Universidad de las Ciencias Informáticas

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

Título: Módulo de Control de Acceso Extendido para el sistema de verificación de seguridad de documentos de viajes de lectura mecánica (DocSec).

Autores:

Daynelis Valdes Monrabal.

Rafael David Portilla Santiáguez.

Tutores:

Msc. Alina Surós Vicente.

Ing. Enrique Almeida Maldonado.

La Habana, Cuba.

Junio, 2013.

Declaración de Autoría

Declaramos que somos los únicos autores de este trabajo y autorizamos al Centro de Identificación y Seguridad Digital (CISED) de la Universidad de las Ciencias Informáticas (UCI) a hacer uso del mismo en su beneficio, con carácter exclusivo.

Para que así conste firmamos la presente a los __ días del mes de _____de 2013.

_____ _____

Autor: Daynelis Valdes Monrabal. Autor: Rafael David Portilla Santiáguez.

_____ _____

Tutor: Alina Surós Vicente. Tutor: Enrique Almeida Maldonado.

Agradecimientos

Agradecemos a todos aquellos que de una forma u otra nos ayudaron al desarrollo de este trabajo, en especial a nuestros tutores: Alina por habernos dedicado con tantas horas extras, por todo el apoyo y el cariño que nos ha brindado, y a Enrique por guiarnos siempre por el lugar correcto, por su preocupación y ayuda. Además quisiéramos agradecer a todo los profes que una forma u otra nos dieron su ayuda incondicional: Adonis, Lourdes, Reynier, Yayneris, Yendry, José Carlo, Félix y Wilhem.

Los autores.

Al gran amor de mi vida: mis papitos lindos, por estar siempre a mi lado sin importar las circunstancias, por ser mi guía, mi ejemplo cada día, por haberme apoyado en cada una mis decisiones, por su amor, cariño, esfuerzo y dedicación. Cada día le doy gracias a la vida por haberme regalado a dos padres tan maravillosos, a la mejor mamá del mundo y mi adorado papá, que aunque no está hoy aquí físicamente conmigo, yo sé que siempre está a mi lado protegiéndome y ayudándome. Hoy puedo darles lo que tanto han querido, hoy nuestro sueño se hace realidad. Muchas gracias, porque a ustedes les debo quien soy. A mi hermana querida Daylene por ser la mejor de todas y por siempre estar a mi lado cuando más la he necesitado, por ser mi mejor amiga y mi confidente. A mi sobrinita Karina por ser mi motor impulsor, por generar en mí el deseo de ser cada día una mejor persona, para que ella se sienta orgullosa de mí. A mis dos abuelitos queridos mima y papi por ser tan buenos y generosos conmigo, por todos esos días en que me regalaron su amor y comprensión. A mis suegros Grisel y José por haberme acogido como una hija y por todo el cariño que me han brindado. A Roinel que aunque no sea mi hermano de verdad, lo quiero como tal. A mi compañero de tesis Rafa, sin el cual no hubiera podido cumplir este sueño, por todo el tiempo que me ha dedicado, por su comprensión, por haberse convertido en mi amigo, y por siempre tener una palabra de aliento cuando todo se veía perdido. Hay muchas personas que a lo largo de estos 5 años han formado parte de mi vida y me han ayudado a superarme y a ser una mejor persona. Le agradezco a mi equipo de siempre por haber compartido tantos días de preocupación y de alegrías cuando todo nos salía bien Olber, Junior y Quiles mi gran amigo. A mis amigos Julié, Liliett, Grettel, Leanni, Celia, Maricé, Kenia, Fernando y Alejandro por cada uno de los momentos que compartimos juntos en las buenas y en las malas, a los que voy a extrañar mucho cuando nos vayamos. Que decir de mi amorcito bello, Ronal a ti te tengo que agradecer por estos maravillosos 5 años que hemos vivido juntos, por hacerme la persona más feliz del mundo, por darme tanta fuerza cada vez que me sentía perdida, por toda tu comprensión, tus mimos y por el amor que cada día me das. Gracias por haberme permitido ser parte de tu vida.

Daynelis.

Agradecimientos

A mis padres que han servido de guía e inspiración durante estos 18 años de estudios. A mis tías y a mis abuelas que siempre estuvieron para dar su granito de arena. A Mey por todo el camino que recorrimos juntos. A mi compañera de tesis Day, mejor imposible, cuyo esfuerzo hizo más dulce el agotador trabajo. A mis amistades del grupo Elián, Amed, Carlos, Quiles, Leo, Junior, Fernando, Luis Miguel, Olber. A las muchachitas Heidi, Leanny, Julié, Yaumara. A Elizabeth y a Jenny por su amistad. A Yunn por su ayuda.

Rafael David

Dedicatoria

Dedico este trabajo a las personas más importantes de mi vida, por siempre estar a mi lado y por brindarme cada día tanto amor:

A mi mamá y a papá por ser mi ejemplo cada día y por confiar siempre en mí.

A mi hermana y a mi sobrina por estar siempre junto a mí.

A mis abuelitos por todo su amor.

A Ronal por cada uno de los días de felicidad que me ha regalado.

Daynelis

Para aquellos que me guiaron, para aquellos que me trajeron aquí donde estoy. Sea este trabajo resultado de sus esfuerzos y su guía:

A mis padres Martha y Rafael por no dejarme ser deportista,

A mis abuelas Olimpia y Mirtha por su sabiduría y su constancia.

A mis tías Inesita, Idelsis, Mirtha y Marilú por su apoyo y cariño.

A Mailén por tantas cosas compartidas.

Rafael David

Resumen

El presente trabajo aborda el Control de Acceso Extendido y las versiones que implementan los pasaportes electrónicos como medida de seguridad. Se realiza un análisis de los sistemas que permiten la ejecución de esta medida de seguridad a nivel mundial, centrando la investigación en el sistema que está actualmente funcionando en la República Bolivariana de Venezuela, el cual solo posee la tecnología que posibilita la ejecución del Control de Acceso Extendido definido por la Unión Europea en su primera versión. Se propone un sistema que permita la ejecución del Control de Acceso Extendido definido por la Unión Europea en sus versiones 1 y 2 y el implementado por Singapur, que facilitarán el acceso a los datos biométricos sensibles almacenados en los pasaportes electrónicos desde los puntos de control migratorios de Venezuela. Además el sistema posibilitará la gestión de los certificados y de los sistemas de inspección que tendrán acceso a utilizarlo.

Palabras clave: control de acceso extendido, medida de seguridad, pasaporte electrónico

Índice de Contenido

Índice de Contenido

Índice de Contenido

Índice de Figuras

Índice de Tablas

Índice de Tablas

Introducción.

Las tecnologías de la informática y las comunicaciones tienen cada día un espacio más significativo en el mundo moderno, dada su capacidad para el fácil almacenamiento de los datos y la velocidad para procesar gran cúmulo de información. Muchas de las operaciones que hace unos años se hacían manualmente, en la actualidad son realizadas por aplicaciones informáticas, reduciendo drásticamente el tiempo, los costos y brindando mayor eficiencia.

Los procesos de identificación, migración y extranjería no se han visto exceptuados de este avance, una de las entidades cuyo trabajo está vinculado con estos procesos es la Organización de la Aviación Civil Internacional (OACI), la cual establece normas y regulaciones internacionales necesarias para garantizar la seguridad, eficiencia y regularidad del transporte aéreo (1). La OACI en 1968 redactó las recomendaciones para una libreta o tarjeta de pasaporte normalizada que fuera susceptible a la lectura mecánica para acelerar el despacho de pasajeros por los puntos de control. Entre esas recomendaciones estaba el reconocimiento óptico de caracteres dado su eficacia y fiabilidad (2).

En 1980 la OACI creó la primera edición del Doc. 9303, donde se publican las especificaciones y textos de orientación de dicho documento. Actualmente el Doc. 9303 se publica en 3 partes: Parte 1, Pasaporte de lectura mecánica; Parte 2, Visa de lectura mecánica y Parte 3, Documentos oficiales de viaje de lectura mecánica (3). Cada una de estas partes ha recibido la aprobación de la Organización Internacional de Normalización (ISO, por sus siglas en inglés) con carácter de normas ISO-7501-1, ISO-7501-2 y ISO-7501-3, respectivamente (4).

