [En línea] http://repositorio.utc.edu.ec/bitstream/27000/458/1/T-UTC-1027.pdf.

5. Kimaldi. Kimaldi. *Control de Acceso y Presencia.* [En línea] http://www.kimaldi.com/area_de_conocimiento/control_de_acceso_y_presencia/control_de_asistencia.

6. Gordon, Jeffrey I. *Forensic identification using skin bacterial communities.* [En línea] http://www.pnas.org/content/early/2013/02/01/1000162107.

7. ConceptoDefinición. *ConceptoDefinición.* [En línea] 2013. http://conceptodefinicion.de/identificacion/.

8. Quijano Acevedo, Carlos Alberto. *Planigrafo. Control Biométrico de Acceso y Asistencia de Personal i-Huellas Planigrafo.* [En línea] http://www.planigrafo.com/ihuellas.html.

9. Roca, Carlos. Dointech. *Automatización, Seguridad y Control.* [En línea] http://www.dointech.com.co/.

10. Ecured. *Sistema de Control de Asistencia (KONASI).* [En línea] http://www.ecured.cu/index.php/Sistema_de_Control_de_asistencia_%28KONASI%29.

11. Datys. Tecnologia & Sistemas. *Control de asistencia con elementos biométricos.* [En línea] http://www.datys.cu/wpinfproducto.aspx?42.

12. Sommerville, Ian. *Ingeniería del Software. Septima Edición.* Madrid : Pearson Educación.SA, 2005. págs. 76-78. 84-7829-074-5.

13. Solis, Camilo J., Figueroa, Roberth G. y Cabrera, Armando A. *METODOLOGÍAS TRADICIONALES VS. METODOLOGÍAS ÁGILES.*

14. Letelier, Patricio y Penadés, María Carmen. *Metodologías ágiles para el desarrollo de software.* [En línea] [Citado el: Noviembre de 2014.] http://www.funtec.org.ar/pdf/Pdf_semin_completo_MetodologiasAgiles.pdf.

15. Beck, K. *Extreme Programming Explained. Embrace Change.* s.l. : Addison Wesley, 1999.

16. Universidad de Oriente, Venezuela. [En línea] [Citado el: 13 de 11 de 2014.] http://wiki.monagas.udo.edu.ve/index.php/Metodolog%C3%ADas_SCRUM_y_XP.

17. Scrum Manager. *Body of the Knowledge.* [En línea] [Citado el: 13 de 11 de 2014.] http://www.scrummanager.net/bok/index.php?title=Modelo_original_de_Scrum_para_desarrollo_de_software.

18. Calabria, Luis. *Metodología FDD.* [En línea] 2003. http://fi.ort.edu.uy/innovaportal/file/2021/1/metodologiafdd.pdf.

19. Rumbaugh, James, Jacobson, Ivar y Booch, Grady. *El Lenguaje Unificado de Modelado. Manual de Referencia.* [ed.] Addison Wesley. Madrid : Pearson Educación, S.A, 2007. pág. 552. 84-7829-037-0.

20. López González, Pascual, López González, Ana Amelia y Lázaro Gallud, José Antonio. *Herramientas CASE ¿Cómo incorporarlas con éxito a nuestra organización?* España : Universidad de Castilla-La Mancha, 2005.

21. Visual Paradimg. *Visual Paradimg for UML.* [En línea] [Citado el: 18 de 11 de 2014.] http://www.visual-paradigm.com/product/vpuml/.

22. De Nobrega, María. *Herramientas CASE: Rational Rose.* [En línea] [Citado el: 18 de 11 de 2014.] http://curso_sin2.blogia.com/2005/060401-herramientas-case-rational-rose.

23. Gutiérrez, Javier J. *¿Qué es un framework web?*

24. Holovaty, Adrian y Kaplan-Moss, Jacob. *The Definitive Guide to Django: Web Development Done.* Nueva York : Apress, 2009. 978-1-4302-1937-8.

25. Micosoft Developed Network. *Información general acerca de .NET Framework.* [En línea] [Citado el: 21 de Noviembre de 2014.] http://msdn.microsoft.com/es-es/library/zw4w595w%28v=vs.110%29.aspx.

26. Wilson Blackett, Leslie. *Comparative Programming Languages.* s.l. : Addison-Wesley, 1993. 0-201-56885-3.

27. Microsoft Developed Network. *Resumen de las características de C#.* [En línea] [Citado el: 21 de 11 de 2014.] http://msdn.microsoft.com/es-es/library/aa287483%28v=vs.71%29.aspx.

28. Álvarez, Miguel Angel. *DesarrolloWeb.com. Qué es Python.* [En línea] [Citado el: 14 de 11 de 2014.] http://www.desarrolloweb.com/articulos/1325.php.

29. Fergarciac. *Entorno de Desarrollo Integrado (IDE).* [En línea] Enero de 2013. https://fergarciac.wordpress.com/2013/01/25/entorno-de-desarrollo-integrado-ide/.

30. Avery, James. *Visual Studio Hacks: Tips & Tools for Turbocharging the IDE.* s.l. : O'Reilly Media, 2005.

31. Jetbrains. *Pycharm The Most Intelligent Python IDE.* [En línea] https://www.jetbrains.com/pycharm/.

32. Roebuck, Kevin. *Advanced Message Queuing Protocol (Amqp): High-Impact Strategies - What You Need to Know: Definitions, Adoptions, Impact, Benefits, Maturity, Vendors.* 2011.

33. Videla, Alvaro y Williams, Jason J.W. *RabbitMQ in Action Distributed Messaging For.* s.l. : Manning Publications Company, 2012.

34. RabbitMQ. [En línea] RabbitMQ. [Citado el: 4 de Diciembre de 2014.] http://www.rabbitmq.com/features.html.

35. DATE, C. J. *Introducción a los sistemas de bases de datos.* México : Pearson Educación, 2001. pág. 960. 968-444-419-2.

36. PostgreSQL. *PostgreSQL: About.* [En línea] [Citado el: 21 de Noviembre de 2014.] http://www.postgresql.org/about/.

37. Larman, Craig. *UML y patrones.* 1999.

38. Hilliard, Rich. *IEEE-Std-1471-2000 Recommended Practice for Architectural Description of SoftwareIntensive Systems.* 2000.

Glosario de términos

A	**ACID** **API**	Atomicity, Consistency, Isolation and Durability. Applications Programming Interface.
C	**CASE** **CISED** **CRC**	Computer Aided Software Engineering. Centro de Identificación y Seguridad Digital Class, Responsibilities y Collaborators
D	**DATYS**	Empresa de Desarrollo de Aplicaciones, Tecnologías y Sistemas.
F	**FDD**	Feature Driven Development
G	**GPL** **GNU** **GRASP** **GOF**	General Public License. GNU's Not Unix. Patrones de asignación de responsabilidades. Banda de los Cuatro.
H	**HU** **HTTPS**	Historia de usuario. Hiper Text Transfer Protocol.
I	**IDE**	Entorno de desarrollo integrado, también conocido como entorno de diseño integrado o entorno de depuración integrada.
M	**MVT**	Model view template o Modelo vista plantilla.
O	**OTP**	Open Telecom Platform.

S	**SGBD** **SSL** **Sprints**	Sistema gestor de bases de datos. Secure Socket Layer.
U	**UML** **UCI**	Unified Modeling Language. Universidad de las ciencias informáticas.
X	**XP**	EXtreme Programing o Programación extrema.

Tabla 13. Glosario de términos.