Universidad de las Ciencias Informáticas

Facultad 1

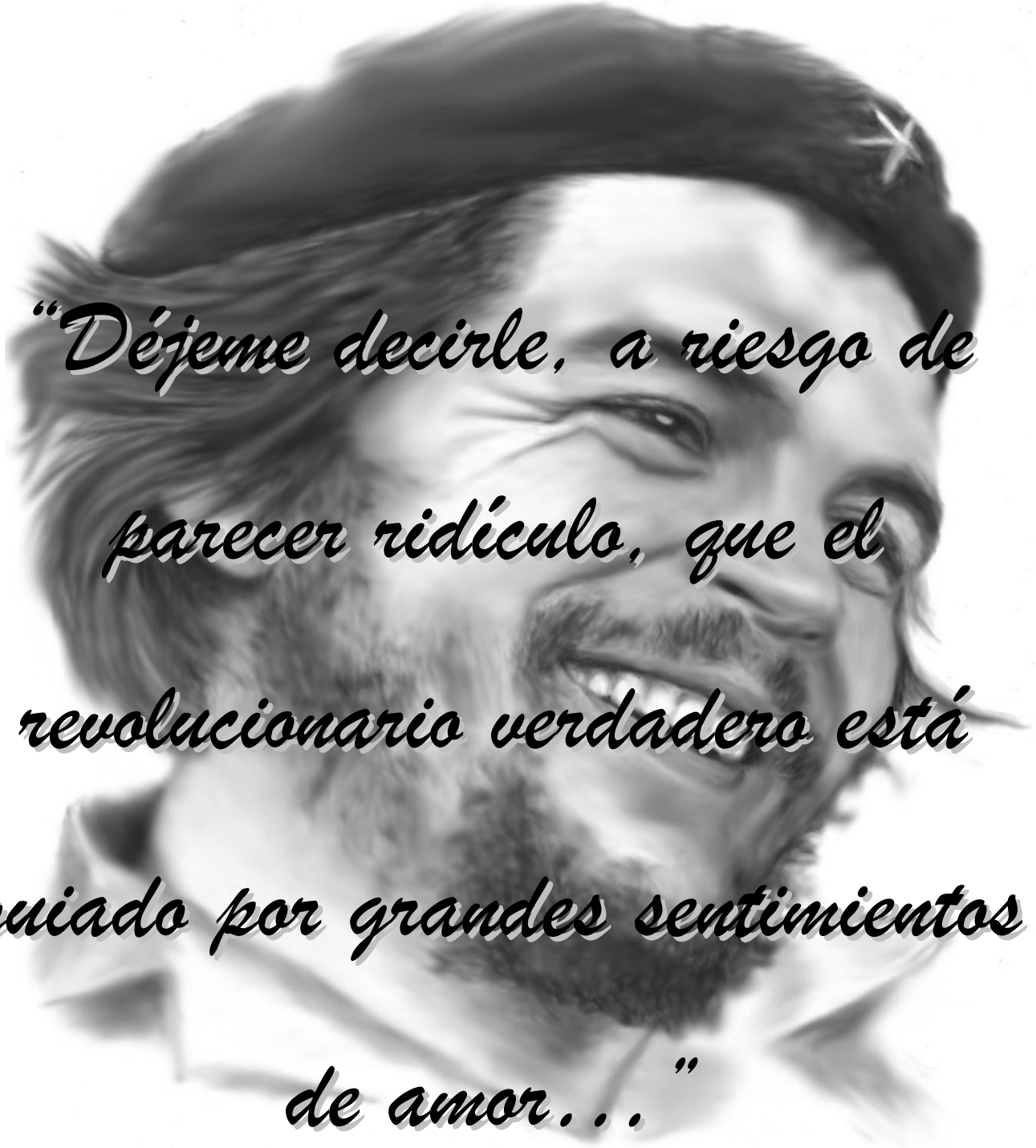
"Applet y Middleware para gestionar la información de historiales clínicos en tarjetas inteligentes"

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

Autores: Diana Rosa Zapata Vizcaino
Raidel Abreu Patterson

Tutores: Ing. Dayron Almeida Sotolongo
Ing. Ander Sánchez Jardines

"Ciudad de la Habana, junio del 2011"


"Déjeme decirle, a riesgo de parecer ridículo, que el revolucionario verdadero está guiado por grandes sentimientos de amor..."

Declaración de auditoría

Por este medio declaramos que nosotros, Diana Rosa Zapata Vizcaino y Raidel Abreu Patterson somos los únicos autores de este trabajo y autorizamos a la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

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

Autores:

_____ _____

Diana Rosa Zapata Vizcaino

Raidel Abreu Patterson

Tutores:

_____ _____

Ing. Ander Sánchez Jardines

Ing.Dayron Almeida Sotolongo

Dedicatoria

Quiero dedicar mi trabajo de Diploma a mis padres, mi hermanita,

A mi familia en general y a mi novia, a todos gracias

Por el amor incondicional

Que me han dado.

Los Quiero…

Raidel

Dedico mi trabajo de diploma, a mi hermano Damián Urbanito, con todo mi amor, para que en el futuro, seas un buen profesional, si alguna vez la vida, te hace pensar que no podrás lograr tus sueños, pues esto es un ejemplo de que todo lo que te propongas, con empeño, podrás hacerlo.

Para ti…

Diana Rosa

Agradecimientos

Agradezco a mis padres por darme su apoyo en todo momento, porque siempre han sido mi fuente de inspiración, y llegar a la altura del hijo que ellos se merecen. Mami gracias por siempre confiar en mí y darme tu apoyo. Papá espero que ahora puedas estar orgulloso de mí, ya que ese es mi mayor deseo. También sé que no voy a poder superarte pero lo seguiré intentando. Los quiero a ambos muchísimo!!!.

A mi hermanita linda por contar conmigo siempre para todo y ser mi confidente. Te quiero tata…

A mis tíos y tías (Cary, Gilbe, America, Leysi en fin todos) que fueron todo cuando mis padres estaban lejos de aquí, que me dieron todo lo que necesitaba y más, que estaban ahí cuando estaba triste, que sufrieron en los primeros años en la universidad cuando tenía los mundiales y no podía estar disfrutando en las casas en la playa cuando tenía que estudiar, entre otras cosas…. Muchas gracias…

A mi abuela por cuidar de mi cuando estuve solo y mudarse conmigo aunque no le gustaba donde vivía. Por favor mejórate ya, que te quiero ver como antes.

Además quiero darles las gracias a Orly y a Iliana por estar siempre ahí cuando los necesité sin ser de mi sangre, siempre estaban ahí para todo sin importar lo que estuviesen haciendo, no sé cómo lo hacían pero siempre tenían tiempo para mi, por eso y por cosas que no debo decir aquí ya son como mis tíos, son de mi familia.

En fin quiero agradecer a mis primos (Victor, Indiana, Waldo) que me apoyaron en todo y a su manera me aconsejaron ya que hubo un tiempo en que estaba casi sin rumbo. A todos muchas gracias por estar ahí….

Agradecimientos

A Claudia por quererme tanto y amarme tal y como soy, por darme esa estabilidad que siempre quise tener pero que nunca pude encontrar, Clau eres sin duda alguna, una de las mejores cosas que me llevo de la UCI, mi niña linda Te AMO!!!!. También quiero agradecer a su familia (Rita y Celia) que me han querido como a un hijo.

A la Universidad por darme la oportunidad de graduarme y de conocer personas extraordinarias de todas las provincias de nuestro país.

A mi equipo de futbol (OneReal) Benito, Randy, Lairon, Alejandro, Jose Andres, Sergio, Addiel, Guillermo y Nelson por darme la oportunidad de mostrar mis habilidades como portero y contar conmigo.

Al antiguo piquete (Los Ardientes) Noel, Anier, Lama, David, Yasser, Leorgis, Ismel entre otros. A mis esposas las negronas más sexys de la UCI (Mairelys y Taimi).

A mis antiguos compañeros de grupo que fueron la primera familia que tuve aquí en la escuela, y que por otros motivos nos hemos distanciado (Frank, Juan, Liuber, Madelin, Yaisi, Rolando, y muchos que actualmente ya no están aquí).

A mis tutores (Ander y Dayron) por soportarme cuando yo los molestaba y apoyarme en las decisiones que se tomaron.

A mi compañera de tesis (Diana Rosa) por comprenderme, soportarme y saber cómo mandarme a hacer las cosas sin que yo creyera eso.

A todos los que no menciono que me ayudaron en la realización de este trabajo y en que yo fuera mejor persona… Muchas Gracias

Raidel

Agradecimientos

Agradezco infinitamente a mis padres Ifrain Y Graciela, por apoyarme en todos estos años de estudio para formarme como profesional. Gracias por confiar en mí y por darme a demostrar que el mundo no se iba a acabar cuando en verdad se me iba encima, con los estudios. Mi mayor deseo es que se sientan orgullosos, y convencidos de que todo el esfuerzo no fue en vano, espero que la vida nos siga sonriendo a todos juntos…Los quiero mucho mucho mucho….

Agradezco a mi me hermano Damián por decirme siempre, "muchacha todo está bien", por querer que yo esté bien siempre…Tu eres el mayor tesoro que tengo en la vida…

Agradezco a mi familia en general por quererme y apoyarme principalmente a mis abuelos, Martha y Angel Luis, que aunque no esté en vida, siempre me llamaba para darme ánimos y saber de mi…Muchas Gracias…

Agradecer a mi primo Frank José por llamarme y darme ánimos, espero que sigas con la idea de ser un profesional….

Quiero agradecer a mi nueva familia, por quererme y darme tanto apoyo, por darme fuerzas para que esto se realizara, Ana María y a Erminda….Muchas gracias por todo…

Agradezco a mi novio Onnier, que aunque no le gusten "estas partes de la historia", por quererme tanto, y darme mucha fuerza para terminar, por llamarme, y hacer todo lo posible por de una forma u otra estar cerca de mí, para darme deseos de vivir y de amarte…

Agradezco a mi AMIGA y más que eso a mi hermana Yailen, por darme tanto apoyo, tanto profesionalmente como en la vida personal, tú eres una de las mejores cosas que tengo en la vida.

A mis amistades de Santiago, Leonardo, Leslie, Gracila, Tikito y todas mis amistades de la vocacional, no me olvido de ninguno…

Agradecimientos

Agradezco a la UCI por darme la oportunidad de graduarme acá, por conocer tantas personas, que de una forma u otra prendimos a convivir y ser mejores personas.

Agradezco a mis más cercanas amistades, Antonio Yeni e Ivis, por compartir los mejores momentos acá con ustedes, le doy gracias a la vida por haberlos conocido…

Agradezco a mis compañeros de grupo de todos los años al igual que a todos los profesores que me impartieron las asignaturas.

Agradezco a Lissi, por ayudarme siempre que pudiste, a Yuliet, a José Eduardo, a Susel, a Liliana, a Juan Andrés, Alexander, a la China y a Sulen,

Agradezco a las personas que hicieron posible este trabajo conjuntamente con nosotros, a mis tutores mi amigo Ander, a Dayron y a mi oponente Yurdik.

Agradezco a mi compañero de tesis Raidel por poner todo su empeño en esto y dar lo mejor, fuiste el mejor compañero de tesis…

A todos los que de una forma u otro hicieron posible este sueño, MUCHAS GRACIAS….

Diana Rosa

Resumen

En el centro de Tarjetas Inteligentes de la Facultad 1 de la Universidad de las Ciencias Informáticas, aún se investiga y se estudia acerca del tema de las *Smart Card*, su aplicación en muchas de las esferas, su estructura e importancia para el desarrollo de la informática en Cuba. El objetivo de este trabajo es desarrollar una aplicación *applet* y su componente *middleware*, para gestionar, la información de los historiales clínicos utilizando estas tarjetas inteligentes, para proporcionar un mejor manejo de la información de estos datos. Para esto se hizo un estudio de las tarjetas, las herramientas, metodologías, lenguajes de programación, referente a ellas, para realizar un sistema con la calidad requerida.

Conjuntamente con la propuesta, y a partir del estudio realizado, en este documento se describe el proceso de desarrollo de una aplicación que gestiona la información contenida en la tarjeta inteligente referente a las historias clínicas guiándose por la metodología de Programación extrema, especificando los artefactos que brinda, finalmente se obtuvo los resultados esperados.

Palabras Claves: tarjetas inteligentes, applet, middleware, herramientas, metodologías, artefactos, calidad.

Resumen

En el centro de Tarjetas Inteligentes de la Facultad 1 de la Universidad de las Ciencias Informáticas, aún se investiga y se estudia acerca del tema de las *Smart Card*, su aplicación en muchas de las esferas, su estructura e importancia para el desarrollo de la informática en Cuba. El objetivo de este trabajo es desarrollar una aplicación *applet* y su componente *middleware*, para gestionar, la información de los historiales clínicos utilizando estas tarjetas inteligentes, para proporcionar un mejor manejo de la información de estos datos. Para esto se hizo un estudio de las tarjetas, las herramientas, metodologías, lenguajes de programación, referente a ellas, para realizar un sistema con la calidad requerida.

Conjuntamente con la propuesta, y a partir del estudio realizado, en este documento se describe el proceso de desarrollo de una aplicación que gestiona la información contenida en la tarjeta inteligente referente a las historias clínicas guiándose por la metodología de Programación extrema, especificando los artefactos que brinda, finalmente se obtuvo los resultados esperados.

Palabras Claves: tarjetas inteligentes, *applet*, *middleware*, herramientas, metodologías, artefactos, calidad.

Resumen

En el centro de tarjetas inteligentes de la Facultad 1 de la Universidad de las Ciencias Informáticas (UCI), aún se investiga y se estudia acerca del tema de las *Smart Card*, su aplicación en muchas de las esferas, su estructura e importancia para el desarrollo de la informática en Cuba. El objetivo de este trabajo es desarrollar una aplicación *applet* y su componente *middleware*, para gestionar, la información de los historiales clínicos utilizando estas tarjetas inteligentes, para proporcionar un mejor manejo de la información de estos datos. Para esto se hizo un estudio de las tarjetas, las herramientas, metodologías, lenguajes de programación, referente a ellas, para realizar un sistema con la calidad requerida.

Conjuntamente con la propuesta, y a partir del estudio realizado, en este documento se describe el proceso de desarrollo de una aplicación que gestiona la información contenida en la tarjeta inteligente referente a las historias clínicas guiándose por la metodología de programación extrema, especificando los artefactos que brinda, finalmente se obtuvo los resultados esperados.

Palabras Claves: tarjetas inteligentes, *applet*, *middleware*, herramientas, metodologías, artefactos, calidad.

Índice

Introducción 1

Capítulo 1: Fundamentación teórica 6

1.1. Introducción 6

1.1.1 Historiales clínicos 6

1.1.2 Tarjetas inteligentes 10

1.1.3 Tarjeta de salud 17

1.1.4 Applet 18

1.1.5 Middleware 18

1.2 Estado del arte 19

1.2.1 Nivel internacional 19

1.2.2 Nivel nacional 20

1.3 Metodologías para el desarrollo del software 20

1.4 Herramientas para el desarrollo del software 24

1.4.1 Developer Suite Gemalto 24

1.4.2 UML como lenguaje de modelación visual 24

1.4.3 UModel Altova 24

1.4.4 Visual Studio 25

1.5 Plataforma .NET 25

1.5.1 Microsoft .NET Framework 25

1.6 Lenguajes de programación 26

1.6.1 C# 26

1.6.2 JavaCard 27

1.7 Pruebas unitarias 27

1.8 Pruebas de aceptación 28

1.9 Necesidad de la implementación de una aplicación applet y middleware 28

1.10 Conclusiones del capítulo 29

Capítulo 2: Caracterización y diseño del sistema a implementar 30

2.1 Introducción 30

2.2 Modelo de dominio 30

2.3 Diagrama de clases del modelo de dominio 30

2.4 Glosario de conceptos del modelo de dominio 31

2.5 Historias de usuario. 31

2.6 Requerimientos no funcionales 35

2.7 Propuesta de solución 35

2.7.1 Metáfora 36

2.7.2 Arquitectura del applet 36

2.7.3 Arquitectura del middleware 38

2.7.4 Estilo arquitectónico 40

2.7.5 Patrones de diseño 42

2.8 Plan de entrega 42

2.9 Estimación de esfuerzo 43

2.10 Costos y beneficios 44

2.12 Conclusiones del capítulo 44

Capítulo 3: Implementación y pruebas del sistema 46

3.1 Introducción 46

3.2 Iteraciones a primera liberación 46
3.2.1 Tarjetas CRC.....46
3.3 Implementación del sistema.....49
3.3.1 Iteración 1. 49
3.3.2 Iteración 2. 49
3.3.3 Descripción de los principales flujos de procesos.....50
3.4 Pruebas.....52
3.4.1 Pruebas unitarias 52
3.4.2 Pruebas de aceptación 53
3.5 Conclusiones del capítulo.....57
3.6 Conclusiones generales 58
Recomendaciones.....59
Referencias bibliográficas.....60
Bibliografías consultadas.....63
Glosario de términos.....64

Índice de Tablas

Tabla 1: HU_1 Inicializar comunicación 32

Tabla 2: HU_ 2 Establecer canal seguro.....32

Tabla 3: HU_ 3 Autenticación al *applet* de HC.....33

Tabla 4: HU_ 4 Gestionar almacenamiento información 33

Tabla 5: HU_5 Enviar y recibir comandos 34

Tabla 6: HU_ 6 Finalizar comunicación. 34

Tabla 7: Plan de entrega 43

Tabla 8: Plan de Iteraciones. 44

Tabla 9: Tarjeta CRC (Applet Historia Clínica) 47

Tabla 10: Tarjeta CRC (Middleware SaludCard) 48

Tabla 11: Tarjeta CRC (Útil) 48

Tabla 12: Tarjeta CRC (Paciente) 48

Tabla 13: Tareas de ingeniería (Iteración 1) 49

Tabla 14: Tareas de ingeniería (Iteración 2) 49

Tabla 15: Prueba unitaria (verificar PIN) 53

Tabla 16: CP1-HU1 Prueba de funcionalidad para listar los lectores conectados 54

Tabla 17: CP2- HU2 Prueba de funcionalidad para establecer la conexión. 55

Tabla 18: CP3-HU3 Prueba de funcionalidad para verificar PIN y autenticarse. 55

Tabla 19:CP4-HU4 Proceso llenar datos de los pacientes 56

Tabla 20: CP5-HU4 Proceso obtención de la información 57

Índice de Ilustraciones

Ilustración 1: Modelo de dominio. 30

Ilustración 2: Algunos datos que contendrá el applet 37

Ilustración 3: Arquitectura de applet 38

Ilustración 4: Arquitectura del middleware 39

Ilustración 5: Diagrama de paquetes (Arquitectura base en tres capas) 41

Ilustración 6: Sub-fases de la fase Iteración 46

Ilustración 7: Diagrama de secuencia (proceso autenticación de usuario) 50

Ilustración 8: Diagrama de secuencia (proceso llenar datos de los pacientes) 51

Ilustración 9: Diagrama de secuencia (proceso obtención de la información) 52

Introducción

Debido al auge y crecimiento de las Tecnologías de la Informática y de las Comunicaciones (TIC[1]), que permiten la continua innovación y mejora de la asistencia sanitaria, se están incorporando hace algunos años en los sistemas públicos de salud avances en la gestión de la información. Ejemplo de estos perfeccionamientos es la inclusión en el sector de salud pública de una historia clínica electrónica (HCE[2]). Este tipo de proceso es implementado principalmente en países desarrollados. El uso de sistemas que gestionan información sanitaria es una de las principales aplicaciones de las TIC por sus beneficios en cuanto a calidad y seguridad y por la gran capacidad que tiene de almacenar datos. (1)