Con la sexta edición de dicho documento, se observa una considerable modernización respecto a versiones anteriores. Esta incluye una nueva norma de interfuncionamiento mundial y su almacenamiento en el circuito integrado sin contacto (ICC, por sus siglas en inglés) según la norma ISO/IEC 14443, brinda nuevas opciones para el almacenamiento de los datos conexos y para la identificación biométrica del titular (2).

Los datos almacenados en el ICC deben poderse leer y ser interpretados por cualquier nación. Con ese fin la estructura lógica de los datos (LDS, por sus siglas en inglés) está compuesta por grupos de datos

estructurados lógicamente que pueden tener carácter obligatorio u opcional. Se hace necesaria además la implementación de una serie de medidas de seguridad que velen por el correcto acceso a la información que se encuentra en el ICC. Algunos de los protocolos que rigen estas medidas no han sido estandarizados internacionalmente, provocando que existan diferentes implementaciones (3).

Las medidas de seguridad que han sido definidas por la OACI son: la Autenticación Pasiva (PA, por sus siglas en inglés), la Autenticación Activa (AA, por sus siglas en inglés), el Control de Acceso Básico (BAC, por sus siglas en inglés), el Control de Acceso Extendido (EAC, por sus siglas en inglés) y el Cifrado (3). El Control de Acceso Extendido es el responsable por la protección de los datos biométricos adicionales, llamados datos sensibles, a los cuales solo podrán tener acceso los sistemas de inspección[1] que hayan sido autorizados por el emisor del pasaporte electrónico (pasaporte-e).

Del Control de Acceso Extendido se han identificado dos implementaciones, una en Singapur (EAC-Singapur) y otra en la Unión Europea (EAC-UE). El EAC-Singapur se basa principalmente en almacenar en el grupo de datos 13 del ICC la llave simétrica necesaria para la comunicación, encriptada con la llave pública del sistema de inspección autorizado para su lectura (5). Mientras que el EAC-UE basa su implementación en dos mecanismos: la autenticación del ICC y la autenticación del terminal de lectura. El primero de estos mecanismos demuestra la veracidad del ICC basado en una fuerte encriptación mientras que el segundo le permite comprobar al ICC que el sistema de inspección está autorizado a leer la información sensible que contiene (3).

Actualmente, para la República Bolivariana de Venezuela se desarrolla en la Universidad de las Ciencias Informáticas (UCI), en el Centro de Identificación y Seguridad Digital (CISED), el proyecto de Desarrollo de Solución Integral para la Transformación y Modernización del Sistema de Identificación, Migración y Extranjería. Este proyecto en el área de *software* tiene como objetivo modernizar los sistemas de Identificación, Migración y Extranjería. En función de lograr este objetivo, se ha desarrollado el sistema DocSec para proveer un sistema que permita a los puntos de inspección de la República Bolivariana de Venezuela verificar las medidas de seguridad de los pasaportes electrónicos. En una primera versión se

[1] Sistema de inspección: Terminal oficial que siempre es operado por una organización gubernamental y se encuentra en el control fronterizo de una nación, donde se realizan los procedimientos de inspección de los pasaportes-e.

implementó la autenticación pasiva, única medida de seguridad de carácter obligatorio de las cinco establecidas por la OACI.

El proyecto técnico Puntos de Control Migratorios de la República Bolivariana de Venezuela fase III, tiene como uno de los objetivos proveer un sistema para la lectura de los pasaportes electrónicos (6). Para ello se adquiere el lector Regula, este cuenta en su *Software Development Kit* (SDK) con funcionalidades para verificar las medidas de seguridad de los pasaportes. Para ejecutar el EAC, el SDK cuenta con varios comandos, para realizar la autenticación del *chip* y del terminal en ese orden. Por lo que solo permite la ejecución del EAC- UE en su versión 1, no permitiendo el acceso a los datos sensibles si los pasaportes-e tienen implementado el EAC-UE en su versión 2 o el EAC-Singapur. El EAC cuenta con una infraestructura de clave pública por lo que hay que tener en cuenta la fecha de expiración de los certificados emitidos por cada uno de los niveles que interfieren en esta infraestructura, debido a que los sistemas de inspección tienen un período de validez de un mes, el del verificador de documento de tres meses y el de la autoridad verificadora de certificación del país de un año. Esta gestión de certificados se debe realizar de forma local, tornando lento y complicado este proceso para todos los puntos de inspección donde será desplegado el sistema, teniendo en cuenta la frecuencia de actualización antes mencionada.

Así mismo en el caso de que se adquieran nuevos lectores, no garantizan que el SDK, en el caso de precisarlo, contenga funcionalidades para realizar el EAC implicando que sea una condición para nuevas adquisiciones que impliquen costos mayores. Teniendo en cuenta la situación problemática planteada se define como **problema de investigación**: ¿Cómo mejorar la ejecución del EAC de los pasaportes-e en los sistemas de inspección de la República Bolivariana de Venezuela de manera que se pueda acceder a los datos biométricos adicionales de todos los pasaportes-e que implementen el EAC y se agilice la gestión de los certificados independientemente del lector?

El **objeto de estudio** de esta investigación queda enmarcado en el proceso de ejecución del Control de Acceso Extendido de los pasaportes-e.

El **objetivo general** de esta investigación es desarrollar un sistema para la ejecución del EAC que brinde las operaciones criptográficas para permitir el acceso a los datos sensibles de los pasaportes-e desde los puntos de inspección de la República Bolivariana de Venezuela. El **campo de acción** se ha delimitado en

el proceso de ejecución del Control de Acceso Extendido de los pasaportes-e, en los sistemas de inspección de la República Bolivariana de Venezuela.

Los **objetivos específicos** que se derivan del objetivo del trabajo son:

- Analizar los sistemas que implementan el EAC, así como las herramientas, tecnologías y metodología a utilizar en el desarrollo del sistema.
- Diseñar el módulo de EAC.
- Implementar el módulo de EAC.
- Realizar las pruebas unitarias y las funcionales para validar el correcto funcionamiento del sistema desarrollado.

Las **tareas de investigación** para darle cumplimiento al objetivo del trabajo son:

1. Realización del estudio del estado del arte de los sistemas existentes para la ejecución del Control de Acceso Extendido.
2. Análisis de las normas de la Organización de Aviación Civil Internacional (OACI), referente a la lectura mecánica de los pasaportes electrónicos.
3. Análisis de metodologías y herramientas para el desarrollo de la aplicación.
4. Análisis de las soluciones existentes que permiten la ejecución del EAC.
5. Análisis y diseño de la solución.
6. Generación de los artefactos descritos por el proceso de ingeniería de *software* según la metodología seleccionada.
7. Implementación de la solución.
8. Realización de pruebas.

Los **métodos teóricos** utilizados:

- **Analítico – Sintético**: Permitió organizar toda la información obtenida en las bibliografías consultadas referente al tema de EAC, para hacer uso de la información adecuada para la realización del presente trabajo.
- **Análisis Histórico – Lógico**: Posibilitó realizar un estudio de la evolución de todo lo referente al EAC y del nivel que han alcanzado los sistemas desarrollados en el mundo referente al tema.

- **Modelación:** Permitió hacer una representación, mediante la elaboración de diagramas especificados por la metodología que se utilizará para guiar el proceso de desarrollo del *software*.

El **método empírico** utilizado:

- **Entrevista:** Permitió interactuar con el personal que tiene conocimiento sobre el tema de EAC, para aumentar el conocimiento sobre esta medida de seguridad**.**

Justificación de la investigación:

Debido a la necesidad de resolver la problemática planteada se decidió realizar el Módulo Control de Acceso Extendido para el sistema DocSec, que posibilite la lectura de los datos sensibles protegidos en el pasaporte-e, mediante la verificación del EAC, permitiendo a las autoridades migratorias de Venezuela realizar una verificación biométrica más exacta a partir de los datos obtenidos. Garantizando el acceso a estos datos en los pasaportes de la comunidad europea y de Singapur.

El documento queda estructurado en tres capítulos:

- **Capítulo 1: Fundamentación teórica**. Se realiza un análisis sobre el estado del arte de las principales implementaciones del Control de Acceso Extendido, de cada una de las tecnologías, herramientas y metodologías a emplear durante la investigación.

- **Capítulo 2: Propuesta de solución.** Se aborda al análisis y diseño del sistema cumpliendo con los patrones de diseños y de arquitectura que se seleccionaron para el desarrollo del sistema. Se generarán los artefactos necesarios durante el ciclo de vida del proceso de desarrollo, especificando las características del sistema, desglosadas en escenarios y requisitos de calidad del servicio. Además se diseña el diagrama de clases.

