Universidad de las Ciencias Informáticas

Facultad 4

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

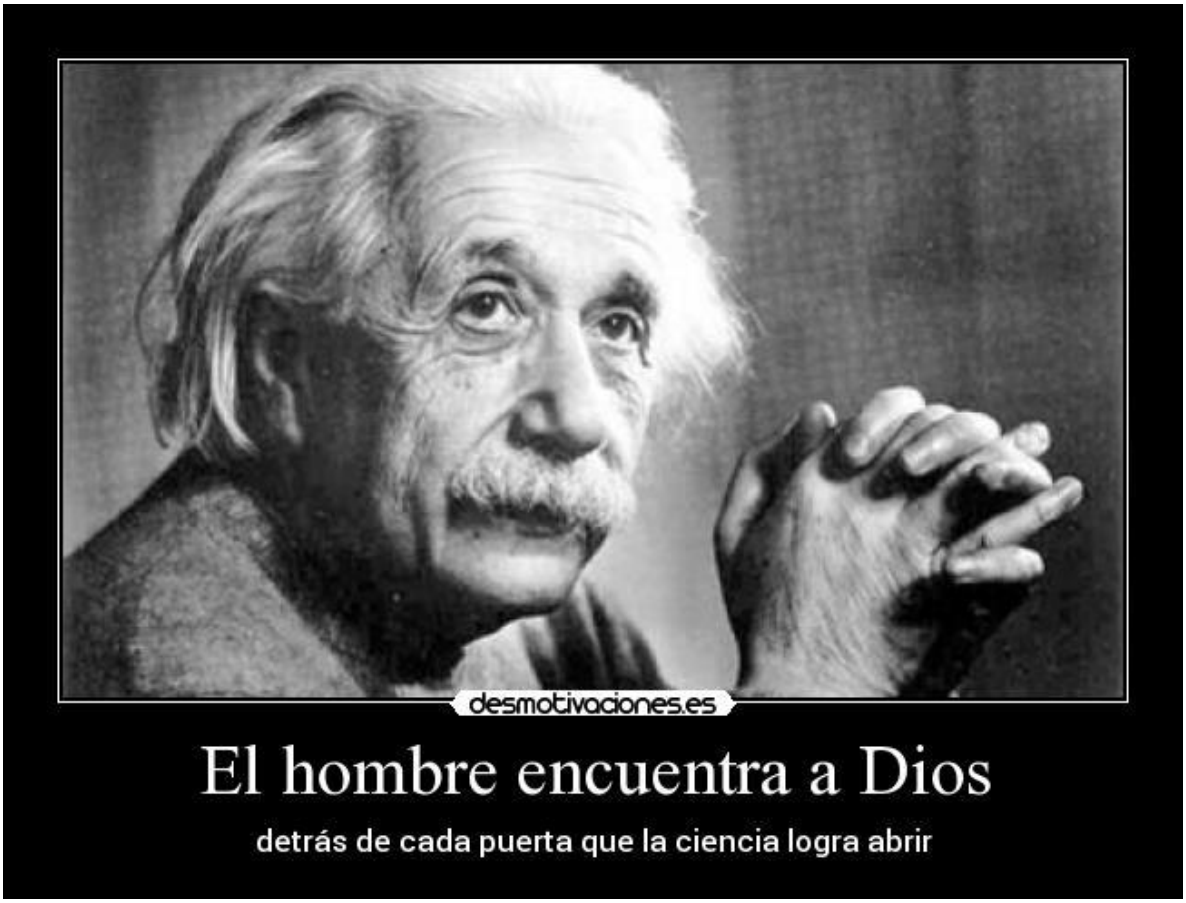
Título: Sistema de generación de hojas de firma mediante identificación biométrica.

Autor: Daniel Miguel González Mora

Tutores:

Msc. Luis Manuel Vidal Piña

Ing. Alberto Arístides Puentes González


desmotivaciones.es
El hombre encuentra a Dios
detrás de cada puerta que la ciencia logra abrir

Declaración de autoría

Hago constar que el presente Trabajo de Diploma fue realizado en la Universidad de las Ciencias Informáticas como parte de la culminación de sus estudios de Ingeniería en Ciencias Informáticas, autorizando a que el mismo sea utilizado por la Institución para los fines que estime conveniente, tanto de forma parcial como total.

Para que así conste, firmo el presente Trabajo de Diploma a los __ días del mes de _____ del año 2019.

Daniel Miguel González Mora

_____

Firma del autor

Msc. Luis Manuel Vidal Piña

_____

Firma del tutor

Ing. Alberto Arístides Puentes González

_____

Firma del tutor

Dedicatoria

Dedico este trabajo al Dios Todopoderoso por cuidar siempre de mí y por haberme ayudado a llegar hasta este momento tan especial para mí. A mis padres por haberme dado tanto amor, por haberme apoyado siempre, tanto en los momentos buenos como en las circunstancias más difíciles, y luchar tanto para que yo pudiera estar aquí ahora. Al resto de mi familia por su gran apoyo y el amor que han depositado en mí. A los mejores amigos que he tenido en esta vida por su imprescindible apoyo para que yo pudiera estar aquí y por esos buenos momentos que pasamos juntos. A mis hermanos de la fe por haberme ayudado en estos cinco años y por los buenos tiempos que hemos estado compartiendo juntos.

Agradecimientos

Primeramente, me gustaría darle las gracias a Dios porque sin Él no hubiera sido posible llegar hasta aquí. Gracias por su amor, por haber sido el principal apoyo que he tenido en este tiempo y por haberme enseñado tantas cosas que eran imprescindibles para mi crecimiento espiritual y personal.

A mi familia por apoyarme en todo momento en especial a mis padres por su apoyo incondicional. A mis tutores Alberto y Luis Manuel agradezco su apoyo en estos meses de continuo trabajo. Al tribunal por todas sus sugerencias. A todos los profesores de los que recibí excelentes clases. A los ingenieros del Centro de Informática Industrial por sus aportes a la investigación. A mis compañeros de aula que me acompañaron en estos cinco años. Y a los grandes amigos que he encontrado en la universidad.

Resumen

En la actualidad los sistemas biométricos son una alternativa confiable para la autenticación de la identidad, existiendo diversas técnicas para su desarrollo. Entre las técnicas de identificación biométrica más utilizadas se encuentra el reconocimiento de personas a través de la huella dactilar, siendo este un método efectivo y seguro para la implementación de un sistema para el control de presencia debido a su facilidad de uso y la gran aceptación que tiene por parte de los usuarios.

El presente trabajo describe las diferentes herramientas, tecnologías y artefactos generados en el proceso de desarrollo de un sistema para el control de asistencia de los trabajadores del Centro de Informática Industrial (CEDIN), perteneciente a la Universidad de las Ciencias Informáticas (UCI), a través de la generación automática de sus hojas de firma, tomando como base las experiencias y tecnologías mediante el uso de la identificación de personas por huella dactilar. En la investigación se realiza el diseño de todos los elementos que componen el sistema de generación de hojas de firma y, tanto el *hardware* como el *software* dan respuesta a las necesidades planteadas, obteniéndose un reporte de hoja de firma por cada trabajador del CEDIN, posibilitando un control detallado y eficiente de la asistencia de sus trabajadores.

Palabras clave: hojas de firma, huella dactilar, identificación biométrica, sistema de control de asistencia.

Abstract

At present, biometric systems are a reliable alternative for identity authentication, and there are several techniques for their development. Among the most used biometric identification techniques is the recognition of people through the fingerprint, this being an effective and safe method for the implementation of a system for presence control due to its ease of use and the great acceptance that has on the part of the users.

The present work describes the different tools, technologies and artifacts generated in the development process of a system for the attendance control of the workers of the Industrial Computing Center, belonging to the University of Informatics Sciences (UIS) through the automatic generation of their signature sheets, based on experiences and technologies through the use of people identification by fingerprint. In the research, the design of all the elements that make up the system for generating signature sheets is done, and both the hardware and the software respond to the needs raised, obtaining a report of signature sheet for each worker of the Industrial Computing Center, enabling a detailed and efficient control of the assistance of its workers.

Key words: *signature sheets, fingerprint, biometric identification, presence control system.*

Índice

Introducción

Los orígenes de la biometría se remontan hasta el antiguo Egipto, donde se usaba la huella dactilar para confirmar la identidad de las personas. También existen evidencias del uso de identificación por huella dactilar en China en el siglo XIV, en las que se dejaba la huella dactilar marcada sobre arcilla (1).

El Genoma Humano aporta el material genético y contribuye a formar las características propias de las huellas dactilares, por tanto, es natural que entre personas de la misma familia se encuentren similitudes en la forma de la huella, pero nunca serán iguales; y por lo anterior, es que los gemelos no poseen huellas iguales, sino solo patrones similares (2). Por lo anterior expresado es que las huellas dactilares son un método eficaz y prácticamente único para identificar a una persona.

La biometría como medio de identificación evoluciona constantemente ampliando sus usos y los dispositivos a los que se incorporan. De los distintos indicadores sobre los que se basa la tecnología biométrica los más extendidos son la huella digital, la lectura del iris, el reconocimiento facial y el reconocimiento de voz, teniendo un papel decisivo en la *"IoT"* (Internet de las Cosas) por sus aplicaciones en la vida doméstica y laboral (3).

Es cada vez más frecuente la necesidad de implementar sistemas que permitan de forma precisa la identificación y/o validación de personas para fines como la seguridad informática, control de acceso y control de asistencia, siendo la identificación biométrica el mejor método de identificación humana para lograr esto. En la actualidad se utilizan los sistemas de reconocimiento biométricos con distintos objetivos, siendo una alternativa confiable y versátil para autentificación de identidad, lo que permite que puedan ser utilizados en cualquier aplicación que requiera seguridad, control de acceso y asistencia e identificación o comprobación de usuarios.

Uno de los mayores problemas en la administración de empresas es el control de asistencia de los empleados. Incluso para empresas que tienen pocos empleados, la diversidad de horarios y los distintos permisos e incidencias que pueden ocurrir hacen que la inversión en la adquisición de un *software* de control de asistencia resulte de suma importancia para el control de los horarios (4). Es por ello que el control de asistencia garantiza no solamente el pago de los empleados, sino también contribuye a mejorar la disciplina en el centro

laboral y el adecuado empleo del tiempo para garantizar que se realicen de una forma adecuada las tareas planificadas.

El control de asistencia tiene como objetivo determinar la ausencia o presencia de personal en un momento determinado y su importancia radica en el hecho de controlar la entrada y salida del mismo. Desde el punto de vista productivo y financiero, para una empresa, los empleados son los principales protagonistas para el cumplimiento de sus objetivos y metas trazadas.

Las técnicas biométricas se basan en los rasgos y características únicas de los seres humanos, de forma tal que las mismas son viables para identificar correctamente a una persona, dejando obsoletos el uso de los métodos tradicionales de identificación. Además de esto, los sistemas biométricos cuentan con características de universalidad, unicidad, estabilidad, facilidad de captura, aceptación por los usuarios, rendimiento y son cuantificables. Por tanto, por las características que presentan, los sistemas biométricos permiten mejorar el control de asistencia de los empleados.

El Centro de Informática Industrial (CEDIN), perteneciente a la Universidad de las Ciencias Informáticas (UCI), tiene la necesidad de incrementar la confiabilidad en el control de la asistencia de sus trabajadores, siendo este aspecto de gran importancia para aumentar la productividad en el trabajo y el aprovechamiento de la jornada laboral. En dicho centro se encuentra instalado el lector de huellas dactilares ZKTeco F702-S, con el objetivo de realizar el control de acceso a sus trabajadores. Este lector trabaja de conjunto con un *software* que permite la identificación de sus trabajadores a través del empleo de un identificador biométrico por huella dactilar y provee la opción de exportar un archivo en diferentes formatos donde el usuario puede seleccionar los campos que desee exportar y que contiene como información el nombre del trabajador, el departamento al que pertenece, un identificador numérico, la fecha y la hora en que el trabajador marcó con su huella en el sistema. No obstante, el proceso de control de los horarios de los trabajadores en el CEDIN se realiza de forma manual, permitiendo la ocurrencia de errores en las hojas de firma de los trabajadores. Las hojas de firma son el modelo que se utiliza para el control de la entrada y salida, recogiendo como información la fecha, hora de entrada y salida de cada trabajador al centro. Es utilizada para el control del personal dentro de una organización con el objetivo de mejorar el uso del tiempo de la jornada laboral y el pago de la nómina de los trabajadores.

Dentro de los problemas que genera el proceso manual de control de horarios a través de las hojas de firma se encuentran:

- El libro de las hojas de firma no siempre se encuentra disponible en el horario y lugar correspondiente.
- Los trabajadores ponen horarios que no se corresponden con el horario real de su entrada y salida.
- Los trabajadores firman en los lugares incorrectos dentro de su hoja de firma.
- En ocasiones las firmas y horarios de un trabajador son falsificados por otro.

Todo esto conlleva a que se realicen pagos indebidos, desaprovechamiento de la jornada laboral, atraso en los proyectos e incumplimiento de los cronogramas. Además de todos estos problemas anteriormente mencionados, el *software* de identificación del lector de huellas dactilares fue desarrollado para el sistema operativo *Windows*, y por cuestiones de la nueva política de gobernabilidad tecnológica de la Universidad no es factible el empleo de este *software* para la generación de reportes personalizados de hojas de firma.

Por tanto, se hace necesario desarrollar un sistema multiplataforma de control de asistencia que permita supervisar el horario de entrada y salida de los trabajadores del CEDIN a través de la generación automática de sus hojas de firma utilizando la información que proporciona la base de datos del *software* que contiene los datos registrados por el lector óptico de huellas digitales instalado en el CEDIN.

Teniendo en cuenta la situación problemática anteriormente descrita, se identifica como **problema de investigación:** ¿Cómo automatizar el proceso de generación de hojas de firma del Centro de Informática Industrial?

Teniendo en cuenta el problema anterior, se define como **objeto de estudio** la generación automática de hojas de firma mediante la identificación por huella dactilar.

Por ello, la presente investigación se encuentra enmarcada en el **campo de acción** proceso de generación automática de hojas de firma del Centro de Informática Industrial utilizando el lector de huellas dactilares ZKTeco F702-S.

Como **objetivo general** se desea desarrollar un sistema de generación de hojas de firma utilizando el lector de huellas dactilares ZKTeco F702-S.

Para dar cumplimiento al objetivo general antes mencionado, se dará cumplimiento a los siguientes **objetivos específicos:**

- Analizar los principales fundamentos teóricos y tecnológicos relacionados con el proceso de generación de hojas de firma mediante la identificación biométrica por huella dactilar.
- Analizar las principales tendencias actuales en el desarrollo de sistemas de generación automática de hojas de firma a través del empleo de la identificación biométrica por huella dactilar.
- Definir las tecnologías, herramientas y metodología a utilizar para la implementación de la propuesta de solución.
- Realizar el análisis y diseño de la propuesta de solución para su posterior implementación.
- Implementar el sistema de generación de hojas de firma mediante identificación biométrica por huella dactilar.
- Validar el sistema implementado para determinar el grado de cumplimiento del objetivo planteado en la investigación realizada mediante la realización de pruebas de *software*.

En la presente investigación se emplearon los siguientes **métodos científicos:**

Métodos teóricos:

- **Histórico-lógico:** Se emplea para el estudio del desarrollo y evolución de los diferentes sistemas biométricos de generación de hojas de firma similares existentes en la actualidad, ya sea en el ámbito nacional o internacional, así como el estudio de las herramientas y tecnologías para el desarrollo de la propuesta de solución como son los lenguajes de programación, *framework* de desarrollo, metodologías y herramientas *CASE* empleadas en la implementación de la misma.
- **Analítico-sintético:** Se utiliza para el análisis de la información empleada en la investigación, identificando así los conceptos, definiciones y avances más relevantes relacionados con el objeto de estudio acerca del proceso de generación automática de hojas de firma a través de la identificación por huella dactilar.
- **Modelación:** Se utiliza en la modelación de los diagramas de clases necesarios que se realizan dentro de la metodología de desarrollo de *software* seleccionada para llevar a cabo la propuesta de solución.

Métodos empíricos:

- **Observación:** Se emplea para obtener información de las necesidades existentes en los laboratorios productivos del CEDIN.
- **Entrevista:** Se utiliza para la recopilación de información, ideas y opiniones de los directivos y trabajadores del CEDIN acerca de la investigación que se lleva a cabo, contribuyendo así al desarrollo del futuro sistema informático.

La siguiente investigación está conformada por:

Resumen, Introducción, 3 Capítulos, Conclusiones, Recomendaciones, Referencias Bibliográficas y Anexos. Los capítulos de la presente investigación se fundamentan en:

- **Capítulo 1: "Fundamentación Teórica".** En este capítulo se exponen los elementos teóricos empleados en la investigación para una mejor comprensión del tema que define el objeto de estudio. Se analizan las principales características de los sistemas similares estudiados. Se describen las herramientas, tecnologías, metodología de desarrollo de *software* y el lenguaje de programación utilizados en el desarrollo de la propuesta de solución.
- **Capítulo 2: "Análisis y Diseño de la solución propuesta".** En este capítulo se exponen los elementos que describen la propuesta de solución, como son el modelo de dominio, los requerimientos funcionales y no funcionales, las historias de usuario y la descripción textual de cada una de ellas, los patrones arquitectónicos y de diseño empleados en la propuesta de solución, incluyendo los diagramas de clases y los prototipos de interfaz de usuario para lograr un mejor entendimiento del sistema a desarrollar.
- **Capítulo 3: "Implementación y Prueba de la solución propuesta".** En este capítulo se describe el modelado de las funcionalidades y la implementación de la propuesta de solución, donde se obtiene un sistema que permite la generación automática de las hojas de firma de los trabajadores del CEDIN. Se describen además las pruebas realizadas al sistema con el objetivo de comprobar el correcto funcionamiento de la propuesta de solución y que cumpla con las especificaciones requeridas por el cliente.

Capítulo 1: Fundamentación Teórica.

Los sistemas de control de asistencia se han visto envueltos en una notable evolución gracias al avance de la tecnología. En la actualidad numerosas empresas e instituciones requieren de la implementación de un sistema de control de asistencia en sus instalaciones, utilizados habitualmente para controlar y registrar los horarios de entrada y salida del personal a sus locales con el objetivo de mejorar el aprovechamiento de su jornada laboral.

El presente capítulo aborda los principales conceptos asociados al problema a resolver, para ello se explica el proceso de control de asistencia laboral y se realiza una investigación de sistemas similares utilizados para la generación automática de reporte de hojas de firma mediante la identificación por huella dactilar. También se describen las diferentes herramientas de *software*, metodologías y lenguajes de programación que se emplean en el proceso de diseño y desarrollo de la propuesta de solución planteada.

1.1 Proceso de control de asistencia laboral

Los controles de asistencia laboral son sistemas de control de acceso que permiten llevar un registro de las entradas y salidas de los empleados en una empresa. La supervisión de la asistencia permite comprobar la puntualidad y asistencia de los trabajadores a su puesto de trabajo (5). Con el control de asistencia laboral la empresa puede monitorizar su productividad empresarial para saber si cumplirá sus objetivos (5).

1.2 Beneficios del control de asistencia laboral (5)

La implementación de sistemas de control sobre la asistencia laboral aporta ventajas para la empresa y los trabajadores. Cada vez más empresas implementan sistemas de control de asistencia por las ventajas que ofrecen, las cuales son:

- Permiten controlar los horarios de los empleados de una forma más eficiente.
- Fomentan una mayor seguridad y control de las visitas a la empresa.
- Controlar los accesos y la asistencia permite ahorrar costes en el personal de la empresa, puesto que la productividad empresarial se monitoriza de forma adecuada.
- Permite calcular la nómina de los empleados con respecto a los horarios de trabajo.
- Los sistemas de control de asistencia permiten monitorizar las horas extras para después pagarlas a los empleados.

La selección de un sistema de control de asistencia en una empresa es un proceso que requiere del análisis de diversos factores. Inicialmente se debe tener en cuenta el grado de seguridad que la empresa requiere, y si se necesita algún sistema para complementar la seguridad se implementa un sistema de control de acceso que permita tener un control más estricto de las entradas y salidas de personal. También es importante conocer la cantidad de usuarios van a utilizar el sistema de control de asistencia, debido a que el soporte será diferente para empresas con una gran cantidad de empleados, para no ralentizar la entrada y salida de los empleados (5).

Existen diversos métodos de control de asistencia laboral, pero los más empleados son las hojas de firma, las tarjetas perforadas, las tarjetas de código de barras y las tarjetas de banda magnética. Estos métodos son muy vulnerables y fácilmente pueden ser falsificados por cualquier individuo. Uno de los métodos más seguros para realizar el control de asistencia laboral es a través del empleo de la identificación biométrica, que luego de analizar el problema de investigación se determinó, atendiendo a las necesidades y recursos de los que dispone el CEDIN, que este sería el método a utilizar en conjunto con la generación automática de las hojas de firma de los trabajadores.

1.3 Identificación biométrica

La identificación biométrica es la verificación de la identidad de una persona basada en las características de su cuerpo como pueden ser la retina, el iris, la voz, el rostro o las huellas dactilares. Para llevar a cabo el proceso de identificación de una persona los biológicos de la misma son medidos y escaneados por sensores y almacenados en una base de datos para su posterior procesamiento. De esta forma, las huellas dactilares son la forma más sencilla, rápida y segura de identificación biométrica (6).

1.4 Técnicas biométricas comunes

Actualmente las técnicas y tecnologías biométricas desarrolladas han facilitado soluciones efectivas tanto para empresas de menor o mediana escala que precisan de precios económicos y facilidad de uso como para grandes empresas que requieren máxima seguridad y robustez según sus características. El desarrollo correcto de un *software* de control de presencia biométrico permitiría emplear un buen control de horarios, ingresos y salidas del personal, garantizando además la rapidez, confiabilidad, seguridad y eficiencia (7).

Actualmente existen una gran variedad de características de la persona tanto fisiológicas como conductuales que se consideran como técnicas biométricas. Algunas de estas técnicas son: la geometría de la mano, los patrones de las venas, el termograma facial, la huella dactilar, los patrones de la retina, el iris, la voz, la firma y el rostro (ver Figura 1.). Además de otras que también se emplean en la actualidad como la forma de la oreja, la forma de caminar, la dinámica de la escritura en un teclado, el ADN y el olor. Partiendo de estas características se han desarrollado dispositivos que han tenido mayor o menor éxito en el mercado. En la actualidad, las técnicas más usadas son:

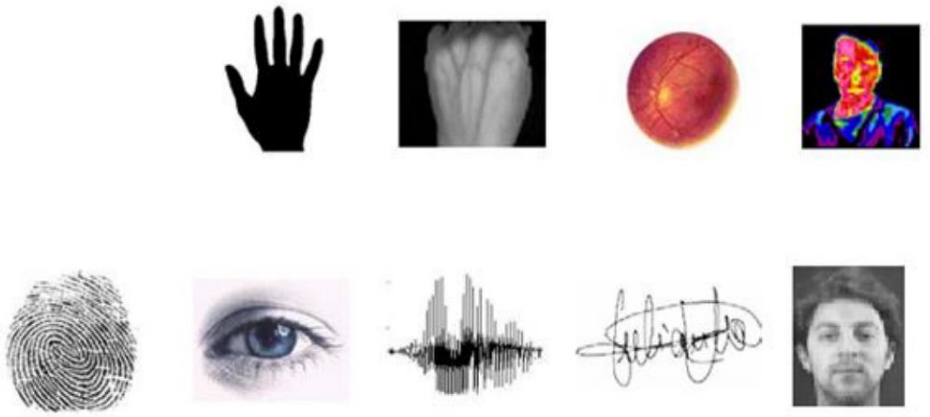


Figura 1. Técnicas biométricas.