La HCE pretende introducir las Tecnologías de la Informática y de las Comunicaciones en el núcleo de la actividad sanitaria, dejando plasmado en ella el registro de la relación entre pacientes, médicos y demás profesionales involucrados, permitiendo además compartir información sanitaria de manera sencilla, segura e íntegra, evolucionando de ser un mero registro de antecedentes de salud de una persona a un sistema integrado de información. (1)

Se puede decir que un registro que integra toda la información de salud de una persona a lo largo de su vida, referida a los diferentes estados de salud y enfermedad, además generada por todos los profesionales de atención a la salud con los que se relaciona esa persona en cualquier nivel asistencial, presenta indudables ventajas en comparación con el método tradicional de registros de salud, puesto que mejora la seguridad, integridad y persistencia de los datos referentes a la comunicación entre pacientes que solicitan el servicio asistencial y los profesionales que lo proveen. (2)

Con la inserción de historias clínicas electrónicas en los procesos de gestión de información sanitaria se empieza a eliminar la antigua costumbre de almacenar los historiales escritos a mano. Estos historiales en su mayoría se encontraban en lugares donde los datos se podían perder o deteriorar con los años. Las ventajas de las HCE con respecto a la historia clínica (HC[3]) convencional puede sintetizarse en tres apartados: acceso simultáneo y remoto, seguridad y confidencialidad de la historia y procesado de la información para adquirir información y conocimiento. (1)

[1] TIC: Tecnologías de la Información y la Comunicación

[2] HCE: Historia Clínica Electrónica

[3] HC: Historias Clínicas

Introducción

No fue hasta principios de los años 90 que se desarrolló una seria conceptualización de la necesidad del desarrollo de estas HCE; a partir de ese momento comienza una carrera entre los desarrolladores de HCE. Esta situación conllevó a un descubrimiento importante: las historias clínicas tradicionales son un documento de fuerza legal y como tal, es obligatoria su conservación en formato de papel para poder ser evaluadas por las autoridades cuando se les requiera. Incluso, en muchos países, éstas HC son frecuentemente sometidas a auditorías. Sin embargo, el reconocimiento de la necesidad de disponer de registros electrónicos ha llevado a algunos de los estados de los EE.UU[4] y de los países de la Unión Europea a dar su anuencia[5] para que este tipo de HC tenga fuerza legal, siempre y cuando cumpla con los estándares que van dirigidos a esta, para lograr la interoperabilidad de los sistemas informáticos y entre la información que recogen estos. (3)

Para cumplir con normas de seguridad que deben tener las HCE, evitar que a través de la red se suplante la identidad de un paciente así como que se cambien sus datos en las fichas médicas, se requiere de almacenarlos en medios seguros, donde no puedan acceder individuos sin la debida autorización. Cumpliendo con todo lo anterior planteado y además, para un rápido acceso a la información y lograr la portabilidad de ésta, se decide el uso de tarjetas inteligentes como solución a estos problemas.

Las tarjetas inteligentes tienen un circuito integrado incrustado en la tarjeta, ésta transmite, almacena y procesa datos, ya que poseen más capacidad que las tarjetas de banda magnética. (4)

¿Por qué se utilizan las tarjetas inteligentes para almacenar las historias clínicas de los pacientes?

El empleo de estas tarjetas para la gestión de la información en el área de la salud ha sido una solución inteligente e innovadora ya que las búsquedas de datos serán más rápidas y eficientes, así como el procesamiento estadístico de éstos mediante fórmulas o herramientas informáticas destinadas a este fin; además, el paciente puede llevar consigo su tarjeta de tal forma que, si se aleja de su entorno habitual por cualquier motivo, pueda contar con su información médica en “el bolsillo” y permitir a cualquier médico consultar sus datos en caso de emergencia u otro tipo de eventualidad. El objetivo final de la tarjeta médica es que pueda ser utilizada por cualquiera de los profesionales de la salud. (3)Otro motivo es que el precio de las tarjetas inteligentes en el mercado mundial es de dos hasta siete euros y el de las hojas

[4] EE.UU: Estados Unidos de América

[5] Anuencia: Consentimiento, permiso para realizar algo.

(paquetes) oscila entre los cuatro y diez; con este cambio se pueden hacer grandes ahorros a la economía del sector. Países como España, Alemania, Francia, Canadá han optado por este cambio.

Por todo lo antes expuesto se puede afirmar que las HCE con tarjetas inteligentes son una solución eficiente y factible.

En el mundo existen muchos sistemas informáticos comercializables, que gestionan la información de historiales clínicos

Después de analizados los procesos de gestión de información de historias clínicas se define la siguiente **situación problémica**:

En Cuba, a pesar de no ser este uno de los aspectos donde se hayan alcanzado logros reconocidos, ya que no se encuentra entre las necesidades prioritarias del país, se comienza a notar un discreto desarrollo en la digitalización de esta información debido a que no existe una red de salud como existe en países del primer mundo, donde están conectados todos los hospitales y medios de atención sanitaria. Con el uso de tarjetas inteligentes los pacientes llevarían, a todas las instalaciones hospitalarias que asistan, la información pertinente para su atención y obtendrían además una mayor confidencialidad y seguridad de sus datos médicos. Existe la necesidad de implementar mecanismos de gestión de información de historiales clínicos más seguros, confiables y portables en el sector de la salud.

Para dar solución a la situación problémica descrita anteriormente se planteó el siguiente **problema científico**: ¿Cómo garantizar la gestión de la información de historiales clínicos utilizando tarjetas inteligentes?

El **objeto de estudio** se enmarca en el proceso de gestión de historiales clínicos, siendo el **campo de acción** el proceso de gestionar historiales clínicos utilizando las tarjetas inteligentes.

Para solucionar lo antes expuesto se ha propuesto como **objetivo general**: desarrollar una aplicación *applet*[6] y su componente *middleware*[7], para gestionar la información de los historiales clínicos utilizando tarjetas inteligentes.

[6] *Applet*: es un componente de una *aplicación* que se ejecuta en el contexto de otro programa, por ejemplo un navegador web. El *applet* debe ejecutarse en un *contenedor*, que lo proporciona un programa anfitrión, mediante un *plug-in,* o en aplicaciones como teléfonos móviles que soportan el modelo de programación por '*applet*s'.

[7] *Middleware*: es un software de conectividad que ofrece un conjunto de servicios que hacen posible el funcionamiento de aplicaciones distribuidas sobre plataformas heterogéneas.

Idea a defender: la realización de un *applet* y un componente *middleware* proporcionará un mejor manejo de la información de historiales clínicos en tarjetas inteligentes.

En el proceso de desarrollo de dichos componentes se realizaron las siguientes **tareas**:

- Realizar una caracterización sobre la arquitectura de las tarjetas inteligentes como base para la implementación de la aplicación *applet*.
- Investigar sobre los estándares asociados a este tipo de tecnología para su puesta en práctica en la implementación a realizar.
- Investigar acerca de los procedimientos y estrategias utilizados en el sistema de salud para el almacenamiento de historiales clínicos.
- Realizar un estudio de sistemas homólogos que utilicen las tarjetas inteligentes usadas en reportes médicos.
- Seleccionar la metodología de desarrollo de la solución.
- Desarrollar la aplicación *applet* y su componente *middleware*.
- Realizar las pruebas de calidad a la solución. **(Ver Ilustración 6)**

La indagación estuvo sustentada en los siguientes **métodos científicos:**

Métodos Teóricos

- Método histórico-lógico: se utiliza para la realización de la investigación donde se estudia la evolución del problema y la existencia de modelos de prueba similares al que se pretendía elaborar. También posibilita conocer los antecedentes y tendencias actuales de las herramientas que posibilitan realizar el *applet* y el componente *middleware* para gestionar la información en las tarjetas inteligentes.
- Analítico-sintético: permite realizar investigaciones en diferentes consultas bibliográficas, para darle solución al objeto del estudio. También permite realizar el procesamiento de toda la información pudiendo sintetizar y diferenciar cada una de ellas y, de esta forma, enfocarlas hacia la investigación, confeccionando así el estado del arte en el primer capítulo.
- Método de modelación: se utiliza debido a la necesidad de crear abstracciones y la necesidad de revelar la unidad del objetivo y lo subjetivo, además para representar los diagramas que se confeccionen para un mejor entendimiento del sistema a desarrollar.

Métodos Empíricos

- Método de observación: este método permite percibir cómo se hacen los *applets* y *middleware* en la práctica para así poder llevar esos conocimientos al tema propuesto, se emplea en todo momento.

Se muestra a continuación un breve resumen de los capítulos que contendrá el documento, haciendo referencia a los principales aspectos que se abordarán en ellos.

Capítulo 1: Fundamentación teórica para la gestión de historiales clínicos.

Se muestra la fundamentación general que soporta teóricamente la solución del problema. Se hace un análisis de las herramientas y lenguajes de programación para el desarrollo del *applet* y del *middleware* que componen el sistema.

Capítulo 2: Caracterización y diseño del sistema a implementar.

Se realiza una descripción detallada del problema existente, así como las particularidades del sistema a desarrollar. En este capítulo además, está contenida la descripción de los principales conceptos mediante un modelo de dominio, así como las historias de usuarios, las descripciones de las mismas y los requisitos no funcionales. Además de la estimación de esfuerzo correspondiente al sistema propuesto.

Capítulo 3: Implementación y pruebas del sistema.

En este se muestra el diseño de la solución, las clases que posee el *applet* y el *middleware* mediante las tarjetas CRC, y las tareas de ingeniería, además se desarrollan las pruebas al sistema para verificar que la implementación de las historias de usuarios han sido desarrolladas correctamente.

Capítulo 1: Fundamentación teórica

1.1 Introducción

Hoy en día, con el desarrollo de la medicina, la historia clínica no se limita a narrar la evaluación médica de un paciente, sino que también deja plasmado en este documento, datos, valoraciones e informaciones de cualquier índole sobre su situación a lo largo del proceso asistencial. Desde el punto de vista clínico puede entonces afirmarse que la historia clínica se origina con el primer episodio de la enfermedad o el control de salud en cuyo contesto se atiende el paciente. (5)

Este capítulo refleja un estudio crítico y valorativo, de la situación actual en el mundo y en Cuba acerca de algunos sistemas que se utilicen con el fin de gestionar información de historiales clínicos electrónicos y que hagan uso de tarjetas inteligentes, así como las tendencias tecnológicas más utilizadas por los desarrolladores de este tipo de soluciones. Además, se definen las herramientas y tecnologías que son utilizadas en el desarrollo del sistema.

1.1.1 Historiales clínicos

La historia clínica se define como el conjunto de documentos que contienen los datos, valoraciones e informaciones de cualquier índole sobre la situación y evolución clínica de un paciente a lo largo del proceso asistencial. Está constituida por el conjunto de documentos, tanto escritos como gráficos, que hacen referencia a los episodios de salud y enfermedad de una persona y a la actividad sanitaria que se genera con motivo de esos episodios.(Carnicero, 2010)

La historia clínica electrónica supone incorporar las Tecnologías de la Informática y las Comunicaciones en el núcleo de la actividad sanitaria. Esto trae como consecuencia que la historia deje de ser un registro de información generada en la relación entre un paciente y un profesional o un centro sanitario y se convierta en un soporte electrónico con toda la información referente al paciente y a su atención. Es accesible, con las limitaciones apropiadas, en todos los casos en los que se precisa asistencia clínica. (6) La misión de esta es proteger toda la información del paciente, para una mejor atención, ya que esta es única para cada uno de ellos.

- ✓ **¿Por qué utilizar la historia clínica electrónica?**
- ✓ **Problemas de la historia clínica en papel**

La historia clínica escrita a mano, es decir en papel, plantea algunos problemas, dentro de los que se encuentran:

- Desorden y falta de uniformidad en los documentos.
- Información ilegible.
- La información no es variable.
- Fácil acceso a la información.
- Errores de archivado parciales.
- Dudosa garantía de confidencialidad (la historia circula por todo el centro sanitario).
- Deterioro del soporte documental debido, por ejemplo, a accidentes.
- Dificultad para separar lo datos de filiación de los clínicos.

Estas necesidades son más fáciles de resolver en el caso de la historia clínica electrónica. La informatización de la historia clínica debe facilitar la solución de los problemas anteriores. Además, es la oportunidad de llevar a cabo la integración de la información clínica y revisar la organización de los servicios y profesionales, mejorando así la asistencia sanitaria.(Carnicero, 2010)

✓ **Ventajas de la historia clínica electrónica**

- Permite la actualización desde cualquier puesto de trabajo y desde cualquier lugar, evitando los tiempos de traslado por el centro sanitario.
- Facilita la explotación de datos para medir y mejorar la calidad de la asistencia sanitaria.
- Imposibilidad de alteración o manipulación por parte de personas no autorizadas.
- Disponibilidad de uso para varios profesionales a la vez.(7)

✓ **Arquitectura de la historia clínica electrónica**

Una arquitectura de historia clínica electrónica debe proporcionar constructores para capturar el significado de la información y asegurar que lo escrito sea comunicable. (2)

Esta arquitectura debe cumplir con los siguientes requisitos:

- Capturar el significado original de las anotaciones de la historia clínica.
- Proporcionar un marco apropiado para analizar e interpretar las historias.
- Permitir la comunicación fidedigna de información clínica entre distintos profesionales, con independencia de su ubicación y cumpliendo los requisitos legales.(2)

✓ **Los estándares en la historia clínica electrónica**

Un estándar o norma, es un documento establecido por consenso y aprobado por un organismo reconocido, que provee para un uso repetido y habitual, reglas, guías o características para las actividades o sus resultados.(2)

✓ **Estándar ISO sobre requisitos de la arquitectura de la historia clínica electrónica**

La especificación técnica ISO 18308 *Requirements for an Electronic Health Record Reference Arquitecture* soporta el uso, compartimento e intercambio de registros electrónicos a través de diferentes sectores de salud, países, y diferentes modelos de asistencia sanitaria.(2)

✓ **Estándar CEN TC 251 (prENV 13606-1*)***

Comunicación con la historia clínica electrónica: esta norma específica la arquitectura de información requerida para las comunicaciones interoperables entre sistemas y servicios que proveen o necesitan datos de la HCE. No pretende especificar la arquitectura interna o el diseño de la base de datos de tales sistemas. Más bien trata de:

- Capturar el significado original pretendido por el autor de un registro o conjunto de entradas.
- Facilitar un marco apropiado a las necesidades de los profesionales y las organizaciones, para analizar la HCE sobre una base individual o poblacional.
- Incorporar los instrumentos médico legales necesarios para soportar la comunicación segura y relevante de elementos de la HCE entre profesionales que trabajen en el mismo o en distintos lugares. (2)

✓ **Estándar *Health Level Seven* (HL7)**

Health Level Seven (HL7) está acreditado por el Instituto Americano de Estándares Nacionales (ANSI). HL7 establece los criterios para representación y comunicación de datos relacionados con el cuidado de la salud. HL7 se concentra en la clínica y aspecto administrativo de los datos generados dentro y fuera de las instituciones de salud.(8)

Se analiza dentro de este los siguientes estándares:

- Mensajería HL7 Versión 2: Estándar de mensajería para el intercambio electrónico de datos de salud.
- Mensajería HL7 Versión 3: Estándar de mensajería para el intercambio electrónico de datos de salud basada en el RIM (*Reference Information Model*).

- CDA HL7: (*Clinical Document Architecture*) Estándar de arquitectura de documentos clínicos electrónicos.
- SPL HL7: (*Structured Product Labeling*) Estándar electrónico de etiquetado de medicamentos.
- HL7 *Medical Records*: Estándar de administración de Registros Médicos.
- GELLO: Estándar para la expresión de reglas de soporte de decisiones clínicas.
- Arden Sintax: Estándar sintáctico para compartir reglas de conocimiento clínico.
- CCOW: Es un estándar *framework* para compartir contexto entre aplicaciones.(9)

En la realización del sistema se utiliza el estándar de arquitectura de documentos clínicos electrónicos de HL7 y el estándar CEN TC 251 (prENV 13606-1) ya que este cumple con las siguientes características:

- Persistencia: La estructura de los datos que se definen en el sistema es por la necesidad de la localidad donde se encuentra, y persistirá en su estado natural por un período de tiempo mientras la necesidad sea la misma.
- Administración: El documento clínico estará gestionado por la Universidad de las Ciencias y esta se encargará de su cuidado.
- Potencial para la autenticación: Para acceder a la historia clínica de un paciente se necesitará de la autenticación para acceder a la información.
- El sistema facilitará un marco apropiado para que los profesionales, analicen la HCE como base individual.

✓ **Principales cuestiones legales de la historia clínica electrónica**

Las principales cuestiones legales de la historia clínica son:

✓ **Admisión de la historia clínica, información y consentimiento del interesado**(9)

Es obligatorio informar al interesado de la existencia la historia clínica informatizada y de la identidad y dirección de su responsable.

✓ **Validez legal**

La historia clínica electrónica puede sustituir a la de papel, y tendrá el mismo valor que la de papel.

✓ **Acceso del paciente**

El paciente tiene todo el derecho de acceder a la información de su historia clínica, y sus familiares en caso de que este haya muerto si antes de hacerlo este no se opuso.

✓ **Conservación y cancelación**

El tratamiento legal de la conservación de la historia clínica y, correlativamente, de las condiciones de cancelación de los datos contenidos en ella no resuelve satisfactoriamente las cuestiones suscitadas en relación con esta materia.

✓ **Acceso a los profesionales**

Los profesionales que participen en el diagnóstico y tratamiento del paciente pueden acceder a los datos de la historia clínica en las siguientes condiciones:

- Previa constancia e identificación en el correspondiente documento de seguridad.
- Deben existir procedimientos de identificación y autentificación, así como controles, para el acceso a la información.
- Debe existir un registro de accesos que conserve la información de detalle relativa a cada acceso.(9)

1.1.2 Tarjetas inteligentes

La tarjeta inteligente surge ante nuevas necesidades del mercado, las cuales no pueden ser satisfechas por la tarjeta de banda magnética. No necesita la conexión en línea, ya que el chip interno, es capaz de realizar operaciones por sí mismo, a diferencia de la banda magnética que necesita la conexión, y esto acarrea un alto costo en la conexión, a través de las telecomunicaciones.

Esta tecnología tiene su origen en la década del 70 cuando inventores de Alemania, Japón y Francia inscribieron las patentes originales. Debido a varios factores que se presentaron, y de los cuales la inmadura tecnología de semiconductores tuvo un mayor peso, muchos trabajos sobre tarjetas inteligentes (*Smart Cards*) estuvieron en investigación y desarrollo hasta la primera mitad de los años ochenta.