- **Capítulo 3: Implementación y pruebas.** Se describen las características de la solución propuesta, cumpliendo con los estándares de codificación para el desarrollo del sistema, se muestran los diagramas de componentes y despliegue, así como la descripción de las pruebas realizadas al sistema.

Capítulo 1: Fundamentación Teórica

1.1. Introducción

En el presente capítulo se muestran los resultados de la investigación realizada sobre el estado del arte de los sistemas informáticos que desarrollan el EAC. Se hace referencia a los conceptos más importantes relacionados con el tema, necesarios para entender el objetivo y el alcance del sistema a desarrollar. Se realiza un estudio sobre las principales metodologías de desarrollo de *software* existentes y se selecciona la que guiará el proceso de desarrollo del sistema. Además se selecciona las herramientas y tecnologías que serán utilizadas para el desarrollo de la solución.

1.2. Pasaporte electrónico

El pasaporte-e surge por la necesidad de incluir datos biométricos para mejorar la verificación de la identidad de los portadores. Es un documento de alta tecnología de lectura mecánica que contiene un circuito integrado donde son guardados los datos, como lo especifica la OACI. Contiene la información biométrica necesaria para autenticar la identidad de los propietarios, los métodos actualmente estandarizados usados para estos tipos de sistemas de identificación son: reconocimiento facial, reconocimiento de la huella dactilar y reconocimiento del iris, pero de todos ellos solamente la imagen fácil del portador es la que se almacena de forma obligatoria en el ICC. La comparación de características biométricas es realizada fuera del ICC de los pasaportes-e por los sistemas de inspección.

1.2.1. Estructura lógica de datos

Los datos al ser almacenados en el *chip* requieren una estructura de datos estandarizada para habilitar una interoperabilidad global, facilitando que todas las naciones tengan conocimiento de cómo está estructurado el pasaporte. La estructura lógica de datos (LDS, por sus siglas en inglés) está conformada por elementos de datos de uso obligatorio u opcional.

Dentro de la LDS, los elementos de datos se agrupan según su organización lógica y son definidos como grupo de datos (DG, por sus siglas en inglés), cada uno de ellos está identificado con un número, como se muestra en la Figura 1 (7).

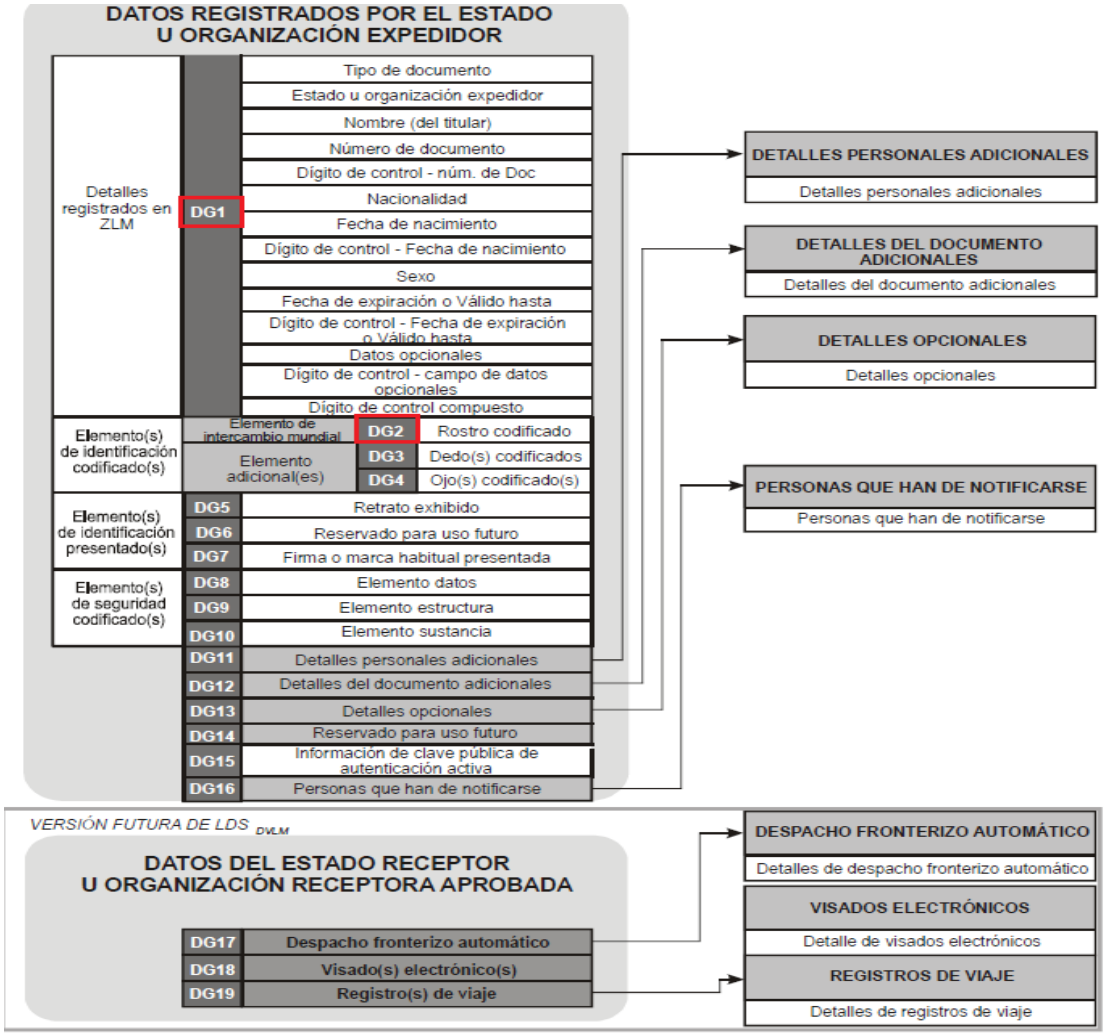


Figura 1. Datos definidos por la OACI para la LDS del ICC (2).

1.2.2. Mecanismos de seguridad del pasaporte-e

Los mecanismos de seguridad establecidos por la OACI a implementar para el acceso a los datos almacenados en el ICC son:

Medida Obligatoria:

- Autenticación Pasiva.

Capítulo 1: Fundamentación Teórica

La autenticación pasiva utiliza firma digital para autentificar los datos almacenados en los grupos de datos del ICC. Esta firma es generada por el emisor del documento en la fase de personalización del *chip* dentro del objeto de seguridad conteniendo un resumen de todos los datos del documento. En los puntos de control simplemente se realiza un nuevo resumen de los datos y se compara con el que se encuentra en el *chip*. La autenticación pasiva posibilita a los sistemas de inspección detectar grupos de datos manipulados, pero no previene contra la clonación del documento (3).

Medidas Opcionales:

- Autenticación Activa.

La autenticación activa previene contra la sustitución del ICC, determinando su autenticidad a partir de un protocolo de retro-respuesta el cual utiliza dos llaves, una pública almacenada en el grupo de datos 15 del ICC y preservada por la autenticación pasiva, y una privada guardada en memoria de forma segura en el *chip* del pasaporte, la cual solo es usada internamente (3).

- Control de Acceso Básico.

El control de acceso básico comprueba que el sistema de inspección tiene acceso físico al documento de viaje al ser leído ópticamente (3).

- Control de Acceso Extendido.

El EAC aunque es una de las medidas opcionales establecidas por la OACI, tiene gran importancia ya que se encarga de la protección del acceso a los datos biométricos adicionales (diferentes a la imagen facial, elemento obligatorio), llamados datos sensibles los cuales son la huella dactilar y el iris (3). En este proceso interviene el emisor del documento, encargado de autorizar la lectura de los datos, el sistema de inspección que intentará leer los datos y podrá hacerlo si se encuentra previamente autorizado y el ICC es el responsable de verificar que el sistema de inspección que intenta tener acceso a los datos sensibles está autorizado.

- Cifrado:

La restricción del acceso a las características biométricas adicionales puede efectuarse mediante el cifrado de las mismas. Para poder descifrar los datos cifrados, el sistema de inspección deberá contar con una clave de descifrado. La definición del algoritmo de cifrado/descifrado y las claves que han de utilizarse queda a discreción del estado que las implante.

1.3. Control de Acceso Extendido

A partir del estudio realizado fue posible identificar dos implementaciones de Control de Acceso Extendido, una en Singapur (EAC-Singapur) y otra en la Unión Europea (EAC-UE), cada una con diferentes formas de implementación.