1.4.1 Reconocimiento facial

El reconocimiento facial permite la identificación de personas en movimientos y el reconocimiento de personas que no estén dispuestos a ser reconocidos por el sistema. Es un método no invasivo y no requiere de acciones manuales. El rendimiento y eficacia de los sistemas de reconocimiento facial disponibles comercialmente es razonable, pero la naturaleza compleja y mutable de los rasgos faciales impone una serie de restricciones tanto en la forma y el ángulo de obtención de la imagen, como en el fondo de iluminación. Esta técnica presenta inconvenientes, es susceptible a problemas de iluminación y el sistema de captura necesita de una fuente de luz auxiliar. Estos sistemas son vulnerables al reconocimiento de sujetos que se han sometido a operaciones de cirugía plástica y no tiene en cuenta en todos los casos los efectos de la edad, por lo que su fiabilidad, facilidad de uso y aceptación por parte de los usuarios es baja.

1.4.2 Iris

La identificación de personas a través del ojo humano ha dado lugar a dos tipos de sistemas biométricos totalmente diferentes: una basada en las características del iris ocular y otra que utiliza las características distintivas de la retina.

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

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

1.4.3 Voz

La voz es una característica que las personas utilizan comúnmente para identificar a los demás. Es posible detectar patrones en el espectro de la frecuencia de voz de una persona que son casi tan distintivos como las huellas dactilares.

Tan solo basta recordar las veces en que se reconoce a alguien conocido por teléfono para comprender la importancia de esta característica como método de reconocimiento. Mediante el análisis de los sonidos que se emiten, los tonos bajos y agudos, la vibración de la laringe, los tonos nasales y de la garganta, los sistemas que usan esta técnica crean modelos de la anatomía de la tráquea, cuerdas vocales y cavidades. Muchos de ellos operan independientemente del idioma o el acento de la persona. No obstante, la voz cambia por aspectos como la edad, enfermedades y estados de ánimo. Es una técnica que

se lleva estudiando durante varias décadas, existiendo innumerables métodos para realizar, tanto para la extracción de características como para la comparación. Algunos métodos son dependientes del texto pronunciado, mientras que otros son independientes del mismo (8).

1.4.4 Geometría de la mano

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

1.4.5 Huella dactilar

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

La huella dactilar de todas las técnicas biométricas es la más extendida por su madurez, coste, usabilidad y rapidez en identificación (9). Es la técnica por excelencia de identificación, siendo confiable, fácil de adquirir, fácil de usar y por ende goza de una gran aceptación por parte de los usuarios.

El empleo de la técnica biométrica por huella dactilar evita el uso de tarjetas magnéticas, llaves, memorizar números identificativos y evitar el uso de contraseñas como métodos de identificación de personas, debido al nivel de confiabilidad que presenta dicha técnica con respecto a los métodos de identificación anteriormente mencionados.

A continuación, se muestra una tabla que representa una comparación entre las distintas técnicas biométricas teniendo en cuenta un conjunto de parámetros predefinidos:

Característica	Aceptación del usuario	Facilidad de uso	Costo	Identificación	Verificación	Estabilidad	Fiabilidad
ADN	Baja	Baja	Alto	SI	SI	Alta	Alta
Dinámica de escritura	Alta	Alta	Bajo	NO	SI	Baja	Baja
Firma	Media	Alta	Bajo	NO	SI	Media	Baja
Geometría de la mano	Media	Alta	Alto	NO	SI	Media	Media
Huella dactilar	Media	Alta	Bajo	SI	SI	Alta	Alta
Iris	Media	Media	Alto	SI	SI	Alta	Alta
Reconocimiento facial	Media	Media	Bajo	NO	SI	Media	Media
Retina	Media	Baja	Alto	SI	SI	Alta	Alta
Voz	Alta	Alta	Bajo	NO	SI	Media	Baja

Tabla 1. Comparación entre las técnicas biométricas más conocidas en la actualidad.

Característica	Nivel de seguridad	Rango de error	Precisión	Errores	Falso positivo	Falso negativo

ADN	Alto	Sin datos	4	No conocidos.	5	5
Dinámica de escritura	Medio	Sin datos	1	Lesiones de mano, cansancio.	3	1
Firma	Medio	1/50	2	Cambios de escritura.	2	1
Geometría de la mano	Medio	1/500	3	Edad, lesiones varias.	4	2
Huella dactilar	Alto	1/500+	4	Sequedad, suciedad, edad.	5	5
Iris	Alto	1/131000	4	Iluminación inadecuada.	4	4
Reconocimiento facial	Medio	Sin datos	3	Pelo, gafas, edad, iluminación.	3	1
Retina	Alto	$1/10^6$	4	Gafas, lentillas.	5	5
Voz	Medio	1/50	2	Ruidos, ronquera, resfriados.	2	1

Tabla 2. Comparación entre las técnicas biométricas más conocidas en la actualidad - Continuación.

En el desarrollo de la propuesta de solución al problema de investigación anteriormente planteado se hará uso de la técnica biométrica mediante huella dactilar, considerada como la técnica más utilizada y extendida de las tecnologías biométricas en la actualidad, que posee una alta precisión y goza de una gran aceptación por parte de los usuarios.

1.5 Sistema biométrico

Son sistemas automatizados que realizan labores biométricas, es decir, el reconocimiento se establece mediante una característica física de una persona con un patrón conocido. Estos sistemas están compuestos por dos dispositivos, uno de captura, que escanea una característica personal y crea una imagen digital o análoga del rasgo biométrico analizado, y otro de almacenamiento de datos, que almacena dicha imagen en una base de datos interna o externa para luego ser verificada al realizar la identificación de la persona.

Existen una serie de requisitos para definir un sistema biométrico. Estos requisitos dependen de las características que se utilizan como parámetros de identificación y clasificación. Para poder decir que una característica sea biométrica ésta debe cumplir las siguientes condiciones:

- **Universalidad:** Existe para todos los individuos.
- **Singularidad:** Debe tener carácter distintivo, es decir, identifica unívocamente a cada individuo.
- **Inmutabilidad:** Se mantiene invariable en el tiempo y ante condiciones ambientales adversas.
- **Facilidad de captura:** Si existen mecanismos sencillos para su adquisición, medición y almacenamiento.
- **Rendimiento:** Comprueba que tan preciso, veloz y robusto es el sistema en su manejo.
- **Aceptabilidad:** Si la tecnología utilizada cuenta con una alta aprobación por parte de los usuarios.
- **Confiabilidad:** El sistema debe reconocer características de una persona viva, es decir, que reconozca datos falsos capturados como fotos falsas o dedos de látex. En fin, esta característica mide el nivel de seguridad que satisface el sistema.

Actualmente los sistemas biométricos son utilizados en muchos lugares debido a que la identidad de una persona puede establecerse de manera rápida y confiable. Brindan un alto grado de seguridad, debido a que otros métodos de identificación como una contraseña o una tarjeta de identificación pueden ser copiados o robados por cualquier persona, mientras que para este sistema se necesita la presencia del individuo para su identificación. Entre los tipos de sistemas biométricos que existen en la actualidad el sistema biométrico dactilar es uno de los más utilizados.

1.6 Sistema biométrico dactilar

Un sistema biométrico dactilar es básicamente un sistema de reconocimiento de patrones en el que la identidad de un individuo se determina a partir de la captura de su huella dactilar, extrayendo un conjunto de características y comparándola con otro conjunto de características almacenadas en base de datos. Dependiendo de su finalidad, un sistema biométrico dactilar puede actuar de tres modos: registro, verificación e identificación (10).

En el **modo registro**, el usuario introduce su huella dactilar en el sistema a través de sus sensores correspondientes y se hace la extracción de sus datos personales y tras realizar el procesado que sea conveniente, se introducen en la base de datos (10).

En el **modo verificación**, el sistema valida la identidad de un usuario comparando su huella dactilar con sus rasgos previamente almacenados en una base de datos. El usuario generalmente indicará su identidad mediante algún tipo de código, identificador, nombre de usuario o una tarjeta inteligente. Posteriormente el sistema realizará una comparación para determinar si su huella dactilar se corresponde realmente con el usuario (10).

En el **modo identificación**, el sistema reconoce a un usuario a partir de su huella dactilar dentro de una base de datos de huellas previamente almacenados. El usuario no introduce un identificador, sino que es el sistema el que debe determinar su identidad. Para ello el sistema realiza una comparación para determinar la identidad del usuario o simplemente decide que no se encuentra almacenado en la base de datos (10).

Ventajas

Algunas de las ventajas que ofrecen el uso de los Sistemas Biométricos Dactilares son: (11)

- Los sujetos tienen múltiples huellas.

- Fácil de usar.
- Requieren poco espacio en disco.
- Permiten almacenar grandes cantidades de datos existentes para realizar chequeos de antecedentes.
- Se ha probado su efectividad en disímiles sistemas a lo largo de años de uso.
- Las huellas digitales son únicas para cada dedo de cada individuo y se mantienen invariables durante toda una vida.

Desventajas

Algunas de las desventajas que trae consigo el uso de los Sistemas Biométricos Dactilares son (11) :

- Preocupaciones de salud o sociales en relación a tocar un sensor que es utilizado por incontables cantidades de personas.
- Una colección de imágenes de una por cada dedo requiere de entrenamiento y destreza, porque la tecnología de lectura básica actual es muy robusta.
- La edad de un individuo y su ocupación pueden producir algunas dificultades en el sensor para capturar una imagen dactilar completa y correcta.
- Para la utilización de sistemas biométricos dactilares se emplean los lectores ópticos de huellas dactilares para capturar imágenes digitales de dichas huellas para ser posteriormente procesadas.

1.7 Lector de huellas dactilares

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

El lector de huella digital tiene una función de seguridad, ya que por medio de él es posible identificar personas dentro de una empresa y llevar un estricto control como su hora de llegada y salida exactas, también para el acceso a ciertos lugares restringidos, para realizar transacciones bancarias, etc. Los sistemas de control de acceso y asistencia por huella

dactilar brindan seguridad en cuanto a la autenticidad de las personas, tiene una alta fiabilidad, alta aceptación por los usuarios mundialmente y alta facilidad de uso.

La utilización de un dispositivo biométrico proporciona unos bajos costos de administración, ya que solo se le debe dar mantenimiento al lector de huellas dactilares y una persona se encarga de mantener la base de datos del sistema actualizada. El CEDIN cuenta actualmente con el lector ZKTeco F702-S para llevar a cabo el control de acceso y asistencia de sus trabajadores; a continuación se describen las principales características de dicho lector.

ZKTECO F702-S

Este lector de huellas dactilares ofrece la flexibilidad de instalarse de manera independiente o con paneles de terceros que admitan *Wiegand* de 26 bits. El rendimiento rápido y confiable y la asequibilidad refuerzan la ventaja competitiva del lector ZK. Cada lector tiene la capacidad de almacenar hasta 2,200 plantillas de huellas dactilares y 50,000 transacciones. La inscripción de usuarios es rápida y sencilla. Dado que la mayoría de estos lectores tienen un teclado y una pantalla fáciles de usar, inscribir usuarios es rápido y fácil y son de fácil acceso. Junto con un puerto TCP / IP o *serial*, los lectores pueden actualizarse y sincronizarse automáticamente (12).

Especificaciones Técnicas (12)**:**

- Equipado con sensor óptico ZK duradero y reconocible.
- Reconocimiento de empleados de 1 toque de 1 segundo.
- Almacena hasta 2200 plantillas de huellas digitales y 50000 transacciones.
- Ingresar PINs no es necesario.
- Admite 50 zonas horarias, 5 grupos y 10 combinaciones de desbloqueo.
- Interruptor a prueba de alteraciones y salida de alarma.
- Contactos de solicitud de salida y alarma.
- Operación autónoma sin una computadora.
- Interfaces TCP/IP, Serie y *Wiegand*.
- Exportar datos de 1 toque en tiempo real a aplicaciones alojadas o no alojadas de terceros.
- Indique señales audiovisuales para aceptación o rechazo debido a huellas digitales válidas o no válidas.

- SDK disponible para clientes OEM y desarrolladores de software.

Plataforma de hardware	ZEM565
Versión del algoritmo de identificación	ZKFinger V10.0
Puertos de Comunicación	RS232/485, TCP/IP, USB-host
Interfaz de Control de Acceso	Cerradura eléctrica de terceros, sensor de puerta, botón de salida, alarma y timbre de puerta
Funciones Estándar	SMS, DLST, servidor web, código de trabajo, *anti-passback*
Funciones Opcionales	Mifare, dispositivo de interfaz humana, ID de usuario de 9 dígitos, Interruptor automático de estado, SOAP, impresora
Señal de *Wiegand*	Entrda/Salida
Monitor	Pantalla LCD B & W
Fuente de alimentación	12 V DC
Temperatura de funcionamiento	0°C – 45°C
Humedad de funcionamiento	20% – 80%
Dimensiones (Ancho × Alto × Profundidad) mm	140.5 × 116 × 43.5

Peso bruto	0.85 kg

Tabla 3: Especificaciones técnicas del ZKTeco F702-S.

Figura 2. Lector de huellas dactilares ZKTeco F702-S.

1.8 Estudio del arte de los sistemas de control de asistencia

Los avances en las tecnologías de la información y las comunicaciones han permitido en los últimos años la evolución de las tecnologías biométricas hasta convertirse, hoy en día, en un método definitivo y legalmente aceptado para la identificación de personas en casi cualquier país del mundo (13). Las tecnologías biométricas están siendo aplicadas en variados lugares para aumentar la seguridad y comodidad de la sociedad.

En la actualidad, los métodos más aceptados de identificación se basan en la recolección de huellas dactilares y, últimamente, en las muestras de ácido desoxirribonucleico (ADN), cuyos grados de confiabilidad resultan casi infalibles. La mayoría de los países del mundo utiliza las huellas digitales como sistema práctico y seguro de identificación.

1.8.1 Soluciones existentes en el mundo

Con el desarrollo de las tecnologías y el avance de la informática en el mundo ha surgido la necesidad de informatizar procesos de forma más rápida y sencilla. Los sistemas de control de asistencia biométricos constituyen hoy en día un avance necesario para cualquier

organización existente. En la actualidad existen diferentes empresas y compañías internacionales con varios años de experiencia en brindar soluciones para el control de presencia biométrico, centradas fundamentalmente en el control de los horarios. Algunas de ellas son: Kimaldi, Dointech, SCSSA, INBIOSYS, entre otras. Como punto de partida de la investigación fue necesario analizar sistemas similares a nivel internacional para el desarrollo de la propuesta de solución. A continuación, se exponen los resultados de la investigación:

La empresa mexicana TSB ha logrado implementar un esquema de autentificación que permite registrar con precisión las entradas y salidas del personal, identificar con certeza a la persona entra a las instalaciones, lograr revertir en corto tiempo la cartera vencida de sus clientes, y contar con un mejor control de visitantes. Tiene como elementos para su solución:

- Control biométrico, con terminales de huella dactilar para garantizar que no exista suplantación de identidad y con tecnología que puede almacenar hasta 50 mil personas con 2 huellas cada una, con capacidad de enlace a su red local, para instalar proyectos con cientos de terminales biométricas, si es necesario.

Sistema EP–300 para el control de ingreso y salida del personal de la Empresa INBIOSYS (Medellín, Colombia):

El Sistema EP-300 controla la entrada, la salida, los tiempos y la asistencia de todo el personal de la Empresa INBIOSYS. Es un equipo que tiene capacidad de 2.000 huellas digitales, almacena hasta 50.000 registros, viene con pantalla LCD, teclado, puertos USB y una red TCP/IP. El sistema de control de tiempos y asistencia EP-300 viene con un completo programa en español que permite configurar los horarios de los empleados de la Empresa INBIOSYS, emitir reportes y exportar la información en archivos planos a otros programas de nómina (14).

El sistema incluye un *software* que permite producir los reportes de las horas laboradas por cada uno de los funcionarios, crear horarios y turnos, controlar ausencias y producir un archivo plano con las horas laboradas. Éste *software* se instala en un computador normal, donde se procesan las horas laboradas. Con el *software* de gestión se pueden monitorear cada uno de los equipos, cargar las huellas y la información de los funcionarios, descargar

cada uno de los registros de asistencia de los diferentes equipos o terminales para producir los informes (14).

El software de gestión permite (14):

- La configuración del sistema.
- Configurar parámetros del administrador.
- La comunicación de datos.
- Agregar equipos (terminales) al sistema.
- Configurar cada equipo (terminal).
- Crear departamentos.
- Permite procesos de inicialización, sincronización de tiempos y copias de seguridad.
- Manejo de personal: adicionar, modificar y eliminar.
- Permite la configuración de privilegios, para que un funcionario esté autorizado en una o dos terminales, y de esta forma puede registrar la entrada en una sede y la salida en otra.
- Configurar registro de exportación.
- Configurar la interfaz del usuario. Por medio de esta interfaz se pueden visualizar todos los eventos.
- Recoge los registros de asistencia de los diferentes equipos (terminales), realiza el cálculo de acuerdo a la configuración y finalmente genera los diferentes reportes.

Sistema K-20 que controla el ingreso, salida y el acceso de personas de la Empresa INBIOSYS (Medellín, Colombia).

Este sistema brinda una buena confiabilidad y precisión al momento del registro. Su velocidad de verificación es menor a 2 segundos y permite registrar los accesos mediante tres opciones: huella digital, tarjeta y contraseña de manera independiente o combinada (15).

Cuenta con una pantalla a color TFT, pudiendo así mostrar de manera óptima la información del dispositivo, incluyendo la calidad de imagen de la huella digital y el resultado de la verificación (15).

Su comunicación estándar es a través del puerto TCP/IP, logrando un monitoreo en tiempo real de las marcaciones del personal, tiene la capacidad de abrir puertas y conectar un

botón de salida, convirtiéndolo en la mejor opción de control de entrada y acceso de toda Colombia (15).

El Sistema incluye un *software* que permite producir los reportes de las horas laboradas por cada uno de los funcionarios, crear horarios y turnos, controlar ausencias y producir un archivo en texto plano con las horas laboradas. Éste *software* se instala en un computador normal, donde se procesan las horas laboradas por los empleados. Con el *software* de gestión se pueden monitorear cada uno de los equipos, cargar las huellas y la información de los funcionarios, descargar cada uno de los registros de asistencia de los diferentes equipos (terminales), para producir los informes (15).

1.8.2 Soluciones existentes en Cuba

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

Sistema biométrico para el control de asistencia de los empleados de la Empresa de Telecomunicaciones de Cuba

El Sistema Biométrico de Control de Asistencia de los empleados de la Empresa de Telecomunicaciones de Cuba (ETECSA) tiene un control estricto del personal que labora en la misma debido a que controla la entrada/salida del personal que labora en las diferentes instalaciones de la empresa y dentro de sus funciones permite emitir reportes relativos a la asistencia, el tiempo de trabajo, los ausentes, etc. Este sistema utiliza Fingerprint SDK como software para la comparación de huellas dactilares, que, por su fácil implementación, por tener soportes para docenas de lenguajes de programación y no utilizar ningún sistema de bases de datos en particular determina la efectividad de la identificación del empleado mediante su huella dactilar.

El sistema permite además llevar los registros de horarios de asistencia, los datos de cada trabajador y la elaboración de reportes útiles que permiten la gestión eficiente de la asistencia del capital humano por parte de la empresa.

GeReport: Sistema de Gestión de Reportes Dinámicos de la Universidad de Cienfuegos

GEReport es una herramienta destinada al diseño, generación y configuración de los reportes relacionados con los datos históricos almacenados en una fuente de datos. El sistema se creó sobre un entorno web y se desarrolló siguiendo lo establecido por el Proceso Unificado de Desarrollo (RUP), utilizando UML como lenguaje de modelado. Para la gestión de la base de datos se seleccionó *PostgreSQL*, en la implementación se utilizaron los lenguajes de programación *PHP*, con *CodeIgniter 2.0* como marco de trabajo del lado del servidor y JavaScript con *Dojo Toolkit 1.8* para el trabajo del lado del cliente. Se utiliza en la Universidad de Cienfuegos por analistas y programadores del Grupo de Estudios y Desarrollo de Ingeniería y Sistemas, perteneciente a la Facultad de Ingeniería.

1.8.3 Soluciones existentes en la UCI

Fingerprint Attendance System Versión 4.8.8

Este sistema trabaja con el lector de huellas dactilares ZKTeco F702-S, que se encuentra instalado en la oficina del Jefe de Departamento del CEDIN, el mismo presenta un rendimiento rápido y confiable. Es un *software* desarrollado para el sistema operativo Windows y dentro de sus funcionalidades se encuentran:

- Es compatible con los gestores de bases de datos *MySQL, Microsoft Server SQL y Oracle.*
- Permite gestionar múltiples empresas y múltiples usuarios.
- Incluye calendario y perfil privados, así como horarios de trabajo flexibles.
- Todos los eventos de asistencia se pueden definir en un calendario privado, incluidos los eventos ausentes.
- Puede almacenar huellas dactilares e imágenes faciales de los empleados para el reconocimiento en terminales biométricos.
- Soporta turnos diurnos, nocturnos, rotativos y flexibles.

- Permite evaluaciones de nómina por turno, tiempos de asistencia obligatorios, períodos de validez de eventos y soporte para días laborables de más de 24 horas.
- Posibilita crear horarios de cambio estimados o fijos.
- Los calendarios públicos se pueden definir y luego personalizar para cada empleado.
- Permite establecer las fechas de vacaciones estándar anuales y los tipos de días establecidos para todos los cálculos de nómina.

A partir del estudio realizado sobre los sistemas de control de presencia biométrico existentes tanto en el ámbito internacional como en el nacional queda evidenciado que estos sistemas usan dispositivos con mejores prestaciones que el utilizado para el desarrollo de la propuesta de solución son propietarios, lo que implica una gran inversión económica para el Centro de Informática Industrial. Además, se suman otros factores que evidencian que no son factibles de utilizar para dar solución a la situación problemática, entre los que se pueden mencionar la utilización de tarjetas de proximidad y puertas electrónicas.

Aunque la integración con varias tecnologías son formas de autenticación más fiable, necesitan de otras tecnologías con las que no se cuentan para el desarrollo del sistema y estos sistemas no permiten la generación de reportes personalizados como se desea en este caso para generar el reporte de hoja de firma del Centro de Informática Industrial.

1.9 Herramientas y Tecnologías

Para el desarrollo de la presente investigación se requiere de la utilización de diversas herramientas y tecnologías necesarias para la confección de la solución. Entre estas herramientas se encuentran las bibliotecas que facilitan el desarrollo, el Entorno de Desarrollo Integrado que se utilizará para programar la solución y el sistema gestor de bases de datos que facilitará la gestión y el almacenamiento de la información que se genera a lo largo del proceso de desarrollo del sistema. Estas herramientas se utilizarán para el desarrollo de la propuesta de solución. A continuación, se mencionan las herramientas y tecnologías empleadas para el desarrollo de la solución propuesta.

1.9.1 Lenguaje de programación.

Los lenguajes de programación son la vía de comunicación entre el hombre y la computadora. Un lenguaje de programación es una técnica estándar de comunicación que

permite expresar las instrucciones que han de ejecutarse en una computadora. Consiste en un conjunto de reglas sintácticas y semánticas que definen un programa informático (Lenguajes de Programación, 2010). A continuación, se describen las principales características del lenguaje de programación elegido para el desarrollo de la aplicación:

1.9.2 Lenguaje de programación C++

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

Posteriormente se añadieron facilidades de programación genérica, que se sumó a los otros dos paradigmas que ya estaban admitidos (programación estructurada y la programación orientada a objetos). Por esto se suele decir que el *C++* es un lenguaje de programación multiparadigma. Actualmente existe un estándar, denominado *ISO C++*, al que se han adherido la mayoría de los fabricantes de compiladores más modernos.

Una particularidad de *C++* es la posibilidad de redefinir los operadores y de poder crear nuevos tipos que se comporten como tipos fundamentales.

Las ventajas del lenguaje de *C++* son:

- **Difusión:** Al ser uno de los lenguajes más empleados en la actualidad, posee una gran comunidad de usuarios y existe gran cantidad de bibliografía dedicados a él.
- **Versatilidad:** C++ es un lenguaje de propósito general, por lo que se puede emplear para resolver cualquier tipo de problema.
- **Portabilidad:** El lenguaje está estandarizado y un mismo código fuente se puede compilar de diversas plataformas.
- **Eficiencia:** *C++* es uno de los lenguajes más rápidos en cuanto a la ejecución de programas.
- **Herramientas:** Existe una gran cantidad de compiladores, depuradores y librerías que se utilizan para este lenguaje.