Una tarjeta inteligente (*Smart Card*) es del mismo tamaño que una tarjeta de crédito, que almacena y procesa información a través de circuitos electrónicos incrustados en el plástico de la tarjeta. Estas tarjetas son portable y resistente, y tienen poder de procesamiento e información y no requieren estar conectadas constantemente a una base de datos.(10)

1.1.2.1 Seguridad en las tarjetas inteligentes

La seguridad es una de las propiedades más importantes de las tarjetas inteligentes y se aplica a múltiples niveles y con diferentes mecanismos. Las medidas de protección van desde los elementos empleados en el material de la tarjeta, así como las características usadas en la impresión gráfica que

dificultan la reproducción y alteración de las mismas; hasta reglas que se establecen para el acceso a la información contenida en el chip, según sus niveles de confidencialidad.(11)

1.1.2.1.1 Seguridad física

El conjunto de medidas de seguridad que se pueden implementar para un documento de identidad se dividen en diferentes niveles:

- Nivel 1: Perceptibles mediante la vista al observar el documento. No requieren de herramientas especiales, por lo cual no es necesario entrenamiento. Pueden ser de conocimiento público.
- Nivel 2: Características escondidas que son visibles mediante equipos simples, luz ultravioleta. No requieren de un entrenamiento especial de los oficiales de seguridad.
- Nivel 3: Características de seguridad que requieren de un entrenamiento al personal y de un equipamiento especial para detectarlas, por ejemplo: un microscopio o lupas especiales.
- Nivel 4: Características de un alto nivel de seguridad sobre las que sólo tienen conocimiento el personal requerido y que únicamente pueden verse mediante un equipamiento en un laboratorio especializado. (11)

1.1.2.1.2 Seguridad lógica

En las tarjetas se implementan distintos niveles de seguridad sobre los ficheros en dependencia de la importancia de la información contenida en ellos. Algunos elementos de seguridad son:

- Especificación de un conjunto de reglas para ejecutar comandos o acceder a datos, las cuales se denominan condiciones de acceso.
- Mensajería segura para la autenticación e integridad del intercambio de datos entre la tarjeta y los lectores.
- Protección de la información por claves que deben presentarse a la tarjeta y son verificadas por su sistema operativo.
- Contador de confirmación para evitar agresiones repetidas contra los valores secretos.
- Mecanismos de autenticación mutua de la tarjeta y los terminales para garantizar que sólo elementos autorizados puedan tener acceso a la información.
- Funciones criptográficas que permiten los procesos de firma digital para prevenir repudios.
- Utilización de la infraestructura de llave pública (PKI).
- Verificación biométrica por parte de la tarjeta. (11)

Las tarjetas inteligentes se pueden clasificar en dos tipos atendiendo a su interfaz:

1.1.2.2 Tarjetas inteligentes sin contacto

El primer tipo es la tarjeta inteligente sin contacto mediante etiquetas RFID[8] en el cual el chip se comunica con el lector de las tarjetas mediante inducción a una tasa de transferencia de 106 a 848 Kb/s. El estándar de comunicación de tarjetas inteligentes sin contacto es el ISO/IEC 14443. Define dos tipos de tarjetas sin contacto, permitidos para distancias de comunicación de hasta 10 cm. Ha habido propuestas para la ISO 14443 [9] de otros tipos que todavía tienen que completar el proceso de estandarización. Un estándar alternativo de tarjetas inteligentes sin contacto es el ISO 15693[10], el cual permite la comunicación a distancias de hasta 50 cm. Las más abundantes son las tarjetas de la familia MIFARE[11] de Philips, las cuales representan a la ISO/IEC 14443-A. (12)

1.1.2.3 Tarjeta inteligente de contacto

El segundo tipo de estas tarjetas disponen de unos contactos metálicos visibles y debidamente estandarizados en el ISO/IEC 7816[12]. Estas tarjetas, por tanto, deben ser insertadas en una ranura de un lector para poder operar con ellas. A través de estos contactos el lector alimenta eléctricamente a la tarjeta y transmite los datos oportunos para operar con ella conforme al estándar. Cada vez es más común ver tarjetas inteligentes del gobierno o de los proveedores de servicios en los bolsillos de sus ciudadanos o clientes, gracias a sus tres características principales:

[8] RFID :siglas de Radio Frequency IDentification, en español identificación por radiofrecuencia es un sistema de almacenamiento y recuperación de datos remoto que usa dispositivos denominados etiquetas, tarjetas, transpondedores o tags RFID.

[9] ISO 14443: es un estándar internacional relacionado con las tarjetas de identificación electrónicas, en especial las tarjetas inteligentes, gestionado conjuntamente por la Organización Internacional de Normalización (ISO) y Comisión Electrotécnica Internacional (IEC).

[10] ISO 15693: es un estándar ISO para "Tarjetas de Vecindad" (Vicinity Cards), como por ejemplo las tarjetas que pueden ser leídas desde una mayor distancia que las tarjetas de proximidad.

[11] MIFARE: es la tecnología de tarjetas inteligentes sin contacto (TISC) más ampliamente instalada en el mundo.

[12] ISO/IEC 7816: es un estándar internacional relacionado con las tarjetas de identificación electrónicas, en especial las tarjetas inteligentes, gestionado conjuntamente por la Organización Internacional De Normalización (ISO) y Comisión Electrotécnica Internacional RUPI (IEC). Se trata de una extensión de la ISO 7810.

- Las tarjetas inteligentes son personales y la mayoría de las veces están protegidas por un código PIN[13].
- Las tarjetas inteligentes permiten una movilidad total: uno puede guardárselas en la cartera y llevarlas a cualquier lugar.
- Las tarjetas inteligentes son muy seguras: integran prestaciones criptográficas y guardan las claves secretas con un alto nivel de seguridad. (12)

1.1.2.4 Comunicación de la tarjeta

Cuando dos computadoras se comunican mutuamente, ellas intercambian paquetes de datos, los cuales son construidos siguiendo un conjunto de protocolos. En forma similar, las tarjetas inteligentes se comunican al mundo exterior usando sus propios paquetes de datos llamados APDU (*Application Protocol Data Units*). Los APDU pueden contener un mensaje de comando o un mensaje de respuesta. En el mundo de las tarjetas, el modelo amo – esclavo, es usado. Por lo cual una tarjeta inteligente siempre juega el rol pasivo. En otras palabras, una tarjeta inteligente siempre espera por un APDU comando desde una terminal. Esta entonces ejecuta la acción especificada en el APDU y responde a la terminal con un APDU respuesta. APDUs comandos y APDUs respuestas son intercambiados alternativamente entre una tarjeta y una terminal. (13)

1.1.2.4.1 Estructura de los APDU[14]

Unidad de datos de protocolo de aplicación (*Application Protocol Data Unit*), es la unidad de comunicación entre un lector y una tarjeta. Su estructura está definida en el estándar ISO 7816, existiendo dos tipos de categorías de APDU, APDU comandos y APDU Respuesta. (14) **(Ver Ilustración 11)**

1.1.2.5 Organización para el intercambio de información

Para organizar el intercambio de información entre la tarjeta inteligente y la interfaz de comunicación, por ejemplo, el lector de tarjetas inteligentes, se especifican los siguientes rasgos básicos: (14)

[13] PIN: (Personal Identification Number o Número de Identificación Personal en español) es un valor numérico usado para identificarse y poder tener acceso a ciertos sistemas o artefactos, como un teléfono móvil o un cajero automático.

[14] El Application Protocol Data Unit (APDU) es la unidad de comunicación entre un lector de tarjetas inteligentes y una tarjeta inteligente. La estructura de un APDU está definida en los estándares ISO/IEC 7816.

✓ **Parejas de comando-respuesta** (14)

Los comandos y respuestas se envían o se reciben de la tarjeta inteligente en forma de APDU, que es la unidad de comunicación entre un lector de tarjetas inteligente y la propia tarjeta. Su estructura es la siguiente:

- **Cabecera:**

1 byte para denotar la clase (CLA).

1 byte para denotar la instrucción (INS).

2 bytes para denotar los parámetros (P1-P2).

- **Campo Lc:** Se compone 0, 1 o 3 bytes si la longitud del campo de datos es mayor que cero. En caso contrario estará ausente.
- **Campo de datos:** bytes correspondientes a la longitud de la cadena de datos a enviar. Si no hay datos a enviar, no se especificará ningún valor en este campo.
- **Campo Le:** número de bytes esperados en la respuesta. Si no se espera ninguna cadena de bytes en la respuesta, no se especificará ningún valor en este campo. (14)

1.1.2.6 Estándares utilizados en tarjetas inteligentes

1.1.2.6.1 ISO 7816

- Estándares internacionales para tarjetas con circuito integrado (tarjetas inteligentes). El objetivo de estos estándares es lograr la interoperabilidad entre distintos fabricantes de tarjetas inteligentes y lectores de las mismas, en lo que respecta a características físicas, comunicación de datos y seguridad. Estos estándares son basados en los ISO 7810 e ISO 7811, los cuales definen características físicas de tarjetas de identificación. (14)

La apariencia física de las tarjetas inteligentes está especificada por la Organización Internacional de Normalización (ISO) y Comisión Electrotécnica Internacional (IEC). La ISO 7816 presenta partes de la norma, las cuales son:

- 7816-1: Características físicas.
- 7816-2: Tarjetas con contactos - Dimensiones y localización de los contactos.
- 7816-3: Características eléctricas.
- 7816-4: Pares comando-respuesta.

- 7816-5: Registro de la solicitud de los proveedores.
- 7816-6: Interoperabilidad en los elementos de datos para el intercambio.
- 7816-7: Interoperabilidad en los comandos de la tarjeta (SCQL).
- 7816-8: Comandos para operaciones de seguridad.
- 7816-9: Comandos para la gestión de la tarjeta.
- 7816-10: Señales electrónicas para operación síncrona.
- 7816-11: Verificación de la identidad personal a través de métodos biométricos.
- 7816-12 Tarjetas con contactos. Interfaz eléctrica USB y procedimientos operativos.
- 7816-13: Comandos de administración de aplicaciones en múltiples aplicaciones entorno.
- 7816-15: Aplicación de información criptográfica.(10)

Para la implementación de nuestra solución nos centramos en el 7816-4 que define:

- El contenido de los pares comando-respuesta que se intercambian a nivel de interfaz.
- Estructuras para aplicaciones y datos en la tarjeta.
- La estructura y contenido de los caracteres históricos de la respuesta de Reset[15], los cuales describen las características de operación de la tarjeta inteligente.
- Métodos para extracción de objetos y elementos de datos de la tarjeta.
- Mecanismos para la identificación y direccionamiento de aplicaciones en la tarjeta.
- Estructuras de archivos y métodos de acceso.
- Arquitectura de seguridad para derechos de acceso a los archivos y datos en la tarjeta.
- Comandos orientados a objetos de datos.
- Métodos de acceso a los algoritmos que procesa la tarjeta inteligente (no describe los algoritmos).
- Métodos para el intercambio seguro de mensajes. (14)

[15] Reset: Se conoce como la puesta en condiciones iniciales de un sistema. Este puede ser mecánico, electrónico o de otro tipo. Normalmente se realiza al conectar el mismo, aunque, habitualmente, existe un mecanismo, normalmente un pulsador, que sirve para realzar la puesta en condiciones iniciales manualmente.

1.1.2.6.2 PC/SC

PC/SC es un grupo de desarrollo cuyo objetivo es el de promover una especificación estándar, que asegure la interoperabilidad entre tarjetas inteligentes, lectores de tarjetas inteligentes y computadoras. PC/SC desarrolló una especificación independiente de la plataforma, que puede ser implementada sobre cualquier sistema operativo. Fue construido sobre los estándares actuales de *Smart Card* (ISO 7816), definiendo interfaces de bajo nivel para dispositivos y APIs independientes del dispositivo. La especificación actual es la PC/SC *Specification* 1.0. La misma incluye los siguientes temas, entre otros:

- Provee la arquitectura del sistema y de sus componentes.
- Detalla las características y requerimientos de compatibilidad de las tarjetas y los dispositivos
- Presenta una discusión con consideraciones de diseño para los dispositivos, y recomendaciones de implementación.
- Describe los componentes con los que debe contar el sistema.
- Presenta consideraciones de diseño para desarrolladores de aplicaciones, indicando cómo hacer uso de los componentes. (14)

1.1.2.6.3 GlobalPlatform

Es una organización independiente enfocada a gestionar una infraestructura estandarizada para el desarrollo y despliegue de tarjetas inteligentes. Proporciona un conjunto de especificaciones universalmente reconocidas e implementadas, junto con configuraciones de mercado, aplicación de esas especificaciones y documentos de apoyo. Cubriendo toda la infraestructura de tarjetas inteligentes (las tarjetas, dispositivos y sistemas) estos documentos técnicos ofrecen una plataforma tecnológica dinámica y completa para el desarrollo de programas de tarjetas inteligentes, para poder establecer una conexión segura con la misma y administrar sus aplicaciones.

Las tarjetas, dispositivos y sistemas *Global Platform*, son interoperables, independientemente de la tecnología del proveedor y la flexibilidad de su infraestructura técnica, garantizan que pueda responder a las necesidades básicas en el instante del despliegue inicial. Ofreciendo a los emisores la seguridad de que la infraestructura que han elegido será capaz de adaptarse y crecer a medida que cambian las condiciones de negocios. (15)

1.1.2.6.4 ISO/IEC 14443

El estándar ISO 14443 está relacionado con las tarjetas de identificación electrónicas, en especial con las tarjetas inteligentes, gestionado conjuntamente por la Organización Internacional de Normalización (ISO) y Comisión Electrotécnica Internacional (IEC). Este estándar define una tarjeta de proximidad, utilizada para identificación y pagos que por lo general utiliza el estándar tarjeta de crédito definida por ISO 7816 - ID 1 (aunque otros formatos son posibles).

El sistema RFID utiliza un lector con un micro controlador incrustado y una antena que opera a 13,56 MHz (frecuencia RFID). El lector mantiene a su alrededor un campo electromagnético de modo que al acercarse una tarjeta al campo, ésta se alimenta eléctricamente de esta energía inducida y puede establecerse la comunicación lector-tarjeta.

El estándar ISO 14443 consta de cuatro partes y se describen dos tipos de tarjetas: tipo A y tipo B. Las principales diferencias entre estos tipos son los métodos de modulación, codificación de los planes (parte 2) y el protocolo de inicialización de los procedimientos (parte 3). Las tarjetas de ambos tipos (A y B) utilizan el mismo protocolo de alto nivel (llamado T=CL) que se describe en la parte 4. El protocolo T=CL especifica los bloques de datos y los mecanismos de intercambio:

1. Bloque de datos de encadenamiento.
2. Tiempo de espera de extensión.
3. Múltiple activación. (16)

1.1.3 Tarjeta de salud

Con el desarrollo de sistemas informáticos clínicos se ha puesto en práctica la utilidad de las tarjetas de salud, esta ofrece recursos adicionales para la aplicación de tarjetas inteligentes sanitarias. Estas contienen toda la información referente al paciente, durante su proceso asistencial. Cuando se inició la emisión de tarjetas sanitarias, se advertía una preocupación mayor por la creación de la base de datos y por la acreditación de las prestaciones, que por los problemas de identificación de las personas. Estas prioridades eran consecuencia lógica de las necesidades de planificación y gestión que se tenían en ese momento.

La utilidad de la tarjeta sanitaria como un instrumento de identificación de los usuarios se ha puesto en evidencia con el desarrollo de los sistemas de información clínica, sobre todo al relacionar sistemas distintos.(2)

1.1.4 Applet

Un *applet* es un pequeño programa que no pretende ser ejecutado por su propia cuenta, sino más bien ser incorporado dentro de otra aplicación.

La clase *Applet* debe ser la superclase de cualquier *applet* que se incrusta en una página Web o visto por el visor de *applet* de Java. La clase *Applet* proporciona una interfaz estándar entre *applets* y su entorno. Entre sus características podemos mencionar, un esquema de seguridad que permite que las aplicaciones que se ejecutan en el equipo, no tengan acceso a partes sensibles (por ejemplo. no pueden escribir archivos), a menos que uno mismo le dé los permisos necesarios en el sistema; la desventaja de este enfoque es, que la entrega de permisos es engorrosa para el usuario común, lo cual juega en contra de uno de los objetivos de los Java *applets*: proporcionar una forma fácil de ejecutar aplicaciones desde el navegador web.(17)

1.1.5 Middleware

El *middleware* es un software es un software destinado a proporcionar conectividad, interoperalidad o integración entre diferentes aplicaciones, normalmente distribuidas, y en el peor de los casos, sobre recursos heterogéneos. Funciona como una capa de abstracción de software distribuida, que se sitúa entre las capas de aplicaciones y las capas inferiores (sistema operativo y red). El *middleware* abstrae de la complejidad y heterogeneidad de las redes de comunicaciones subyacentes, así como de los sistemas operativos y lenguajes de programación, proporcionando una API[16] para la fácil programación y manejo de aplicaciones distribuidas. Dependiendo del problema a resolver y de las funciones necesarias, serán útiles diferentes tipo de servicios de *middleware*.

Por lo general el *middleware* del lado cliente está implementado por el sistema operativo subyacente, el cual posee las bibliotecas que implementan todas las funcionalidades para la comunicación a través de la red. (18)

[16]API: Una interfaz de programación de aplicaciones o API (del inglés *Application Programming Interface*) es el conjunto de funciones y procedimientos que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción. Son usados generalmente en las bibliotecas.

1.2 Estado del arte

1.2.1 Nivel internacional

1.2.1.1 Health Care Smart Card (Taiwan, República de China)

El proyecto de atención de salud de Taiwán de la tarjeta inteligente es uno de los más grandes de salud soluciones de tarjetas inteligentes en el mundo y el primero en Taiwan, República de China. La infraestructura de tarjetas inteligentes del proyecto original responde a la historia clínica en papel.

La tarjeta se renueva después de que el paciente utiliza los servicios médicos hasta seis veces. Contiene fuertes requisitos de seguridad para la información de salud, utilizando el PIN para proteger dicha información, y se guía por los estándares de HL7 para conformar los datos almacenados. (19)

1.2.1.2 EMedix, México

EMedix es un sistema de información para Médicos, utilizado desde el 2005, en el cual al momento de atender a un paciente, pueden registrar toda la información generada en la consulta, y se tiene el acceso a la información generada durante las visitas anteriores del paciente.

En eMedix, durante la consulta el Médico elabora la Nota Médica, genera la receta o prescripción, registra los signos vitales, agrega anotaciones a la Historia Clínica del paciente, registra resultados de estudios radiológicos o análisis clínicos y consulta la información generada durante las visitas anteriores del paciente. Dado que eMedix está desarrollado con tecnologías de vanguardia de internet, usted estará siempre actualizado ya que es un servicio de suscripción. EMedix, es más que un sistema médico, es una herramienta invaluable para la administración de la información médica de sus pacientes y consultas, todo desde una clara y sencilla interfaz que se puede acceder desde cualquier computadora que tenga acceso a internet. (20)

1.2.1.3 Health Card de LifeNexus(Registros Médicos en una Tarjeta personal de salud) (EE.UU)

Los EE.UU presentó hace poco su tarjeta electrónica de salud que combina el registro médico de una persona con la opción de una tarjeta de crédito. Antes de que un paciente pueda usar la tarjeta, el sistema LifeNexus requiere la descarga de un software a la computadora para su funcionamiento. La funcionalidad múltiple de esta tarjeta proporciona otro beneficio, si la persona desea colocar su tarjeta de salud en una tarjeta de pago, puede manejar su información médica y a la vez realizar sus compras generales diarias, algo en verdad muy interesante. La Tarjeta Personal de Salud de LifeNexus está activada por una tecnología *iChip* (*Individually Controlled Health Information Platform*), que utiliza la tecnología de un

servidor móvil integrada a una tarjeta chip que está codificada y protegida por una clave, proporcionando un ambiente de alta seguridad para almacenar información comprensiva y que puede salvar la vida de individuos y de sus familias. (21)

1.2.2 Nivel nacional

En Cuba en este momento se encuentra en un nivel 1, o sea, sólo tiene automatizada la labor de algunos departamentos aislados, en algunos centros médicos. (Ejemplo, en el Hospital "Hermanos Almejeiras"). El Ministerio de Salud Pública (MINSAP[17]) no está exento de tal política, en este sentido un elemento básico en la informatización de la salud pública cubana lo constituye la Historia Clínica Electrónica (HCE). La creación de un sistema de HCE, por lo tanto, debe ser un objetivo prioritario para el país. (22)

Después de haber realizado el estudio de los sistemas descritos anteriormente que utilizan HCE en tarjetas inteligentes y sin ellas. Se han seleccionado varias funcionalidades que estos presentan para que se encuentren en el sistema a desarrollar, como son:

- Almacenar los datos del paciente en su historia clínica en un contenedor de información.
- Observar los datos de la historia clínica del paciente que fueron almacenados en consultas anteriores.
- Proteger la información de la historia clínica, utilizando un Número de Identificación Personal (PIN).