1.3.1. EAC-Singapur

El Control de Acceso Extendido que ha implementado la República de Singapur se encuentra en el documento *"Singapore Standard SS 529: 2006, Specifications for SmartCard ID"* (5) el cual describe la estructura, medidas de seguridad y condiciones de acceso para las estructuras de datos que se encuentran en dispositivos como las tarjetas inteligentes o *microchip* habilitados. El EAC es llevado a cabo mediante el almacenamiento en el grupo de datos 13 del ICC de la llave simétrica necesaria para la comunicación, el cual está encriptado con la llave pública de cada sistema de inspección autorizado para la lectura de los datos sensibles (5), como se muestra en la Figura 2. La llave utilizada durante su ejecución es triple-DES de 16-bytes, denominada EAC *key*. El EAC de Singapur, es un esquema poco complejo, su uso permite adicionar un nivel de protección a los datos sensibles, sin embargo, un sistema de inspección autorizado no puede ser adicionado una vez personalizado el documento, además no cuenta con un mecanismo de revocación si una llave se encuentra comprometida y un sistema corrupto puede compartir la llave simétrica obtenida (8).

Llave EAC encriptada usando la llave pública asimétrica 1. (para el IS 1)	Llave EAC encriptada usando la llave pública asimétrica 2. (para el IS 2)	...	Llave EAC encriptada usando la llave pública asimétrica n-1. (para el IS n-1)	Llave EAC encriptada usando la llave pública asimétrica n. (para el IS n)

Figura 2. Representación del DG13 de los pasaportes-e de Singapur (9).

1.3.2. EAC-UE

El EAC implementado por la Unión Europea se encuentra estandarizado en el documento *"Technical Guideline (TR-03110) Advanced Security Mechanisms for Machine Readable Travel Documents - Extended Access Control (EAC)"* (3), el cual trata temas referentes a mecanismos de seguridad avanzados para pasaportes electrónicos y especificaciones de protocolos de comunicación. La implementación que propone la Unión Europea está compuesta por dos mecanismos de seguridad

fundamentales: la autenticación del ICC, la cual provee una fuerte encriptación que permite al sistema de inspección demostrar que el circuito es genuino, y la autenticación del terminal que le demuestra al ICC que el terminal puede tener acceso a los datos sensibles que contiene en el DG3 y DG4. Basado en una infraestructura de clave pública, dotando al sistema de inspección de una cadena de certificados que son enviados al ICC, que luego de verificados permite el acceso a los datos (3).

Del EAC de la UE se han identificado dos versiones, las cuales tienen como diferencia fundamental quien comienza la comunicación entre el *chip* y el terminal.

En la versión 1 comienza la comunicación la autenticación del *chip* y luego la del terminal para poder ejecutar el EAC, como se muestra en las Figura 3 y Figura 4.

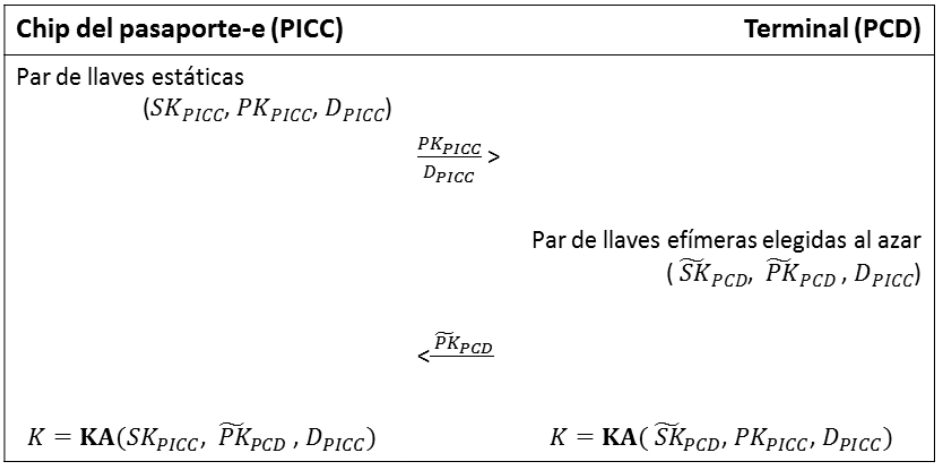


Figura 3. Autenticación del *chip* en la versión 1 del EAC-UE.

Los siguientes pasos son ejecutados por el terminal y el *chip* para la ejecución de la autenticación del *chip* del EAC-UE versión 1:

1. El *chip* le envía al terminal la llave estática pública (PK_{PICC}) y los parámetros DH(D_{PICC}) al terminal.
2. El terminal genera un par de llaves efímeras ($\widetilde{SK_{PCD}}$, $\widetilde{PK_{PCD}}$) con los parámetros del dominio recibidos y le envía la llave pública ($\widetilde{PK_{PCD}}$) al *chip*.
3. El *chip* y el terminal calculan:
    a. El secreto compartido (*K*) con las funciones para los acuerdos entre las llaves (*KA*).
    K= **KA** ($\widetilde{SK_{PICC}}$, $\widetilde{PK_{PCD}}$, D_{PICC}) = **KA**($\widetilde{SK_{PCD}}$,PK_{PICC},D_{PICC}).

b. Las llaves de sección (K_{MAC}, K_{Enc}) utilizando la función de derivación de llaves (*KDF*) a partir de *K*. K_{MAC}=KDF_{MAC}(*K*), K_{Enc}=KDF_{Enc}(*K*).
c. El terminal comprime al llave pública (**Comp** ($\widetilde{PK_{PCD}}$)) para la posterior ejecución de la autenticación del terminal.

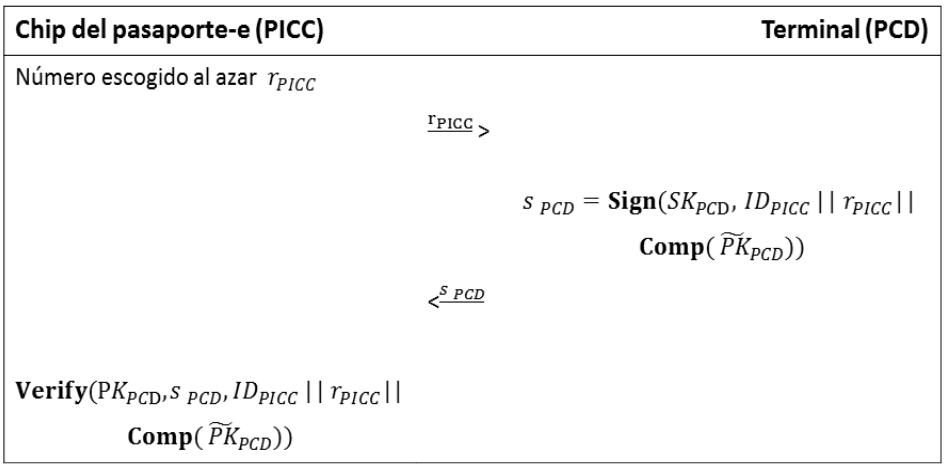


Figura 4. Autenticación del terminal en la versión 1 del EAC-UE.

Los siguientes pasos son ejecutados por el terminal y el *chip* para la ejecución de la autenticación del terminal del EAC-UE versión 1:

Para la ejecución de este mecanismo debe de realizarse previamente el BAC, de donde se obtiene el identificador del *chip* (ID_{PICC}).

1. El terminal envía una cadena de certificados al *chip*. La cadena está compuesta por el certificado del verificador de documento y el certificado del sistema de inspección.
2. El *chip* verifica los certificados y extrae la llave pública del terminal PK_{PCD}.
3. El *chip* escoge un valor aleatorio r_{PICC} y se lo envía al terminal.
4. El terminal responde con la firma: s_{PCD}=**Sing** (SK_{PCD}, ID_{PICC}||r_{PICC}||**Comp** ($\widetilde{PK_{PCD}}$)).
5. El *chip* verifica la firma: **Verify** (PK_{PCD}, s_{PCD}, ID_{PICC}||r_{PICC}||**Comp** ($\widetilde{PK_{PCD}}$))= verdadero.

Mientras que en la versión 2 del EAC-UE comienza la comunicación la autenticación del terminal y luego la del *chip*. Este proceso se puede apreciar en las Figura 5 y Figura 6 respectivamente.