1.9.3 Marco de Trabajo (*Framework*).

Para el desarrollo de software se hace necesario la utilización de un marco de trabajo o *framework* (en inglés). Se consideran una estructura conceptual y tecnológica de soporte definido, con la cual un proyecto de *software* puede ser fácilmente organizado y desarrollado. Un *framework* puede incluir soporte de programas, bibliotecas, lenguaje interpretado, entre otras herramientas, para así ayudar a desarrollar los diferentes componentes de un proyecto. Su utilización tiene como ventajas garantizar la reutilización del código, minimizar los riesgos y los costos de producción y mejorar la calidad del producto final. A continuación se describe el marco de trabajo o *framework* definido por los directivos del CEDIN para la realización de la propuesta de solución.

1.9.4 Framework de desarrollo Qt

El *framework* utilizado para el desarrollo de la propuesta de solución es *Qt* en su versión 5.11.0, el cual es multiplataforma para el desarrollo de aplicaciones; dichas aplicaciones pueden realizarse con o sin interfaz gráfica de usuario (16). Para el desarrollo de la propuesta de solución se hace uso de este *framework* debido a que utiliza el lenguaje de programación C++ de forma nativa y es precisa para realizar todas las interfaces gráficas necesarias.

Qt es un *framework* multiplataforma orientado a objetos ampliamente usado para desarrollar software que utilicen interfaz gráfica de usuario, así como también diferentes tipos de herramientas para la línea de comandos y consolas para servidores que no necesitan una interfaz gráfica de usuario. Incluye un amplio conjunto de componentes gráficos que proporcionan las funcionalidades estándar de interfaz gráfica de usuario, introduce una innovadora alternativa para la comunicación entre objetos, llamados señales y ranuras. Puede soportar las funcionalidades de interfaz de usuario que requieren las aplicaciones modernas, tales como menús, menús contextuales, arrastrar, soltar, y barras de herramientas. Propone el patrón de diseño Modelo/Vista para la representación gráfica de los datos, con la separación de las funcionalidades introducidas por esta arquitectura. El marco de trabajo *Qt* ofrece a los desarrolladores una mayor flexibilidad para personalizar la presentación de elementos y proporciona una interfaz de modelo estándar que permite una amplia gama de fuentes de datos.

1.9.5 Entorno de Desarrollo Integrado (IDE).

Para el desarrollo de la propuesta de solución se requiere de un Entorno de Desarrollo Integrado o *IDE* (por sus siglas en inglés, *Integrated Development Environment*), el cual es un entorno de programación que ha sido empaquetado como un programa de aplicación, es decir, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica. Proveen un marco de trabajo amigable para la mayoría de los lenguajes de programación tales como *C++, PHP, Python, Java, C#, Delphi, Visual Basic*, entre otros. (fergarciac, 2013)

Existen diversos entornos integrados de desarrollo (*IDE*) tales como, *NetBeans, MS Visual Studio, Visual C++, Eclipse, Qt Creator*, entre otros.

1.9.6 Entorno de Desarrollo Integrado Qt Creator

El *IDE* seleccionado para el desarrollo de la propuesta de solución es *Qt Creator* en su versión 4.6.1. *Qt Creator* es un *IDE* multiplataforma para el desarrollo de aplicaciones que pueden o no tener una interfaz gráfica de usuario. Este se centra en proporcionar características que ayudan a los nuevos usuarios del *IDE* a aprender y comenzar a desarrollar rápidamente (17).

Existen otras características importantes que presenta este *IDE* como la disponibilidad de código fuente, la excelente documentación organizada que provee en el *Qt Assistant* y un editor para el diseño de formularios denominado *Qt Designer* (17).

Qt Creator cuenta con:

- Un editor de código con soporte para *C++*.
- Herramientas para la rápida navegación por el código.
- Resaltado de sintaxis y auto-completado de código.
- Soporte para refactorización de código.
- Paréntesis coincidentes y modos de selección.
- Plegado de código.
- Ayuda sensitiva al contexto.

Qt Creator es un entorno de programación que ha sido empaquetado como un programa de aplicación; es decir, está compuesto por un editor de código, un compilador, un

intérprete, un depurador, un cliente, un constructor de interfaces gráficas de usuario y ofrece la posibilidad de utilizar un sistema de control de versiones.

1.9.6 Sistema Gestor de Bases de Datos (SGBD).

Los Sistemas Gestores de Bases de Datos (SGBD) son un conjunto de programas cuya función principal es permitir el almacenamiento, la modificación y extracción de datos en una Base de Datos (BD), proporcionando un acceso controlado a la misma. Entre sus servicios se encuentran:

- Definición y creación de Bases de Datos.
- Manipulación de los datos realizando consultas, inserciones y actualizaciones.
- Acceso controlado a los datos mediante mecanismos de seguridad de acceso a los usuarios.
- Mantener la integridad de los datos.
- Controlar la concurrencia en las Bases de Datos.
- Mecanismos de copias de respaldo y recuperación para restablecer la información en caso de fallos en el Sistema (Ramírez, 2008).

Por parte de los directivos del CEDIN se decidió emplear para el desarrollo del sistema *SQLite* en su versión 3.10.

1.9.7 Sistema Gestor de Bases de Datos SQLite

SQLite es un sistema de gestión de bases de datos relacional, el cual es un proyecto de dominio público. A diferencia de los sistemas de gestión de bases de datos cliente-servidor, el motor de *SQLite* no es un proceso independiente con el que el programa principal se comunica. En lugar de eso, la biblioteca *SQLite* se enlaza con el programa pasando a ser parte integral del mismo. El programa utiliza la funcionalidad de *SQLite* a través de llamadas simples a subrutinas y funciones. Esto reduce la latencia en el acceso a la base de datos, debido a que las llamadas a funciones son más eficientes que la comunicación entre procesos. El conjunto de la base de datos (definiciones, tablas, índices, y los propios datos), son guardados como un sólo fichero estándar en la máquina host. Este diseño simple se logra bloqueando todo el fichero de base de datos al principio de cada transacción. En su versión 3, *SQLite* permite bases de datos de hasta 2 Terabytes de tamaño. Entre sus características más atractivas e importantes se encuentran (*SQLite*, 2011):

- **Tamaño:** *SQLite* tiene una pequeña memoria y una única biblioteca es necesaria para acceder a bases de datos, lo que lo hace ideal para aplicaciones de bases de datos incorporadas.
- **Rendimiento de base de datos:** *SQLite* realiza operaciones de manera eficiente y es más rápido que *MySQL* y *PostgreSQL*.
- **Portabilidad:** Se ejecuta en muchas plataformas y sus bases de datos pueden ser fácilmente portadas sin ninguna configuración o administración.
- **Estabilidad:** *SQLite* es compatible con *ACID*, reunión de los cuatro criterios de Atomicidad, Consistencia, Aislamiento y Durabilidad.
- **SQL:** Implementa un gran subconjunto de la *ANSI – 92 SQL7* estándar, incluyendo sub-consultas, generación de usuarios, vistas y *triggers*.
- **Interfaces:** Cuenta con diferentes interfaces del *API*, las cuales permiten trabajar con *Java, C++, PHP, Python, Ruby*.
- **Costo:** *SQLite* es de dominio público y por tanto, es libre de utilizar para cualquier propósito sin costo y se puede redistribuir libremente.

1.9.8 Sistema Gestor de Bases de Datos Microsoft Access

Microsoft Access es un programa utilizado en los sistemas operativos *Microsoft Windows* para la gestión de bases de datos, creado y modificado por *Microsoft* y orientado a ser usado en entornos personales o en pequeñas organizaciones. Es un componente del paquete de programas de *Microsoft Office*. Entre sus principales características se encuentran (18):

- Permite crear ficheros de bases de datos relacionales que pueden ser fácilmente gestionadas por una interfaz gráfica sencilla.
- Las bases de datos creadas pueden ser consultadas por otros programas.
- Facilita la administración de datos, ya que sus posibilidades de consulta y conexión le ayudan a encontrar rápidamente la información deseada, cualquiera que sea su formato o lugar de almacenamiento.
- Es posible producir formularios e informes sofisticados y efectivos, así como gráficos y combinaciones de informes en un solo documento.
- Permite lograr un considerable aumento en la productividad mediante el uso de los asistentes y las macros. Estos permiten automatizar fácilmente muchas tareas sin necesidad de programar.

- Permite manipular los datos en forma de tablas (formadas por filas y columnas), crear relaciones entre tablas, consultas, formularios para introducir datos e informes para presentar la información.

Con el uso de *SQlite* se garantiza tener contenida dentro de la misma aplicación la base de datos que incluye las plantillas de formularios necesarias para realizar el proceso de captura de datos. Además, al ser *SQLite* independiente, ya que no se comunica con un motor de base de datos, sino que las librerías de *SQLite* pasan a integrar la aplicación, se pueden gestionar las bases de datos sin necesidad de tener instalado o conectarse con un servidor de base de datos. Por estas razones, *SQLite* es el sistema gestor de base de datos que mejor se ajusta a la solución propuesta, empleándose fundamentalmente para la gestión de los usuarios de la aplicación. Además, *Microsoft Access* se emplea para la recolección de los datos que genera el lector óptico de huellas dactilares ZKTeco F702-S, debido a que el software con que cuenta este dispositivo utiliza este sistema gestor de bases de datos.

1.9.9 Administrador gráfico de Bases de Datos.

Como administrador gráfico de *SQLite* se decide utilizar *DB Browser for SQLite* que es una aplicación gratuita y de código abierto diseñada para facilitar la creación y administración de las bases de datos con *SQLite*. Mientras que para poder trabajar con estas bases de datos es necesario aprenderse una gran cantidad de comandos *SQL*, aumentando la probabilidad de que algo salga mal y hagamos que nuestra base de datos deje de funcionar correctamente.

DB Browser for SQLite funciona con una interfaz muy clara y sencilla de utilizar, similar basada en tablas como las que podemos encontrar en *Excel* de manera que tanto usuarios sin mucha experiencia en la creación y administración de bases de datos, como los desarrolladores más avanzados puedan trabajar cómodamente con sus bases de datos.

Algunas de las características que nos ofrece *DB Browser for SQLite* son:

- Permite crear archivos de bases de datos y compactar archivos ya creados con *SQLite*.
- Permite crear, definir y eliminar tablas.
- Permite crear, definir y eliminar índices.
- Permite buscar, editar, añadir o eliminar entradas.

- Cuenta con un potente buscador de entradas.
- Importa y exporta entradas en modo texto.
- Importa y exporta tablas en ficheros *CSV*.
- Importa y exporta bases de datos en volcados *SQL*.

1.9.10 Lenguaje de modelado.

Para el desarrollo de la propuesta de solución es necesario seleccionar un lenguaje de modelado que permita una visión constructiva del *software* y que se encuentre en correspondencia con la metodología de desarrollo a utilizar.

Un lenguaje para el modelado de objetos es un conjunto estandarizado de símbolos y de modos de disponerlos para modelar un diseño de *software*. El uso de un lenguaje de modelado es más sencillo que la auténtica programación, pues existen menos medios para verificar efectivamente el funcionamiento adecuado del modelo. Esto puede suponer también que las interacciones entre partes del programa den lugar a sorpresas cuando el modelo ha sido convertido en un *software* que funciona.

1.9.11 Lenguaje de modelado UML 2.1

Entre los lenguajes de modelado visual más conocidos se encuentra el Lenguaje de Modelado Unificado (*UML*, en inglés *Unified Model Language*), el cual se usa para entender, diseñar, hojear, configurar, mantener y controlar la información sobre un sistema de software. El lenguaje de modelado pretende unificar la experiencia pasada sobre técnicas de modelado e incorporar las mejores prácticas actuales en un acercamiento estándar. *UML* incluye conceptos semánticos, notación y principios generales. Tiene partes estáticas, dinámicas, de entorno y organizativas. Captura decisiones y conocimientos sobre los sistemas que se deben construir. Con la utilización de *UML* es posible llevar a cabo las siguientes funciones (Jacobson, 2000):

- **Visualizar:** Permite expresar de forma gráfica un sistema de forma que otro lo pueda entender.
- **Especificar:** Permite especificar cuáles son las características de un sistema antes de su construcción.
- **Construir:** A partir de los modelos especificados se pueden construir los sistemas diseñados.

- **Documentar:** Los propios elementos gráficos sirven como documentación del sistema desarrollado y pueden servir para su futura revisión.

La especificación de *UML* no define un proceso estándar, pero está pensado para ser útil en un proceso de desarrollo iterativo. Pretende dar apoyo a la mayoría de los procesos de desarrollo orientados a objetos. *UML* también contiene construcciones organizativas para agrupar los modelos en paquetes, lo que permite a los equipos de *software* dividir grandes sistemas en piezas de trabajo, para entender y controlar las dependencias entre paquetes, y para gestionar las versiones de las unidades del modelo, en un entorno de desarrollo complejo (19). Conocidas las funciones que trae consigo el empleo de un lenguaje de modelado se decidió, por parte del equipo de desarrollo, utilizar *UML* en su versión 2.1 para facilitar al programador la implementación de la propuesta de solución una vez que sea diseñado.

1.9.12 Herramienta CASE

Las herramientas de Ingeniería de Software Asistida por Computadoras, conocida por sus siglas en inglés como (*CASE*) de modelado con *UML* permiten representar el *software* mediante diagramas que se generan durante las diferentes etapas del proyecto. Entre las facilidades más apremiadas que proporcionan estas herramientas se encuentran: el diseño de proyectos, cálculo de costos, implementación automática de parte del código dado un diseño previo, compilación automática, documentación, detección de errores, entre otras. Por parte de los directivos del CEDIN se decidió emplear para el desarrollo del sistema la herramienta *CASE Visual Paradigm* en su versión 8.0.

Visual Paradigm es una poderosa herramienta *CASE* que utiliza *UML* (Lenguaje Unificado de Modelado) como lenguaje de modelado. Soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. Permite crear tipos diferentes de diagramas en un ambiente totalmente visual. Es muy sencillo de usar, fácil de instalar y actualizar. Genera código para varios lenguajes.

Posibilita la representación gráfica de los diagramas permitiendo ver el sistema desde diferentes perspectivas como: componentes, despliegue, secuencia, casos de uso, clase, actividad, estado, entre otros. Además, identifica requisitos y comunica información, se centra en cómo los componentes del sistema interactúan entre ellos, sin entrar en detalles excesivos. También permite ver las relaciones entre los componentes del diseño y mejorar

la comunicación entre los miembros del equipo usando un lenguaje gráfico. Soporta la realización de ingeniería tanto directa como inversa. Para el desarrollo de la propuesta de solución se utilizará la versión 8.0 de esta herramienta.

1.9.13 Bibliotecas para la Generación de Reportes

En los proyectos de informatización de los últimos años, la generación de reportes ha constituido una característica esencial, con gran peso para la toma de decisiones. Compañías de gran prestigio han invertido en desarrollar herramientas que, de una forma amigable, permiten al usuario personalizar sus reportes e informes (20). A continuación se muestran algunas de las bibliotecas de generación de reportes investigadas como parte del objeto de estudio para la generación del reporte de hoja de firma de los trabajadores del CEDIN.

LimeReport

LimeReport es una biblioteca multiplataforma de de generación de reportes escrita en *C++* usando el framework *Qt* y diseñada para desarrolladores de software que deseen agregar a su aplicación la capacidad de crear informes o imprimir formularios generados utilizando plantillas. El diseñador de informes incluido en la biblioteca permite crear o imprimir de forma rápida e intuitiva plantillas de formularios que pueden guardarse en formato *XML* y usarse para generar páginas de informes. Las páginas creadas pueden enviarse a una vista previa, a un archivo *PDF* o a una impresora. Como desarrollador, se puede usar una base de datos o los datos que se pasan de la aplicación mediante la interfaz *QAbstractTableModel* del *framework Qt*. Además se pueden inicializar variables que estén disponibles como parámetros de solicitud en la base de datos. El objetivo de *LimeReport* es proporcionar a la aplicación de funcionalidades abundantes y, al mismo tiempo, es una herramienta fácil de usar para que la generación de informes sea utilizada incluso por usuarios inexpertos en las Tecnologías de la Información (21).

Principales características de LimeReport:

- Código compatible puro *Qt4 / Qt5*
- Soporte multiplataforma.
- Diseñador de informes incrustado.
- Vista previa incrustada.

- Configuraciones de fuente de datos externa e interna.
- Parámetros externos para solicitudes de base de datos.
- Varios tipos de bandas para un informe de cualquier complejidad.
- Encabezado y pie de página.
- Agrupación de datos (*GroupHeader, GroupFooter, Subdetail, SubdetailHeader, SubdetailFooter*).
- Funciones de agregación (*SUM, COUNT, AVG, MIN, MAX*).
- Elementos del informe: Texto, Geométrico (Línea, Elipsis, Rectángulo), Imagen.
- Grupos de elementos horizontales.
- *HTML* para formatear campos de entrada.
- *Scripts* para formatear datos de salida.
- Un ajuste automático de la altura de la banda.

CuteReport:

CuteReport es una solución de reportes fácil, potente y ampliable basada en el *framework Qt*. En general, *CuteReport* consta de dos partes: la biblioteca central y el diseñador de plantillas. Ambos son totalmente modulares y su funcionalidad se puede ampliar simplemente escribiendo módulos adicionales. Es totalmente abstracto de los datos utilizados y se puede usar como almacenamiento: sistema de archivos, base de datos, sistemas de control de versiones, etc. El objetivo del *CuteReport* es proporcionar un sistema de reportes potente, pero fácil de usar para usuarios sin experiencia y diseñadores de reportes (22).

Principales características de CuteReport:

- Varios orígenes de datos: base de datos *SQL*, texto, sistema de archivos, modelo de datos externo (*QAbstractTableModel*).
- Varios tipos de almacenamientos para mantener plantillas de informes y objetos de informes como imágenes, etc.: Sistema de archivos, GIT, base de datos SQL, almacenamiento integrado.
- Variedad de elementos de dibujo para construir un excelente informe: texto (Memo), Imagen, Código de barras, Arco, Gráfico, Acorde, Elipse, Línea, Empanada, Rectángulo.
- Fuentes de imagen: estática, conjunto de datos, almacenamiento.
- Número ilimitado de detalles dentro de un informe.

- Encabezados y pies de página.
- Funciones agregadas: *SUM, COUNT, AVG, MIN, MAX*.
- Parámetros que se pueden pasar desde una aplicación personalizada.
- El motor de secuencias de comandos de la aplicación completa para gestionar cualquier aspecto de la representación del informe.
- Unidades de medida soportadas: Milímetros y Pulgadas.
- Algunos complementos de *Designer* preinstalados: editor de *ReportProperty*, editor de páginas, editor de scripts, editor de conjuntos de datos, vista previa.
- Informes multilingües.
- Tabla de contenidos automática.
- Multiplataforma.
- Procesamiento de informes por lotes.

QtRPT:

QtRPT es un motor de informes de impresión fácil de usar escrito en *C++*. Permite combinar varios informes en un archivo *XML*. Para el campo tomado por separado, puede especificar alguna condición, según la cual este campo se mostrará con diferentes fuentes y color de fondo, etc. El proyecto consta de dos partes: la biblioteca de informes *QtRPT* y la aplicación de diseño de informes *QtRpt Designer*. El informe de archivo es un archivo en formato *XML*. El diseñador de informes facilita la creación de archivos de informes en el formato *XML*. Gracias a la biblioteca *Qt*, esta herramienta de reportes se puede utilizar para trabajar en los sistemas operativos *Windows, Linux y MacOS* (23).

Principales características de QtRPT:

- Los formatos de salida compatibles son: Impresora, *PDF* y *HTML*.
- Tipo universal de fuente de datos.
- Fuente de datos y Modelado visual de consultas *SQL*.
- Los elementos del informe son: Campo de etiqueta, Imágenes, Diagrama.
- Informe de página / Encabezado / Pie de página / Pie de grupo.
- Las funciones y matemáticas agregadas son: *AVG, SUM, COUNT*.
- Mostrar / ocultar campos por condiciones lógicas.
- Diagramas con datos manuales o con funciones agregadas.
- Campos de texto enriquecido.
- Control total de la creación de informes desde la aplicación del usuario.

- Desarrollado en Código *Qt4 / Qt5* puro.

La herramienta para generar reportes seleccionada para la realización de la propuesta de solución es *QtRPT* en su versión 1.5.2, desarrollada por el programador *Aleksey Osipov* para diferentes plataformas como Windows y Linux, por ser una biblioteca desarrollada completamente en el framework *Qt* y usando el lenguaje de programación C++, por lo que es compatible con las tecnologías que se emplean para el desarrollo de la propuesta de solución.

1.10 Metodología de desarrollo de software

La metodología de desarrollo de *software* constituye un enfoque estructurado para el desarrollo de *software* que incluye modelos de sistemas, notaciones, reglas, sugerencias de diseño y guías de procesos (24).

Particularmente, una metodología se basa en una combinación de los modelos de procesos genéricos para obtener como beneficio un *software* que solucione un problema. Adicionalmente, una metodología debería definir con precisión los artefactos, roles y actividades, junto con prácticas, técnicas recomendadas y guías de adaptación de la metodología al proyecto. La complejidad del proceso de creación de *software* es netamente dependiente de la naturaleza del proyecto mismo (25).

Una metodología es un conjunto integrado de técnicas y métodos que permite abordar de forma homogénea y abierta cada una de las actividades del ciclo de vida de un proyecto de desarrollo. Es un proceso de *software* detallado y completo. Las metodologías se dividen en dos grupos: **tradicionales o robustas** y **ágiles o ligeras**. A continuación, se muestra una breve comparación entre las metodologías ágiles y tradicionales (Hernández Sampieri, 1998):

Metodologías Ágiles	**Metodologías Tradicionales**
Especialmente preparados para cambios durante el proyecto. Proceso menos controlado, con pocos principios.	Cierta resistencia a los cambios. Proceso mucho más controlado, con numerosas políticas y normas.

El cliente es parte del equipo de desarrollo.	El cliente interactúa con el equipo de desarrollo mediante reuniones.
Grupos pequeños (menos de 10 integrantes) y trabajando en el mismo sitio.	Grupos grandes y posiblemente distribuidos.
Pocos artefactos.	Muchos artefactos.
Pocos roles.	Muchos roles.
XP, Open UP, SCRUM, Crystal, Feature-Driven Development (FDD), Adaptive Software Development, DSDM.	*RUP*, Métrica V3.

Tabla 4. Comparación entre las metodologías ágiles y las metodologías tradicionales.

A continuación, se describe la metodología de desarrollo de *software* que se emplea en el CEDIN para el desarrollo de soluciones para la industria, de manera que el proceso de desarrollo del *software* cumpla con la calidad requerida y se termine en el tiempo previsto para satisfacer al cliente con una aplicación que cumpla con todas sus expectativas.

Agile Unified Process (AUP) variante UCI

AUP (Proceso Unificado Ágil, en español) es una versión simplificada del Proceso Racional Unificado (*RUP*, en inglés). Descrito en una forma simple, fácil de entender y brinda un enfoque de desarrollo de *software* utilizando técnicas ágiles y conceptos del *RUP*. El Proceso Unificado (*Unified Process* o *UP*) es un marco de desarrollo de *software* iterativo e incremental (26).

Al no existir una metodología de *software* universal, ya que toda metodología debe ser adaptada a las características de cada proyecto (equipo de desarrollo, recursos, etc.) exigiéndose así que el proceso sea configurable. Se decide hacer una variación de la metodología *AUP*, de forma tal que se adapte al ciclo de vida definido para la actividad productiva de la UCI (26).

AUP-UCI divide el ciclo de desarrollo en 3 fases (26):

- **Inicio:** Durante el inicio del proyecto se llevan a cabo las actividades relacionadas con la planeación del proyecto. En esta fase se realiza un estudio inicial de la organización cliente que permite obtener información fundamental acerca del alcance del proyecto, realizar estimaciones de tiempo, esfuerzo y costo y decidir si se ejecuta o no el proyecto.
- **Ejecución:** En esta fase se ejecutan las actividades requeridas para desarrollar el *software*, incluyendo el ajuste de los planes del proyecto considerando los requisitos y la arquitectura. Durante el desarrollo se modela el negocio, se obtienen los requisitos, se elaboran la arquitectura y el diseño, se implementa y se libera el producto.
- **Cierre:** En esta fase se analizan tanto los resultados del proyecto como su ejecución y se realizan las actividades formales de cierre del proyecto.