1.3 Metodologías para el desarrollo del software

Cada metodología de desarrollo de software tiene más o menos su propio enfoque para el desarrollo de software. Este es un marco de trabajo usado para estructurar, planificar y controlar el proceso de desarrollo en sistemas de información. Hoy en día existen numerosas propuestas metodológicas que inciden en distintas dimensiones del proceso de desarrollo. Un ejemplo de ellas son las propuestas tradicionales centradas específicamente en el control del proceso. Estas han demostrado ser efectivas y necesarias en un gran número de proyectos, sobre todo aquellos proyectos de gran tamaño (respecto a tiempo y recursos).(16)

[17] MINSAP: Ministerio de Salud Pública de Cuba es el organismo rector del Sistema Nacional de Salud, encargado de dirigir, ejecutar y controlar la aplicación de la política del Estado y del gobierno en cuanto a la Salud pública, el desarrollo de las Ciencias Médicas y la industria médico-farmacéutica.

Sin embargo, la experiencia ha demostrado que las metodologías tradicionales no ofrecen una buena solución para proyectos donde el entorno es volátil y donde los requisitos no se conocen con exactitud, porque no están pensadas para trabajar con incertidumbre.

Aplicar metodologías tradicionales nos obliga a forzar a nuestro cliente a que tome la mayoría de las decisiones al principio. Luego el coste de cambio de una decisión tomada puede llegar a ser muy elevado. Ejemplo de metodología tradicional es RUP[18]. El Proceso Unificado de Racional (RUP, por sus siglas en inglés *Rational Unified Process*) es un proceso de desarrollo de software y junto con el Lenguaje Unificado de Modelado (UML), constituye la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos. RUP es en realidad un refinamiento realizado por *Rational Software* del más genérico Proceso Unificado. Proporciona una guía en el orden de las actividades de un equipo, dirige las tareas individuales de los desarrolladores, especifica qué productos deberían ser desarrollados y ofrece criterios para monitorear y medir los productos y actividades del proyecto.

Como respuesta a los problemas aplicando metodologías tradicionales surgieron otras metodologías que tratan de adaptarse a la realidad del desarrollo de software. Las metodologías ágiles buscan un justo medio entre ningún proceso y demasiado proceso, proporcionando simplemente suficiente proceso para que el esfuerzo valga la pena. (16)

La metodología XP[19] es una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores, y propiciando un buen clima de trabajo. XP se basa en realimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios. XP se define como especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes, y donde existe un alto riesgo técnico. (16)

[18] RUP: El Proceso Racional Unificado (Rational Unified Process en inglés, habitualmente resumido como RUP) es un proceso de desarrollo de software y junto con el Lenguaje Unificado de Modelado UML, constituye la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos.

[19] XP: La programación extrema o *eXtreme Programming* (XP) es un enfoque de la ingeniería de software formulado por Kent Beck.

✓ **Características fundamentales.**

Las características fundamentales del método son:

- **Desarrollo iterativo e incremental:** pequeñas mejoras, unas tras otras.
- **Pruebas unitarias continuas:** frecuentemente repetidas y automatizadas. Se aconseja escribir el código de la prueba antes de la codificación.
- **Programación en parejas:** se recomienda que las tareas de desarrollo se lleven a cabo por dos personas en un mismo puesto. Se supone que la mayor calidad del código escrito de esta manera el código es revisado y discutido mientras se escribe- es más importante que la posible pérdida de productividad inmediata.
- **Integración del equipo de programación con el cliente o usuario:** se recomienda que el cliente trabaje junto al equipo de desarrollo.
- **Corrección de todos los errores:** antes de añadir nueva funcionalidad. Hacer entregas frecuentes.
- **Refactorización del código:** es decir, reescribir ciertas partes del código para aumentar su legibilidad y mantenibilidad pero sin modificar su comportamiento. Las pruebas han de garantizar que en la refactorización no se ha introducido ningún fallo.
- **Propiedad del código compartida:** en vez de dividir la responsabilidad en el desarrollo de cada módulo en grupos de trabajo distintos, este método promueve que todo el personal pueda corregir y extender cualquier parte del proyecto. Las frecuentes pruebas de regresión garantizan que los posibles errores serán detectados.
- **Simplicidad en el código:** es la mejor manera de que las cosas funcionen. Cuando todo funcione se podrá añadir funcionalidad si es necesario. La programación extrema apuesta que es más sencillo hacer algo simple y tener un poco de trabajo extra para cambiarlo si se requiere. (23)

✓ **Prácticas (Ver Ilustración 12)**

- **Planificación Incremental:** es un espacio frecuente de comunicación entre el cliente y los programadores.
- **Pruebas:** no debe existir ninguna característica en el programa que no haya sido probada.
- **Programación en parejas:** el emparejamiento es dinámico.

- **Refactorización:** prepara nuestro sistema para que en un futuro acepte nuevos cambios y pueda albergar nuevas características
- **Diseño simple:** se debe diseñar la solución más simple que pueda funcionar y ser implementada en un momento determinado del proyecto.
- **Propiedad colectiva del código:** ningún miembro del equipo es propietario del código.
- **Integración continua:** el código se debe integrar como mínimo una vez al día, y realizar las pruebas sobre la totalidad del sistema.
- **Cliente en el equipo:** un cliente real debe sentarse con el equipo de programadores, estar disponible para responder a sus preguntas, resolver discusiones y fijar las prioridades.
- **Entregas pequeñas:** la idea es producir rápidamente versiones del sistema que sean operativas.
- **Semanas de 40 horas:** esto requiere que trabajemos 40 horas a la semana, la regla de XP dice nunca 2 semanas seguidas realizando horas extras.
- **Estándares de codificación:** se debe establecer un estándar de codificación aceptado e implantado por todo el equipo.
- **Uso de metáforas:** las metáforas ayudan a cualquier persona a entender el objetivo del programa. (16)

✓ **Artefactos esenciales en XP**

- Historias del usuario.
- Tareas de ingeniería.
- Pruebas de aceptación.
- Pruebas unitarias y de integración.
- Plan de entrega.
- Código (16)

✓ **Faces de XP (Ver Ilustración 13)**

Un proyecto XP tiene éxito cuando el cliente selecciona el valor de negocio a implementar basado en la habilidad del equipo para medir la funcionalidad que puede entregar a través del tiempo. El ciclo de desarrollo consiste (a grandes rasgos) en los siguientes pasos:

- El cliente define el valor de negocio a implementar.
- El programador estima el esfuerzo necesario para su implementación.

- El cliente selecciona qué construir, de acuerdo con sus prioridades y las restricciones de tiempo.
- El programador construye ese valor de negocio.
- Vuelve al paso 1. (16)

El ciclo de vida ideal de XP consiste de seis fases: Exploración, Planificación de la Entrega (Release[20]), Iteraciones, Producción, Mantenimiento y Muerte del Proyecto.

✓ **Se escoge XP como metodología a utilizar por:**

- Necesidad de resultados tangibles a corto plazo.
- El tamaño del grupo de desarrollo, en este caso, de apenas dos personas.

1.4 Herramientas para el desarrollo del software

1.4.1 Developer Suite Gemalto

Poderosa herramienta que brinda un ambiente favorable para el diseño y la implementación de *applets*, además nos posibilita simular las funcionalidades de los *applets* antes de ser instalados en las tarjetas inteligentes.(14)

1.4.2 UML[21] como lenguaje de modelación visual

Es el lenguaje en el que esta descrito el modelo, para visualizar, especificar, construir y documentar un sistema. UML ofrece un estándar para describir un plano del sistema modelo, incluyendo aspectos conceptuales tales como procesos de negocio y funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes reutilizables.(24)

1.4.3 UModel Altova

Diseña visualmente modelos de aplicación en UML y genera código Java, C #. NET y documentación del proyecto. *UModel Altova* es la herramienta de UML que hace que el diseño de software visual sea práctico para cualquier proyecto. Es la forma más sencilla y rentable para dibujar en UML. (14)

[20] Release: Una versión candidata a definitiva o candidata para el lanzamiento, aunque más conocida por su nombre en inglés *release candidate*, comprende un producto final, preparado para publicarse como versión definitiva a menos que aparezcan errores que lo impidan.

[21] UML: Lenguaje Unificado de Modelado (LUM o UML, por sus siglas en inglés, Unified Modeling Language) es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad; está respaldado por el OMG (Object Management Group). Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema.

1.4.4 Visual Studio

Visual Studio es un entorno de programación que soporta multitud de lenguajes. Esta herramienta permite crear aplicaciones, sitios web, así como servicios web en cualquier entorno que soporte la plataforma .NET. Es un entorno de desarrollo integrado (IDE[22], por sus siglas en inglés) para sistemas operativos *Windows*. (14)

1.5 Plataforma .NET

1.5.1 Microsoft .NET *Framework*

El *Framework* de .Net es una infraestructura sobre la que se reúne todo un conjunto de lenguajes y servicios que simplifican enormemente el desarrollo de aplicaciones. Mediante esta herramienta se ofrece un entorno de ejecución altamente distribuido, que permite crear aplicaciones robustas y escalables. Este organiza toda la funcionalidad del sistema operativo en un espacio de nombres jerárquicos de forma que a la hora de programar resulta bastante sencillo encontrar lo que se necesita.

.Net *Framework* soporta múltiples lenguajes de programación y aunque cada lenguaje tiene sus características propias, es posible desarrollar cualquier tipo de aplicación con cualquiera de estos lenguajes. Existen más de 30 lenguajes adaptados a .Net, desde los más conocidos como C# (C Sharp), Visual Basic o C++ hasta otros lenguajes menos conocidos como Perl o Cobol. (MSDN[23]: Inicio de .NET *Framework*, 2010). (14)

Ventajas:

- Código administrado: El CLR[24] realiza un control automático del código para que este sea seguro, es decir, controla los recursos del sistema para que la aplicación se ejecute correctamente.
- Interoperabilidad multilenguaje: El código puede ser escrito en cualquier lenguaje compatible con .NET ya que siempre se compila en código intermedio o *Microsoft Intermediate Lenguaje* (MSIL).

[22] IDE: Un entorno de desarrollo integrado (en *inglés integrated development environment*) es un programa informático compuesto por un conjunto de herramientas de programación.

[23] MSDN: (Microsoft Developer Network) puede referirse tanto a los servicios web orientados a desarrolladores de software basado en plataformas Microsoft como al conjunto de software que se adjunta con sus compiladores (Visual Studio) y ciertos SDK.

[24] CLR: Las siglas de Common Language Runtime, componente de máquina virtual de la plataforma .Net de Microsoft.

➢ Compilación *just-in-time*: El compilador JIT (*Just In Time*, nombre que recibe ese tipo de compilación porque se realiza en tiempo de ejecución), incluido en el *Framework*, compila el código intermedio generando el código máquina propio de la plataforma. Se aumenta así el rendimiento de la aplicación al ser específico para cada plataforma.

➢ Despliegue: Por medio de los ensamblados resulta mucho más fácil el desarrollo de aplicaciones distribuidas y el mantenimiento de las mismas. El *Framework* realiza esta tarea de forma automática mejorando el rendimiento y asegurando el funcionamiento correcto de todas las aplicaciones. (14)

1.6 Lenguajes de programación

1.6.1 C#

C# es un lenguaje de programación orientado a objetos desarrollado por *Microsoft* y estandarizado como parte de su plataforma .NET. Aunque para la plataforma .NET es prácticamente posible programar en cualquier lenguaje, el C# es el lenguaje de propósito general diseñado por Microsoft para ser utilizado en ella, por lo que programar usando C# es mucho más sencillo e intuitivo que hacerlo con cualquiera de los otros. (14)

✓ **Características principales:**

➢ **Sencillez:** Elimina muchos elementos que otros lenguajes incluyen y que son innecesarios en .NET. Por ejemplo:

- El código escrito en este lenguaje es auto contenido, lo que significa que no necesita de ficheros adicionales a la propia fuente, como ficheros de cabecera.
- El tamaño de los tipos de datos básicos es fijo e independiente del compilador, sistema operativo o máquina para la cual se compile, lo que facilita la portabilidad del código.
- **Orientación a componentes:** la propia sintaxis del lenguaje incluye elementos propios del diseño de componentes que otros lenguajes tienen que simular mediante construcciones más o menos complejas. Es decir, la sintaxis de este permite definir cómodamente propiedades (similares a campos de acceso controlado), eventos (asociación controlada de funciones de respuesta a notificaciones) o atributos (información sobre un tipo o sus miembros).

➢ **Eficiente:** como principio, en C# el código incluye numerosas restricciones para asegurar su seguridad y no permite el uso de punteros. Sin embargo, y a diferencia de Java, en C# es posible saltarse dichas restricciones manipulando objetos a través de punteros, con sólo marcar regiones de código como

inseguras. En estos bloques pueden usarse los punteros de forma similar a cómo se hace en C++, lo que cual resulta vital en situaciones donde se necesite eficiencia y mayor velocidad de procesamiento. (14)

1.6.2 JavaCard

JavaCard es una combinación del lenguaje de Java con un entorno de ejecución para tarjetas inteligentes, permitiendo ejecutar pequeñas aplicaciones, llamadas *applets*, en el chip de estas. Dichos *applets* contienen funcionalidades que son reutilizables para otros componentes. Por lo que esta tecnología aporta beneficios al mundo, desarrollando sistemas para utilizar las tarjetas inteligentes, que son una vía para el desarrollo económico, por su bajo precio. Estos *applets* se ejecutan he interactúan en todo momento con el entorno de ejecución que contiene la máquina virtual de *JavaCard*, junto a las clases y servicios definidos en las API de estas. (14)

1.7 Pruebas unitarias

Las pruebas unitarias son pruebas individuales a un método o a una clase en específica. La base de este método es el hacer pruebas en pequeños fragmentos del sistema, desde el inicio de este hasta el final, durante todo el proceso, a las funcionalidades que tenemos definidas. Es decir:

- Se escribe el código con la funcionalidad deseada.
- Se escribe el código de las pruebas inmediatamente después.
- Se ejecutan las pruebas.
- Se corrigen los errores.
- Al añadir una nueva funcionalidad se repite el ciclo.

Al realizar estas pruebas podemos ir eliminando los errores que posee el sistema.

Para el diseño de las pruebas existe una secuencia a seguir:

- Escribir una prueba.
- Compilarla.
- Ejecutarla y hacer que falle.
- Ejecutarla bien.
- Refactorizar el código.

Se puede decir que:

- Las pruebas unitarias son una herramienta muy útil en el desarrollo y diseño del software ya que ayudan a garantizar que el programa hace justo lo que se especifica en los códigos de pruebas que lo definen.
- Herramientas de pruebas unitarias están implementadas en casi cualquier lenguaje de programación.
- Los códigos de pruebas son útiles en cualquier momento del desarrollo del software aunque se cambie la implementación siempre que se mantenga la interfaz.
- Si los códigos de pruebas son correctos y completos se podrá modificar sin temor el código que seguirá funcionando tal y como debería. (14)

1.8 Pruebas de aceptación

Estas son pruebas funcionales, sobre el sistema completo. No se realizan durante el desarrollo; sino que se realizan sobre el producto terminado. El objetivo de las pruebas de aceptación es validar que un sistema cumple con el funcionamiento esperado y permitir al usuario de dicho sistema que determine su aceptación, desde el punto de vista de su funcionalidad y rendimiento. Las pruebas de aceptación son definidas por el usuario del sistema y preparadas por el equipo de desarrollo, aunque la aprobación final corresponde al usuario.

Las pruebas de aceptación son creadas a partir de las historias de usuario. Una historia de usuario puede tener más de una prueba de aceptación, tantas como sean necesarias para garantizar su correcto funcionamiento y no se considera completa hasta que no supera sus pruebas de aceptación. Es responsabilidad del cliente verificar la corrección de las pruebas y tomar decisiones acerca de las mismas. (14)

1.9 Necesidad de la implementación de una aplicación *applet* y *middleware*

La Introducción de un sistema que permita la gestión de la información de historiales clínicos utilizando las tarjetas inteligentes en Cuba, le facilitará al personal de la salud un mayor control de los pacientes y sus padecimientos, además de que se conservará esta información en caso de cualquier accidente. Por esta razón se hace necesario el desarrollo de una aplicación (*Applet*) que permita guardar dicha información que será insertada dentro de la tarjeta electrónica, además de un *Middleware* que permita la comunicación entre un ordenador y los distintos lectores de las tarjetas inteligentes gestionando así dicha información contenida en el *applet*.

1.10 Conclusiones del capítulo

En el presente capítulo se concluyó con la selección de las herramientas, metodología, y lenguajes de programación para el desarrollo de la aplicación:

- Se va a realizar la implementación de una aplicación (*applet*) la cual soportará escribir y leer datos, dentro de la tarjeta. El *applet* se va a desarrollar utilizando tecnología *JavaCard*, la cual permite implementar aplicaciones que se ejecutan dentro de la tarjeta inteligente, de modo que esta tenga funcionalidades prácticas.
- Se implementará un *Middleware* utilizando el Visual Estudio, que permitirá la comunicación entre un ordenador y los distintos lectores de las tarjetas inteligentes y así permitir la gestión de la información contenida en el *applet* que contendrá la tarjeta inteligente.

A partir del estudio de los *applets* y *middleware* homólogos, tanto en el ámbito internacional como nacional, se escogieron los estándares para conformar la arquitectura, y el diseño para desarrollar una aplicación similar a ellas.

Capítulo 2: Caracterización y diseño del sistema a implementar

2.1 Introducción

En este capítulo se hace una descripción de la problemática existente, abordando además las características y elementos que se relacionan con el tema de *applet* y *middleware*, para su posterior desarrollo. Debido a que no están bien definidos los procesos de negocio, se reflejan los principales conceptos tratados y las relaciones entre ellos mediante un modelo de dominio, identificando para esto las entidades principales que se tendrán y las relaciones entre ellas. También se plantean las características que debe cumplir y tener el *applet* y *middleware*, así como la reseña de cada uno de ellos.

2.2 Modelo de dominio

Un modelo de dominio es un artefacto que se realiza durante el análisis de la solución, construido con las reglas de UML, 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.(25)

A continuación, se muestra el modelo de dominio que conceptualiza los elementos principales de la realidad y sus relaciones entre sí.