Chip del pasaporte-e (PICC)		**Terminal (PCD)**
	$\xleftarrow[A_{PCD}]{\mathbf{Comp}(\widetilde{PK}_{PCD})}$	Par de llaves efímeras elegidas al azar $(\widetilde{SK}_{PCD},\ \widetilde{PK}_{PCD}, D_{PICC})$
[Número escogido al azar r_{PICC}]	$\xrightarrow{r_{PICC}}$	
	$\xleftarrow{s_{PCD}}$	$s_{PCD} = \mathbf{Sign}(SK_{PCD}, ID_{PICC} \mid\mid r_{PICC} \mid\mid \mathbf{Comp}(\widetilde{PK}_{PCD}) \mid\mid A_{PCD})$
$\mathbf{Verify}(PK_{PCD}, s_{PCD}, ID_{PICC} \mid\mid r_{PICC} \mid\mid \mathbf{Comp}(\widetilde{PK}_{PCD}) \mid\mid A_{PCD})$		

Figura 5. Autenticación del terminal en la versión 2 del EAC-UE.

Para la ejecución de este mecanismo debe de realizarse previamente el BAC, de donde se obtiene el identificador del *chip* (ID_{PICC}) así como los parámetros del dominio (D_{PICC}).

1. El terminal envía una cadena de certificados al *chip*. La cadena está compuesta por el certificado del verificador de documento y el certificado del sistema de inspección.
2. El *chip* verifica los certificados y extrae la llave pública del terminal PK_{PCD}.
3. El terminal:
    a. Genera un par de llaves efímeras ($\widetilde{SK_{PCD}}$, $\widetilde{PK_{PCD}}$) con los parámetros del dominio recibidos y le envía el comprimido de la llave pública **Comp** ($\widetilde{PK_{PCD}}$) al *chip*.
    b. Puede enviar un valor auxiliar A_{PCD} al *chip*.
4. El *chip* escoge un valor aleatorio r_{PICC} y se lo envía al terminal.
5. El terminal responde con la firma: s_{PCD}=**Sing**$(SK_{PCD}, ID_{PICC} \| r_{PICC} \| \mathbf{Comp}(\widetilde{PK_{PCD}}) \| A_{PCD})$.
6. El *chip* verifica la firma: **Verify** $(PK_{PCD}, s_{PCD}, ID_{PICC} \| r_{PICC} \| \mathbf{Comp}\ (\widetilde{PK_{PCD}}) \| A_{PCD})$=verdadero.

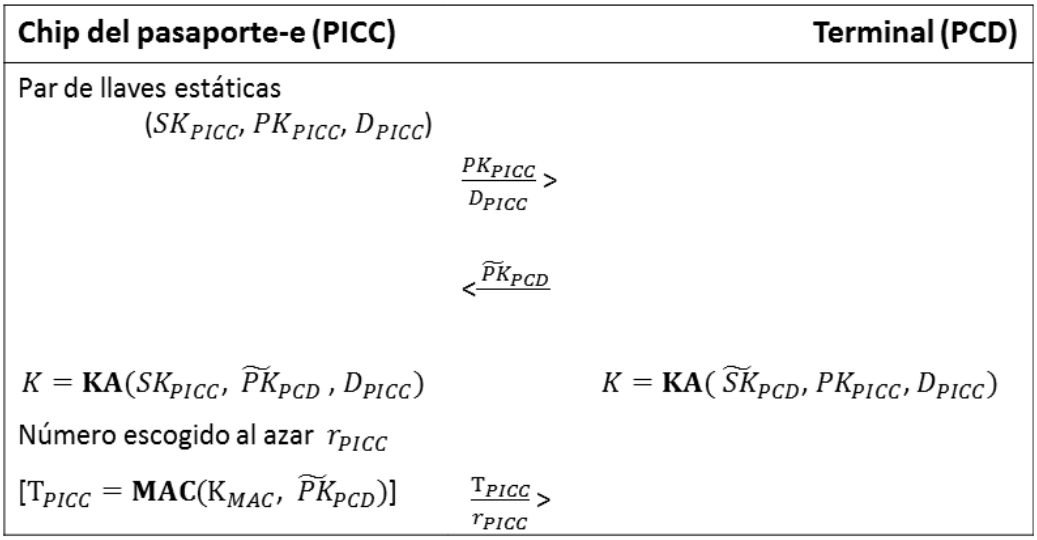


Figura 6. Autenticación del *chip* en la versión 2 del EAC-UE.

Los siguientes pasos son ejecutados por el terminal y el *chip* para la ejecución de la autenticación del *chip* del EAC-UE versión 2:

1. El *chip* le envía al terminal la llave estática pública (PK_{PICC}) y los parámetros DH (D_{PICC}) al terminal.
2. El terminal envía la llave pública ($\widetilde{PK_{PCD}}$) al *chip*.
3. El *chip* comprime la llave recibida y la compara con el comprimido obtenido durante la autenticación del terminal.
4. El *chip* y el terminal calculan:
    a. El secreto compartido (*K*) con las funciones para los acuerdos entre las llaves (*KA*). *K*= **KA** ($\widetilde{SK_{PICC}}$, $\widetilde{PK_{PCD}}$, D_{PICC}))= **KA** ($\widetilde{SK_{PCD}}$,PK_{PICC},D_{PICC}).
5. El *chip:*
    a. Escoge un valor aleatorio r_{PICC} y deriva las llaves de sección K_{MAC}=KDF_{MAC}(*K*, r_{PICC}), K_{Enc}=KDF_{Enc}(*K*, r_{PICC}) utilizando el secreto compartido obtenido y r_{PICC}.
    b. Calcula un *token*[2] de autenticación (T_{PICC}) cifrando la llave pública con la K_{MAC}.
    T_{PICC} = *MAC*(K_{MAC}, $\widetilde{PK_{PCD}}$).

[2] Es utilizado en el EAC-UE versión 2. Representa un arreglo de bytes donde el *chip* almacena el cifrado de la llave pública efímera recibida del terminal, usando la llave de sesión K_{MAC} generada a partir del secreto compartido.

c. Envía r_{PICC} y T_{PICC} al terminal.

6. El terminal:
   a. Deriva las llaves de sección K_{MAC}=KDF_{MAC}(*K*, r_{PICC}), K_{Enc}=KDF_{Enc}(*K*, r_{PICC}) utilizando el secreto compartido obtenido y r_{PICC}.
   b. Verifica la autenticidad del *token* T_{PICC}.
   c. Las llaves de sección (K_{MAC}, K_{Enc}) utilizando la función de derivación de llaves (*KDF*) a partir de *K*.
   d. El terminal comprime al llave pública (**Comp** ($\widetilde{PK_{PCD}}$)) para la posterior ejecución de la autenticación del terminal.

Infraestructura de Clave Pública

La infraestructura de clave pública (PKI, por sus siglas en inglés) definida para el EAC-UE está conformada por la Autoridad Verificadora de Certificación del País (CVCA, por sus siglas en inglés) que es el emisor de los certificados de los verificadores de documentos. Representa el único punto de confianza del estado emisor. La llave pública del CVCA para verificar el certificado del Verificador de Documento (DV, por sus siglas en inglés) es almacenada en la zona segura del ICC. El Verificador de Documento es el emisor de los certificados de los sistemas de inspección y los sistemas de inspección (IS, por sus siglas en inglés) son los que acceden al ICC, como se muestra en la Figura 7 (3).

Un sistema de inspección será autorizado a acceder a los datos confidenciales almacenados en el *chip* a través de la cadena de certificados, empezando por el certificado del DV, emitido por la CVCA del emisor del documento y terminando con el certificado del IS. Como consecuencia de ello, un sistema de inspección contiene una cadena de certificados para cada estado emisor.

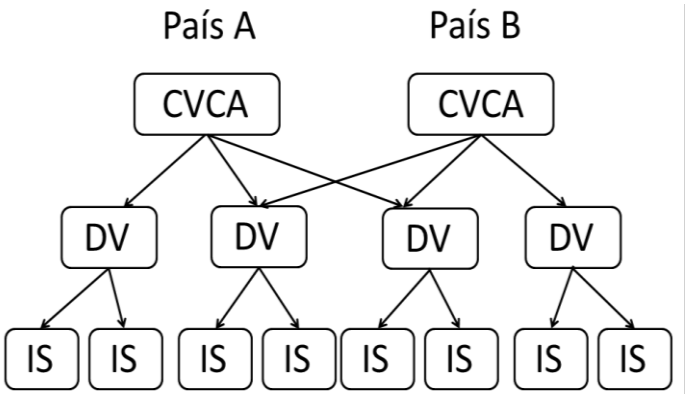


Figura 7. PKI del EAC de la Unión Europea (3).