AUP-UCI propone las siguientes disciplinas (26):

- **Modelado de negocio:** Es la disciplina destinada a comprender los procesos de negocio de una organización. Se comprende cómo funciona el negocio que se desea informatizar para tener garantías de que el *software* desarrollado va a cumplir su propósito.
- **Requisitos:** El esfuerzo principal en la disciplina Requisitos es desarrollar el modelo del sistema que se va a construir. Esta disciplina comprende la administración y gestión de los requisitos funcionales y no funcionales del producto.
- **Análisis y diseño:** En esta disciplina, si se considera necesario, los requisitos pueden ser refinados y estructurados para conseguir una comprensión más precisa de estos, y una descripción que sea fácil de mantener y ayude a la estructuración del sistema (incluyendo su arquitectura). Además, en esta disciplina se modela el sistema y su forma (incluida su arquitectura) para que soporte todos los requisitos, incluyendo los requisitos no funcionales. Los modelos desarrollados son más formales y específicos que el de análisis.
- **Implementación:** En la implementación, a partir de los resultados del Análisis y Diseño se construye el sistema.

- **Pruebas internas:** En esta disciplina se verifica el resultado de la implementación probando cada construcción, incluyendo tanto las construcciones internas como intermedias, así como las versiones finales a ser liberadas.
- **Pruebas de Aceptación:** Es la prueba final antes del despliegue del sistema. Su objetivo es verificar que el *software* está listo y que puede ser usado por usuarios finales para ejecutar aquellas funciones y tareas para las cuales el *software* fue construido.

Entre sus principales características se encuentran:

- Se obtiene una comprensión común cliente - equipo de desarrollo del alcance del nuevo sistema y definir una o varias arquitecturas candidatas para el mismo.
- Permite que el equipo de desarrollo profundice en la comprensión de los requisitos del sistema y en validar la arquitectura.
- Admite que el sistema desarrollado sea probado al completo en el ambiente de desarrollo.

AUP-UCI define actividades que el desarrollador debe realizar para construir, validar y entregar un *software* que satisfaga las necesidades de las partes interesadas. A partir de esta se obtendrán los artefactos necesarios para conformar la propuesta de solución.

Conclusiones parciales

En el presente capítulo se expusieron los elementos teóricos principales de la solución propuesta, arribando a las siguientes conclusiones:

- El análisis del proceso de control de asistencia laboral de los trabajadores del CEDIN permitió constatar la necesidad de desarrollar un sistema de control de presencia y generación de hoja de firma a través del empleo de la identificación biométrica por huella dactilar para la informatización de este proceso.
- Al analizar los principales conceptos relacionados con el dominio del problema se logró un mejor entendimiento del objeto de estudio definido.
- Al realizar un estudio de soluciones similares y analizar dichos sistemas se llegó a la conclusión de que es necesario implementar un sistema de control de asistencia en el Centro de Informática Industrial con el objetivo de informatizar las hojas de firma de los trabajadores a través del uso de la identificación biométrica por huella dactilar.
- Después de realizar un estudio de las herramientas y metodologías para el desarrollo de *software* se seleccionó como lenguaje de programación *C++*, utilizando como marco de trabajo *Qt* en su versión 5.11.0, empleando como entorno de desarrollo *Qt Creator* en su versión 4.6.1, *DB Browser for SQLite* para el sistema gestor de bases de datos *SQLite* en su versión 3.10.1 y otro sistema gestor de base de datos a emplear es *Microsoft Access* 2013. La herramienta *CASE* seleccionada fue *Visual Paradigm* en su versión 8.0 empleando el lenguaje de modelado *UML* en su versión 2.1 y la metodología de desarrollo de *software* AUP-UCI en su escenario número 4: Proyectos que modelan el negocio con Historias de Usuario.

Capítulo 2: Análisis y Diseño de la solución propuesta

En este capítulo se abordarán aspectos fundamentales relacionados con el diseño del sistema a desarrollar. Se modela la propuesta de solución exponiendo los conceptos del dominio de aplicación, la especificación de los requerimientos funcionales y no funcionales del sistema para la generación de hojas de firma de los trabajadores del CEDIN. Se realiza la representación de las historias de usuario y la interacción de los actores con el sistema, así como una descripción detallada de los mismos.

2.1 Modelo de dominio

Un Modelo de dominio es un artefacto de la disciplina de análisis, construido con las reglas de *UML* durante la fase de concepción, presentado como uno o más diagramas de clases y que contiene, no conceptos propios de un sistema de *software* sino de la propia realidad física. Los modelos de dominio pueden utilizarse para capturar y expresar el entendimiento ganado en un área bajo análisis como paso previo al diseño de un sistema, ya sea de *software* o de otro tipo. Similares a los mapas mentales utilizados en el aprendizaje, el modelo de dominio es utilizado por el analista como un medio para comprender el sector industrial o de negocios al cual el sistema va a servir (27).

El modelo de dominio es una representación de los conceptos clave de un área o problema. Se utilizará para la representación de los aspectos del mundo real o físico. A continuación, se muestra el modelo de dominio utilizado para la solución propuesta.

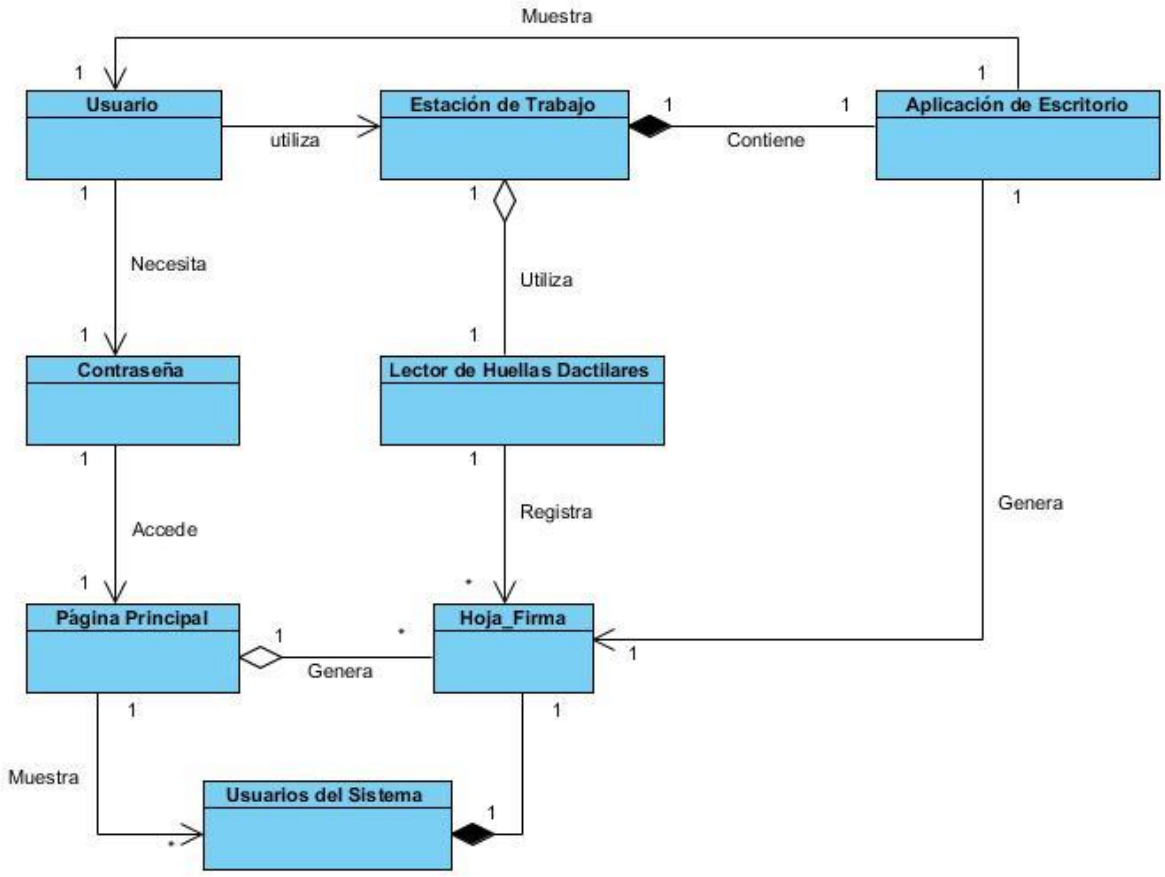


Figura 3. Modelo de dominio.

Descripción del modelo del dominio:

- **Estación de Trabajo:** Computadora con la que el usuario va a trabajar.
- **Aplicación de Escritorio:** Interfaz gráfica que permite visualizar la información que contiene el sistema que interactúa con el usuario.
- **Página Principal:** Página que inicialmente se muestra al usuario que se autentica con éxito en el sistema, permitiéndole realizar todas las operaciones que brinda el mismo.
- **Hoja de Firma:** Modelo que se utiliza para el control de la entrada/salida de los trabajadores y que recoge como información la fecha, hora de entrada y salida de cada trabajador al centro.
- **Usuario:** Persona que interactúa con el sistema que debe tener el rol Administrador para tener acceso a la Página Principal del sistema.
- **Contraseña:** Clave que solo el usuario conoce, para decidir si es quien dice ser.

- **Usuarios del Sistema:** Personas que están registradas en la base de datos del sistema.
- **Lector de Huellas Dactilares:** Dispositivo conectado a la estación de trabajo para realizar los proceso de captura y procesamiento de huellas dactilares.

2.2 Especificaciones de los Requisitos del Sistema.

Los requisitos del sistema no son más que la condición que debe ser alcanzada o poseída por un sistema o componente de un sistema para satisfacer un contrato, estándar u otro documento impuesto formalmente. Se denomina como: condición o capacidad que necesita un usuario para resolver un problema o lograr un objetivo, define qué es lo que el sistema debe hacer, para lo cual se identifican las funcionalidades requeridas y las restricciones que se imponen (Autores, 2009).

Partiendo de la descripción de los conceptos más importantes del sistema representados en el Modelo de Dominio, se determinaron los requisitos funcionales y no funcionales del sistema.

Requisitos Funcionales.

Los requisitos funcionales son declaraciones de los servicios que debe proporcionar el sistema, de la manera en que éste debe reaccionar a entradas particulares y de cómo se debe comportar en situaciones particulares (28). Los requisitos funcionales describen las funciones que el *software* va a ejecutar; por ejemplo, ajustarse a un formato de texto o modular una señal. Se conocen también como capacidades o funciones que el sistema debe cumplir. Para el desarrollo de la aplicación se identificaron los siguientes requisitos funcionales:

Requisitos funcionales	Descripción
1) Autenticar usuario.	El sistema debe permitir que se autentiquen los usuarios que tengan permisos de Administrador y categoría Trabajador.

2) Registrar usuario.	El sistema debe permitir que el Administrador registre los datos de un nuevo usuario en la base de datos del sistema.
3) Modificar usuario.	El sistema debe permitir que el Administrador modifique los datos de un usuario en la base de datos del sistema, excepto del usuario que está autenticado en ese momento.
4) Eliminar usuario.	El sistema debe permitir que el Administrador elimine los datos de un usuario en la base de datos del sistema, excepto del usuario que está autenticado en ese momento.
5) Mostrar usuario (s).	El sistema debe ser capaz de mostrar los datos de los usuarios registrados en la base de datos del sistema, así como información detallada de un usuario específico o del usuario que se encuentre autenticado en el sistema.
6) Buscar usuario	El sistema debe permitir la búsqueda de un usuario que se encuentre registrado en el sistema a través de su nombre de usuario.
7) Cambiar contraseña de usuario	El sistema debe permitir cambiar la contraseña de un usuario específico,

	excepto la del usuario que está autenticado en ese momento.
8) Exportar información de los usuarios.	El sistema debe ser capaz de exportar la información de los usuarios del sistema contenida en una tabla en formato *PDF* o *HTML*.
9) Abrir plantilla.	El sistema debe ser de capaz de abrir una plantilla en formato *XML* para luego mostrarla en una ventana emergente que brinda un conjunto adicional de funcionalidades como imprimir o exportar en formato *PDF* y *HTML*.
10) Configurar reporte de hoja de firma.	El sistema debe ser capaz de configurar el reporte de hoja de firma seleccionando al usuario, el mes y el año en que el Administrador desee para luego mostrar todos los datos que se gestionan en la hoja de firma.
11) Generar reporte de hoja de firma.	El sistema debe ser capaz de realizar la generación del reporte de hoja de firma de cada uno de los trabajadores del CEDIN en un mes y año especificado por el Administrador, tomando además el nombre, el número de expediente y el área a la que pertenece el trabajador.
12) Imprimir reporte de hojas de firma.	El sistema debe ser capaz de imprimir la hoja de firma del trabajador seleccionado por el Administrador.

13) Exportar reporte de hoja de firma.	El sistema debe ser capaz de exportar la información contenida en la hoja de firma generada por el sistema en formato *PDF* y *HTML*.

Tabla 5. Requisitos funcionales del sistema.

Requisitos No Funcionales (RNF)

Los requisitos no funcionales son restricciones de los servicios o funciones ofrecidas por el sistema. Incluyen restricciones de tiempo sobre los procesos de desarrollo y estándares. A menudo se aplican al sistema en su totalidad. Normalmente apenas se aplican a características o servicios individuales del sistema. Los requisitos no funcionales son los que actúan para obligar la solución. Se conocen a veces como apremios o requisitos de calidad. Pueden ser clasificados más a fondo según si son requisitos de funcionamiento, requisitos de capacidad de mantenimiento, requisitos de seguridad, requisitos de confiabilidad, o uno de muchos otros tipos de requisitos del *software*. A continuación se mencionan los requisitos no funcionales definidos para el sistema.

Requisitos de software:

- El software debe ser compatible con cualquier distribución de Sistema Operativo *GNU-Linux* y Windows *XP* en adelante.
- Debe estar instalado el driver para el lector óptico de huellas dactilares ZKTeco F702-S en la *PC*.
- El software debe mostrarse como una aplicación de escritorio.

Requisitos de hardware:

- **Hardware requerido para desplegar y utilizar la aplicación:** Ordenador con *CPU* compatible con *DualCore* o superior, con 1 GHz de velocidad de microprocesador o superior, con memoria *RAM* de 1 GB como mínimo y capacidad de espacio libre en disco duro de 100 MB como mínimo.
- Se necesita el lector óptico de huellas dactilares ZKTeco F702-S conectado al ordenador donde estará desplegada la aplicación.

Requisitos de usabilidad:

- La aplicación debe permitir la incorporación de nuevas funcionalidades que se necesiten agregar.
- La aplicación debe poseer una interfaz fácil de utilizar para cualquier tipo de usuario con conocimientos básicos de informática y en el manejo de ordenadores.

Requisitos de confiabilidad:

- El acceso a la información almacenada en la base de datos de la aplicación estará restringida por usuario, contraseña y por el rol de Administrador.
- La conexión con la base de datos del software del dispositivo ZKTeco F702-S se realizará a través del protocolo de comunicación *TCP/IP*.

Requisitos de soporte:

- El sistema debe funcionar para los sistemas operativos *Windows XP* en adelante y para cualquier distribución de sistemas *GNU/Linux*.

Requisitos de apariencia o Interfaz externa:

- El diseño de la aplicación será agradable para el usuario que lo utilice, respetando el diseño del estándar de codificación del Centro de Informática Industrial.
- El idioma utilizado en la aplicación será el español.
- Diseño sencillo, permitiendo la utilización del sistema sin mucho entrenamiento.
- Se debe informar al usuario sobre el estado de cualquier acción que ejecute en el sistema, informándole mediante notificaciones del resultado de estas, tanto si son mensajes de error, alerta o confirmación.

Requisitos de restricciones de diseño e implementación:

- La herramienta de desarrollo será el *IDE Qt Creator* en su versión 4.6.1.
- Se desarrollará el sistema utilizando el lenguaje *C++* con el framework *Qt* en su versión 5.11.0.
- Se utilizarán como gestor de base de datos *SQLite* en su versión 3.5 y *Microsoft Access* 2013.

- Se utilizará como herramienta *CASE* para la modelación *Visual Paradigm 8.0* empleado como lenguaje de modelado *UML* en su versión 2.1.

2.3 Historias de Usuario

Las Historias de Usuario (HU) son una técnica utilizada en AUP-UCI para especificar los requisitos del *software*. Por cada requisito funcional del sistema se realizó una historia de usuario, las cuales contribuyeron a realizar estimaciones y elaborar el plan de iteraciones. Cada historia de usuario que se muestra a continuación se definió lo más comprensible posible para que pueda ser implementada por el desarrollador.

Descripción de Historias de Usuario

- **Número:** Posee el número asignado a la Historia de Usuario.
- **Nombre de Historia de Usuario:** Atributo que contiene el nombre de la Historia de Usuario (el nombre del requisito funcional correspondiente).
- **Prioridad en el Negocio:** Evidencia el nivel de prioridad de la Historia de Usuario en el negocio.
- **Nivel de Complejidad:** Muestra el nivel de complejidad que presenta el requisito a implementar.
- **Tiempo Estimado:** Este atributo es una estimación hecha por el equipo de desarrollo del tiempo de duración de la Historia de Usuario. Cuando el valor es 1 equivale a una semana ideal de trabajo.
- **Iteración Asignada:** Número de la iteración en la cual se desarrollará la Historia de Usuario.
- **Programador responsable:** Persona o responsable encargado de desarrollar la implementación de la Historia de Usuario.
- **Descripción:** Posee una breve descripción de lo que realizará la Historia de Usuario.
- **Prototipo de Interfaz:** Prototipo del requisito del sistema a implementar.

Historia de Usuario	
Número: 1	**Nombre de HU:** Autenticar usuario

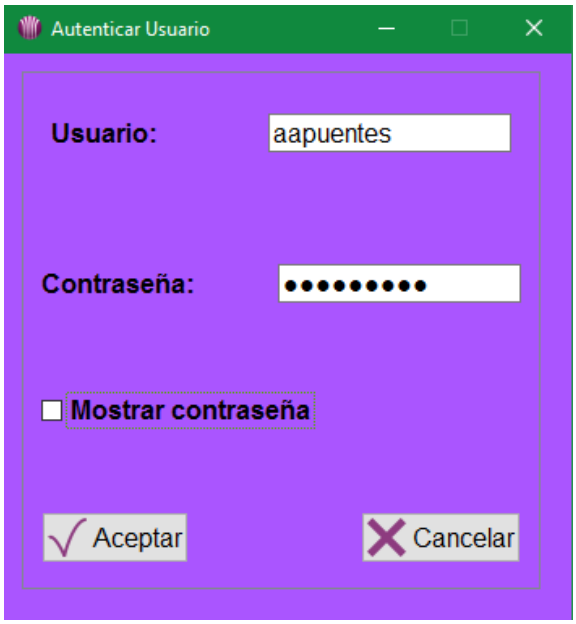
Prioridad en el Negocio: Alta	**Nivel de Complejidad:** Media
Tiempo Estimado: 2	**Iteración Asignada:** 1
Programador responsable: Daniel Miguel González Mora	
Descripción: Consiste en autenticar al usuario en el sistema, introduciendo su usuario y su contraseña para validar su entrada en el sistema.	
Observaciones: En caso de que la autenticación sea fallida, se notifica enviando un mensaje de error, y si es correcta el usuario entra al sistema satisfactoriamente. El sistema está organizado por roles, por lo que cada usuario tiene asignado un rol en el sistema. El rol Administrador es el único que puede realizar todas las operaciones que proporciona el sistema. Además, es necesario que el usuario con rol Administrador tenga asignada la categoría Trabajador.	
Prototipo de Interfaz: 	

Tabla 6. Historia de usuario – Autenticar usuario.

Historia de Usuario	
Número: 10	**Nombre de HU:** Configurar reporte de hoja de firma.
Prioridad en el Negocio: Alta	**Nivel de Complejidad:** Media
Tiempo Estimado: 2	**Iteración Asignada:** 2
Programador responsable: Daniel Miguel González Mora	
Descripción: Consiste en configurar el reporte de hoja de firma de forma tal que se selecciona al trabajador y el mes y año especificado por el usuario, generando una vista donde se muestra una tabla con el registro de los datos de entrada y salida contenidos en la base de datos del lector de huellas dactilares y nombre y apellidos, el número de expediente, la fecha seleccionada y el área que pertenece el trabajador seleccionado.	
Observaciones: Si no se selecciona a un trabajador en particular y la fecha seleccionada no es válida, no es posible configurar el reporte de hoja de firma y el sistema lo notifica a través de un mensaje de error.	

Prototipo de Interfaz:

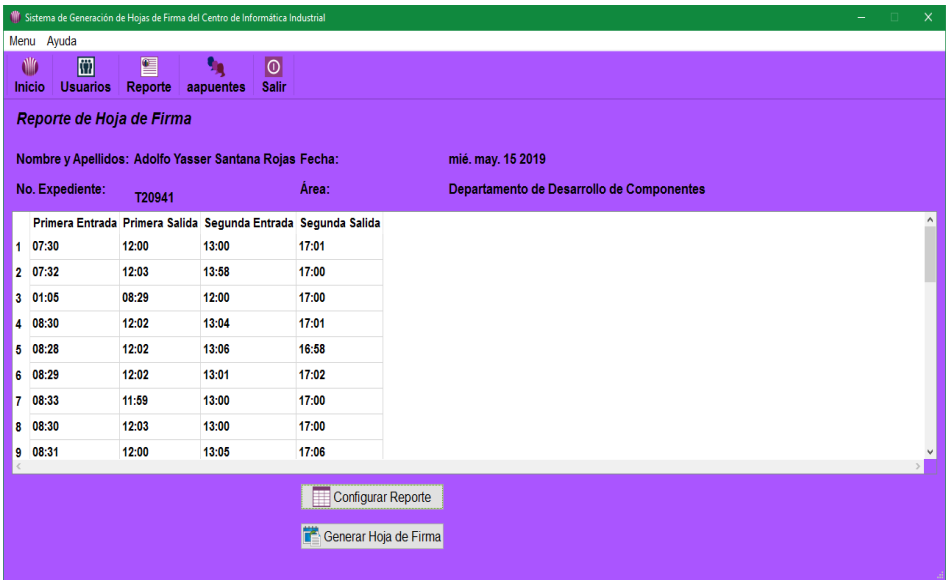


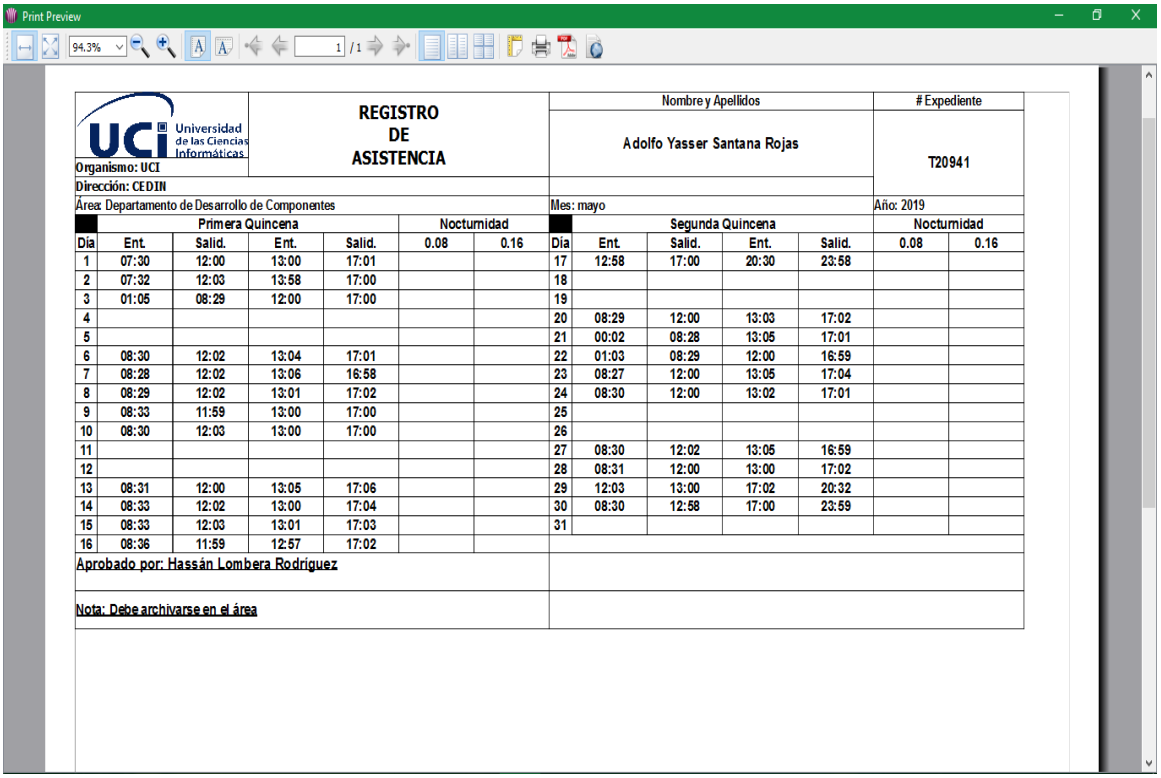
Tabla 7. Historia de usuario – Configurar reporte de hoja de firma.