2.3 Diagrama de clases del modelo de dominio

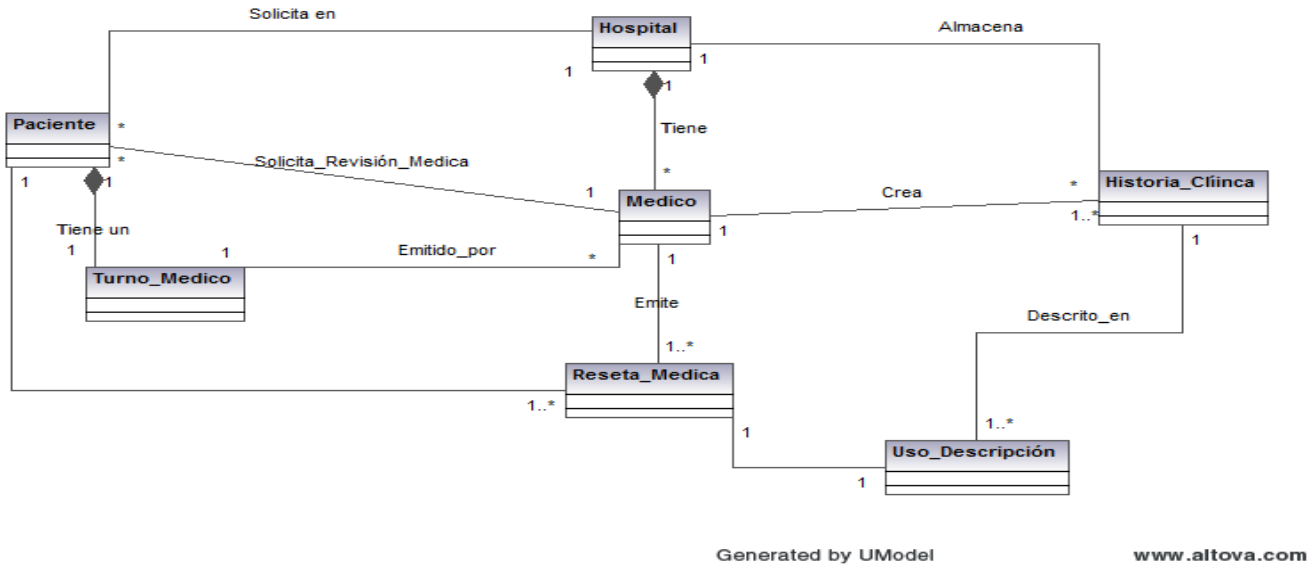


Ilustración 1: Modelo de dominio.

2.4 Glosario de conceptos del modelo de dominio

Historia clínica: es un documento médico legal, que surge de la interacción entre el médico y el paciente. En ella se recoge la información necesaria para la correcta atención de los pacientes.

Médico: es un profesional que practica la medicina y que intenta mantener y recuperar la salud humana mediante el estudio, el diagnóstico y el tratamiento de la enfermedad o lesión del paciente.

Uso y descripción: es el documento donde el médico explica como tomar los medicamentos recetados.

Receta médica: el documento por medio del cual los médicos legalmente capacitados prescriben la medicación al paciente para su dispensación por parte del farmacéutico.

Hospital: es un lugar físico en donde se atiende a los enfermos, para proporcionar el diagnóstico y tratamiento que necesitan.

Paciente: persona a la cual se le realizara la historia clínica y que tendrá la interacción con el médico.

Turno médico: es el documento en el cual el médico remite al paciente para que asista a otra consulta, ya sea realizada por el mismo o con otro médico.

2.5 Historias de usuario.

Las historias de usuario son una forma rápida de administrar los requerimientos de los usuarios, son utilizadas en las metodologías ágiles, ya que estas responden rápidamente a los requisitos cambiantes.(26)

Historia de Usuario	
Número: HU_1	**Usuario:** Desarrollador
Nombre historia: Inicializar comunicación.	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Bajo
Puntos estimados: 1	**Iteración asignada:** 1
Programador responsable: Raidel Abreu Patterson y Diana Rosa Zapata Vizcaino.	
Descripción: Se muestran los lectores que están disponibles, se selecciona uno con el cual se va a establecer la comunicación con la tarjeta inteligente.	

Observaciones: En caso de que no exista ningún lector conectado al ordenador, el sistema muestra un aviso indicando la ausencia de éstos.

Tabla 1: HU_1 Inicializar comunicación

Historia de Usuario	
Número: HU_2	**Usuario:** Desarrollador
Nombre historia: Establecer canal seguro.	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Medio
Puntos estimados: 2	**Iteración asignada:** 1
Programador responsable: Diana Rosa Zapata Vizcaino.	
Descripción: Se establece un canal de intercambio de información entre el *middleware* y el *applet* de historia clínica electrónica, utilizando protocolo de canal seguro especificado en *GlobalPlatform*.	
Observaciones:	

Tabla 2: HU_ 2 Establecer canal seguro

Historia de Usuario	
Número: HU_ 3	**Usuario:** Desarrollador
Nombre historia: Autenticación al *applet* de historia clínica electrónica.	
Prioridad en negocio: Alta	**Riesgo en desarrollo**: Medio
Puntos estimados: 2	**Iteración asignada:** 1
Programador responsable: Raidel Abreu Patterson	

Descripción: El *middleware* autentica al *applet* de historia clínica electrónica mediante el envió de datos para ser firmados y luego descifrar dichos datos y verificarlos, estableciendo así un canal seguro de intercambio de información entre *middleware* y el *applet*.
Observaciones: El *applet* de historia clínica electrónica debe contener la llave privada con la cual firma los datos recibidos.

Tabla 3: HU_ 3 Autenticación al applet de HC

Historia de Usuario	
Número: HU_ 4	**Usuario**: Desarrollador
Nombre historia: Gestionar Almacenamiento Información.	
Prioridad en negocio: Alta	**Riesgo en desarrollo**: Medio
Puntos estimados: 2	**Iteración asignada:** 2
Programador responsable: Diana Rosa Zapata Vizcaino.	
Descripción: Permitirá crear la estructura de ficheros en el *applet* de historia clínica electrónica, donde se va almacenar la información sanitaria de los pacientes y gestionar la seguridad para acceder a los datos almacenados.	
Observaciones:	

Tabla 4: HU_ 4 Gestionar almacenamiento información

Historia de Usuario	
Número: HU_ 5	**Usuario:** Desarrollador
Nombre historia: Enviar y recibir comandos APDU de la tarjeta.	

Prioridad en negocio: Alta	**Riesgo en desarrollo:** Medio
Puntos estimados: 2	**Iteración asignada:** 2
Programador responsable: Raidel Abreu Patterson	
Descripción: Cuando hay una conexión abierta entre un lector y una tarjeta inteligente, el usuario hace una solicitud que es procesada y se envía los correspondientes comandos APDU que son transmitidos a la tarjeta. A este comando la tarjeta siempre devolverá una respuesta que puede ser satisfactoria, la información solicitada o un mensaje de error. La tarjeta procesa el comando APDU enviado y manda la respuesta.	
Observaciones: Si la tarjeta no está en estado conectada en el lector, el sistema notifica un error.	

Tabla 5: HU_5 Enviar y recibir comandos 1

Historia de Usuario	
Número:HU_6	**Usuario:** Desarrollador
Nombre historia: Finalizar comunicación.	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Bajo
Puntos estimados: 1	**Iteración asignada:** 2
Programador responsable: Raidel Abreu Patterson	
Descripción: Consiste en realizar la desconexión entre la tarjeta inteligente, donde se encuentra el *applet* de historia clínica electrónica y el lector.	
Observaciones: Si la tarjeta no está en estado conectada en el lector, el sistema notifica un error.	

Tabla 6: HU_ 6 Finalizar comunicación. 1

2.6 Requerimientos no funcionales

- **Usabilidad**

El *middleware* deberá ser entendible y fácil de usar para una posterior conexión con las diferentes aplicaciones implementadas.

- **Integridad**

Según se define en el estándar *GlobalPlatform*, los datos que se transmitirán y los que se recibirán, podrán ser verificados utilizando código de autenticación de datos.

- **Rendimiento**

El *applet* deberá ser capaz de atender pedidos y enviar respuesta eficientemente, y en el menor tiempo posible, asegurando así su eficiencia.

- **Portabilidad**

El *middleware* debe ser compatible con cualquier lector de tarjetas que cumpla con el estándar PC/SC.

- **Seguridad**

✓ **Confiabilidad**

La información almacenada en las tarjetas estará protegida por su propio sistema operativo, por la tecnología *JavaCard* y por la seguridad que defina el proveedor de las mismas.

- **Requerimientos mínimos de hardware**

Lector de tarjetas incorporado a la PC que cumpla con el estándar PC/SC.

- **Requerimientos de software**

Para el funcionamiento de la plataforma se requiere que estén instalados en la PC cliente los drivers del lector de tarjetas.

2.7 Propuesta de solución

El desarrollo de una aplicación utilizando tarjetas inteligentes, normalmente, consta de tres grandes componentes. Una aplicación del lado de la tarjeta, una aplicación del lado del terminal y un protocolo de comunicación. La aplicación del lado del terminal es la que interactúa con el usuario y a través del protocolo de comunicación interactúa con la aplicación en la tarjeta. Se debe tener esto en cuenta al momento desarrollar la tarjeta de salud.

2.7.1 Metáfora

Cada proyecto XP es guiado por una metáfora global. Estas ayudan a cualquier persona a entender el objetivo del programa, principalmente al equipo ya que aporta un contexto para entender los elementos básicos y sus relaciones proporcionando integridad conceptual. Es de vital importancia que el cliente y los desarrolladores estén de acuerdo y conozcan la metáfora a usar para así poder trabajar y discutir en los mismos términos de una forma más precisa y de dominio de todos.

El sistema está compuesto por un *applet* el cual se usará para personalizar el chip insertado en la tarjeta, así como un *middleware* que funciona como protocolo para la comunicación entre las aplicaciones. El *applet* tiene la funcionalidad de guardar, y mostrar datos del historial médico.

El sistema para gestionar la información de historias clínicas funcionará en todos los centros donde se cuenta con lectores de tarjetas inteligentes. Cuando haya una conexión abierta entre un lector y una tarjeta inteligente, el usuario hará una solicitud que es procesada y se enviarán los correspondientes comandos APDU que serán transmitidos a la tarjeta a través de un componente cliente, que accederá a la capa que se comunica con la tarjeta. A este comando la tarjeta siempre devolverá una respuesta que puede ser satisfactoria, la información solicitada o un mensaje de error.

2.7.2 Arquitectura del *applet*

Los *applets* son las aplicaciones que corren dentro en una *JavaCard*. Dichas aplicaciones interactúan, se nutren en todo momento con el JCRE utilizando los servicios que éste brinda, e implementan la interfaz definida en la clase abstracta javacard.*framework*.*Applet*. Se puede decir que un *applet* comienza su ciclo de vida al ser correctamente cargado en la memoria de la tarjeta. Una vez registrada ante el JCRE un *applet* está en condiciones de ejecutar. Este *applet* normalmente existe durante el resto de la vida de la tarjeta aunque se pueden eliminar tanto los *applet* como los paquetes que los contienen.

Se debe tener presente la información a almacenar dentro de la tarjeta, esta debería contener toda la historia clínica del paciente, de forma de que ante cualquier emergencia el médico tenga la posibilidad de acceder a la mayor cantidad de información disponible del paciente.

En esta fase de diseño se han especificado las funciones del *applet*, estas funciones darán soporte para estos datos y otros:

Categoría	Item	Tipo	Tamaño (bytes)
Datos administrativos	Código del paciente	char	10
	Tipo de documento	char	3
	Documento	char	15
	Nombre	char	50
	Fecha de nacimiento	date	8
	Edad aparente	byte	1
	Departamento de nacimiento	char	2
	País de nacimiento	char	2
	Sexo	char	1
	Dirección	char	100
	Teléfono	char	20
	Nombre del padre	char	50
	Nombre de la madre	char	50
	Situación familiar	char	2
	Color de piel	byte	1
	Nombre del contacto	char	50
	Teléfono del contacto	char	20
	Dirección del contacto	char	100
	Descripción del contacto	char	50
	Fecha actualización	date	8
Subtotal para esta categoría			543

Ilustración 2: Algunos datos que contendrá el applet 1

Para prevenir un acceso no autorizado de la tarjeta, la misma posee un algoritmo de seguridad. Este algoritmo requiere que el usuario ingrese un PIN, una cadena de máximo 4 dígitos. Este algoritmo de seguridad de la tarjeta tiene como efecto el cierre o bloqueo después de tres intentos fallidos de ingresar el PIN. El PIN es inicializado de acuerdo a los parámetros instalados cuando el *applet* es instalado y creado.

Para poder desarrollar el sistema, en este caso, el *JavaCard applet*, primeramente se ha tenido que seguir por una fase de diseño del mismo, definiendo en esta, la arquitectura que tendrá para un posterior desarrollo.

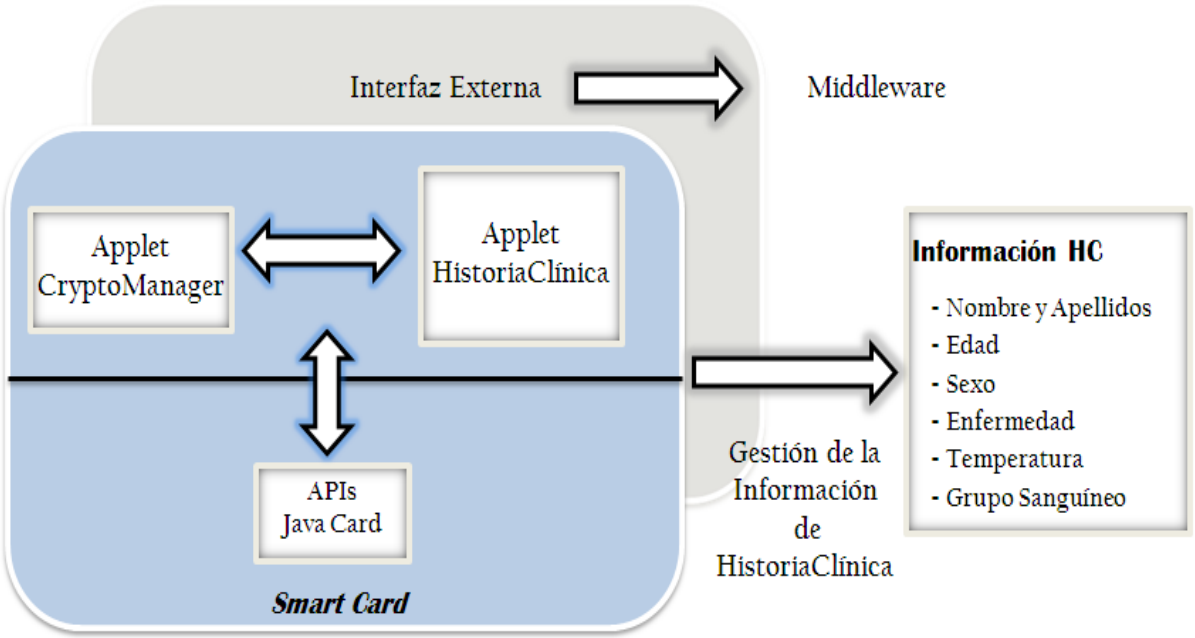


Ilustración 3: Arquitectura de applet

Cuando una *JavaCard* es insertada dentro de un CAD (*Card Acceptance Device* o la terminal lectora), el CAD selecciona el o un *applet* de la tarjeta y le envía una serie de comandos para ejecutarlos. Cada *applet* es identificado y seleccionado por su AID. Comandos como la selección, son formateados y transmitidos en la forma de un APDU (*Application Protocol Data Units*). Los *applets* responden a cada operación APDU con un SW (*status Word*), que indica el resultado de la operación. Un *applet* puede contestar de modo opcional un APDU con otro tipo de dato. (27)

2.7.3 Arquitectura del *middleware*

El *middleware* implementa los procesos definidos en los requerimientos de la solución, utilizando además las operaciones necesarias para verificar la gestión de la seguridad. Este actúa como un componente, que opera como una capa (wrapper), que aísla al humano de la comunicación directa con las operaciones que realiza el *applet*.

La comunicación del *middleware* con el *applet* se realiza a través del *middleware SmartCard Framework*, el cual implementa las especificaciones de los estándares ISO 7816 – 4, y *GlobalPlatform*

para los procesos de encriptación de los datos a enviar, también implementa a nivel de API la comunicación con los lectores de tarjetas electrónicas inteligentes.

Esta comunicación está determinada bajo capas que son necesarias para establecer un canal correcto de transmisión de los datos de información. **(Ver Ilustración 14).**

Para poder desarrollar nuestro sistema, en este caso, el *middleware*, primeramente se ha tenido que seguir por una fase de diseño del mismo, definiendo en esta, la arquitectura que tendrá para un posterior desarrollo. En el siguiente diagrama se muestra la arquitectura que tendrá el *middleware* donde se relacionan paquetes y componentes.

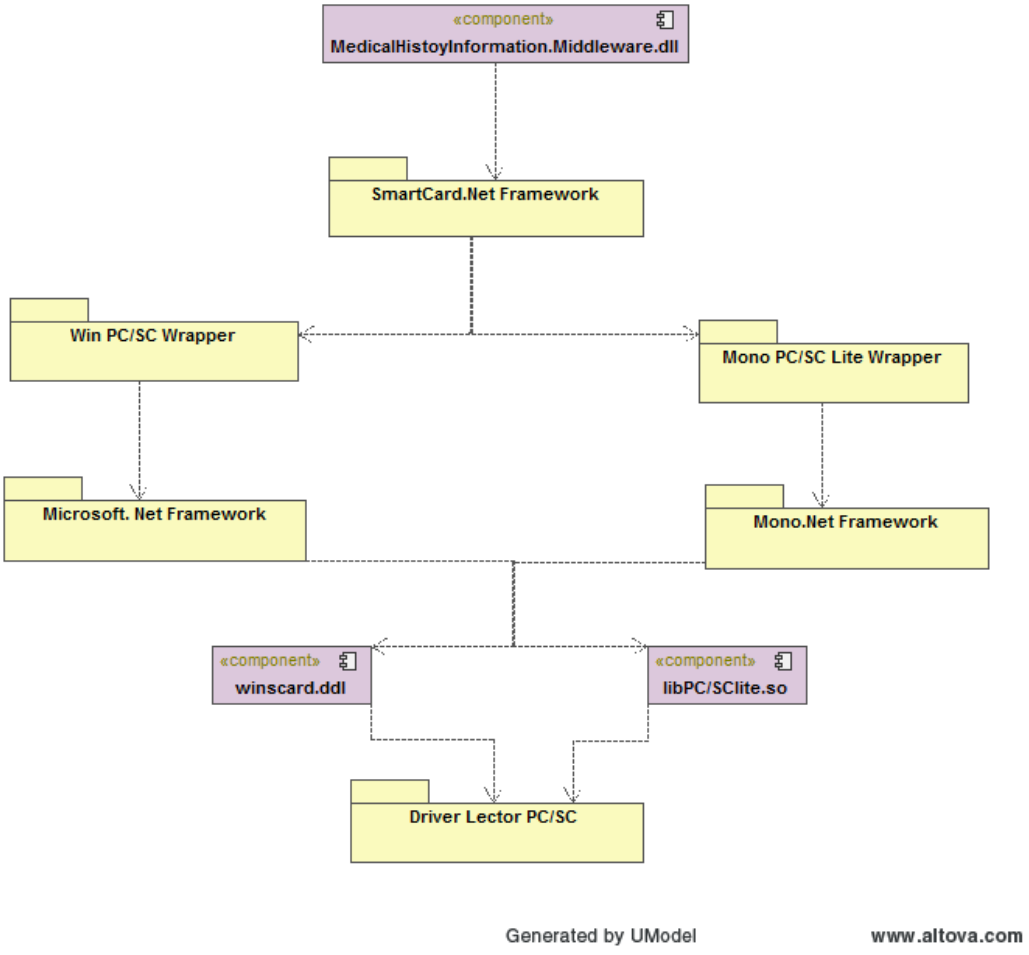


Ilustración 4: Arquitectura del middleware

El mismo está compuesto por la DLL <<MedicalHistoyInformation.dll>> la cual contiene todas las funcionalidades que presenta el *middleware*, este a su vez utiliza el *SmartCard* .Net *Framework* que es un paquete contenedor de los wrapper WinPCSC y Mono PCSCLite.