1.4. Solución de Control Migratorio para los sistemas de inspección de la República Bolivariana de Venezuela

El proyecto técnico Puntos de Control Migratorio de la República Bolivariana de Venezuela, está enmarcado en el Contrato de Desarrollo de Solución Integral para la Transformación y Modernización del Sistema de Identificación, Migración y Extranjería.

El módulo de Control Migratorio permite llevar el control de los movimientos migratorios en los diferentes puntos de entrada y salida del país, integrando la lectura del pasaporte con equipamiento especializado. Realiza el chequeo de las prohibiciones, antecedentes u otros problemas que pueda presentar el ciudadano durante el trámite de migración. Almacena los datos personales de los ciudadanos, incluyendo su fotografía, con posibilidades de ajuste de imagen. Realiza el completamiento de los datos pertenecientes a la tarjeta de migración. Mantiene el control de los vuelos en coordinación con la torre de control, permitiendo incluir vuelos de última hora.

Dicho proyecto está orientado a la modernización de la infraestructura tecnológica integral en los puntos de control migratorio, específicamente en el montaje de seis estaciones de auto chequeo en el Instituto Autónomo Aeropuerto Internacional Simón Bolívar de Maiquetía, además de veintiún lectores de documentos electrónicos. Para lo cual se adquirió el lector Regula, este cuenta en su SDK con funcionalidades para verificar las medidas de seguridad de los pasaportes, incluyendo el EAC.

1.4.1. Lector de documentos "Regula"

El lector de documentos Regula posee un *Software Development Kit* (SDK) con funcionalidades para verificar las medidas de seguridad de los pasaportes, dentro de las cuales se encuentra el EAC. Para ejecutar el EAC, el SDK cuenta con varios comandos, para realizar la autenticación del *chip* y del terminal.

La autenticación del *chip* es llevada a cabo de forma automática después de la configuración de la llave de acceso del BAC y la lectura de los datos del grupo de dato 14. El comando *RFID_Notification_ChipAuthentication* informa a la aplicación del usuario del inicio y del fin del procedimiento de la autenticación del *chip*.

En el caso de la autenticación del terminal, se requiere de una cadena de certificados, la cual comienza por el certificado emitido por el emisor del documento. El comando *RFID_Command_Get_EAC_PKD*, obtiene a través de un número anidado de directorios donde estarán almacenados los recursos para la autenticación del terminal. El comando *RFID_Notification_TerminalAuthentication* es el encargado de notificar a la aplicación del usuario sobre el inicio y fin del procedimiento de autenticación del terminal.

Si la autenticación del terminal es efectuada satisfactoriamente se permite el acceso a la lectura de los datos sensibles del pasaporte y son mostrados descifrados (10). Esta gestión se debe realizar de forma local, tornando lento y complicado el proceso para todos los puntos de inspección donde será desplegado el sistema.

1.5. Criptografía

Según el diccionario de la Real Academia, la palabra criptografía proviene del griego cripto, que significa oculto, y grafía, que significa escritura, y su definición es: "Arte de escribir con llave secreta o de un modo enigmático" (11).

La criptografía hace años que dejó de ser un arte para convertirse en una técnica, o más bien un conglomerado de técnicas, que tratan sobre la protección de la información. Entre las disciplinas que engloba se destacan la Teoría de la Información, la Teoría de Números y la Complejidad Algorítmica. La criptografía se representa a través de un modelo al cual se le llama criptosistema (12).

Cuando se está hablando de criptografía se utilizan términos tales como:

- Encriptar: Se refiere a la acción de aplicar técnicas criptográficas con el objetivo de esconder un mensaje.
- Desencriptar: Es la acción que permite hacer entendible un mensaje cifrado.
- Texto plano o texto en claro: Es el mensaje legible inicial que se quiere transmitir.
- Algoritmo de cifrado: Es el conjunto de pasos que se realizan para cifrar el texto plano.
- Llave o clave: Es una secuencia de caracteres, que puede contener letras, dígitos y símbolos, que es convertida en un número, utilizada por los algoritmos de cifrado para codificar y decodificar mensajes.

1.5.1. Algoritmos simétricos

Son los criptosistemas más sencillos. Se trata de algoritmos que trabajan con una única clave de doble función (encriptado y desencriptado). Dentro de los sistemas simétricos se distinguen dos tipos de algoritmos: los descifrado de bloque, que dividen el texto en claro en bloques de tamaño prefijado (por ejemplo 64 bits) y los cifran bloque a bloque; y los de cifrado de flujo, que cifran bit a bit o byte a byte. Los algoritmos simétricos utilizados para la ejecución del EAC son:

Triple-DES (Triple - *Data Encryption Standard*):

El algoritmo criptográfico DES es utilizado para el cifrado de bloques de hasta 128 bits el cual utiliza mecanismos de las redes de *Feistel* para lo obtención de una cadena lo más aleatoria posible a partir de un mensaje original. Dada la capacidad de cómputo actual y la relativa facilidad que supone romper el algoritmo DES, se desarrolló un sistema de triple aplicación al algoritmo DES, con tres claves diferentes para aplicar sucesivamente (en realidad se usa una clave externa dividida para aplicación intermedia dado que DES matemáticamente no es grupo, y su aplicación repetida ocasionaría un aumento efectivo de tamaño). Mediante este sistema se obtiene un cifrado de 192 bits (168 efectivos y 24 de paridad) con tres claves que resulta mucho más complejo de vulnerar (13). Este algoritmo criptográfico será utilizado en el cifrado del *token* de comprobación durante la autenticación del terminal en EAC-UE versión 2 y en la ejecución del EAC-Singapur.

AES (*Advanced Encryption Standard*):

Se trata de un algoritmo simétrico que puede funcionar mediante el cifrado en bloques de longitud variable o en flujo y que se sirve de claves de longitud variable (128, 192 o 256 bits) (13). Este algoritmo criptográfico se utiliza en el cifrado del *token* de comprobación para la autenticación del terminal en EAC-UE versión 2. El algoritmo puede resumirse en la Figura 8.

.A	Toma el texto en claro y lo cifra en bloques para obtener el punto intermedio llamado "estado", que se representa como una matriz de bytes de cuatro filas.
B	A partir del "estado" se realizan las siguientes operaciones en forma de bucle durante un número determinado de iteraciones.
b1	Sustitución de bytes no lineal, operando independientemente sobre cada uno de los bytes del "estado".
b2	Desplazamiento de las filas del "estado" cíclicamente con offsets diferentes.
b3	Mezcla de columnas, que se realiza multiplicando las columnas del "estado" por un polinomio fijo *c(x)*.
b4	Adición de la clave de vuelta, en la que se aplica al "estado" por medio de un simple XOR. La clave de cada vuelta se deriva de la clave de cifrado mediante el esquema de clave.
C	El esquema de clave consiste en dos operaciones, expansión de clave y selección de clave de vuelta de cifrado, y el proceso de cifrado consta de tres pasos: una adición inicial de la clave de vuelta, n-1 vuelta de cifrado y una vuelta final.

Figura 8. Resumen del algoritmo AES (13).

1.5.2. Algoritmos hash

Los criptosistemas de resumen, conocidos familiarmente como funciones o algoritmos *hash*, constituyen un tipo especial de criptosistemas. Muchos manuales de criptografía los sitúan como un subgrupo de los criptosistemas simétricos. En los algoritmos *hash* no existe el concepto de clave criptográfica, ni tampoco el concepto de descifrado; el concepto de algoritmo criptográfico se mantiene (13).

Un algoritmo tipo *hash* acepta como entrada un mensaje de longitud arbitraria, y tras efectuar sobre él los cálculos determinados por el algoritmo, devuelve una cadena de caracteres que representa el *hash* del mensaje al que aplicamos el algoritmo. Este *hash* no puede ser denominado criptograma dado que no es posible el proceso de descifrado que nos devolvería el mensaje original (13). El algoritmo *hash* utilizado para la ejecución del EAC-UE es:

SHA-1 (*Secure Hash Algorithm – 1*):

SHA-1 fue ideado por el NIST en 1994 como ampliación al algoritmo SHA. Se trata de una función criptográfica de tipo *hash* que acepta una entrada de 2^64 *bits* como máximo (2048 *Terabytes*) y devuelve como salida una cadena de 160 *bits*. SHA-1 es ligeramente más lento que MD5 (otro de los algoritmos *hash*), pero también es computacionalmente más complejo y su salida es de mayor longitud, por lo que se considera de forma global más seguro (13). Esta función resumen será utilizada para realizar la firma y comprobación de los *token* en la autenticación del terminal en el EAC-UE.