Historia de Usuario	
Número: 11	**Nombre de HU:** Generar reporte de hoja de firma.
Prioridad en el Negocio: Alta	**Nivel de Complejidad:** Alta
Tiempo Estimado: 4	**Iteración Asignada:** 3
Programador responsable: Daniel Miguel González Mora	
Descripción: Consiste en generar una vista previa de la hoja de firma del trabajador seleccionado una vez configurado el reporte, mostrando su nombre y apellidos, su número de expediente, el organismo, dirección y área a la que pertenece dicho trabajador y el mes y el año seleccionado por el usuario.	

Observaciones:

No se puede generar el reporte de hoja de firma si antes no se ha configurado, la vista previa del reporte que se genera proporciona una serie de funcionalidades para configurar la forma en la que se desea mostrar el reporte, las dimensiones y opciones para imprimir y exportar la hoja de firma del trabajador seleccionado en formato *PDF* y *HTML*.

Prototipo de Interfaz:



Universidad de las Ciencias Informáticas

REGISTRO DE ASISTENCIA

Nombre y Apellidos	# Expediente
Adolfo Yasser Santana Rojas	T20941

Organismo: UCI
Dirección: CEDIN
Área: Departamento de Desarrollo de Componentes
Mes: mayo
Año: 2019

	Primera Quincena				Nocturnidad			Segunda Quincena				Nocturnidad	
Día	Ent.	Salid.	Ent.	Salid.	0.08	0.16	Día	Ent.	Salid.	Ent.	Salid.	0.08	0.16
1	07:30	12:00	13:00	17:01			17	12:58	17:00	20:30	23:58		
2	07:32	12:03	13:58	17:00			18						
3	01:05	08:29	12:00	17:00			19						
4							20	08:29	12:00	13:03	17:02		
5							21	00:02	08:28	13:05	17:01		
6	08:30	12:02	13:04	17:01			22	01:03	08:29	12:00	16:59		
7	08:28	12:02	13:06	16:58			23	08:27	12:00	13:05	17:04		
8	08:29	12:02	13:01	17:02			24	08:30	12:00	13:02	17:01		
9	08:33	11:59	13:00	17:00			25						
10	08:30	12:03	13:00	17:00			26						
11							27	08:30	12:02	13:05	16:59		
12							28	08:31	12:00	13:00	17:02		
13	08:31	12:00	13:05	17:06			29	12:03	13:00	17:02	20:32		
14	08:33	12:02	13:00	17:04			30	08:30	12:58	17:00	23:59		
15	08:33	12:03	13:01	17:03			31						
16	08:36	11:59	12:57	17:02									

Aprobado por: Hassán Lombera Rodríguez

Nota: Debe archivarse en el área

Tabla 8. Historia de usuario – Configurar reporte de hoja de firma.

Con la descripción de cada uno de los requisitos funcionales en historias de usuario se permite comprender mejor la respuesta que debe dar el sistema. Además, la realización de los prototipos de interfaz representa una guía visual de la forma en que será visualizado cada uno de los requisitos funcionales.

2.4 Diagrama de paquetes

Un diagrama de paquetes en el Lenguaje Unificado de Modelado representa las dependencias entre los paquetes que componen un modelo. Es decir, muestra cómo un sistema está dividido en agrupaciones lógicas y las dependencias entre las agrupaciones (29).

El diagrama de paquete muestra la vista general de la arquitectura, representa la relación entre la vista el modelo y el controlador. Teniendo en cuenta la arquitectura seleccionada, el controlador demanda información al modelo, y este a su vez responde con la información solicitada al controlador, enviándola a la vista.

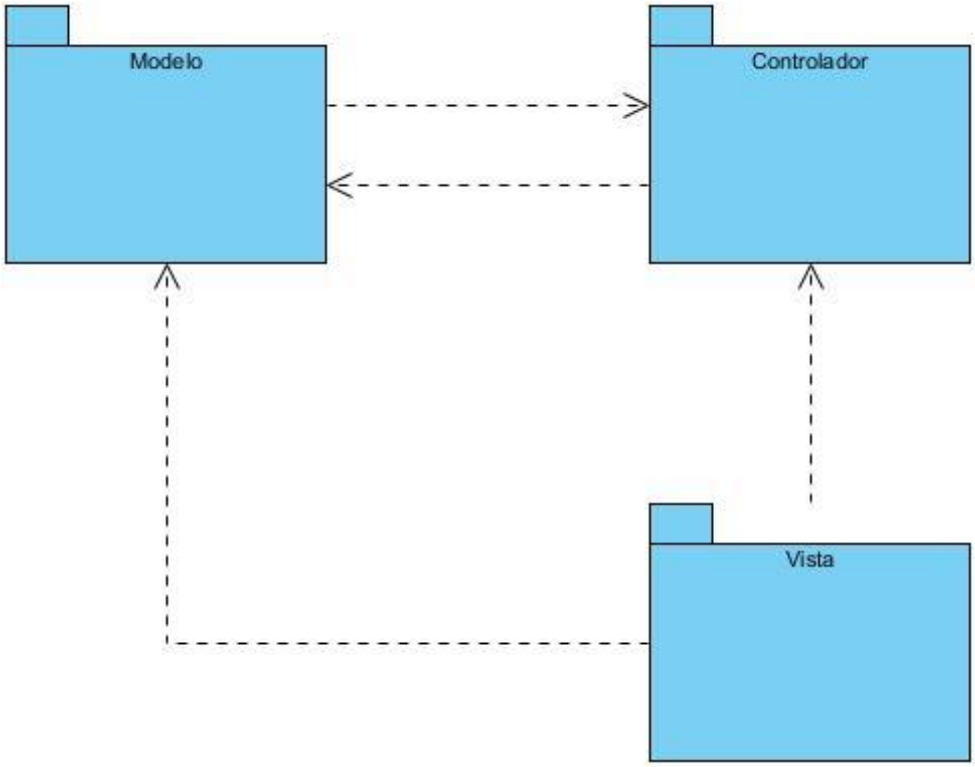


Figura 4. Diagrama de paquetes.

2.5 Diagrama de clases del diseño

Un diagrama de clases es utilizado para visualizar las relaciones entre las clases que involucran el sistema, las cuales pueden ser asociativas, de herencia, de uso y de agregación, ya que una clase es una descripción de conjunto de objetos que comparten los mismos atributos, operaciones, métodos, relaciones y semántica; mostrando un conjunto

de elementos que son estáticos, como las clases y tipos junto con sus contenidos y relaciones. Cuando se crea un diagrama de clases, se está modelando una parte de los elementos y relaciones que configuran la vista de diseño del sistema. A continuación se muestra el diagrama de clases del sistema teniendo en cuenta las entidades, sus atributos y relaciones.

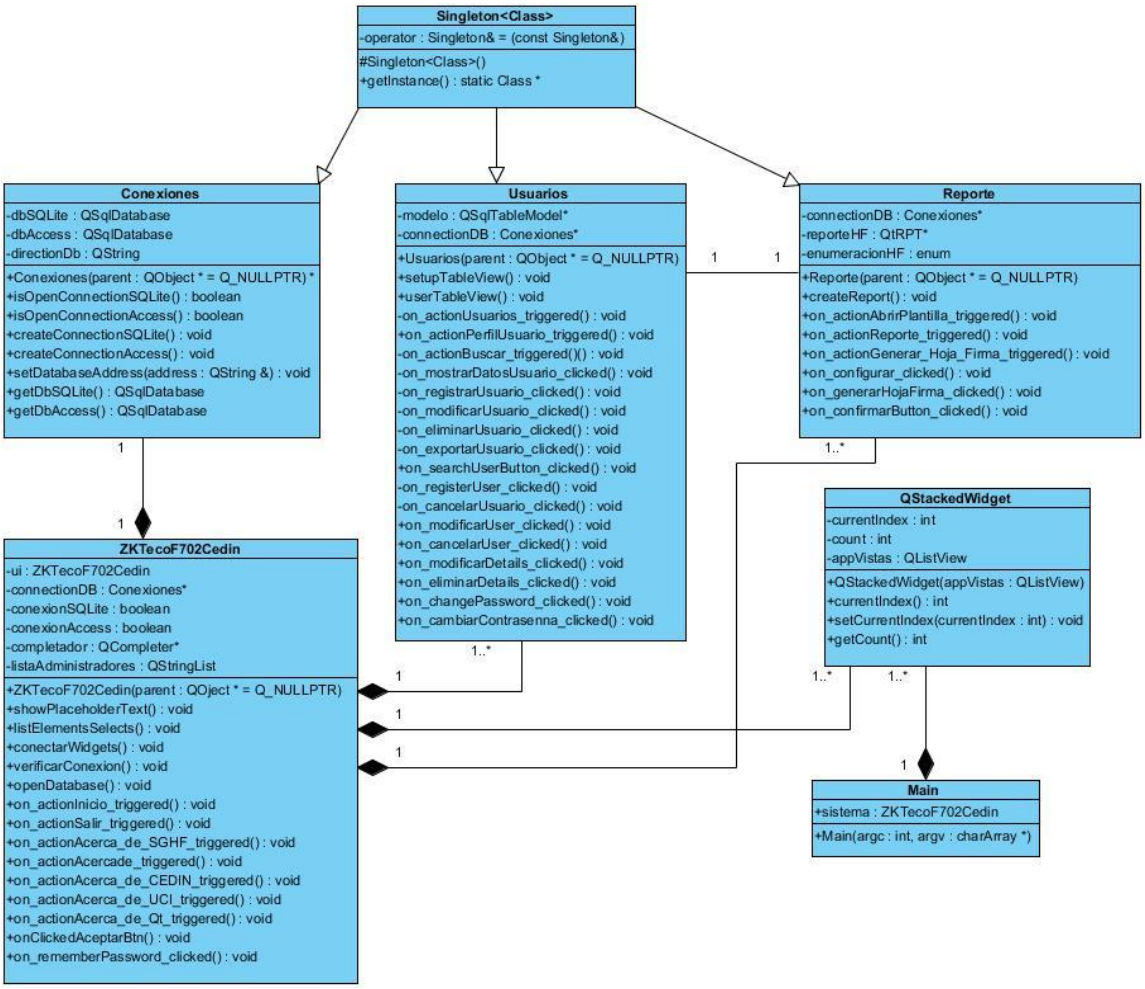


Figura 5. Diagrama de clases del diseño.

2.6 Arquitectura del Sistema

La arquitectura de *software* de un programa o sistema de cómputo es la estructura o estructuras del sistema, lo que comprende a los componentes del *software*, sus propiedades externas visibles y las relaciones entre ellos (28).

La arquitectura no es el sistema operativo, sino es una representación que permite analizar la efectividad del diseño para cumplir los requerimientos establecidos, considerar alternativas arquitectónicas en una etapa en la que hacer cambios al diseño todavía es relativamente fácil y reducir los riesgos asociados con la construcción del *software* (28). Es una vista estructural de alto nivel que ocurre muy tempranamente en el ciclo de vida del *software* y define los estilos o grupos de estilos adecuados para cumplir con los requerimientos no funcionales.

El *framework Qt* contiene un conjunto de vistas de elementos que utilizan una arquitectura Modelo/Vista para administrar la relación entre los datos y la forma en que se presentan al usuario. La separación de la funcionalidad introducida por esta arquitectura ofrece a los desarrolladores una mayor flexibilidad para personalizar la presentación de los elementos y proporciona una interfaz de modelo estándar para permitir el uso de una amplia gama de fuentes de datos con las vistas de elementos existentes.

2.7 Patrones de arquitectura

Los patrones de arquitectura son soluciones acertadas a un problema general, que establecen pautas para definir estructuras de diseño en el desarrollo de un *software*. Estos están más enfocados a los diseños orientados a objetos, pero según dichos patrones a menudo tienen en cuenta características de los objetos tales como la herencia y el polimorfismo que proporciona generalidad (30).

El patrón de diseño Modelo-Vista-Controlador (MVC) consiste en tres tipos de objetos. El Modelo es el objeto de la aplicación, la Vista es la presentación en pantalla y el Controlador define la forma en que la interfaz de usuario reacciona a la entrada del usuario. Antes de MVC, los diseños de interfaz de usuario tendían a agrupar estos objetos. MVC los desacopla para aumentar la flexibilidad y la reutilización.

Si los objetos de la vista y del controlador se combinan, el resultado es la arquitectura Modelo/Vista. Esta arquitectura separa la forma en que se almacenan los datos que se presentan al usuario, proporcionando un marco más sencillo basado en los mismos principios del patrón Modelo-Vista-Controlador. Esta separación hace posible mostrar los mismos datos en varias vistas diferentes, e implementar nuevos tipos de vistas, sin cambiar las estructuras de datos subyacentes. Para permitir un manejo flexible de las opiniones de los usuarios, se introduce el concepto de delegado. La ventaja de tener un delegado en

este marco es que permite que los elementos de datos se representen y editen para personalizarlos (31).

El patrón de arquitectura usado es el Modelo/Vista, debido a que es la arquitectura base que propone el marco de trabajo *Qt*, utilizado en el desarrollo de la misma, donde el modelo comunica con una fuente de datos, proporcionando una interfaz para los otros componentes en la arquitectura. La naturaleza de la comunicación depende del tipo de fuente de datos, y la forma en que se implementa el modelo. La vista obtiene índices de modelo; estos son referencias a objetos de datos. Mediante el suministro de los índices del modelo, la vista puede recuperar los elementos de datos del origen de datos. En vistas estándar, un delegado hace que cuando se edite un elemento, el delegado se comunica con el modelo usando directamente los índices de modelo (31).

En general, las clases del patrón Modelo/Vista se pueden separar en los grupos: modelos y vistas. Cada uno de estos componentes se define por las clases abstractas que proporcionan interfaces comunes y en algunos casos, las implementaciones por defecto de sus características. Las clases abstractas están destinados a ser una subclase con el fin de proporcionar el conjunto completo de funcionalidades esperadas por otros componentes; esto también permite que los componentes especializados puedan ser escritos.

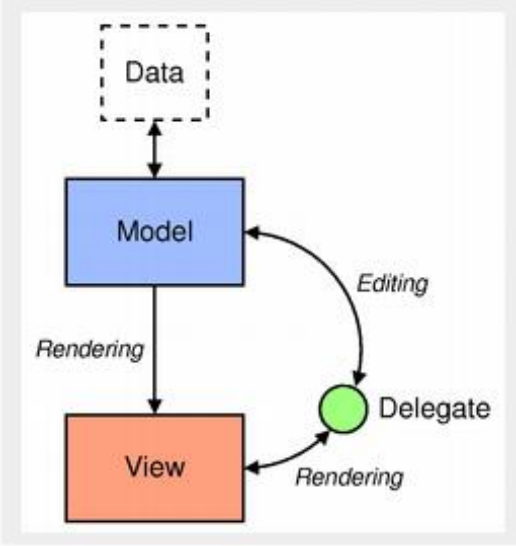


Figura 6. Representación del patrón arquitectónico Modelo/Vista.

2.8 Patrones de diseño

Los patrones de diseño tratan los problemas que se repiten y que se presentan en situaciones particulares del diseño, con el fin de proponer soluciones a ellas. Se encargan de identificar clases, instancias, roles, colaboraciones entre estas, así como la distribución de responsabilidades. En resumen, es una descripción de clases y objetos comunicándose entre sí, adaptada para resolver un problema de diseño general en un contexto particular (32). Estos patrones de diseño Banda de los Cuatro [*Gang of Four*, (*GOF* por sus siglas en inglés)] son clasificados principalmente por tres grupos como son:

- **Patrones de comportamiento:** Son utilizados a la hora de definir como las clases y objetos interaccionan entre ellos.
- **Patrones creacionales:** Son utilizados para instanciar objetos, y así separar la implementación del cliente de la de los objetos que se utilizan. Con ellos se intenta separar la lógica de creación de objetos y encapsularla.
- **Patrones estructurales:** Son utilizados para crear clases u objetos incluidos dentro de estructuras más complejas.

Para la asignación general de responsabilidades en el software existen patrones denominados Patrones Generales de Asignación de Responsabilidades (*GRASP*, por sus siglas en inglés), que describen los principios fundamentales de asignación de responsabilidades a objetos, expresados como patrones. Seguidamente se exponen algunos patrones utilizados dentro del contexto de los módulos a implementar (32):

- **Alta cohesión:** Facilita la solución al problema ¿Cómo mantener manejable la complejidad? mediante la asignación de responsabilidades de manera que la información que almacena una clase sea coherente y esté relacionada con la clase. El uso de este patrón se ve reflejado en todo el diagrama debido a que cada clase almacena la información de ella misma de manera coherente.
- **Experto:** Asigna una responsabilidad al experto en información, es decir, es la clase que cuenta con la información necesaria para cumplir la responsabilidad. De forma general en el diseño del sistema se basa en asignar a cada clase la responsabilidad que solo ellas pueden realizar, pues cada una cuenta con la información necesaria para llevarlas a cabo. Se utiliza este patrón en todas las clases del sistema.

- **Bajo acoplamiento:** Facilita la solución al problema ¿Cómo dar soporte a una dependencia escasa y a un aumento de la reutilización?
- **Creador:** Permite asignar el responsable de la creación de una nueva instancia de alguna clase. Explica que clase es la encargada de crear objetos, en determinados escenarios de ejecución y guía la asignación de responsabilidades relacionadas con la creación de objetos. El propósito general de este patrón es encontrar un creador que se debe conectar con el objeto producido en cualquier evento.

Conclusiones parciales

En el presente capítulo se realizó el proceso de análisis y diseño de la solución propuesta, arribando a las siguientes conclusiones:

- Para un correcto funcionamiento del sistema se realizó un levantamiento de requisitos, donde se identificaron 13 requisitos funcionales que deben ser cumplidos y 18 requisitos no funcionales que se deben tener en cuenta a la hora de desplegar el sistema.
- Las descripciones de las historias de usuario permitieron lograr una mayor comprensión de los requisitos funcionales que deben ser implementados.
- La arquitectura Modelo / Vista que propone el framework *Qt* permite un desarrollo desacoplado de fácil adaptación a cambios y la inclusión de nuevas funcionalidades.

Capítulo 3: Implementación y Prueba de la propuesta de solución

En el presente capítulo se describen las disciplinas de implementación y prueba de la propuesta de solución, realizando un modelado de las funcionalidades y la implementación de la aplicación propuesta. Se muestra el estándar de codificación utilizado para el desarrollo de la solución, así como el diagrama de componente y el diagrama de despliegue. Con el objetivo de comprobar la calidad del *software* se realizan pruebas funcionales al sistema de generación de hojas de firma para detectar posibles fallas en la aplicación a desarrollar. Finalmente, una vez corregidos los errores podrá ser desplegado el sistema para la generación de hojas de firma de los trabajadores del CEDIN a través de los datos que se obtienen de la base de datos del lector óptico de huellas dactilares ZKTeco F702-S para su posterior utilización.

3.1 Diagrama de componentes

Un diagrama de componentes representa la separación de un sistema de *software* en componentes físicos (por ejemplo, archivos, módulos, paquetes, base de datos, código fuente, binario o ejecutable) y muestra las dependencias y organización existente entre estos componentes. Los diagramas de componentes prevalecen en el campo de la arquitectura de *software*, pero pueden ser usados para modelar y documentar cualquier arquitectura de sistema (29).

Muestra la organización y las dependencias entre un conjunto de componentes. No es necesario que un diagrama incluya todos los componentes del sistema, normalmente se realizan por partes. El diagrama de componentes tiene en consideración los requisitos relacionados con la facilidad de desarrollo, la gestión de *software*, la reutilización, y las restricciones impuestas por los lenguajes de programación y las herramientas utilizadas en el desarrollo (Sparks, 2015) (Giraldo, Leyva, & Moreno, 2011). A continuación se muestra el diagrama de componentes de la solución propuesta.

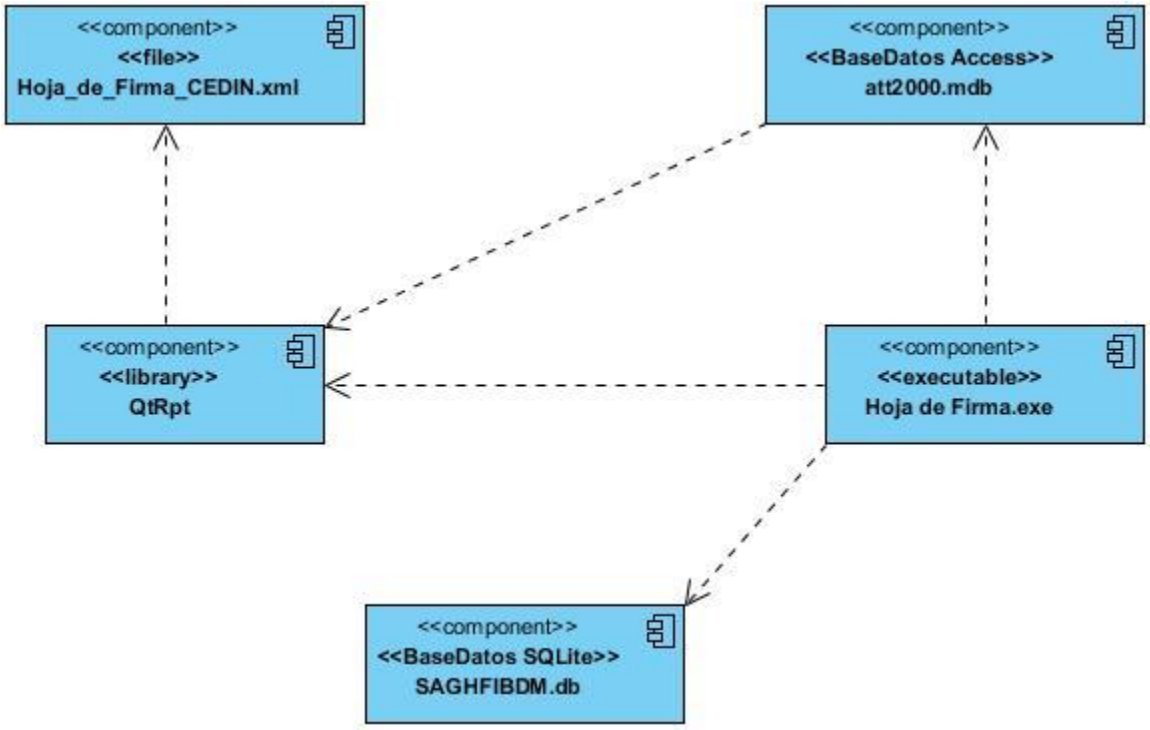


Figura 7. Diagrama de componentes.

Descripción del diagrama de componentes

Hoja de Firma CEDIN: Plantilla de hoja de firma de los trabajadores del CEDIN donde se registran todos los datos de entrada y salida que tuvo un trabajador durante un mes.

QtRpt: Biblioteca de generación de reportes utilizada para crear el modelo de hoja de firma de los trabajadores del CEDIN.