El WinPCSC Wrapper es la solución para la ejecución de aplicaciones de *Smart Card* que cumplan las especificaciones PC/SC para el sistema Operativo Windows, la cual utiliza el paquete *Microsoft.Net Framework* que contiene todas las librerías bases que provee la tecnología .Net, así como de la DLL <<winscard.dll>> que permite la comunicación con cualquier equipo que cumpla con las especificaciones PC/SC.

El MonoPCSCLite es la solución para la ejecución de aplicaciones de *Smart Card* que cumplan las especificaciones PC/SC para software libre, este wrapper incluye todas las características para ser ejecutado bajo ambiente Linux o Windows, está desarrollado sobre el Mono.Net *Framework* y utiliza la biblioteca <<winscard.dll>>para implementar PC/SC para Windows y de la librería <<libpcsclite.so>> para implementar PC/SC para entorno Linux.(14)

2.7.4 Estilo arquitectónico

Dentro de la amplia clasificación de estilos arquitectónicos se utilizará arquitectura en capas debido a que organiza el modelo de diseño.

Las arquitecturas en capas son muy utilizadas para el desarrollo de aplicaciones en la actualidad por las grandes ventajas que proporcionan. El principal objetivo que persigue es reducir dependencias entre artefactos, situándolos en capas lógicas, donde cada capa depende del servicio prestado por la inferior y presta un servicio a la superior, proporcionando a los desarrolladores ventajas en cuanto al mantenimiento y reutilización de estos componentes.

Cada capa se ocupa de un nivel del problema y debe tener poco acoplamiento con las demás de manera que el cambio en una capa, no altera en gran medida los cambios en la otra capa. Si se desea en un futuro incorporar una capa de presentación distinta, no debe alterar a la capa operacional ya desarrollada, es decir, cambiar su implementación, debe introducir los mínimos efectos en el resto de la aplicación. (28)

La arquitectura a utilizar constará de las siguientes capas:

2.7.4.1 Capa de presentación

Esta es la capa que interactúa con el usuario, es la encargada de modelar como se recogerán y mostrarán los datos del proceso asistencial, así como de la apariencia que tendrá la interfaz visual. Esta

se comunica con la capa controladora o *middleware* a la cual envía todas las solicitudes, y las muestra cuando estas hayan sido procesadas. En el sistema que será desarrollado posteriormente esta capa se llamará *Middleware*HCE.

2.7.4.2 Capa controladora

La capa controladora representa las clases del sistema, es la encargada de darle solución a las historias de usuario, en esta es donde se implementan las restricciones que deberá cumplir la aplicación. Se comunica con la capa de presentación para recibir las solicitudes del usuario, las envía a la capa de acceso a datos y envía las respuestas a la capa de presentación después del procesamiento de la solicitud. En el sistema que será desarrollado posteriormente esta capa se llamará: *Middleware*ParaHistorialesClínicosEnTarjetasInteligentes.

2.7.4.3 Capa de acceso a datos

Esta capa contendrá una aplicación donde estará almacenada la información referente al paciente, y esta será la encargada del manejo de dicha información. La misma se comunicará con la capa intermediaria, de donde recibe la solicitud de guardar información o de mostrarla. En el sistema que será desarrollado posteriormente esta capa se llamará *applet SaludCard*.

A continuación se muestra un diagrama de la arquitectura base en tres capas del sistema:

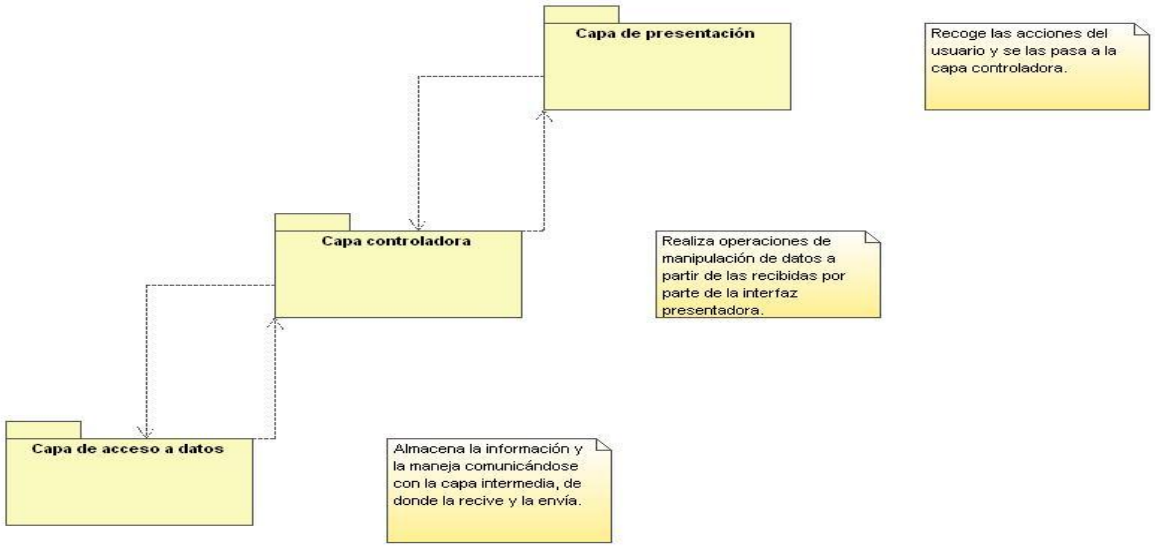


Ilustración 5: Diagrama de paquetes (Arquitectura base en tres capas)

Se escoge el estilo arquitectónico 3 capas sobre Modelo Vista Controlador (MVC) debido a que el MVC la lógica de negocio y el acceso a datos van juntos en los componentes del modelo y separados de la lógica de proceso que se implementa en los controladores, y en el caso del sistema a implementar estas van separadas.

2.7.5 Patrones de diseño

Se utilizaron para el diseño los patrones GRASP los cuales describen los principios fundamentales de diseño de objetos para la asignación de responsabilidades y aplica el razonamiento para el diseño de una forma sistemática, racional y explicable.

Creador: Permite decidir cuáles serán las clases creadoras de otras clases. La creación de instancias es una de las actividades más comunes en un sistema orientado a objetos. En consecuencia, es útil contar con un principio general para la asignación de las responsabilidades de creación. Este patrón se utiliza en las clases entidades, cuando estas se instancian y crean instancias de clases contenidas en las mismas, ejemplo: Configurar.cs.

Alta cohesión: Cada elemento del diseño debe realizar una labor única dentro del sistema, no desempeñada por el resto de los elementos y auto-identificable. Una clase cohesionada facilita el cambio. Al realizar un cambio en una clase muy cohesionada, todos los métodos que pueden verse afectados, toda la información que necesitamos controlar, estará a la vista, en el mismo fichero. Se pone de manifiesto: Paciente.cs.

Bajo acoplamiento: Uno de los principales síntomas de un mal diseño y alto acoplamiento es una herencia muy profunda. Debe haber pocas dependencias entre las clases. Uno de los principios para proteger al software frente al cambio es mantener bajo el acoplamiento entre clases. Ejemplo, (ComandoAlmacenarNombreCompleto.cscs -> APDUCommand.cs).

2.8 Plan de entrega

Después de tener ya definidas las historias de usuarios es necesario crear un plan de publicación. En este plan se hace los desarrolladores y clientes establecen los tiempos de implementación ideales de las historias de usuario y la prioridad con la que serán implementadas. Al final se propone una fecha para entregar el producto, con el tiempo esta fecha puede variar, pero lo más seguro es que esta se acerque mucho a la planificación. En la siguiente tabla se muestra la planificación de la primera versión del sistema de gestión de historiales clínicos utilizando las tarjetas inteligentes.

Entregable	Iteración	Fecha de entrega
Applet y *Middleware* para gestionar la información de HC en las tarjetas inteligentes.	Iteración 1	21/02/2011 - 3/04/2011
Applet y *Middleware* para gestionar la información de HC en las tarjetas inteligentes.	Iteración 2	4/04/2011 - 15/05/2011

Tabla 7: Plan de entrega

2.9 Estimación de esfuerzo

Los desarrolladores estiman cuanto tiempo es necesario para implementar cada una de las historias de usuarios. Si ese tiempo supera en alguna las tres semanas se debe desglosar en otras más pequeñas. Si es inferior a una semana, la historia de usuario ha descendido a un nivel de detalle excesivo y habrá que combinarla con otra. Esto se hace concluida la fase de exploración. Como se ha dicho anteriormente, este valor es estimado y se irá acercando a la realidad con el transcurso de las iteraciones. En la presente solución se han identificado seis historias de usuarios y quedan definidas las dos iteración con un total de 10 semanas de duración; de ser necesaria otra iteración se valoraría.

Historia de Usuario	Estimación (semanas)
Inicializar comunicación.	1
Establecer canal seguro.	2
Autenticación al *applet* de historia clínica electrónica.	2
Gestionar almacenamiento información.	2
Enviar y recibir comandos APDU de la tarjeta.	2

Finalizar comunicación.	1
Total	10

Tabla 8: Plan de Iteraciones.

2.10 Costos y beneficios

La realización de un análisis de los costos y beneficios del proyecto es imprescindible a la hora de asumir una tarea, pues se realiza con el objetivo de demostrar la viabilidad o factibilidad del desarrollo del sistema propuesto, planteándose los beneficios tangibles e intangibles que reportaría la aplicación.

A partir de la información obtenida como resultado de la fase de inicio se tienen los conocimientos necesarios para tener una idea de lo que debe hacer el producto. A continuación se hará un análisis de los principales recursos que se necesitarán para que los desarrolladores le den cumplimiento a la realización del sistema así como sus precios.

Aunque para la realización de sistemas se contará con 2 PC Dell Optiplex 760, su costo en el mercado libre de Venezuela es de aproximadamente 468 CUC, las tarjetas que se utilizarán tienen un valor de 6 CUC, y los lectores de tarjeta tienen un precio de 25 CUC aproximadamente y los dos desarrolladores que trabajarán en la solución adquirirán un sueldo de 625 MN, se llega a la conclusión de que el desarrollo del sistema no supone un gasto considerable debido al uso, en su mayoría, de herramientas gratis. Los gastos por concepto de tecnologías son mínimos, pues ya la universidad cuenta con una infraestructura productiva, que favorece la realización de un proyecto con estas características.

La utilización de un sistema que gestione la información de historias clínicas en las tarjetas inteligentes utilizando estos recursos, traerá consigo una buena atención sanitaria, ya que la comunicación entre profesionales de la salud, incluso entre estos y los pacientes será mejorada y los datos de los pacientes tendrán mayor seguridad, por lo antes planteado se concluye que es factible desarrollar un sistema que reporte estos beneficios.

2.12 Conclusiones del capítulo

Después de analizar los principales conceptos relacionados con el sistema que se desea implementar, se llegó a la conclusión de que es necesario realizar un modelo de dominio, quedando

reflejado en este la relación entre los conceptos, brindando así, una visión más clara para implementar los componentes de la solución.

Además al describir las historias de usuario se amplió la visión para desarrollarlas posteriormente, se estructuró la arquitectura del *applet* y la del *middleware* separadas, organizando una arquitectura de tres capas entre el *applet*, la interfaz el componente *middleware* que los unirá.

Se describieron los beneficios que representará el sistema al ser implementado y se concluye que el producto no incurrirá en grandes gastos y será terminado en el tiempo estimado.

Capítulo 3: Implementación y pruebas del sistema

3.1 Introducción

A partir del marco de trabajo definido en la sección anterior, se define a continuación la evolución del procedimiento, durante las fases de Iteraciones a primera liberación y producción, además de desarrollar las pruebas del sistema para verificar la implementación de las historias de usuarios, ya que las pruebas funcionales permiten verificar que el sistema en desarrollo satisface a éstas. Se explica también el diseño de la solución así como los principales componentes y sus relaciones.

3.2 Iteraciones a primera liberación

Al culminar esta fase se le da cumplimento al plan de iteraciones. En cada iteración se desarrollan las sub-fases de diseño, realización de pruebas unitarias, codificación de la solución y refactorización. Al terminar esta, el cliente estará listo para realizar las pruebas de aceptación.(16)

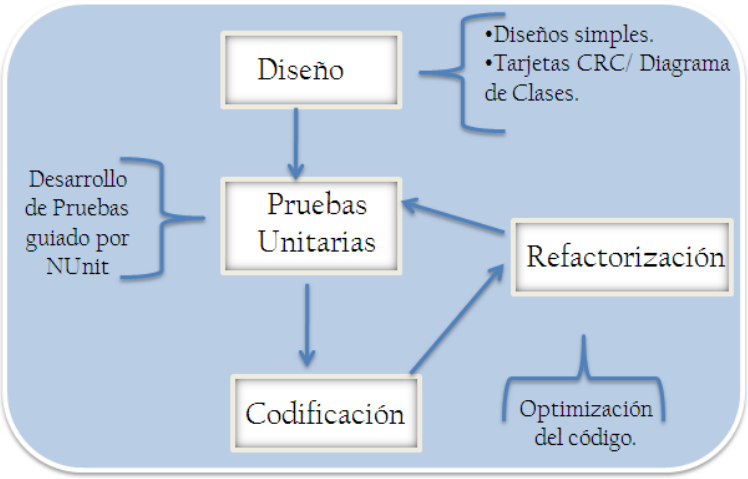


Ilustración 6: Sub-fases de la fase Iteración 1

3.2.1 Tarjetas CRC

La técnica de las tarjetas CRC, se puede usar para guiar el sistema a través de análisis encaminados por la responsabilidad. Las clases se examinan, se filtran y se refinan en base a sus responsabilidades con respecto al sistema, y las clases con las que necesitan colaborar para completar sus responsabilidades. (32)

***Applet*_Historia_Clínica**	
Responsabilidades	**Colaboradores**
- Establecer el canal seguro de comunicación con el *middleware*.	- Javacard.*framework*
- Cerrar el canal seguro de comunicación.	
- Comprobar que el PIN del usuario es correcto.	
- Cambiar el PIN por uno nuevo.	
- Obtener toda la información referente al paciente.	
- Almacenar toda la información referente al paciente.	

Tabla 9: Tarjeta CRC (Applet Historia Clínica) 1

***Middleware*_Historia_Clínica**	
Responsabilidades	**Colaboradores**
- Listar lectores conectados. - Establecer comunicación con la tarjeta. - Seleccionar *applet* historia clínica. - Configurar el canal seguro por *GlobalPlatform* para el *applet* historia clínica. - Permitir introducir el PIN. - Permitir verificar el PIN en el *applet* historia clínica. - Permitir verificar el nivel de Acceso del	- Usuario - Útil - Común - APDU - *SmartCard*.Client - *SmartCard*.Core - *SmartCard*.Core.Common - *SmartCard*.Core.Utils - *SmartCard*.*GlobalPlatform*

profesional que atienda al paciente. - Permitir almacenar toda la información referente al paciente. - Permitir obtener toda la información referente al paciente.	- *SmartCard*.*GlobalPlatform*.Client - *SmartCard*.Devices.CardReaders - *SmartCard*.ISO7816.APDU - *SmartCard*.ISO7816.APDU.Commands - *SmartCard*.*GlobalPlatform*.Security - *SmartCard*.*GlobalPlatform*.Commands - AlmacenarAlergias.cs - AlmacenarApararoHematologico.cs - AlmacenarAparatoCardiovascular.cs - AlmacenarAparatoDigestivo.cs - Entre otras

Tabla 10: Tarjeta CRC (Middleware SaludCard) 1

Útil	
Responsabilidades	**Colaboradores**
Convertir el arreglo de byte a byte Convertir de byte a decimal Convertir de decimal a byte Convertir de byte a arreglo de byte Verificar comando APDU de respuesta Codificar cadena Decodificar cadena	*SmartCard*.ISO7816.APDU

Tabla 11: Tarjeta CRC (Útil) 1

Paciente	
Responsabilidades	**Colaboradores**
Permitir obtener o cambiar información del usuario.	No tiene

Tabla 12: Tarjeta CRC (Paciente) 1

3.3 Implementación del sistema

3.3.1 Iteración 1.

Después de un análisis a las historias de usuarios seleccionadas para el desarrollo del sistema, se escogieron cuales de estas eran más significativas para desarrollarlas primeramente en la iteración 1. En XP el trabajo de cada iteración se divide en tareas de ingeniería, las tareas de ingeniería se derivan de las historias de usuario; se realizan para especificar las acciones que realizan los desarrolladores para darle cumplimiento a las mismas. Todo el trabajo durante las dos iteraciones en el que se planificó el mismo es expresado en tareas de programación. A continuación se muestran las tareas de ingeniería derivadas de cada historia de usuario a desarrollar en la primera iteración:

	Historia de Usuario	Tarea
1	Inicializar comunicación	Listar lectores conectados.
	Establecer canal seguro.	Intercambio de Información, por canal seguro, entre el *middleware* y el *applet* de historia clínica electrónica.
	Autenticación al *applet* de historia clínica electrónica.	Autenticar al *applet* historia clínica electrónica, mediante el envío de datos.

Tabla 13: Tareas de ingeniería (Iteración 1) 1

Para ver los detalles de cada tarea de ingeniería perteneciente a la primera iteración **(Ver Anexo 3).**

3.3.2 Iteración 2.

En la segunda iteración se implementarán las historias de usuarios menos importantes para el cumplimiento de los objetivos de la aplicación.

Iteración	Historia de Usuario	Tarea
2	Gestionar almacenamiento Información.	Crear la estructura de ficheros en el *applet* de historia clínica electrónica.
	Enviar y recibir comandos APDU de la tarjeta.	Enviar comandos APDU a la tarjeta. Recibir respuesta APDU de la tarjeta.
	Finalizar comunicación	Desconectar la tarjeta del lector.

Tabla 14: Tareas de ingeniería (Iteración 2) 1

Para ver los detalles de cada tarea de ingeniería perteneciente a la segunda iteración **(Ver Anexo 3).**

3.3.3 Descripción de los principales flujos de procesos

3.3.3.1 Proceso autenticación de usuario

El objetivo del proceso de autenticación de usuario es validar las condiciones de seguridad que existen a la hora de escribir o leer información en la Historia Clínica Electrónica (HCE). Este proceso inicia cuando la terminal de servicio a través del *middleware*, solicita el Número de Identificación Personal (PIN) del portador de la tarjeta, este introduce código solicitado y el *middleware* envía los datos de autenticación al *applet* de la tarjeta para que sean verificados; luego el *applet* envía la respuesta obtenida al *middleware* y este a su vez se la muestra a la terminal de servicio.