1.5.3. Algoritmos asimétricos

Se trata de criptosistemas más modernos y complejos que los simétricos, a la vez que mucho más seguros. Se fundamentan en la existencia de un par de claves complementarias que denominamos clave pública y clave privada respectivamente. Un criptograma generado por una de las claves puede ser descifrado únicamente por la otra clave, y viceversa (13). Los algoritmos asimétricos utilizados para la realización del sistema son:

RSA (*Rivest - Shamir – Adleman*):

El algoritmo RSA nació en 1978 de la mano de Ron Rivest, Adi Shamir y Leonard Adleman. Se trata de un algoritmo de cifrado asimétrico basado en el problema de la factorización entera, y aunque la descripción de este algoritmo fue propuesta en 1973 por Clifford Cocks, fue secreta hasta 1978 cuando se publicó RSA. Aunque el algoritmo fue patentado, la patente expiró en el año 2000 y actualmente se trata de un algoritmo libre. RSA es el algoritmo asimétrico de cifrado más usado, tanto en conexiones de Internet y protocolos seguros, como en cifrado de datos (13). Este algoritmo criptográfico será utilizado para la ejecución de la autenticación del terminal en el EAC-UE y para la obtención de la llave simétrica en EAC-Singapur.

DH (Diffie-Hellman):

Este algoritmo se emplea fundamentalmente para acordar una clave común entre dos interlocutores, a través de un canal de comunicación inseguro. La ventaja de este sistema es que no son necesarias llaves públicas en el sentido estricto, sino una información compartida por los dos comunicantes (14). Este algoritmo es utilizado en la ejecución del EAC-UE, puesto que se establece como mecanismo para acordar claves simétricas empleadas para el cifrado de una sesión entre el *chip* y el terminal.

ECDSA (Elliptic Curve Digital Signature Algorithm):

Es una modificación del algoritmo DSA que emplea operaciones sobre puntos de curvas elípticas en lugar de las exponenciaciones que usa DSA (problema del logaritmo discreto). La principal ventaja de este esquema es que requiere números de tamaños menores para brindar la misma seguridad que DSA o RSA (14). Este algoritmo criptográfico es utilizado en la ejecución de la autenticación del terminal del EAC-UE durante la firma y comprobación del *token* de seguridad.

ECDH (Elliptic Curve Diffie - Hellman):

Este algoritmo está basado en la utilización de curvas elípticas junto con las operaciones de Diffie-Hellman, posibilitando generar claves más cortas dado el aumento significativo de la complejidad de las operaciones que realiza. Este algoritmo es utilizado en la ejecución del EAC-UE en la creación de llaves de sección a partir de un secreto compartido.

1.6. Soluciones existentes sobre el EAC

Después de haber enunciado los conceptos fundamentales para la presente investigación, es necesario abordar en cuanto a las soluciones existentes. A continuación se mencionarán diferentes funcionalidades y características de determinados productos que permitan la ejecución de esta medida de seguridad.

1.6.1. Golden Reader

Golden Reader Tool (TRB) es una aplicación para la lectura de documentos electrónicos de identificación. Fue desarrollado por la BSI[3] y se utiliza para verificar la correcta aplicación de las directrices y especificaciones técnicas durante la emisión de documentos de identidad electrónicos y para probar la interoperabilidad de los documentos de identidad de varias naciones. La base para la interoperabilidad de los pasaportes electrónicos requiere soluciones estandarizadas, que cumplan con las necesidades de alta seguridad para cualquier país. El *Golden Reader Tool* es compatible con los siguientes mecanismos de seguridad de la ICAO: autenticación pasiva, control de acceso básico, autenticación activa, control de acceso extendido, autenticación *chip* y autenticación del terminal (15).

[3]BSI: Oficina Federal Alemana para la Seguridad de la Información.

Es por ello que se reconoce el sistema *Golden Reader* como uno de los sistemas más potentes dedicados a la realización de verificaciones de los mecanismos de seguridad de los pasaportes-e pero funciona con licencia privativa e implica costos su adquisición, por lo que se decide no utilizarlo.

1.6.2. D SCAN Master

El *D SCAN* es un lector utilizado para verificar la autenticidad de tarjetas identificadoras, pasaportes electrónicos y otros documentos. Este lector tiene la habilidad de implementar todos los protocolos aprobados de la OACI incluyendo autenticación pasiva, autenticación activa, control de acceso básico, y control de acceso extendido (16). Con el desarrollo del presente trabajo se pretende independizar el proceso de inspección de las medidas de seguridad de los pasaportes-e, de los SDK de los lectores por lo que se decide no utilizar esta solución.

1.6.3. JMRTD

JMRTD es una implementación de código abierto en Java del MRTD[4] del *chip*, que cumple con los estándares de la OACI. JMRTD provee ambos lados de la aplicación de la tarjeta, el *applet* del pasaporte y la API[5] del lado del servidor para el acceso al pasaporte-e.

La API del lado del servidor puede ser usada en diferentes escenarios. En los sistemas de inspección debe posibilitar leer, decodificar y validar la información en el *chip* necesaria para el acceso a los datos. En la inscripción/personalización del sistema el API permite codificar la información a través del cumplimiento de las normas pertinentes. Además JMRTD fue desarrollado inicialmente para probar la conformidad y la seguridad de las implementaciones de pasaportes electrónicos (17).

Esta implementación no permite la gestión de los sistemas de inspección, ni la gestión de los certificados, además solo puede ejecutar el EAC-UE versión 1 por lo que se decide no utilizar.

[4] Documentos de viaje de lectura mecánica.

[5] API (Programación de Aplicaciones de Interfaz): es un lenguaje y un formato de mensaje usado por un programa de aplicación para comunicarse con el sistema operativo o algún otro programa de control.

1.7. Tecnologías, Lenguajes, Herramientas y Metodologías

Luego de analizar la panorámica referente al estado del arte de las soluciones de Control de Acceso Extendido existentes, es importante abordar sobre las principales tecnologías, lenguajes, herramientas que se utilizarán en la realización de este sistema y la metodología que guiará su proceso de desarrollo. Como el sistema que se implementará será un módulo de DocSec, se deberán utilizar las mismas tecnologías, lenguajes y herramientas que fueron utilizadas en la realización de este sistema, aunque podrían incluirse nuevas según la necesidad que exista para poder darle solución al problema existente.

1.7.1. Tecnologías

1.7.1.1. Plataforma de desarrollo .NET

.NET es un conjunto de tecnologías en las que *Microsoft* ha estado trabajando durante los últimos años con el objetivo de obtener una plataforma sencilla y potente para distribuir el *software* que conecta información, usuarios, sistemas y dispositivos. Creada para lograr un desarrollo de *software* con especial énfasis en la rápida creación de aplicaciones y la independencia de lenguaje, busca una mejor comunicación y distribución de las mismas (18).

Figura 9. Estructura interna del entorno de ejecución en lenguaje común.

.NET soporta múltiples lenguajes de programación y aunque cada lenguaje tiene sus características propias, es posible desarrollar cualquier tipo de aplicación con cualquiera de ellos. El Lenguaje Común de Ejecución (CRL, por sus siglas en inglés) se encarga de gestionar la ejecución de las aplicaciones .NET. Además de la integración de lenguajes, también se encarga del cumplimiento de las normas de seguridad y la administración de la memoria, los procesos y los subprocesos entre otros. Posee un compilador JIT

(*Just-In-Time*), el cual genera el código máquina real que se ejecuta en la plataforma que tenga la computadora. La compilación JIT la realiza el CLR a medida que se invocan los métodos en el programa y, el código ejecutable obtenido, se almacena en la memoria caché de la computadora, siendo recompilado sólo cuando se produce algún cambio en el código fuente (19).