SGHFIBDM.db: Base de datos *SQLite* que registra toda la información referente a los usuarios registrados en el sistema.

Att2000.mdb: Base de datos *Access* que contiene toda la información del lector óptico de huellas dactilares ZKTeco F702-S, incluyendo las entradas y salidas de todos los trabajadores del CEDIN.

ZKTecoF702Cedin.exe: Archivo que se genera al compilar la aplicación y la ejecuta fuera del entorno de desarrollo de *Qt Creator*.

Diagrama de despliegue

Con el objetivo de proveer una descripción de la distribución física del sistema se realiza el modelo de despliegue, mediante el cual se muestran las relaciones físicas de los distintos nodos que componen el sistema. Los diagramas de despliegue son capaces de describir la arquitectura física del sistema durante la ejecución en términos de procesadores, dispositivos y componentes de *software*. Permite una mejor comprensión entre la correspondencia de la arquitectura de *software* y la arquitectura de *hardware* (33). A continuación se muestra el diagrama de despliegue asociado a la propuesta de solución.

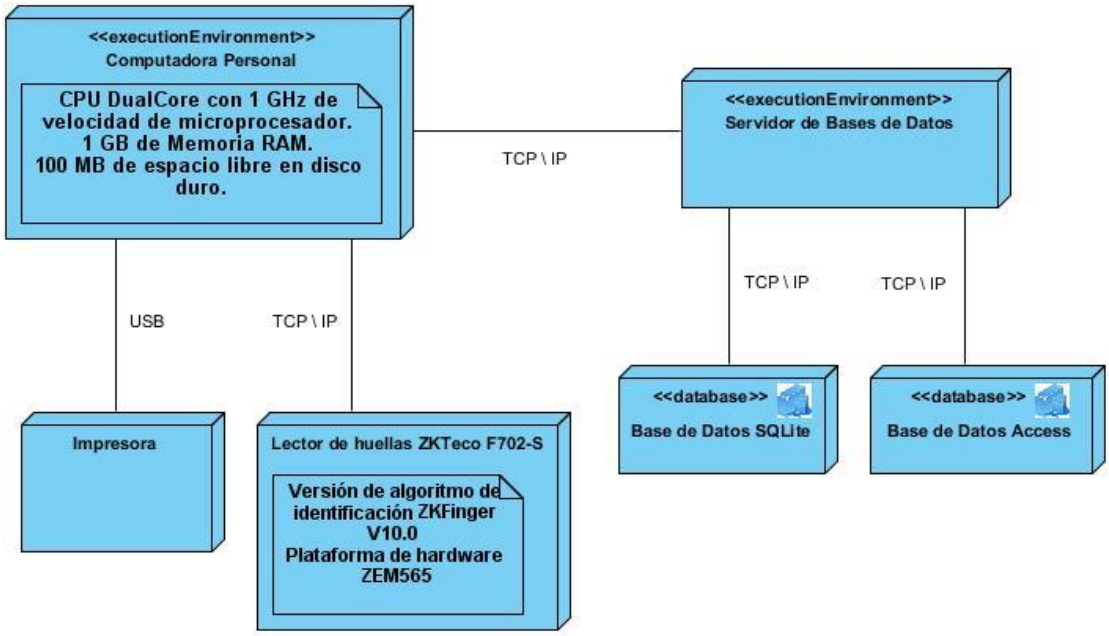


Figura 8. Diagrama de despliegue.

3.2 Estándar de codificación

Uno de los instrumentos que facilitan el trabajo y la calidad del software son la adopción de estilos y estándares de codificación. El uso de estos estándares tiene innumerables ventajas entre ellas lograr un estilo de código homogéneo asegurando su legibilidad y proveer una guía para el encargado del mantenimiento y actualización del sistema, con código claro y bien documentado. Además, ayuda a mejorar el proceso de codificación haciéndolo en gran medida eficiente y en muchos casos reutilizables (34). La solución propuesta en este trabajo utiliza el estándar de codificación definido por el proyecto. Algunas de las pautas definidas por el proyecto son:

- El código será escrito en inglés y la documentación en español.

- Las variables y funciones comienzan con letra minúscula. Cada palabra consecutiva en el nombre comienza con letra mayúscula.
- Los atributos de las clases deben empezar con letra inicial minúscula, en el caso de atributos compuestos, la inicial de la segunda palabra debe comenzar con mayúscula.
- Los parámetros que recibe una función deben empezar con letra inicial minúscula.
- Ninguna función debe tener más de 200 líneas.
- Los valores de los numerativos deben ser con letras mayúsculas.
- Las secciones *public*, *protected* y *private* son declaradas en el orden expuesto.

3.3 Solución del problema

En este epígrafe se muestran las interfaces del Sistema de Generación de Hojas de Firma del CEDIN, el cual integra al sistema la biblioteca de generación de reportes *QtRpt*, utilizada para la generación de la plantilla del reporte de hoja de firma de los trabajadores del CEDIN. En la siguiente imagen se muestra la ventana para autenticar a los usuarios, donde los únicos que tendrán acceso al sistema serán los usuarios que cumplan el rol de Administrador en el sistema y la categoría Trabajador.

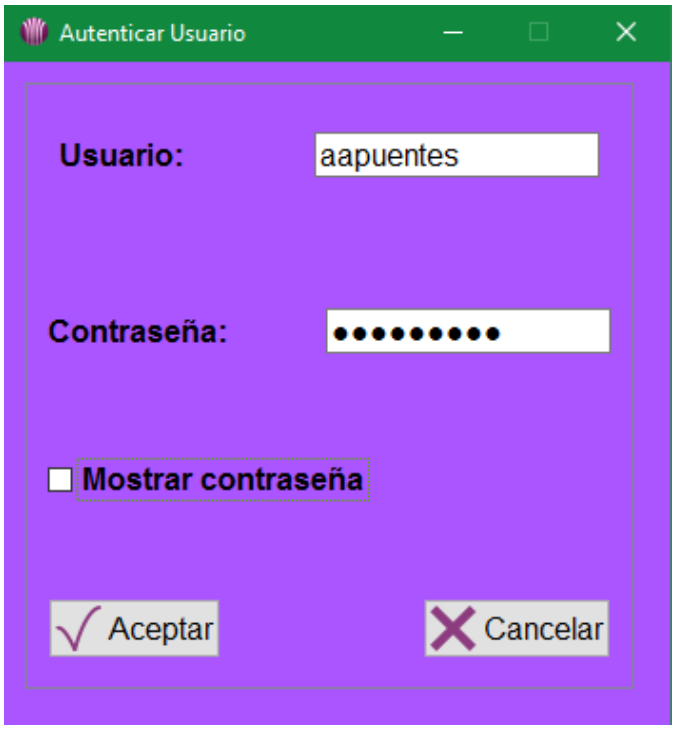
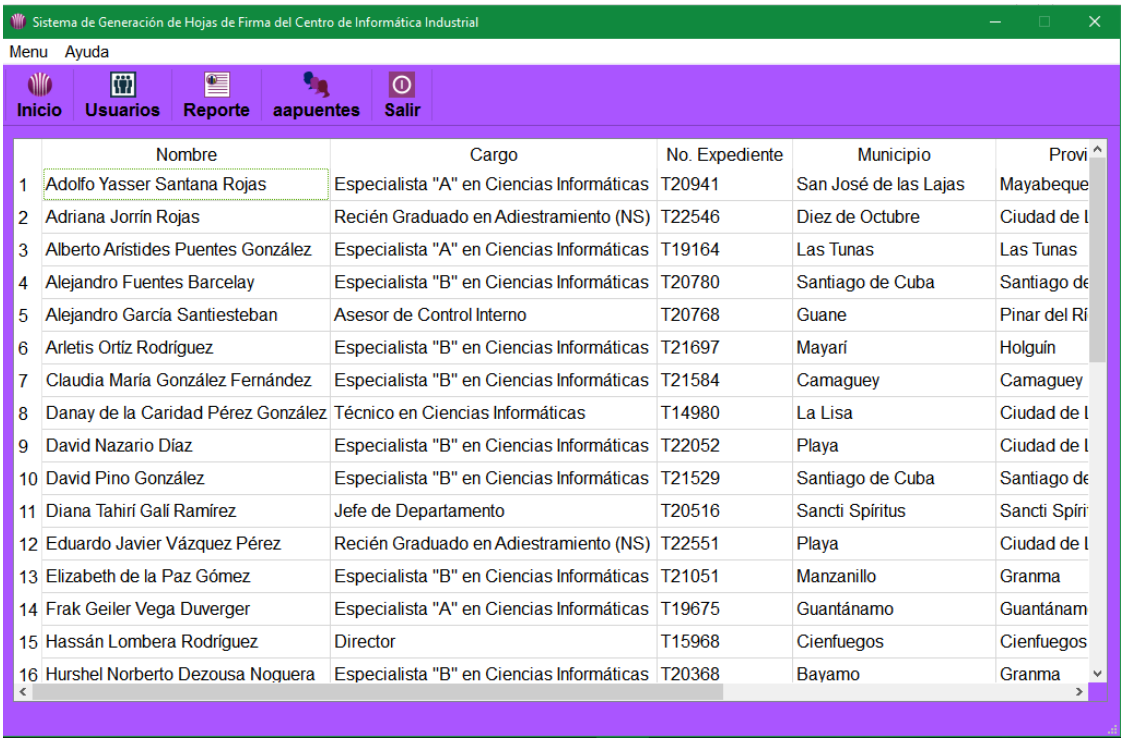


Figura 9. Ventana para la autenticación de los usuarios.

En la siguiente imagen se muestra la ventana principal del sistema en la que se encuentran todos los usuarios que están registrados en el sistema con algunos de sus datos como nombre y apellidos, cargo, número de expediente y área a la que pertenece. Además, brinda un conjunto de funcionalidades presentes en una barra de herramientas superior para acceder a esta ventana principal cuando el usuario lo desee, una opción para realizar la gestión de los usuarios del sistema, una opción para mostrar los datos personales del usuario autenticado en ese momento, una opción para configurar y generar el reporte de hoja de firma del trabajador que se seleccione en una fecha especificada por el usuario y una opción para salir del sistema, mostrando nuevamente la vista para autenticar usuarios. Además, presenta una barra de menú para acceder a otras funcionalidades que brinda el sistema y un menú que muestra información adicional acerca del sistema en cuestión, el lector óptico de huellas dactilares utilizado para el desarrollo de la aplicación, el CEDIN, la UCI y la versión del framework *Qt* con la que fue desarrollada la aplicación.



	Nombre	Cargo	No. Expediente	Municipio	Provi
1	Adolfo Yasser Santana Rojas	Especialista "A" en Ciencias Informáticas	T20941	San José de las Lajas	Mayabeque
2	Adriana Jorrín Rojas	Recién Graduado en Adiestramiento (NS)	T22546	Diez de Octubre	Ciudad de l
3	Alberto Arístides Puentes González	Especialista "A" en Ciencias Informáticas	T19164	Las Tunas	Las Tunas
4	Alejandro Fuentes Barcelay	Especialista "B" en Ciencias Informáticas	T20780	Santiago de Cuba	Santiago de
5	Alejandro García Santiesteban	Asesor de Control Interno	T20768	Guane	Pinar del Rí
6	Arletis Ortíz Rodríguez	Especialista "B" en Ciencias Informáticas	T21697	Mayarí	Holguín
7	Claudia María González Fernández	Especialista "B" en Ciencias Informáticas	T21584	Camaguey	Camaguey
8	Danay de la Caridad Pérez González	Técnico en Ciencias Informáticas	T14980	La Lisa	Ciudad de l
9	David Nazario Díaz	Especialista "B" en Ciencias Informáticas	T22052	Playa	Ciudad de l
10	David Pino González	Especialista "B" en Ciencias Informáticas	T21529	Santiago de Cuba	Santiago de
11	Diana Tahirí Galí Ramírez	Jefe de Departamento	T20516	Sancti Spíritus	Sancti Spíri
12	Eduardo Javier Vázquez Pérez	Recién Graduado en Adiestramiento (NS)	T22551	Playa	Ciudad de l
13	Elizabeth de la Paz Gómez	Especialista "B" en Ciencias Informáticas	T21051	Manzanillo	Granma
14	Frak Geiler Vega Duverger	Especialista "A" en Ciencias Informáticas	T19675	Guantánamo	Guantánam
15	Hassán Lombera Rodríguez	Director	T15968	Cienfuegos	Cienfuegos
16	Hurshel Norberto Dezousa Noguera	Especialista "B" en Ciencias Informáticas	T20368	Bayamo	Granma

Figura 10. Ventana principal del Sistema.

En la siguiente imagen se muestra una vista previa del reporte de hoja de firma para un trabajador especificado por el usuario donde se registran todas las entradas de ese trabajador que fueron registradas por el lector de huellas ZKTeco F702-S, con el nombre

completo del trabajador, su número de expediente, mes y año seleccionado por el usuario y otros datos necesarios para la hoja de firma. Esta vista previa brinda una serie de funcionalidades para configurar la forma en la que se muestra el reporte y las opciones para imprimir y exportar dicho reporte. De esta manera se da cumplimiento al problema planteado sobre la generación automática de las hojas de firma de los trabajadores del CEDIN mediante la utilización de la identificación biométrica por huella dactilar.

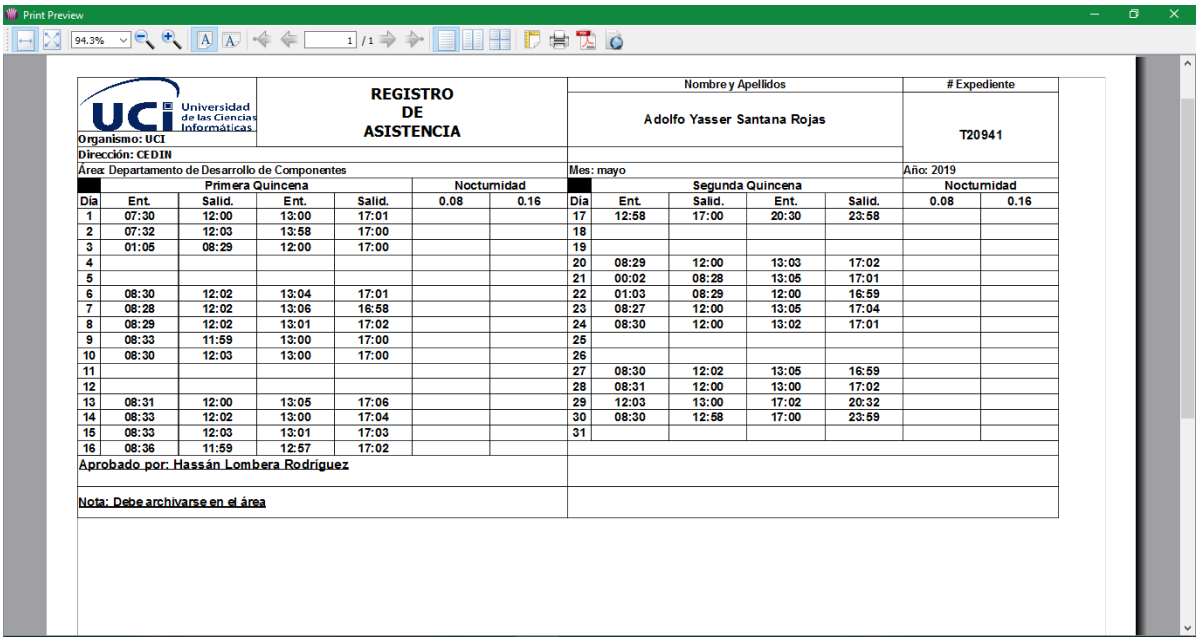


Figura 11. Vista previa del reporte de hoja de firma.

3.4 Pruebas del sistema

Las pruebas de *software* comprenden el conjunto de actividades que se realizan para identificar posibles fallos de funcionamiento, configuración o usabilidad de un programa o aplicación, por medio de pruebas sobre el comportamiento del mismo (35).

Las pruebas de *software* son muy importantes en la obtención de un producto de alta calidad y buen funcionamiento, por lo que su objetivo fundamental es verificar y validar que realmente el *software* realice lo que está planificado que haga. Forman un punto indispensable para cerciorarse de que un sistema es realizado con calidad y para reducir el número de errores que no fueron detectados en la etapa de implementación. Cuando se aplican pruebas a un *software* es necesario tener en cuenta el objetivo que se persigue, debido a que las pruebas son agrupadas por niveles que se encuentran en distintas etapas del proceso de desarrollo. Dentro de las estrategias de pruebas que existen se encuentran

las pruebas unitarias, las pruebas de integración, las pruebas de sistema, las pruebas de aceptación y las pruebas de regresión (Pressman, 2012).

Pruebas Unitarias: Pretenden probar que las unidades que forman el sistema cumplen las especificaciones y tienen el comportamiento esperado. Cumplen el objetivo de un componente de *software* (Método - Clase) (GAVIRIA, 2017). El primer nivel de las pruebas consiste en la verificación de unidades de *software* de forma aislada, es decir, probar el correcto funcionamiento de una unidad de código. Este tipo de prueba suelen ser realizadas por los desarrolladores, ya que es muy recomendable conocer el código fuente del programa y generalmente se realizan pruebas de caja blanca o se analiza el código para comprobar que cumple con las especificaciones del componente (Peño, 2015).

Pruebas de Integración: Los módulos o componentes funcionan integrados. Los datos enviados de un módulo o componente se reciben de manera consistente en el otro (GAVIRIA, 2017). Tienen el objetivo de verificar que las interfaces entre los usuarios y las aplicaciones funcionan correctamente. Son utilizadas para determinar cómo la base de datos de prueba será cargada. Verifican que las especificaciones de diseño sean alcanzadas. Algunas de sus características son:

- Describe cómo verificar que las interfaces entre las componentes de *software* funcionan correctamente.
- Determina cómo la base de datos de prueba será cargada.
- Determina el enfoque para avanzar desde un nivel de integración de las componentes al siguiente.
- Decide qué acciones tomar cuando se descubren problemas (Mera, 2016).

Pruebas de Sistemas: Los procesos soportados por la aplicación se cumplen completamente, es decir, los procesos fluyen desde su inicio hasta el final (GAVIRIA, 2017). Se utilizan para validar que todas las funciones y componentes del sistema trabajen correctamente. Estas pruebas se ejecutan una vez concluidas las pruebas unitarias y las de integración. A diferencia de estas dos pruebas, las pruebas de sistemas se realizan desde el punto de vista del usuario y validan en su totalidad los requerimientos del usuario especificados e incluso aquellos no especificados que afectan el funcionamiento del sistema (Mera, 2016).

Pruebas de Aceptación: Las pruebas de aceptación son realizadas con los clientes y define su aceptación del *software* (GAVIRIA, 2017).

Método de Caja Negra

Para la realización de las pruebas de la solución propuesta se empleó el Método de Caja Negra. Las pruebas de caja negra se refieren a las pruebas que se llevan a cabo en la interfaz del *software*. Una prueba de caja negra examina algunos aspectos fundamentales de un sistema con poca preocupación por la estructura lógica interna del *software* (Pressman, 2012).

La técnica de partición equivalente pertenece a las técnicas del método de caja negra consiste en dividir en subconjuntos equivalentes con respecto a una relación específica del dominio de las entradas. La prueba de un valor representativo de una clase permite suponer que el resultado obtenido será el mismo que para otro valor de la clase. Se realiza un conjunto representativo de casos de prueba para cada clase de equivalencia (Blanco, 2015).

3.4.1 Aplicación de las pruebas de caja negra. Casos de Prueba

Para comprobar el correcto funcionamiento del sistema de generación de hojas de firma se aplicó el método de caja negra empleando la técnica de partición equivalente, que permitió evaluar los valores de entrada al sistema. Los casos de prueba diseñados pretenden demostrar que el sistema funciona de manera correcta, las entradas válidas y no válidas se aceptan de forma adecuada y los resultados obtenidos son correctos. Para representar las pruebas de funcionamiento se definieron los siguientes elementos:

- **Número:** Representa al caso de prueba, incluye el número de HU, de la prueba y si posee diferentes escenarios.
- **HU:** Número de la HU a la cual pertenece.
- **Nombre:** Junto al código conforma el identificador del caso de prueba.
- **Descripción general:** Acción que debe realizar el sistema.
- **Condiciones de Ejecución:** Describe las características y elementos que debe contener el sistema para realizar el caso de prueba.
- **Entrada/Pasos de Ejecución:** Incluye las entradas necesarias para realizar el sistema, además de los pasos para realizar el caso de prueba.

- **Resultados Esperados:** Descripción de la respuesta del sistema ante el caso de prueba.
- **Evaluación de la Prueba:** Clasificación de la prueba en satisfactoria o insatisfactoria.

La siguiente tabla muestra las pruebas funcionales de la Historia de Usuario #1: Autenticar usuario, perteneciente a la primera iteración de sistema:

Caso de Prueba Funcional	
Número: 1	**No. Historia de Usuario:** 1
Nombre: Autenticar usuario	
Descripción: Comprobar que el usuario se autentique correctamente en el sistema.	
Condiciones de Ejecución: El usuario que desee autenticarse debe tener permisos de Administrador para poder entrar en el sistema.	
Entradas / Pasos de Ejecución: ➢ Introducir usuario y contraseña en los campos correspondientes. ➢ Presionar el botón de Aceptar para verificar que los campos fueron introducidos correctamente.	
Resultado esperado: El usuario se autentica correctamente en el sistema y se muestra un mensaje de confirmación y se accede a la página principal del sistema.	
Evaluación de la prueba: Prueba satisfactoria.	

Tabla 9. Caso de Prueba Funcional #1: Autenticar usuario.

Caso de Prueba Funcional

Número: 10	**No. Historia de Usuario:** 10
Nombre: Configurar reporte de hojas de firma.	
Descripción: Comprobar que el reporte de hoja de firma sea configurado correctamente para un trabajador específico y su registro de entrada y salida se muestren en una tabla.	
Condiciones de Ejecución: El usuario selecciona un trabajador y una fecha específica.	
Entradas / Pasos de Ejecución: ➢ Seleccionar uno de los trabajadores registrados en el sistema ➢ Seleccionar una fecha específica. ➢ Presionar el botón Confirmar. ➢ El sistema cambia de vista y muestra las entradas y salidas del trabajador seleccionado, en conjunto con su número de expediente, nombre y apellidos, fecha seleccionada y el área a la que pertenece dicho trabajador.	
Resultado esperado: El usuario configura el reporte completo de hoja de firma del mes y del trabajador seleccionado, mostrando todos los datos anteriores distribuidos en sus respectivas casillas.	
Evaluación de la prueba: Prueba satisfactoria.	

Tabla 10. Caso de Prueba Funcional #11: Configurar reporte de hoja de firma.

Caso de Prueba Funcional	
Número: 11	**No. Historia de Usuario:** 11
Nombre: Generación de reporte de hojas de firma.	

Descripción: Comprobar que el reporte de hoja de firma para un trabajador especificado por el usuario se genere correctamente desde el sistema.
Condiciones de Ejecución: El usuario debe haber configurado antes el reporte antes de poder generar la hoja de firma del trabajador seleccionado.
Entradas / Pasos de Ejecución: ➢ Extraer los datos de las entradas y salidas del trabajador seleccionado. ➢ Extraer nombre y apellidos, fecha, número de expediente y área a la que pertenece. ➢ Mostrar dichos datos en una tabla auxiliar. ➢ Presionar el botón Generar Hoja de Firma para generar una ventana emergente donde se muestra la plantilla de hoja de firma cargada con todos los datos necesarios.
Resultado esperado: El usuario genera el reporte completo de hoja de firma del mes y del trabajador seleccionado, mostrando todos los datos anteriores distribuidos en sus respectivas casillas.
Evaluación de la prueba: Prueba satisfactoria.

Tabla 11. Caso de Prueba Funcional #12: Generar reporte de hoja de firma.