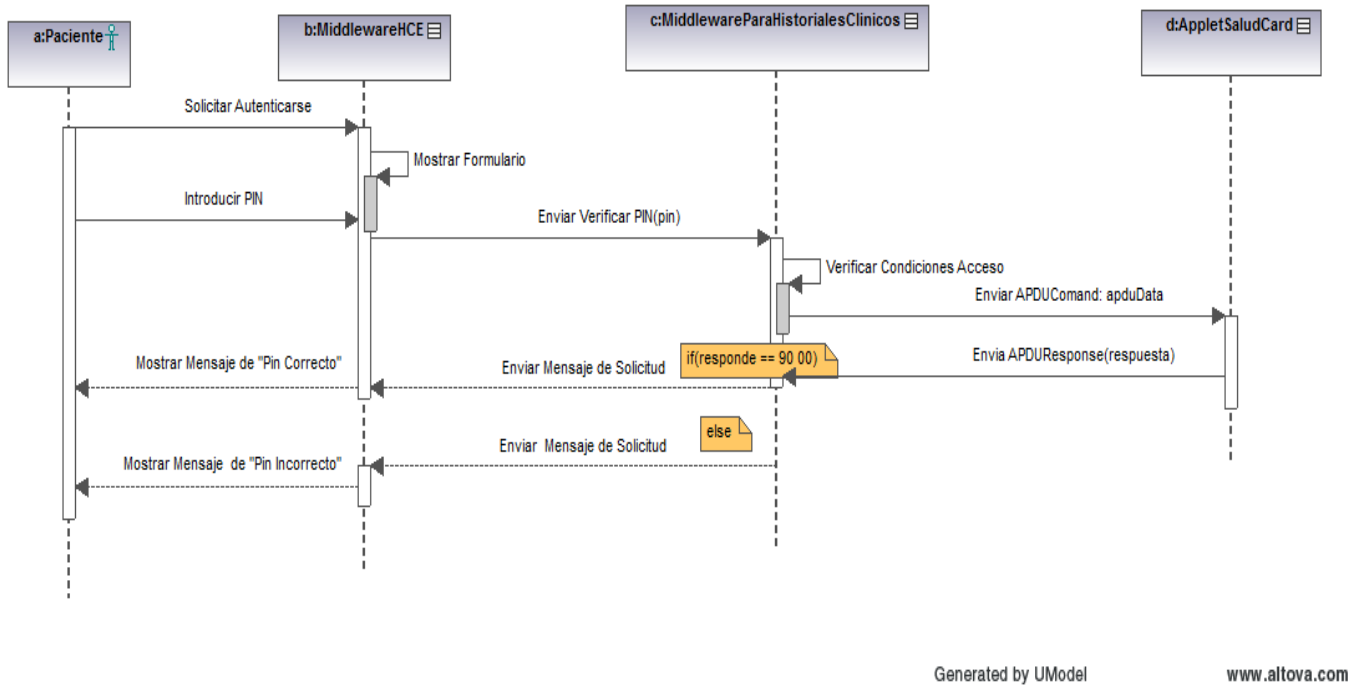


Ilustración 7: Diagrama de secuencia (proceso autenticación de usuario) 1

3.3.3.2 Proceso llenar datos de los pacientes

Para darle inicio al proceso de Escribir Información, primeramente el profesional que atiende al paciente debe insertar su nivel de acceso con el cual el sistema le permite acceder a la opción de llenar los datos de los pacientes en la HCE, enviando la información que se desea persistir en ella. El *middleware* envía dicha solicitud de escritura y la información que se desea guardar en la HCE, al *applet*.

El *applet* realiza una verificación de las condiciones de acceso antes establecidas para su escritura, el *applet* procede a almacenar la información deseada y envía la respuesta obtenida al *middleware*, luego el *middleware* se la muestra a la terminal de servicio.

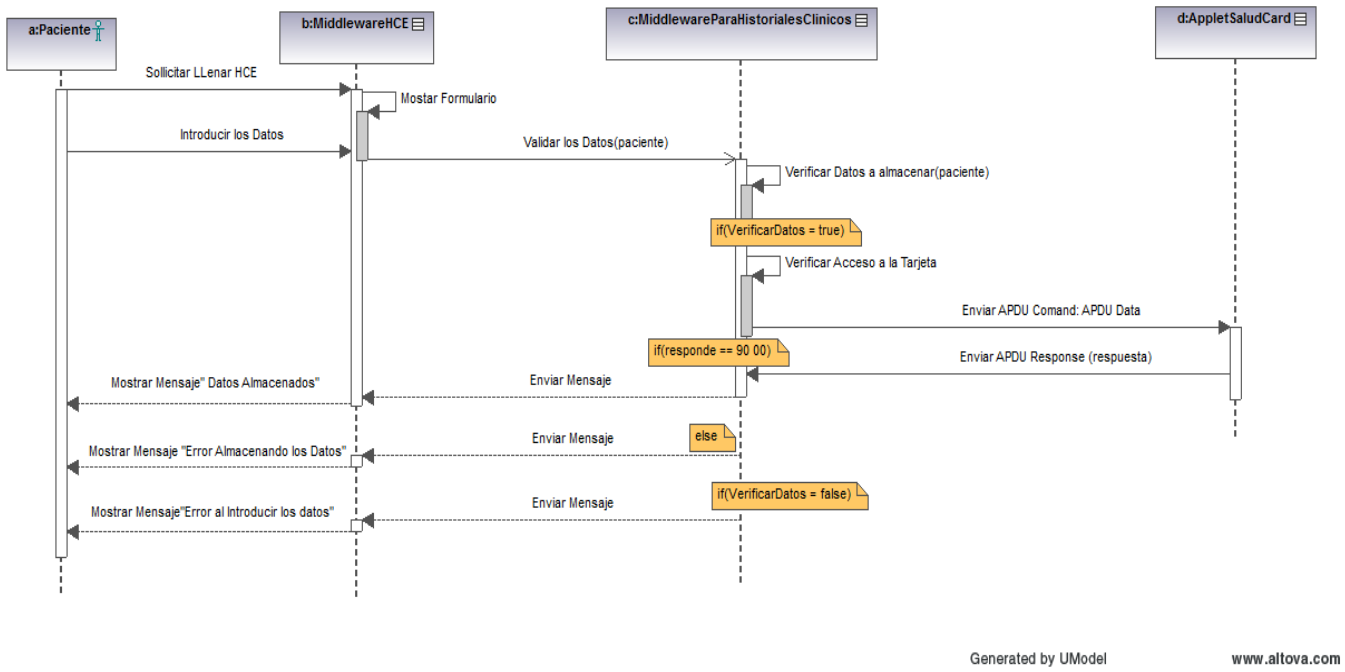


Ilustración 8: Diagrama de secuencia (proceso llenar datos de los pacientes) 1

3.3.3.3 Proceso obtención de la Información

Para iniciar el proceso de obtención de la Información de los pacientes, primeramente el profesional que atiende al paciente debe insertar su nivel de acceso con el cual el sistema le permite acceder a la opción de acceder a la información, el *middleware* envía dicha solicitud de lectura al *applet* de la HCE. El *applet* realiza una verificación de las condiciones de acceso y si estas son las necesarias para realizar la lectura, procede a retornar un cuerpo de datos (DATA), donde se encuentra la información deseada. El *Middleware* recibe la información y la transforma para mostrársela a la terminal de servicio.

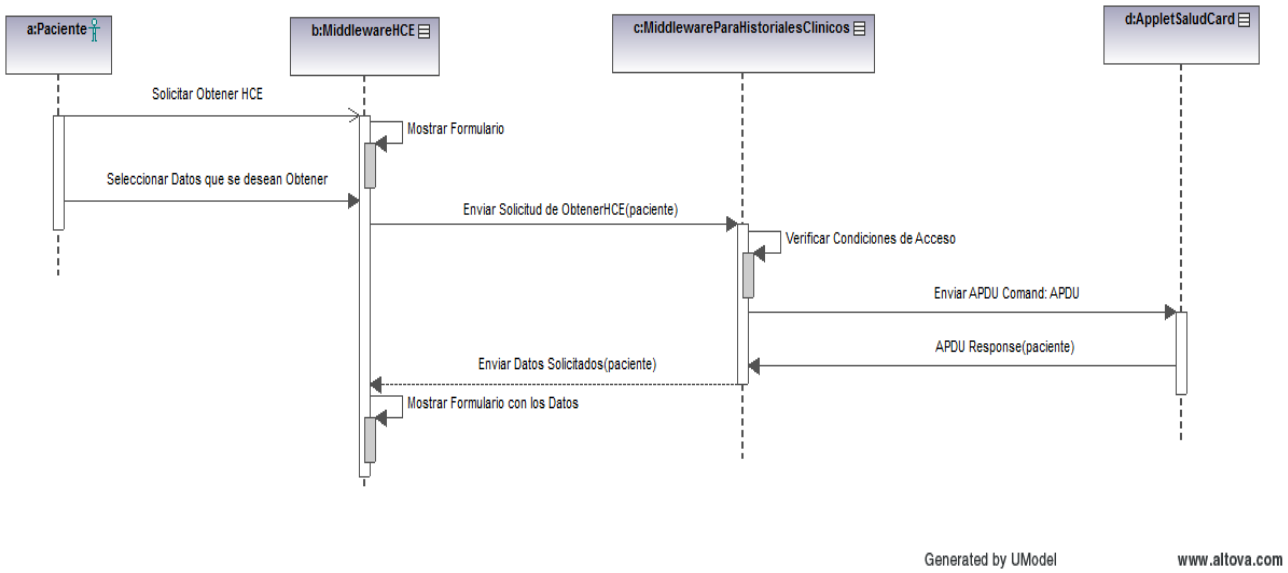


Ilustración 9: Diagrama de secuencia (proceso obtención de la información) 1

3.4 Pruebas

Las pruebas del sistema tienen como objetivo verificar la funcionalidad del sistema a través de sus interfaces externas comprobando que dicha funcionalidad sea la esperada en función de los requisitos del sistema. XP divide las pruebas del sistema en dos grupos: pruebas unitarias, encargadas de verificar el código y diseñada por los programadores, y pruebas de aceptación o pruebas funcionales destinadas a evaluar si al final de una iteración se consiguió la funcionalidad requerida.

3.4.1 Pruebas unitarias

A continuación se muestra la prueba unitaria más significativa ya que a través del proceso verificar PIN se acceden a los demás procesos del sistema.

Prueba de Unidad		
Nombre Prueba: Verificar PIN		
Estado: Satisfactoria	**Tipo:** Caja Blanca	**Última ejecución:**
Ejecutado por: Diana Rosa Zapata	**Verificado por:** Raidel Abreu Patterson	
Descripción: Para el desarrollo de esta prueba previamente se debe de haber introducido el PIN a verificar, si el		

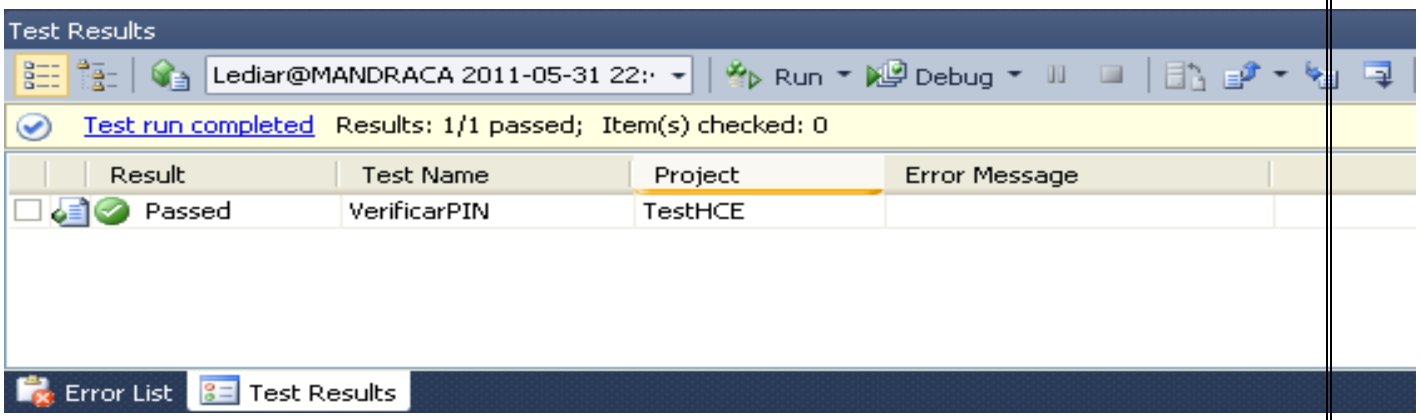
mismo es correcto, se autentica satisfactoriamente, de lo contrario se lanza un mensaje de PIN incorrecto.
Entrada: Número de Identificación personal(PIN)
Resultado: 

Tabla 15: Prueba unitaria (verificar PIN) 1

3.4.2 Pruebas de aceptación

Las pruebas de aceptación mostraron que el sistema realizaba los diferentes procesos correctamente, estas se hicieron a partir de las historias de usuario, a continuación se muestran los casos de prueba y los resultados, que todos fueron satisfactorios.

Caso de prueba de aceptación	
Código de caso de prueba: HU1_CP1	**Nombre de la historia de usuario:** Inicializar Comunicación.
Responsable de la prueba: Diana Rosa Zapata Vizcaino.	

Descripción de la prueba: Prueba de funcionalidad para listar los lectores conectados.
Condiciones de ejecución: Debe existir algún lector conectado.
Entrada/Pasos de ejecución: • Aparece en la página una lista desplegable con los nombre de los lectores conectados.
Resultado esperado: Mostrar satisfactoriamente el listado con los nombres de los lectores conectados.
Evaluación de la prueba: Prueba satisfactoria.

Tabla 16: CP1-HU1 Prueba de funcionalidad para listar los lectores conectados 1

Caso de Prueba de aceptación	
Código de caso de prueba: HU2_CP2	**Nombre de la historia de usuario:** Establecer Canal Seguro.
Responsable de la prueba: Diana Roza Zapata Vizcaino	
Descripción de la prueba: Prueba de funcionalidad para establecer la conexión entre el lector y la tarjeta.	
Condiciones de ejecución: Debe haber una tarjeta insertada en el lector.	
Entrada/Pasos de ejecución: • El usuario se registra en la aplicación. • Selecciona un lector para establecer la conexión. • El lector establece conexión con la tarjeta inteligente.	
Resultado esperado: Se muestra una notificación al usuario que se ha establecido la conexión.	

Evaluación de la prueba: Prueba satisfactoria.

Tabla 17: CP2- HU2 Prueba de funcionalidad para establecer la conexión.1

Caso de Prueba de Aceptación	
Código de caso de prueba: HU3_CP3	**Nombre de la historia de usuario:** Autenticación al *applet* de historia clínica electrónica.
Responsable de la prueba: Diana Roza Zapata Vizcaino	
Descripción de la prueba: Prueba de funcionalidad para verificar PIN y autenticarse.	
Condiciones de ejecución: Debe haber una tarjeta insertada en el lector.	
Entrada/Pasos de ejecución: • El usuario introduce el PIN. • El *middleware* inicia la comunicación enviando los datos. • La tarjeta procesa el comando APDU y devuelve una respuesta que será enviada al *middleware*. • Este muestra que el dato es correcto y permite la gestión de datos del *applet*.	
Resultado esperado: Se muestra una notificación al usuario que se ha realizado la autenticación correctamente.	
Evaluación de la prueba: Prueba satisfactoria.	

Tabla 18: CP3-HU3 Prueba de funcionalidad para verificar PIN y autenticarse. 1

Caso de Prueba de Aceptación	
Código de caso de prueba: HU4_CP4	**Nombre de la historia de usuario:** Gestionar almacenamiento Información.
Responsable de la prueba: Raidel Abreu Patterson.	

Descripción de la prueba: Proceso llenar datos de los pacientes
Condiciones de ejecución: Debe haber una tarjeta insertada en el lector. Debe haberse autentificado.
Entrada/Pasos de ejecución: • Se solicita llenar datos de los pacientes. • El *middleware* envía dicha información al *applet*. • El *applet* realiza una verificación de las condiciones de acceso antes establecidas para su escritura. • El *applet* procede a almacenar la información. • El *applet* le envía al *middleware* la respuesta obtenida. • El *middleware* muestra información al usuario.
Resultado esperado: Se llenan los datos deseados del paciente. Se muestra un mensaje avisándote que se almacenó la información.
Evaluación de la prueba: Prueba satisfactoria.

Tabla 19:CP4-HU4 Proceso llenar datos de los pacientes 1

Caso de Prueba de Aceptación	
Código de caso de prueba: HU4_CP5	**Nombre de la historia de usuario:** Gestionar almacenamiento Información.
Responsable de la prueba: Raidel Abreu Patterson.	
Descripción de la prueba: Proceso obtención de la información	
Condiciones de ejecución: Debe haber una tarjeta insertada en el lector. Debe haberse autentificado.	

Entrada/Pasos de ejecución: • Se solicita mostrar datos de los pacientes. • El *middleware* envía dicha información al *applet*. • El *Applet* procede a retornar los datos. • El *Middleware* recibe la información y la muestra al usuario.
Resultado esperado: Se muestran los datos solicitados.
Evaluación de la prueba: Prueba satisfactoria.

Tabla 20: CP5-HU4 Proceso obtención de la información 1

3.5 Conclusiones del capítulo

En la realización de este capítulo se llevó a cabo todo el proceso de implementación:

- Con el uso de las tarjetas CRC se permitió ver las clases no solo como almacenadora de los datos, sino también, el comportamiento de estas, sus colaboradores y sus responsabilidades.
- Las tareas de ingeniería mostraron a los programadores, las funcionalidades específicas a desarrollar, con más claridad.
- Las pruebas al sistema permitieron verificar y relevar la calidad del sistema, durante todo el proceso de implementación.
- Las pruebas unitarias sirvieron de mucha ayuda durante el desarrollo del sistema, permitiendo así que este culminara sin errores y cumpliendo con las historias de usuarios.
- Las pruebas de aceptación le dieron seguridad y satisfacción al cliente, ya que estas se produjeron correctamente.

3.6 Conclusiones generales

Producto a la investigación llevada a cabo, y para dar cumplimiento a las tareas se arribó a las siguientes conclusiones:

- Se analizaron las características de la arquitectura del *applet*, estas fueron el punto de partida para la implementación de dicha aplicación.
- El sistema desarrollado cumple con algunos estándares definidos para el trabajo con tarjetas inteligentes e historias clínicas.
- Se realizó un estudio de los procedimientos que se utilizan para almacenar información en las historias clínicas, esto sirvió de apoyo para almacenar la información seleccionada del paciente en el *applet*.
- Se realizó un estudio de las tarjetas inteligentes y como estas son utilizadas para almacenar información referente a la salud y los reportes médicos.
- El proceso de desarrollo de *software* estuvo guiado satisfactoriamente por la metodología XP, cumpliendo con la obtención de los artefactos definidos para dicha metodología y el tiempo previsto para el desarrollo de la solución.
- Se implementó la aplicación *applet* y su componente *middleware*, teniendo una cómoda comunicación entre ellos y una interfaz que permite al usuario gestionar la información contenida en dicho *applet*.
- El desarrollo guiado por pruebas aseguró el cumplimiento de los objetivos trazados en las historias de usuario.
- El sistema a partir de los resultados obtenidos durante la fase de prueba se encuentra listo para desplegarse.

Recomendaciones

A partir de las experiencias obtenidas en el desarrollo del trabajo y con vista a lograr un aprovechamiento óptimo del resultado alcanzado se recomienda:

- Organizar una infraestructura económica lo suficientemente robusta para aplicar esta solución informática en nuestro país.
- Ampliar la visión y alcance de la aplicación e incursionar en la web.
- La utilización de un gestor de base datos para el almacenamiento de las historias clínicas en el hospital, para así poder consultarlas y comparar notas.
- Hacer la solución más compatible con estándares internacionales para poder comercializar.
- Seguir en la implementación del sistema para que pueda logar la compatibilidad con los sistemas de código abierto.

Referencias bibliográficas

1. ANDRÉS FERNÁNDEZ, E. O. Tecnologías de la información y la comunicación en el sector salud: oportunidades y desafíos para reducir inequidades en América Latina y el Caribe. . 2010, nº Disponible en: http://www.cepal.org/cgi-bin/getProd.asp?xml=/publicaciones/xml/3/40953/P40953.xml&xsl=/dds/tpl/p9f.xsl&base=/drni/tpl/top-bottom.xslt.
2. CARNICERO, J. De la historia clínica a la historia de salud electrónica(resumen). 2010., nº Disponible en: http://www.conganat.org/seis/informes/2003/PDF/capitulo1.pdf.
3. SÁNCHEZ, D. H. H. ¿Qué sabe usted acerca de la Historia clínica electrónica? Disponible en: http://www.sld.cu/sitios/otorrino/temas.php?idv=15212.
4. WOLFGANG RANKL , W. E. Smart Card Handbook. 3 ed. 2004.
5. LANZA, I. J. L. A. La historia clínica electrónica: ideas, experiencias y reflexiones. 2005, nº Disponible en: http://bvs.sld.cu/revistas/aci/vol13_5_05/aci02505.pdf.
6. AGUSTÍN, C. C. D. Historia clínica electrónica Disponible en: http://www.exeforum.com/eventos/hce/index.html.
7. DIAS, E. S. Historia clínica electrónica. 2008, nº
8. STUDIES, I. F. H. C. HL7 Introductory Tutorial and Certification. 2008, nº Disponible en: http://www.ihcs.msu.edu/hl7/HL7%20Pamphlet.pdf.
9. STUDIES, I. F. H. C. HL7. 2009, nº Disponible en: www.iguanadoc.googlecode.com/svn/trunk/.../estandares/HL7/HL7_Tesis.pdf.
10. ORTEGA, M. Implementación de Tarjeta Inteligente Java Card para el Control de Acceso a Instalaciones. 2011, nº Disponible en: http://www.eatis.org/eatis2010/portal/paper/memoria/html/files/sistemas/Maria%20Ortega.pdf. .
11. LEYVA., M. R. P. C. L. F. Sistema de Administración de Tarjetas Inteligentes y Aplicaciones para la Cédula de Identidad Electrónica de la República Bolivariana de Venezuela. Trabajo. UCI, 2008.
12. ANDRES. Tarjetas Inteligentes Disponible en: http://aprendamossobretarjetasdelpc.blogspot.com/2010_04_01_archive.html.
13. ERLICH, J. Especificación Formal de la Máquina Virtual Java Card Disponible en: www.fing.edu.uy/.../informacion/.../documentacion_especificacionjavacard.doc.

14. DAYRON ALMEIDA SOTOLONGO, J. S. V. Solución para el control de acceso a la información de las entidades externas, en la cédula de identificación electrónica de la República Bolivariana de Venezuela. 2008, nº
15. *GLOBALPLATFORM*. Card Specification. 2003, nº
16. VIÑOLO, K. P. y SANTANA, V. F. Plataforma para el desarrollo de servicios en línea utilizando tarjetas inteligentes. 2010, nº
17. PLATFORM, J. JavaTM 2 Platform Standard Ed. 5.0. 2010, nº Disponible en: http://download.oracle.com/javase/1.5.0/docs/api/java/*applet*/*Applet*.html.
18. KARL. *Middleware*. 2011, nº Disponible en: http://www.buenastareas.com/ensayos/*Middleware*/2031177.html.
19. ALLIANCE, S. C. The Taiwan Health Care Smart Card Project. 2005, nº
20. ORDAZ, B. D. EMedix. 2005, nº Disponible en: http://www.tiendavirtual.ws/emedix/contenido.cfm?cont=MAIN&CFID=261012&CFTOKEN=70832138
21. SOFTONIC. Softonic. 2004, nº Disponible en: http://medical-2003-st.softonic.com/.
22. ATHOS A SÁNCHEZ MANSOLO1, J. L. I. D., GABRIEL PERDOMO GONZÁLEZ3, JOSÉ y LUIS HERNÁNDEZ CÁCERES4, D. M. Historia Clínica Electrónica en Cuba, quimera o posiblidad real. 2002, nº
23. EXPÓSITO, E. D. Metodologías de desarrollo de software. ¿Cuál es el camino? nº
24. UML. Uml sistemas - Document Transcript 2005, nº Disponible en: http://www.slideshare.net/Giorlysole/uml-sistemas.
25. GARCERANT. Modelo de Dominio. 2010, nº Disponible en: http://synergix.wordpress.com/2008/07/10/modelo-de-dominio/.
26. LODOÑO. Una Introducción a Scrum. nº Disponible. 2005, nº Disponible en: http://www.scribd.com/doc/27034150/002-introduccion-a-SCRUM.
27. NASH, D. Implementing and Managing E-Security. McGraw-Hill, Inc. New York, NY, USA,. 2001,nº
28. INFANTE, J. Arquitectura de Software: patrones de arquitectura y de diseño. 2011, nº Disponible en: http://comunidades.uci.cu/blogs/desarrolloSOA/2011/01/arquitectura-de-software-patrones-de-arquitectura-y-de-diseno/.

29. ADRIANA GÓMEZ, M. D. C. L., SILVINA MIGANI, ALEJANDRA OTAZÚ. UN MODELO DE ESTIMACION DE PROYECTOS DE SOFTWARE. nº

30. JONSSON, K. Cryptography Specifications Version. 2003, nº

Bibliografías consultadas

1. ALBERTO, I. D. TRABAJO PRACTICO FINAL DE CRIPTOGRAFIA Y SEGURIDAD INFORMATICA (66.69). 1998, nº
2. ALIANCE, S. C. Sistemas de Administración de Identidad, Tarjetas Inteligentes y Privacidad. 2010, nº Disponible en: www.smartcardalliance.org/latinamerica.
3. BETARTE, G. Proyecto de Taller V (2000) Programación de JavaCards. 2001, nº
4. GEMALTO. Java Card™ & STK *Applet* Development Guidelines. 2005.
5. GOÑI, C. A. Informes SEIS De la historia clínica a la historia de salud electrónica. 2003.
6. HOLCOMBE, B. GOVERNMENT SMART CARD HANDBOOK. 2004.
7. INFORÁTICAS, I. N. D. T. H. INGENIERÍA DEL SOFTWARE: METODOLOGÍAS Y CICLOS DE VIDA. 2009.
8. JURGENSEN, T. M. y GUTHERY, S. B. Smart Cards: The Developer's Toolkit. 2008, nº
9. MARTÍNEZ, H. Dispositivo de Control de Acceso Mediante Java y Tarjetas Inteligentes. 2008, nº
10. MAYES, K. E. y MARKANTONAKIS, K. Smart Cards, Tokens, Security and Applications. 2008.
11. SÁNCHEZ, A. G. VALIDACIÓN DE UNA HISTORIA CLÍNICA ELECTRÓNICA PARA PACIENTES GRAVES. 2007.
12. SÁNCHEZ, A. G. Alberto Gómez Sánchez. 2008, nº
13. VANDEWALLE, J. J. Smart Cards, *Framework*, and Application Models 2009.

Glosario de términos

API: Una interfaz de programación de aplicaciones o API (del inglés *Application Programming Interface*) es el conjunto de funciones y procedimientos (o métodos, en la programación orientada a objetos) que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción. Son usados generalmente en las bibliotecas.

APDU: El *Application Protocol Data Unit* (APDU) es la unidad de comunicación entre un lector de tarjetas inteligentes y una tarjeta inteligente. La estructura de un APDU está definida en los estándares ISO/IEC 7816.

Applet: es un componente de una aplicación que se ejecuta en el contexto de otro programa, por ejemplo un navegador web. El *applet* debe ejecutarse en un contenedor, que lo proporciona un programa anfitrión, mediante un plug-in, o en aplicaciones como teléfonos móviles que soportan el modelo de programación por '*applet*s'.

Anuencia: Consentimiento, permiso para realizar algo.

EE.UU: Estados Unidos de América.

Fidelización de clientes: consiste en lograr que un cliente (una persona que ya ha adquirido nuestros productos o servicios) se convierta en un cliente fiel a nuestros productos, marca o servicios; es decir, se convierta en un cliente asiduo o frecuente.

HC: Historias Clínicas.

HCE: Historia Clínica Electrónica.

ISO 14443: es un estándar internacional relacionado con las tarjetas de identificación electrónicas, en especial las tarjetas inteligentes, gestionado conjuntamente por la Organización Internacional de Normalización (ISO) y Comisión Electrotécnica Internacional (IEC).

ISO 15693: es un estándar ISO para "Tarjetas de Vecindad" (*Vicinity Cards*), como por ejemplo las tarjetas que pueden ser leídas desde una mayor distancia que las tarjetas de proximidad.

ISO/IEC 7816: es un estándar internacional relacionado con las tarjetas de identificación electrónicas, en especial las tarjetas inteligentes, gestionado conjuntamente por la Organización Internacional De Normalización (ISO) y Comisión Electrotécnica Internacional (IEC). Se trata de una extensión de la ISO 7810.

Middleware: es un *software* de conectividad que ofrece un conjunto de servicios que hacen posible el funcionamiento de aplicaciones distribuidas sobre plataformas heterogéneas.

MINSAP: Ministerio de Salud Pública de Cuba es el organismo rector del Sistema Nacional de Salud, encargado de dirigir, ejecutar y controlar la aplicación de la política del Estado y del gobierno en cuanto a la Salud pública, el

desarrollo de las Ciencias Médicas y la industria médico-farmacéutica.

MIFARE: es la tecnología de tarjetas inteligentes sin contacto (TISC) más ampliamente instalada en el mundo.

PIN: (Personal Identification Number o Número de Identificación Personal en español) es un valor numérico usado para identificarse y poder tener acceso a ciertos sistemas o artefactos, como un teléfono móvil o un cajero automático.

Reset: Se conoce como la puesta en condiciones iniciales de un sistema. Este puede ser mecánico, electrónico o de otro tipo. Normalmente se realiza al conectar el mismo, aunque, habitualmente, existe un mecanismo, normalmente un pulsador, que sirve para realzar la puesta en condiciones iniciales manualmente.

RUP: El Proceso Racional Unificado (*Rational Unified Process* en inglés, habitualmente resumido como RUP) es un proceso de desarrollo de software y junto con el Lenguaje Unificado de Modelado UML, constituye la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos.

Release: Una versión candidata a definitiva o candidata para el lanzamiento, aunque más conocida por su nombre en inglés *release* candidate, comprende un producto final, preparado para publicarse como versión definitiva a menos que aparezcan errores que lo impidan.

UML: Lenguaje Unificado de Modelado (LUM o UML, por sus siglas en inglés, *Unified Modeling Language*) es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad; está respaldado por el OMG (*Object Management Group*). Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema.

RFID: siglas de Radio *Frequency IDentification*, en español identificación por radiofrecuencia es un sistema de almacenamiento y recuperación de datos remoto que usa dispositivos denominados etiquetas, tarjetas, transponedores RFID.

Softel-UCI: Empresa que ofrece soluciones informáticas para el Sistema de Salud.

TIC: Tecnologías de la Información y la Comunicación.

XP: La programación extrema o *eXtreme Programming* (XP) es un enfoque de la ingeniería de software formulado por Kent Beck.

Anexos

Anexo 1. Tabla de responsables por tareas.

Tareas	Responsable
- Realizar una caracterización sobre la arquitectura de las tarjetas inteligentes como base para la implementación de la aplicación applet.	Diana Rosa Zapata Vizcaíno
- Investigar sobre los estándares asociados a este tipo de tecnología para su puesta en práctica en la implementación a realizar.	Raidel Abreu Patterson
- Investigar acerca de los procedimientos y estrategias utilizados en el sistema de salud para el almacenamiento de historiales clínicos.	Diana Rosa Zapata Vizcaíno
- Realizar un estudio de sistemas homólogos que utilicen las tarjetas inteligentes usadas en reportes médicos.	Diana Rosa Zapata Vizcaíno
- Seleccionar la metodología de desarrollo de la solución.	Raidel Abreu Patterson
- Desarrollar la aplicación applet y su componente middleware.	Raidel Abreu Patterson y Diana Rosa Zapata
- Realizar las pruebas de calidad a la solución.	Raidel Abreu Patterson

Tabla 21: Responsables por tareas 1

Anexo 2. Estructura comando APDU

Estructura Comando APDU.		
- Campo	- Tamaño	- Descripción
- CLA	- 1 byte	- Clase de instrucción. Indica la estructura y el formato.
- INS	- 1 byte	- Código de instrucción. Especifica la instrucción del comando.

- P1	- 1 byte	- Parámetros de la instrucción. Proveen más información sobre la instrucción.
- P2	- 1 byte	- Número de bytes en el Data Field del APDU.
- Data Field	- LC bytes	- Secuencia de bytes con información.
- LE	- 1 byte	- Cantidad máxima de bytes esperados como respuesta.

Tabla 22: Estructura comando APDU 1

Anexo 3. Tareas de ingeniería.

Tarea de Ingeniería	
Número Tarea: 1	**Historia de Usuario (1-** Inicializar Comunicación**)**
Nombre Tarea: Listar lectores conectados.	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 21/02/2011	**Fecha Fin:** 27/02/2011
Programador Responsable: Diana Rosa Zapata Vizcaino.	
Descripción: Se muestran los lectores que están disponibles, se selecciona uno.	

Tabla 23: Tarea de ingeniería (HU-1) 1

Tarea de Ingeniería	
Número Tarea: 2	**Historia de Usuario (2-** Establecer canal seguro.**)**
Nombre Tarea: Intercambio de Información, por canal seguro, entre el middleware y el applet de historia clínica electrónica.	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 2
Fecha Inicio: 7/03/2011	**Fecha Fin:** 20/03/2011
Programador Responsable: Diana Rosa Zapata Vizcaino.	
Descripción: Se establece un canal de intercambio de información entre el middleware y el applet de historia clínica electrónica,	

Tabla 24: Tarea de ingeniería (HU-2) 1

Tarea de Ingeniería	
Número Tarea: 3	**Historia de Usuario (3-** Autenticación al applet de historia clínica electrónica**)**
Nombre Tarea: Autenticar al applet historia clínica electrónica, mediante el envió de datos.	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 2

Fecha Inicio: 21/03/2011	**Fecha Fin:** 3/04/2011
Programador Responsable: Raidel Abreu Patterson	
Descripción: El middleware autentica al applet, mediante el envió de datos, estableciendo un canal seguro de intercambio de información entre middleware y el applet.	

Tabla 25: Tarea de ingeniería (HU-3) 1

Tarea de Ingeniería	
Número Tarea: 4	**Historia de Usuario (4-** Gestionar Almacenamiento Información**)**
Nombre Tarea: Crear la estructura de ficheros en el applet de historia clínica electrónica.	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 2
Fecha Inicio: 4/04/2011	**Fecha Fin:** 17/04/2011
Programador Responsable: Raidel Abreu Patterson	
Descripción: Permitirá crear la estructura de ficheros en el applet, donde se va almacenar la información sanitaria de los pacientes.	

Tabla 26: Tarea de ingeniería (HU-4) 1

Tarea de Ingeniería

Número Tarea: 5	**Historia de Usuario (5-** Enviar y recibir comandos APDU de la tarjeta**)**
Nombre Tarea: - Enviar comandos APDU a la tarjeta. - Recibir respuesta APDU de la tarjeta.	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 2
Fecha Inicio: 25/04/2011	**Fecha Fin:** 8/05/2011
Programador Responsable: Raidel Abreu Patterson	
Descripción: el usuario hace una solicitud que es procesada y se envía los correspondientes comandos APDU que son transmitidos a la tarjeta. A este comando la tarjeta siempre devolverá una respuesta que puede ser satisfactoria, la información solicitada o un mensaje de error.	

Tabla 27: Tarea de ingeniería (HU-5) 1

Tarea de Ingeniería	
Número Tarea: 6	**Historia de Usuario (6-** Finalizar Comunicación**)**
Nombre Tarea: Desconectar la tarjeta del lector.	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 9/04/2011	**Fecha Fin:** 15/05/2011

Programador Responsable: Raidel Abreu Patterson
Descripción: se realiza la desconexión entre la tarjeta inteligente, donde se encuentra el applet de historia clínica electrónica y el lector.

Tabla 28: Tarea de ingeniería (HU-6) 1

Anexo 4 Tabla de datos del proceso asistencial (APDU)

Algunos datos del proceso asistencial							
Variable	**Clase**	**Instrucción**	**P 1 Guardar**	**P 1 Mostrar**	**P 2**	**Datos**	**Longitud**
Nombre completo	90	01	G01	M21	0	Datos introducidos	Longitud de datos
Fecha Nacimiento			G02	M22			
Edad			G03	M23			
País nacimiento			G04	M24			
Sexo			G05	M25			
Dirección			G06	M26			
CI			G07	M27			
Tipo sangre			G08	M28			
Estado civil			G09	M29			
Teléfono			G10	M30			
Nombre del padre			G11	M31			
Nombre de la madre			G12	M32			
Situación familiar			G13	M32			

Color de piel			G14	M33			
Contacto de urgencia			G15	M34			
Contacto de urgencia(Telf.)			G16	M35			
Dirección del Contacto			G17	M36			
Fecha de Actualización			G18	M37			

Ilustración 10: Algunos datos del proceso asistencial 1

Ilustración 11: Formato de un APDU. 1

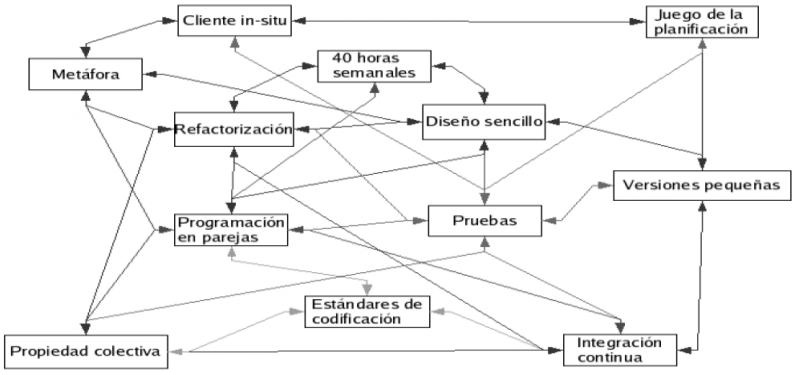


Ilustración 12: Prácticas de XP 1

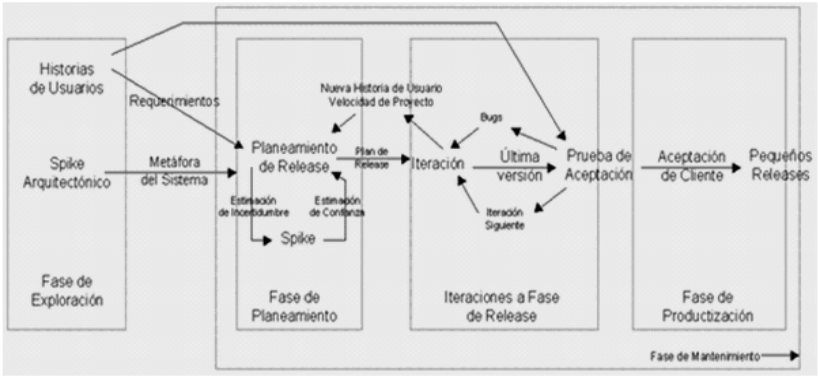


Ilustración 13: Fases de XP 1

Ilustración 14: Diagrama de capas de comunicación