Con las pruebas realizadas se detectaron 13 No Conformidades (NC) para un total de 13 historias de usuario. A continuación, se muestran los resultados de las pruebas funcionales realizadas al sistema:

No. NC	Requisito Funcional	Tipo de NC	Descripción	Iteración	Estado
1	RF1	Validación	El campo de la contraseña no cumple con los parámetros	1	Resuelta

			mínimos de seguridad (mayúsculas y números).		
2	RF2	Funcional	Al crear un usuario no se inserta correctamente en la base de datos.	1	Resuelta
3	RF3	Funcional	Al modificar un usuario no se actualizan los cambios en la base de datos.	1	Resuelta
4	RF4	Funcional	Al eliminar un usuario no se actualizan los cambios en la base de datos.	1	Resuelta
5	RF6	Validación	Al realizar la búsqueda de usuario no realiza ninguna acción.	1	Resuelta
6	RF8	Funcional	Al exportar la información de los usuarios el archivo no se genera correctamente.	2	Resuelta
7	RF10	Validación	Se configura el reporte aun así no esté seleccionado ningún trabajador.	2	Resuelta
8	RF10	Funcional	No se visualizan las entradas del usuario seleccionado.	2	Resuelta
9	RF10	Funcional	No visualiza el botón Configurar Reporte.	2	Resuelta

10	RF11	Funcional	Al generar la hoja de firma esta se encuentra vacía.	3	Resuelta
11	RF11	Funcional	Al editar una entrada manualmente fuera del formato establecido el sistema generaba la hoja de firma con estos errores.	3	Resuelta
12	RF11	Funcional	Al presionar el botón Generar Hoja de Firma no realiza ninguna acción.	3	Resuelta
13	RF11	No Funcional	El campo Fecha se muestra en el estándar numérico.	3	Resuelta

Tabla 12. Relación de no conformidades detectadas por requisito e iteración.

Una vez realizados todos los casos de pruebas para un total de 13 Historias de Usuarios, con tres iteraciones se eliminaron las 13 no conformidades encontradas.

Pruebas del Sistema	**Historias de Usuarios**	**No. Iteración**	**Cantidad de No Conformidades detectadas**	**Cantidad de No Conformidades resueltas.**
Sistema de generación de hojas de firma mediante identificación biométrica	13	1ra.	5	5
		2da.	4	4
		3ra.	4	4
		4ta.	0	0

Tabla 13. Resultados de las pruebas de caja negra.

Como resultado se detectaron que las 13 no conformidades encontradas fueron de tipo Funcionalidad y Validación que no realizaban la acción prevista, errores visuales donde el sistema no mostraba lo requerido y algunas validaciones que fueron realizadas. El resultado de las pruebas realizadas al software fue satisfactorio. Se comprobó que las respuestas fueran las esperadas y que se cumple con las especificaciones definidas a partir de los requisitos funcionales del sistema planteados con anterioridad.

Conclusiones parciales

Durante el desarrollo de este capítulo se expusieron los principales artefactos del modelo de implementación, arribando a las siguientes conclusiones:

- La realización del diagrama de componentes permitió mostrar los componentes del sistema y las relaciones existentes entre ellos.
- El diseño del diagrama de despliegue permitió comprender las relaciones físicas entre los componentes de *hardware* y *software* del sistema.
- Para la validación del correcto funcionamiento del *software* se realizaron pruebas al sistema mediante el método de caja negra utilizando la técnica de partición equivalente, donde se detectaron un total de 13 no conformidades en 4 iteraciones, siendo resuelta satisfactoriamente al final de cada una, contribuyendo a la calidad del producto final.
- La solución propuesta responde a las necesidades del cliente y de los usuarios finales.

.

Conclusiones generales

Con la culminación del presente trabajo se llegó a las siguientes conclusiones:

- El estudio de los sistemas de identificación mediante huellas dactilares permitió demostrar que es una de las técnicas biométricas más confiables desarrolladas en la actualidad, presentado tres de las características de un indicador biométrico, que son universalidad, unicidad y permanencia.
- Los requisitos funcionales y no funcionales garantizaron que la solución propuesta responde a las necesidades del cliente.
- Se obtuvo un sistema de generación automática de las hojas de firma que permitió un mejor control de la asistencia laboral de los trabajadores en el Centro de Informática Industrial.

Recomendaciones

Teniendo en cuenta los resultados obtenidos en el presente trabajo, se proponen las siguientes recomendaciones:

- Se recomienda añadir a la aplicación la funcionalidad de generar prenómina de los trabajadores, con el objetivo de automatizar este proceso.
- La utilización de la aplicación en los centros de producción de la Universidad para llevar a cabo un mejor control del horario de entrada y salida de los trabajadores.

Referencias Bibliográficas

1. Sánchez Redondo, Carlos. *Conectividad Bluetooh con sensores biométircos en dispositivos móviles.* Leganés : s.n., 2014. ISBN.

2. Maya Vargas, Adriana. *Sistema biométrico de reconocimiento de huella dactilar en control de acceso de entrada y salida.* Bogotá D.C : s.n., 2013. ISBN.

3. Blog CucoRent. Qué tecnología biométrica es la más segura? [En línea] https://www.cucorent.com/blog/que-tecnologia-biometrica-es-mas-segura/.

4. SystemPin. Control de empleados - Control de entradas y salidas y presencia. [En línea] https://www.systempin.com/control-presencia.

5. EmprendePyme. Control de asistencia laboral. [En línea] https://www.emprendepyme.net/control-de-asistencia-laboral.html.

6. GrupoRPP. GrupoRPP Noticias. *GrupoRPP Web site.* [En línea] 3 de Enero de 2017. https://rpp.pe/tecnologia/mas-tecnologia/asi-de-claro-que-es-la-identificacion-biometrica-noticia-1020941.

7. Biometria. *Preguntas frecuentes.* [En línea] http://www.biometria.gov.ar/acerca-de-la-biometria/preguntas-frecuentes.aspx.

8. Futronic_Finger. Futronic_Finger. [En línea] 30 de Septiembre de 2012. http:/article.wn.com.

9. UNAM-Facultad de Ingeniería Biometría Informática. Huella dactilar. [En línea] http://redyseguridad.fi-p.unam.mx/proyectos/biometria/clasificacionsistemas/recohuella.html.

10. Martínez Díaz, Marcos. *Vulnerabilidades en sistemas de reconocimiento basados en huella dactilar: Ataques Hill-Climbing.* Madrid : s.n., 2006. ISBN.

11. dCHAIN. www.dchain.com. *dCHAIN Web site.* [En línea] 1996. [Citado el: 23 de Enero de 2019.] http://dchain.com/ventajas-y-desventajas-de-los-diversos-tipos-de-sistemas-biometricos/.

12. Equipos Bancarios Dulon SAS. Terminal Reloj Biometrico ZKTECO F702-S| Equipos Bancarios Dulon. [En línea] [Citado el: 30 de Abril de 2019.] https://dulon.com.co/terminal-reloj-biometrico-zkteco-f702-s/.

13. Umanick. Aplicaciones y usos de las tecnologías biométricas. *Umanick - Soluciones de Identificación Biométrica.* [En línea] UMANICK TECHNOLOGIES, S.L, 2017. http://www.umanick.com/aplicaciones-y-usos-de-las-tecnologias-biometricas/.

14. INBIOSYS Biometria. inbiosys.wordpress.com. *INBIOSYS Biometria Web site.* [En línea] [Citado el: 23 de Enero de 2019.] https://inbiosys.wordpress.com/nuestros-productos/sistema-de-control-de-acceso-y-control-de-tiempos-y-asistencia/.

15. —. inbiosys.wordpress.com. *INBIOSYS Biometria Web site.* [En línea] [Citado el: 23 de Enero de 2019.] https://inbiosys.wordpress.com/nuestros-productos/sistema-de-control-de-ingreso-salida-y-acceso-de-personas.

16. NOKIA CORPORATION. Qt. [En línea] 2015. http://qt.nokia.com/products/developertools/.

17. SOSA, VIÑOLO. *Sistema Gestor de Proceso de Media v2.* [En línea] [Citado el: 12 de Marzo de 2019.] http://publicaciones.uci.cu/index.php/SC | seriecientifica@uci.cu..

18. Blog de WordPress.com. SISTEMAS GESTORES DE BASE DE DATOS LIBRES (MICROSOFT ACCESS) | Estructura de Datos. [En línea] 15 de Noviembre de 2011. [Citado el: 10 de Mayo de 2019.] https://b1m2.wordpress.com/2011/11/15/sistemas-de-gestores-de-base-de-datos-libres-microsoft-access/.

19. James Rumbaugh, Grady Booch, Ivar Jacobson. El Lenguaje Unificado de Modelado. Manual de Referencia. Madrid, España : s.n., 2000. ISBN: 84-7829-037-0.

20. iText. iText. [En línea] 2009. [Citado el: 5 de Mayo de 2019.] http://itextpdf.com.

21. Alexander, Arin. LimeReport. *Lime by Arin Alexander.* [En línea] Simple Machines, 2016. [Citado el: 4 de Junio de 2019.] http://limereport.ru/en/index.php.

22. Mikhalov, Alexander. CuteReport. *CuteReport web site.* [En línea] 2013 - 2016. [Citado el: 4 de Junio de 2019.] https://www.cute-report.com/en.

23. Osipov, Aleksey. Inqlude - The Qt library archive. [En línea] [Citado el: 9 de Mayo de 2019.] https://inqlude.org/libraries/qtrpt.html.

24. Sommerville, Ian. *Ingeniería de Software.* 2002.

25. Academia. [En línea] [Citado el: 8 de Abril de 2019.] http://www.academia.edu/9953322/Metodologias_para_el_desarrollo_de_software..

26. Sánchez, Tamara Rodríguez. *Metodología de desarrollo para la Actividad productiva de la UCI. Programa de mejora.* Universidad de las Ciencias Informaticas, La Habana.

27. Blog de WordPress.com. *Visión de Synergix de los Sistemas de Información y la Ingeniería del Software Tecnología y.* [En línea] 2013. [Citado el: 12 de Marzo de 2019.] https://synergix.wordpress.com/2008/07/10/modelo-de-dominio/.

28. Pressman, Roger S. *Ingeniería del Software. Un enfoque práctico. 7ma. edición.* 2010. ISBN: 978-607-15-0314-5.

29. Sommerville, Ian. *Ingeniería de Software. 7ma edición.* s.l. : Pearson Eduación Madrid, 2005. ISBN: 84-7829·074-5.

30. Wesley. *Ingeniería de Software (Parte IV Desarrollo).* s.l. : 7ma. México : Pearson, 2005. ISBN 84-7829-074-5.

31. The Qt Company. Qt Creator Documentation. [En línea] 2018. http://doc.qt.io/archives/qt-4.8/model-view-programming.

32. Larman. *UML y Patrones.* s.l. : Pearson Educación Madrid, 2002.

33. Pressman, Roger S. *Ingeniería del Software. Un enfoque práctico.* D. F. México : MC GRAW HILL INTERAMERICANA, 2010. ISBN: 978-607-15-0314-5.

34. Lorenzo, Ariel Chávez. *Estándares de codificación para C++.* 2010.

35. Toledo, Federico. *Introducción a las Pruebas de Sistemas de Información.* Montevideo : Kindle Edition, 2014.

36. S.L, Kimaldi Electronics . www.kimaldi.com. *S.L, Kimaldi Electronics Web site.* [En línea] Grupo Kimaldi. [Citado el: 23 de Enero de 2019.]

https://www.kimaldi.com/blog/biometria/que_aplicaciones_pueden_tener_los_sistemas_de_reconocimiento_biometrico/.

37. TECNOSeguro. Acerca de nosotros: TECNOSeguro. *TECNOSeguro Web site.* [En línea] 2011. [Citado el: 23 de Enero de 2019.] https://www.tecnoseguro.com/noticias/control-de-acceso/control-acceso-biometrico-seguridad-fermax.

38. Inc, Bayometric. www.bayometric.com.mx. *Inc, Bayometric Web site.* [En línea] [Citado el: 23 de Enero de 2019.] http://www.bayometric.com.mx/cross-match-verifier-300/.

39. S.L, Fermax Holding Investment. www.fermax.com. *Fermax Holding Investment Web site.* [En línea] Fermax Professional, 1949. [Citado el: 23 de Enero de 2019.] https://blog.fermax.com/esp/control-de-accesos-biometrico-ventajas-y-desventajas.

40. Murdoch Sistemas Peru. *Escáner de Huella* | *Crossmatch Verifier 300* | *1.* Cuzco : s.n. ISBN.

41. INBIOSYS Biometria. inbiosys.wordpress.com. *INBIOSYS Biometria Web site.* [En línea] [Citado el: 23 de Enero de 2019.] https://inbiosys.wordpress.com/nuestros-productos/sistema-de-control-de-acceso-de-personas-y-apertura-de-puertas.

42. Hernández, Sampieri. 1998.

43. Parra, Violeta. Blog Seas. *Blog Seas Web Site.* [En línea] Seas Estudios Superiores Abiertos. [Citado el: 11 de Marzo de 2019.] https://www.seas.es/blog/seas-cursos-online/librerias-de-vision-artificial/.

44. The Qt Company. Qt APIs & Libraries, Tools and IDE. *Qt* | *Cross-platform software development for embedded & desktop.* [En línea] 2018. [Citado el: 1 de diciembre de 2018.] http://doc.qt.io/qt-5/qmlapplications.html.

45. —. Qt APIs & Libraries, Tools and IDE. *Qt* | *Cross-platform software development for embedded & desktop.* [En línea] 2018. [Citado el: 1 de diciembre de 2018.] https://www.qt.io/qt-for-application-development/.

46. —. *Qt APIs & Libraries, Tools and IDE.* [En línea] Qt | Cross-platform software development for embedded & desktop, 2018. [Citado el: 2019 de Marzo de 11.] http://doc.qt.io/qt-5/qmlapplications.html.

47. *Qt* | *Cross-platform software development for embedded & desktop.* [En línea] Qt APIs & Libraries, Tools and IDE, 2018. [Citado el: 12 de Marzo de 2019.] https://www.qt.io/qt-for-application-development/..

48. UNAM-Facultad de Ingeniería Biométrica Informática. Reconocimiento del iris ocular. [En línea] [Citado el: 12 de Abril de 2019.] http://redyseguridad.fi-p.unam.mx/proyectos/biometria/clasificacionsistemas/recoiris.html.

49. Biometria. Reconocimiento del Iris. [En línea] [Citado el: 12 de Abril de 2019.] http://www.biometria.gov.ar/metodos-biometricos/iris.aspx.

50. Biometría. Reconocimiento de Voz. [En línea] [Citado el: 12 de Abril de 2019.] http://www.biometria.gov.ar/metodos-biometricos/voz.aspx.

51. BlogCucoRent. ¿Controlar los horarios por huella dactilar o por reconocimiento facial? [En línea] [Citado el: 30 de Abril de 2019.] https://www.cucorent.com/blog/controlar-los-horarios-huella-dactilar-reconocimiento-facial-2/.

Anexos

Historias de Usuario

Historia de Usuario	
Número: 2	**Nombre de HU:** Registrar usuario
Prioridad en el negocio: Media	**Nivel de Complejidad:** Media
Puntos estimados: 2	**Iteración Asignada:** 1
Programador responsable: Daniel Miguel González Mora	
Descripción: Consiste en registrar a un nuevo usuario en el sistema, introduciendo todos los datos que sean necesarios para su registro en el sistema y el rol que va a desempeñar en el mismo.	
Observaciones: En caso de existir campos vacíos, se notifica enviando un mensaje de error. Solamente pueden registrar usuarios en el sistema los que tengan rol Administrador. Si es correcta la información registrada el nuevo usuario es registrado en el sistema satisfactoriamente y sus datos son registrados en la base de datos.	
Prototipo de Interfaz:	

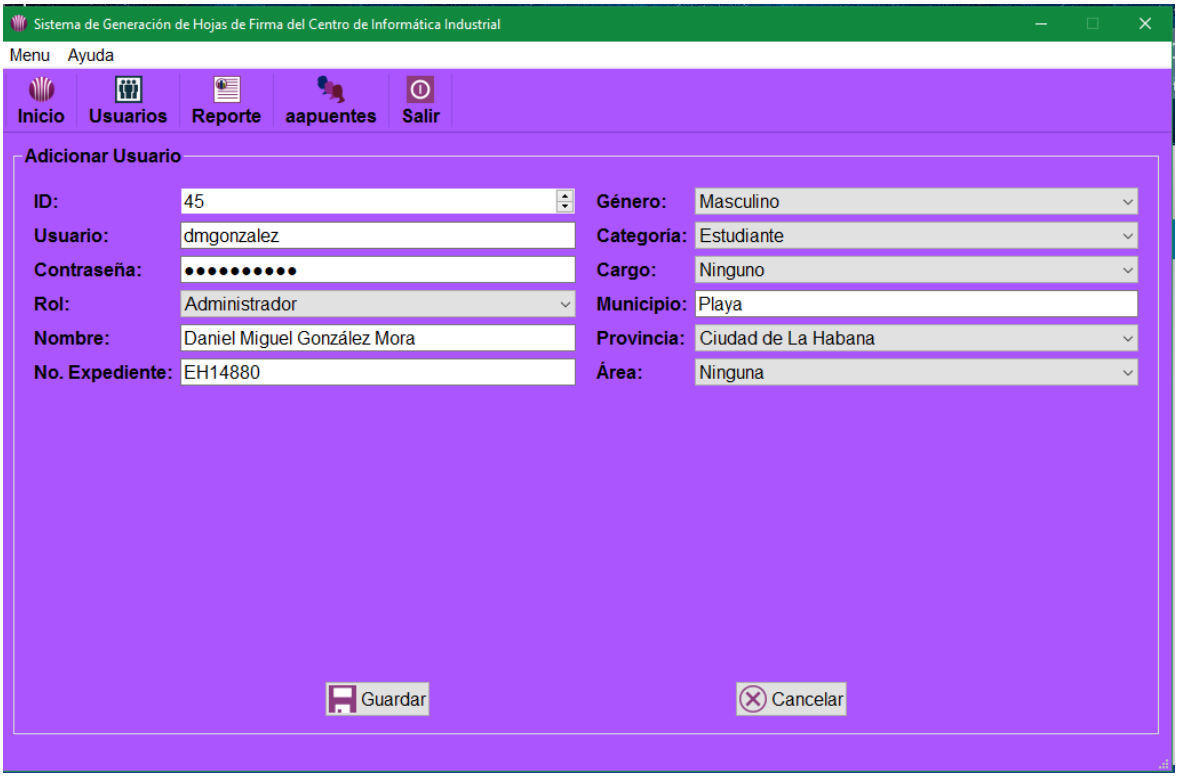


Tabla 14: Historia de usuario – Registrar usuario.

Historia de Usuario	
Número: 3	**Nombre de HU:** Modificar usuario
Prioridad en el negocio: Media	**Nivel de Complejidad:** Media
Puntos estimados: 2	**Iteración Asignada:** 1
Programador responsable: Daniel Miguel González Mora	
Descripción: Consiste en modificar a un usuario registrado en el sistema, modificando todos los datos que sean necesarios para ser registrado nuevamente en el sistema incluso el sistema brinda la posibilidad de cambiar el rol que el usuario desempeña en el sistema.	

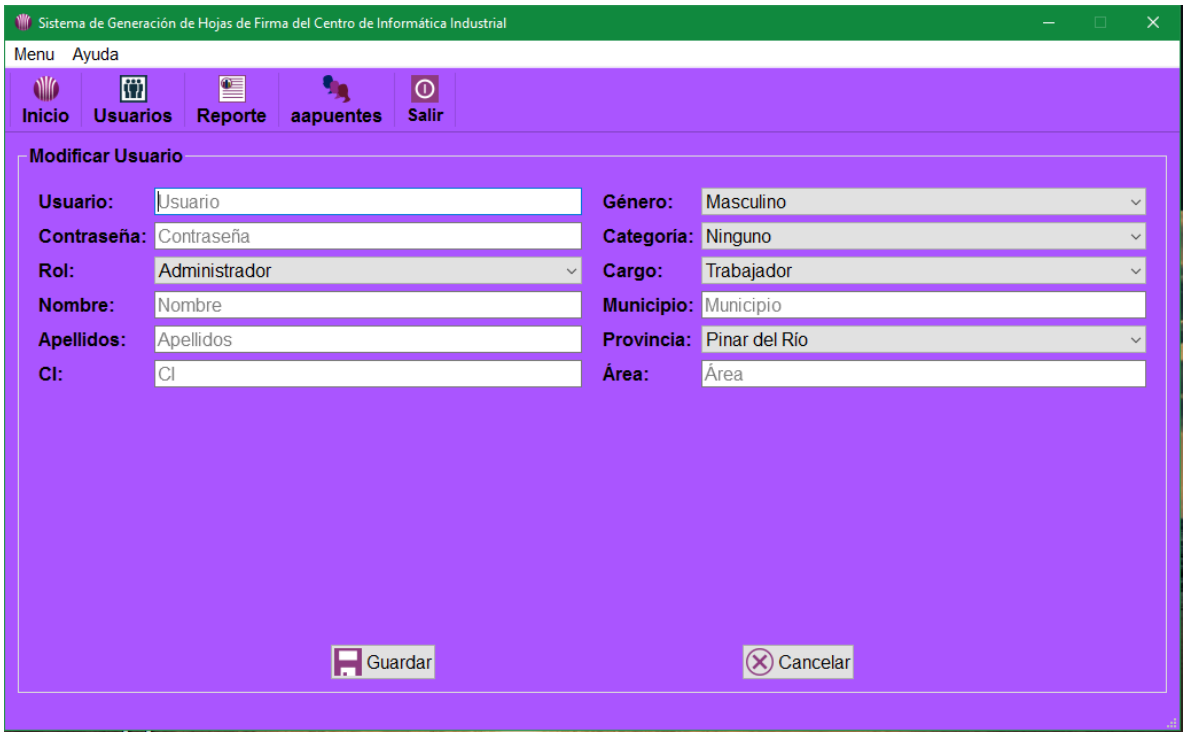
Observaciones: En caso de que la modificación sea fallida porque existan campos vacíos, se notifica enviando un mensaje de error. Solamente pueden modificar usuarios en el sistema los que tengan rol Administrador. Si es correcta la información modificada el usuario es registrado nuevamente en el sistema satisfactoriamente con sus datos actualizados en la base de datos. Si el usuario seleccionado para modificar sus datos es el mismo que está autenticado, entonces no se pueden modificar sus datos.
Prototipo de Interfaz: 

Tabla 15: Historia de usuario – Modificar usuario.

Historia de Usuario	
Número: 4	**Nombre de HU:** Eliminar usuario
Prioridad en el negocio: Media	**Nivel de Complejidad:** Baja

Puntos estimados: 1	**Iteración Asignada:** 1
Programador responsable: Daniel Miguel González Mora	
Descripción: Consiste en eliminar un usuario del sistema, seleccionando al usuario que se desea eliminar del sistema.	
Observaciones: En caso de que la eliminación sea fallida, se notifica enviando un mensaje de error. Solamente pueden eliminar usuarios en el sistema los que tengan rol Administrador. Si es correcta el usuario es eliminado del sistema satisfactoriamente y sus datos de la base de datos son eliminados. Si el usuario seleccionado para eliminar sus datos es el mismo que está autenticado, entonces no se eliminar del sistema.	
Prototipo de Interfaz:	

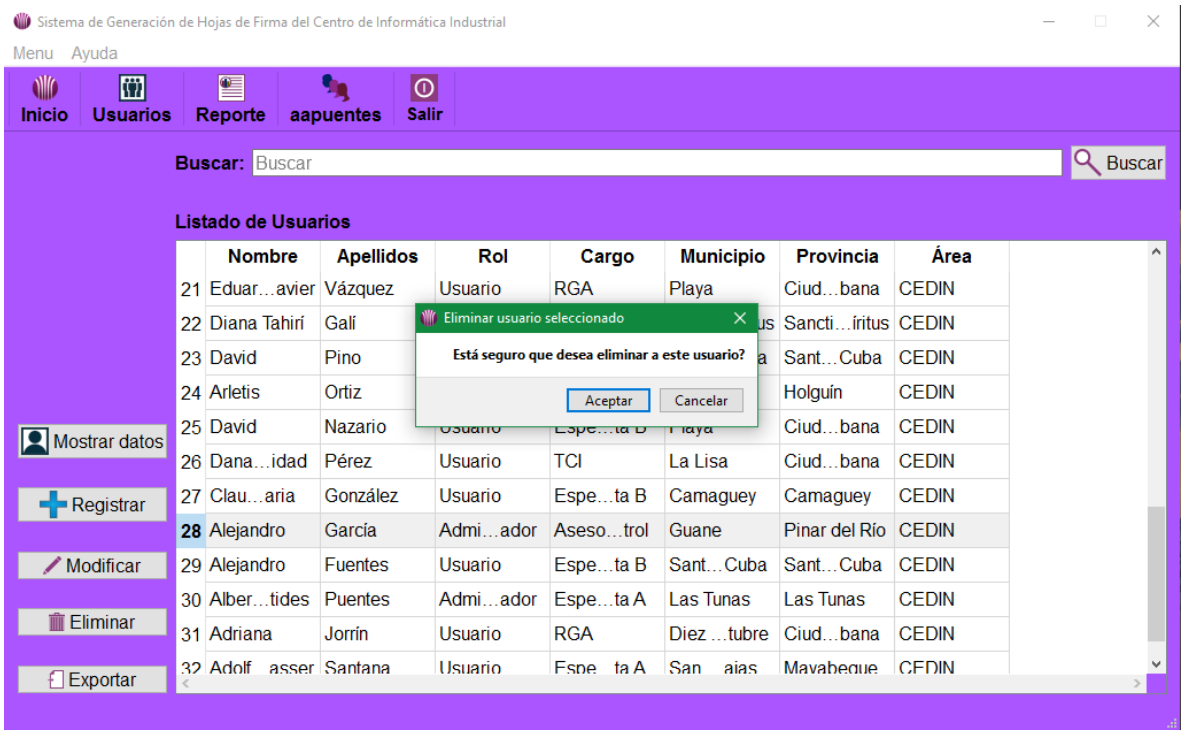
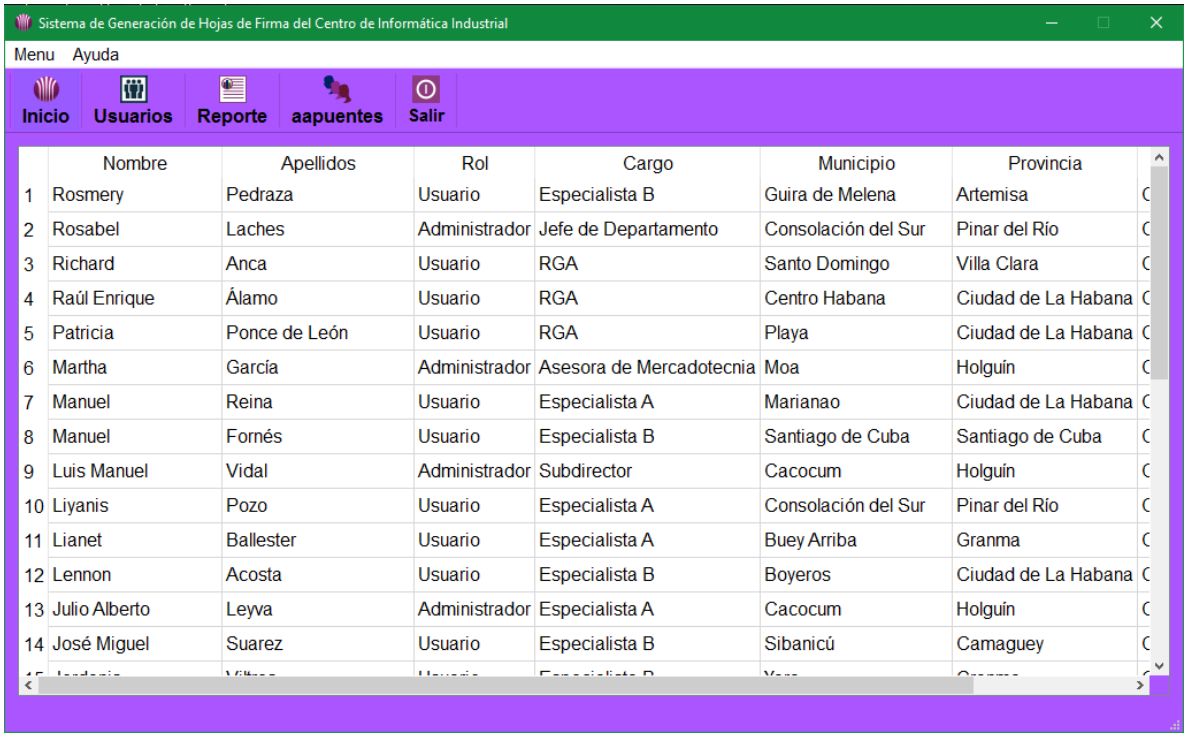


Tabla 16: Historia de usuario – Eliminar usuario (s).

Historia de Usuario	
Número: 5	**Nombre de HU:** Mostrar usuario (s)
Prioridad en el negocio: Baja	**Nivel de Complejidad:** Baja
Puntos estimados: 1	**Iteración Asignada:** 1
Programador responsable: Daniel Miguel González Mora	
Descripción: Consiste en mostrar a todos los usuarios que se encuentran registrados en el sistema.	
Observaciones: En caso de que se desee ver más detalles de un usuario en específico, el sistema muestra una acción que al realizarla muestra una vista con más detalles acerca del usuario que se encuentra seleccionado en ese momento.	
Prototipo de Interfaz:	



	Nombre	Apellidos	Rol	Cargo	Municipio	Provincia
1	Rosmery	Pedraza	Usuario	Especialista B	Guira de Melena	Artemisa
2	Rosabel	Laches	Administrador	Jefe de Departamento	Consolación del Sur	Pinar del Río
3	Richard	Anca	Usuario	RGA	Santo Domingo	Villa Clara
4	Raúl Enrique	Álamo	Usuario	RGA	Centro Habana	Ciudad de La Habana
5	Patricia	Ponce de León	Usuario	RGA	Playa	Ciudad de La Habana
6	Martha	García	Administrador	Asesora de Mercadotecnia	Moa	Holguín
7	Manuel	Reina	Usuario	Especialista A	Marianao	Ciudad de La Habana
8	Manuel	Fornés	Usuario	Especialista B	Santiago de Cuba	Santiago de Cuba
9	Luis Manuel	Vidal	Administrador	Subdirector	Cacocum	Holguín
10	Liyanis	Pozo	Usuario	Especialista A	Consolación del Sur	Pinar del Río
11	Lianet	Ballester	Usuario	Especialista A	Buey Arriba	Granma
12	Lennon	Acosta	Usuario	Especialista B	Boyeros	Ciudad de La Habana
13	Julio Alberto	Leyva	Administrador	Especialista A	Cacocum	Holguín
14	José Miguel	Suarez	Usuario	Especialista B	Sibanicú	Camaguey

Tabla 17: Historia de usuario – Mostrar usuario (s).

Historia de Usuario	
Número: 6	**Nombre de HU:** Buscar usuario
Prioridad en el negocio: Baja	**Nivel de Complejidad:** Baja
Puntos estimados: 1	**Iteración Asignada:** 1
Programador responsable: Daniel Miguel González Mora	
Descripción: Consiste en realizar una búsqueda para hallar a un usuario en específico que se encuentre registrado en la base de datos del sistema.	

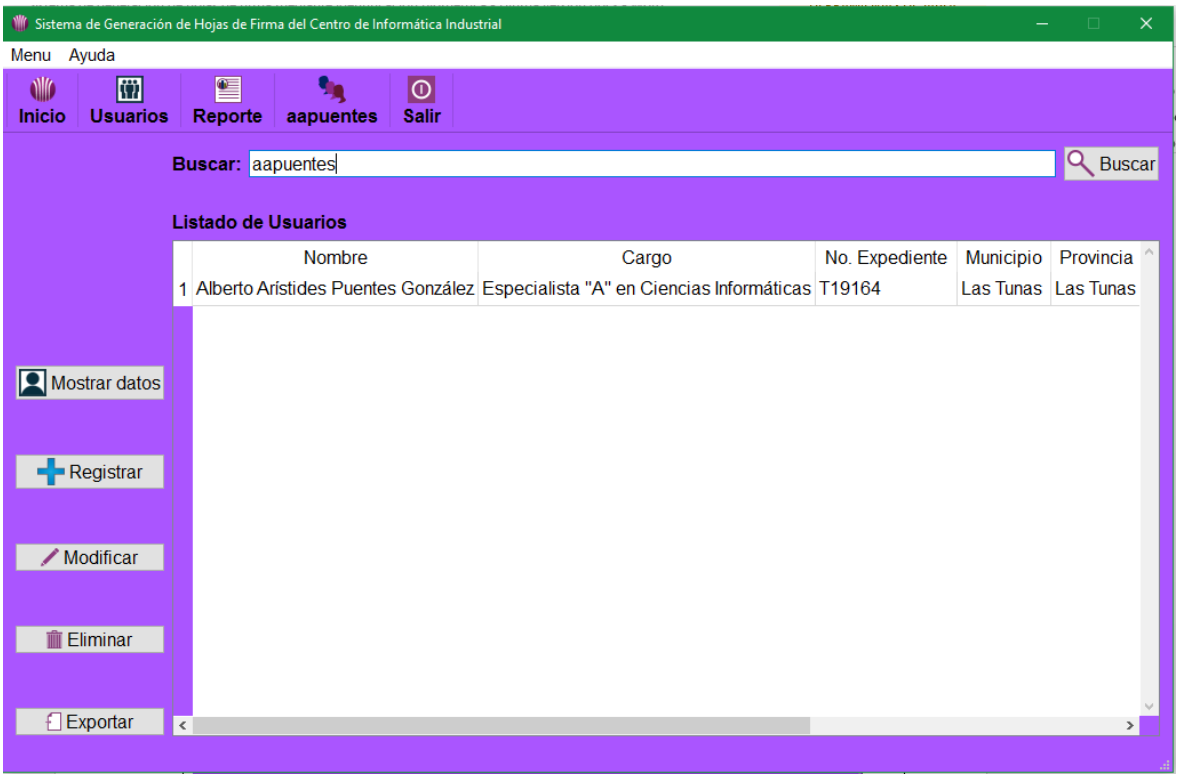
Observaciones: En caso de que el usuario no haya sido encontrado significa que dicho usuario no se encuentra registrado en el sistema, notificando a través de un mensaje de error de que no se encontró ningún resultado.
Prototipo de Interfaz: 

Tabla 18: Historia de usuario: Buscar usuario.

Historia de Usuario	
Número: 7	**Nombre de HU:** Cambiar contraseña de usuario
Prioridad en el negocio: Baja	**Nivel de Complejidad:** Baja
Puntos estimados: 1	**Iteración Asignada:** 2

Programador responsable: Daniel Miguel González Mora
Descripción: Consiste en cambiar la contraseña del usuario del cual se muestran sus datos registrados en el sistema.
Observaciones: No se puede cambiar la contraseña o introducir una nueva si el campo correspondiente se encuentra vacío.
Prototipo de Interfaz:

Tabla 19: Historia de usuario: Cambiar contraseña de usuario.

Historia de Usuario	
Número: 8	**Nombre de HU:** Exportar información de los usuarios
Prioridad en el negocio: Baja	**Nivel de Complejidad:** Baja
Puntos estimados: 1	**Iteración Asignada:** 2
Programador responsable: Daniel Miguel González Mora	
Descripción: Consiste en exportar toda la información que haya sido mostrada en la tabla donde se muestran los usuarios del sistema en formato *PDF*	
Observaciones: En caso de que no se muestre ninguna información en la tabla donde se muestran los usuarios del sistema, se notifica a través de un mensaje de error.	
Prototipo de Interfaz:	

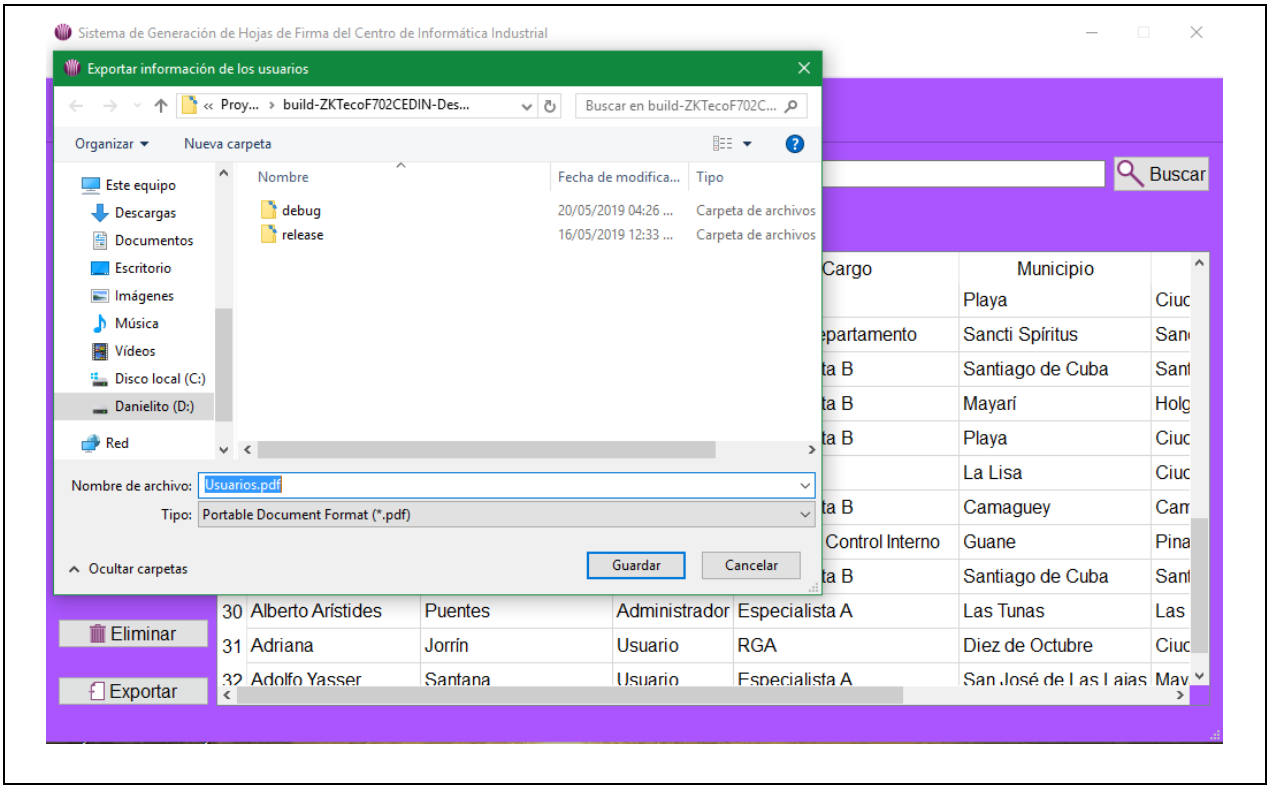


Tabla 20: Historia de usuario: Exportar información de los usuarios.

Historia de Usuario	
Número: 9	**Nombre de HU:** Abrir plantilla
Prioridad en el negocio: Baja	**Nivel de Complejidad:** Baja
Puntos estimados: 1	**Iteración Asignada:** 2
Programador responsable: Daniel Miguel González Mora	
Descripción: Consiste en abrir una plantilla en formato *XML*, mostrando en una ventana emergente la plantilla cargada con un conjunto de funcionalidades proporcionadas por la biblioteca de generación de reportes utilizada.	
Observaciones: Ninguna.	

Prototipo de Interfaz:

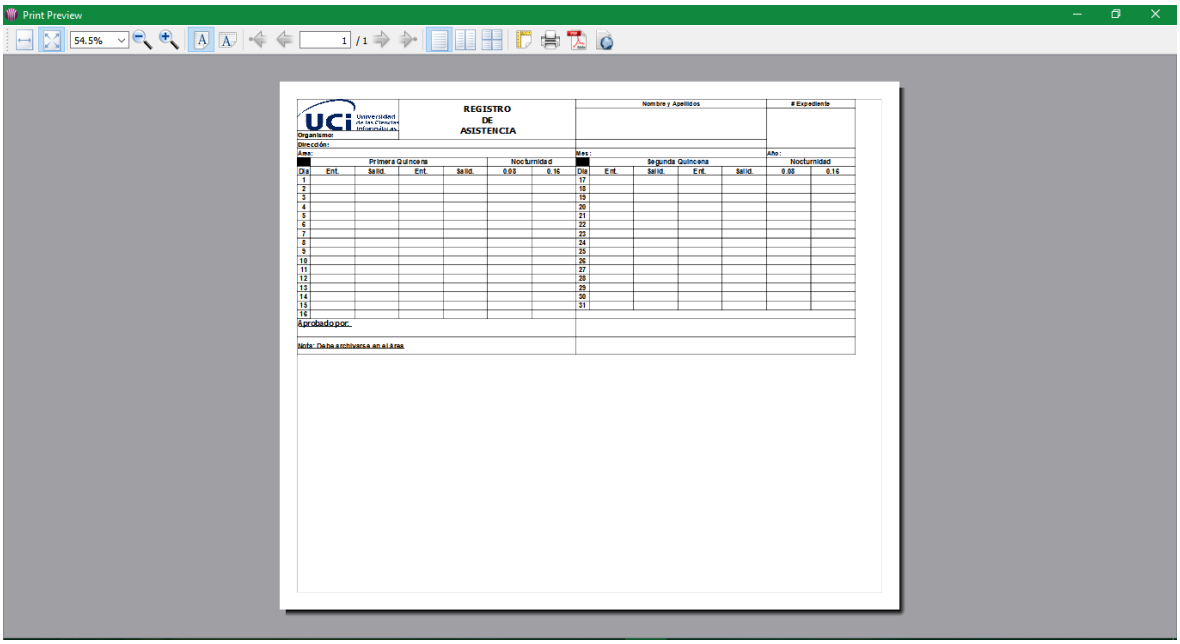


Tabla 21: Historia de usuario: Abrir plantilla.

Historia de Usuario	
Número: 12	**Nombre de HU:** Imprimir reporte de hojas de firma
Prioridad en el negocio: Alta	**Nivel de Complejidad:** Baja
Puntos estimados: 1	**Iteración Asignada:** 3
Programador responsable: Daniel Miguel González Mora	
Descripción: Consiste en exportar toda la información que haya sido mostrada en la tabla donde se muestran los usuarios del sistema en formato *PDF* o *HTML*.	

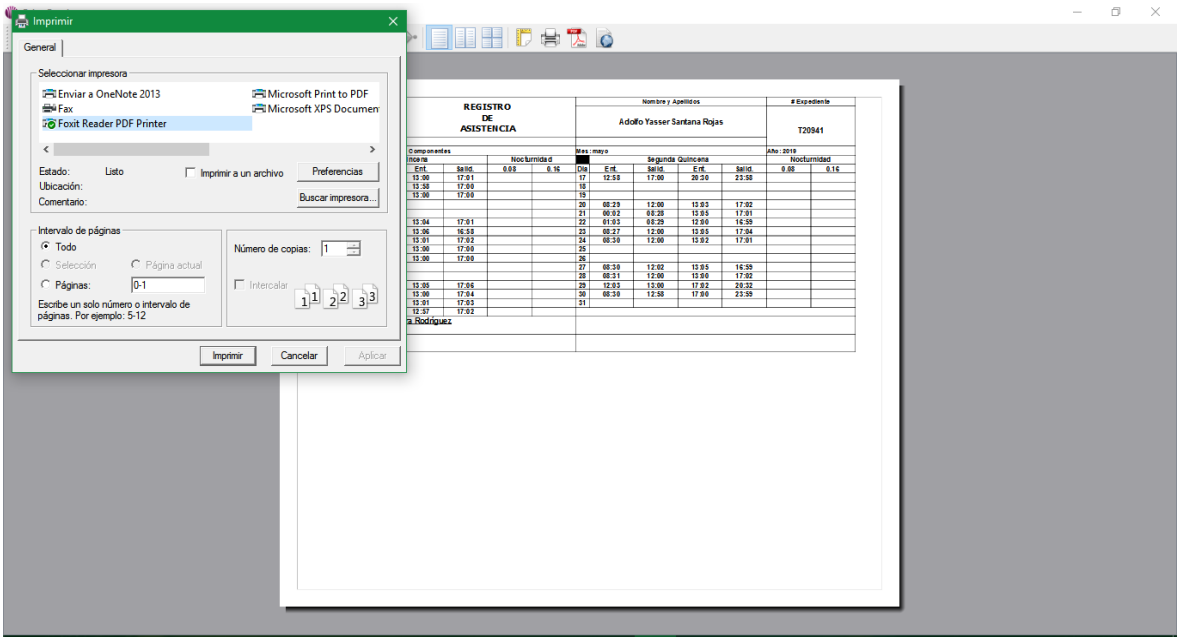
Observaciones: En caso de que no se muestre nada en la tabla donde se muestran los usuarios del sistema, se notifica a través de un mensaje de error de que no se muestra la información que se desea exportar.
Prototipo de Interfaz: 

Tabla 22: Historia de usuario: Imprimir reporte de hoja de firma.

Historia de Usuario	
Número: 13	**Nombre de HU:** Exportar reporte de hoja de firma
Prioridad en el negocio: Alta	**Nivel de Complejidad:** Baja
Puntos estimados: 1	**Iteración Asignada:** 3
Programador responsable: Daniel Miguel González Mora	

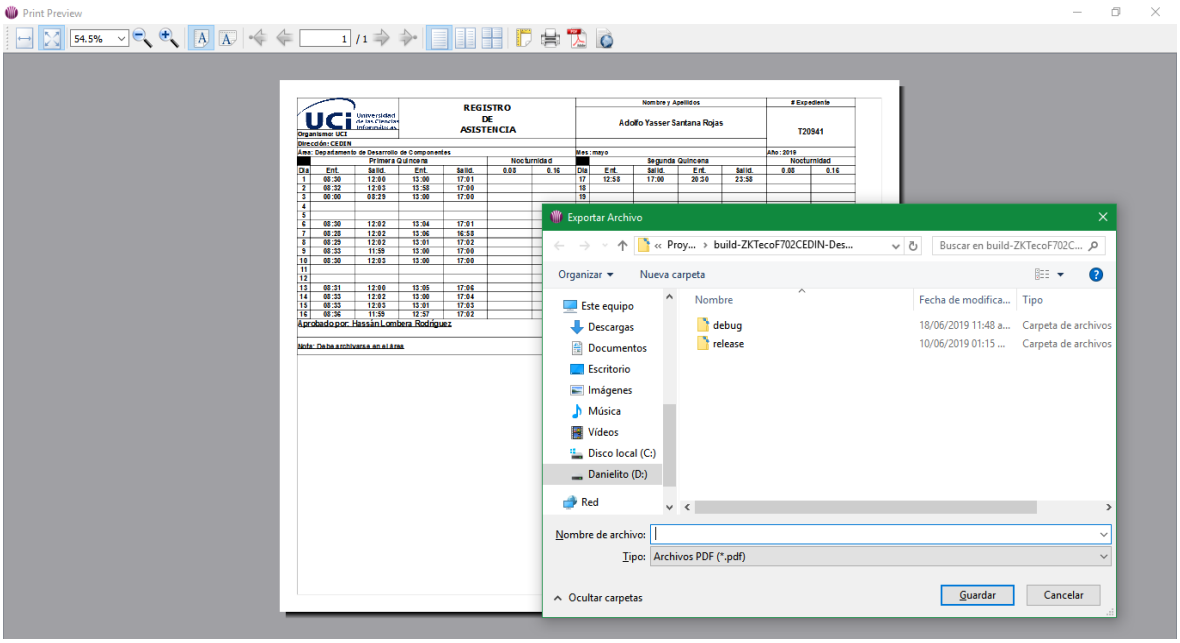
Descripción: Consiste en exportar toda la información que haya sido mostrada en la tabla donde se muestran los usuarios del sistema en formato *PDF* o *HTML*.
Observaciones: En caso de que no se muestre nada en la tabla donde se muestran los usuarios del sistema, se notifica a través de un mensaje de error de que no se muestra la información que se desea exportar.
Prototipo de Interfaz: 

Tabla 23: Historia de usuario: Exportar reporte de hoja de firma.