1.7.1.2. Windows Communication Foundation

Windows Communication Foundation (WCF) es un marco de trabajo para la creación de aplicaciones orientadas a servicios. Con WCF, es posible enviar datos como mensajes asincrónicos de un extremo de servicio a otro. Un extremo de servicio puede formar parte de un servicio disponible continuamente hospedado por IIS[6], o puede ser un servicio hospedado en una aplicación .NET. Un extremo puede ser un cliente de un servicio que solicita datos de otro extremo de servicio. Los mensajes pueden ser tan simples como un caracter o una palabra que se envía como XML[7], o tan complejos como una secuencia de datos binarios. WCF se ha diseñado para ofrecer un enfoque manejable para la creación de servicios web y clientes de servicios web. WCF es orientado a servicios por lo que permite crear aplicaciones orientadas a servicios. SOA[8], la arquitectura orientada a servicios es usada en servicios web para enviar y recibir datos. Los servicios tienen la ventaja de estar débilmente acoplados entre una aplicación y otra en lugar de incluidos en el código. Una relación de acoplamiento débil implica que cualquier cliente creado en cualquier plataforma puede conectar con cualquier servicio siempre y cuando se cumplan los contratos esenciales. La arquitectura de WCF tiene varios puntos de extensibilidad. Si se necesita una función adicional, existen una serie de puntos de entrada que le permiten personalizar el comportamiento de un servicio (20).

1.7.1.3. PKCS # 12: Estándar de Intercambio personal:

El estándar PKCS #12 describe una sintaxis para intercambiar información personal, incluyendo claves privadas, certificados y secretos de diversos tipos. Los objetos PKCS #12 permiten mover información

[6]Internet Information Services: Estos servicios proporcionan las herramientas y funciones necesarias para administrar de forma sencilla un servidor web seguro.

[7]eXtensible Markup Language (Lenguaje de Marcas Extensible): Es una tecnología que permite la compatibilidad entre sistemas para compartir la información de una manera segura, fiable y fácil.

[8]Service Oriented Architecture (Arquitectura orientada a servicios de cliente): Es un concepto de arquitectura de *software* que define la utilización de servicios para dar soporte a los requisitos del negocio.

personal de manera segura y con garantías de integridad entre diferentes sistemas. El estándar define diferentes modelos de uso a la hora de mantener la seguridad y la integridad. Este estándar para mantener la privacidad cuanta con dos modelos. El modelo de privacidad basado en clave pública, encargado de que la información se cifre en la plataforma origen utilizando una clave pública confiable. Y el modelo de privacidad basado en una contraseña en el que la información se cifra con una clave simétrica derivada de un nombre de usuario y la contraseña proporcionada (21).

Para mantener la integridad de los datos también se dispone de dos modelos. El modelo de integridad basado en clave pública en el que la integridad se garantiza mediante una firma digital de los datos, que se genera con la clave privada de la plataforma de origen. En la plataforma de destino se utiliza la correspondiente clave pública para verificar la firma. Y el modelo de integridad basado en contraseñas encargado de garantizar la integridad mediante un código de autenticación de mensajes (MAC, por sus siglas en inglés) derivado de la contraseña secreta de integridad. Cuando se usa en conjunto con el modelo de privacidad basado en contraseñas, ambas contraseñas no tienen por qué ser iguales (21).

El PKCS #12 permite cualquier combinación de modelos de integridad y privacidad, aunque para mejorar la seguridad se deben evitar algunos de ellos, por ejemplo: si se utiliza el modelo de privacidad basado en contraseñas, es deseable disponer de seguridad física a la hora de transportar el contenedor. Como norma general, los modelos basados en clave pública son más seguros que los basados en contraseñas desde el punto de vista de la seguridad. Sin embargo en muchas ocasiones no es posible utilizarlos ya que no se sabe cuál es la plataforma de destino y por lo tanto no se conoce su clave pública.

1.7.1.4. LDAP

Lightweight Directory Access Protocol (LDAP, por sus siglas en inglés), en español Protocolo Ligero de Acceso a Directorios es un protocolo de tipo cliente-servidor para acceder a un servicio de directorio. LDAP funciona sobre TCP/IP u otros servicios de transferencia orientados a conexión y está basado en el estándar X500 (22). Este surge como alternativa al DAP[9].

Directorio LDAP

[9] **DAP** (Directory Access Protocol): Es un protocolo a nivel de aplicación, por lo que, tanto el cliente como el servidor debían implementar completamente la torre de protocolos OSI.

Capítulo 1: Fundamentación Teórica

Un directorio LDAP es una base de datos optimizada para la lectura y búsqueda de información que es almacenada de manera jerárquica. Los directorios soportan opciones avanzadas de filtrado. Generalmente no soportan transacciones complejas que sí ofrecen los sistemas de gestión de bases de datos diseñadas para procesar un gran volumen de actualizaciones. Los cambios en la información almacenada en un directorio suelen ser del tipo "o todo o nada", es decir, cambios de las ramas del árbol de directorio de información (DIT5, por sus siglas en inglés) completamente, pero, aunque no estén optimizados para ello, los directorios pueden permitir cambios muy específicos. Los directorios están preparados para dar una respuesta rápida a un gran volumen de búsquedas. Disponen de mecanismos de replicación de la información en varios servidores para incrementar la disponibilidad y fiabilidad del servicio mientras se reduce el tiempo de respuesta (22).

Un directorio es una base de datos, pero que en general contiene información más descriptiva y basada en atributos de usuarios y recursos de red. Permite hacer una preselección de las políticas mediante la selección de las ramas en función de la información de la red antes de hacer una búsqueda. Ofrece un control dinámico y coordinado de los elementos de la red ya que las decisiones se toman de manera automática basándose en reglas, peticiones de usuarios o de servicios. Esta gestión dinámica de los elementos de la red representa un cambio cualitativo, desde el punto de vista empresarial, ya que permite la gestión de los recursos de la red y de los servicios de manera más eficiente. Ver Figura 10.

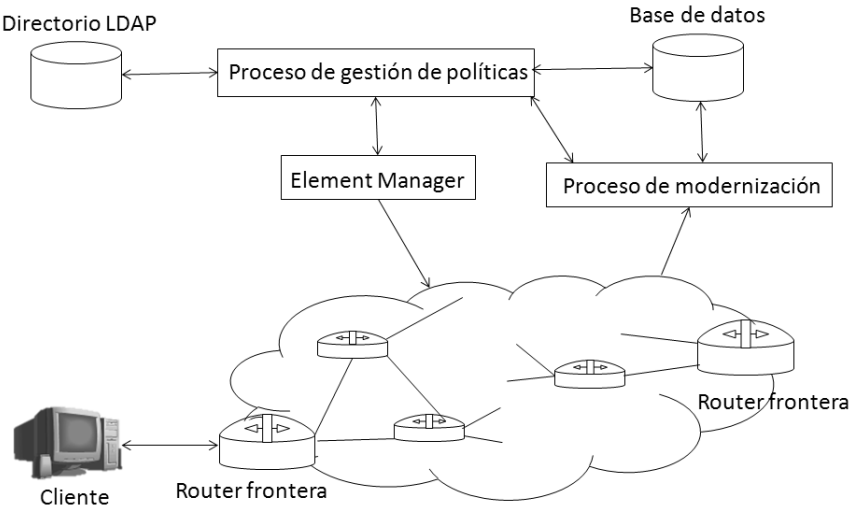


Figura 10. Esquema del directorio LDAP para la gestión de recursos (22).

Open LDAP:

Open LDAP es una aplicación de servidor LDAP, que proporciona la funcionalidad de Servicios de Directorio, tales servicios las incluyen para gestionar identidades y las relaciones entre los ordenadores, usuarios y grupos de ordenadores o usuarios que participan en la red, y proporcionan una forma consistente de describir, localizar y gestionar esos recursos. Ver Figura 11.

El servidor open LDAP posibilita:

- Implementación del protocolo LDAPv3 que está definido para que permita el uso de IPv4 e IPv6.
- Autentificación y nivel de seguridad que permite servicios de autentificación mediante capa de seguridad y autenticación simple (SASL, por sus siglas en inglés). El acceso al directorio ha de ser restringido ya que contiene información de usuarios (nombres de usuarios y contraseñas), información de las políticas y parámetros. Por esta razón sólo el administrador del directorio tiene acceso a la información.
- Control de acceso: OpenLDAP permite definir una serie de filtros para el control de acceso a diferentes DITs, entradas, o atributos de estas entradas.
- Es totalmente gratuito bajo licencia *Open Source*.
- El servidor OpenLDAP hace una implementación *Open Source* del protocolo LDAP y requiere un motor para almacenar y hacer búsquedas de la información (22).

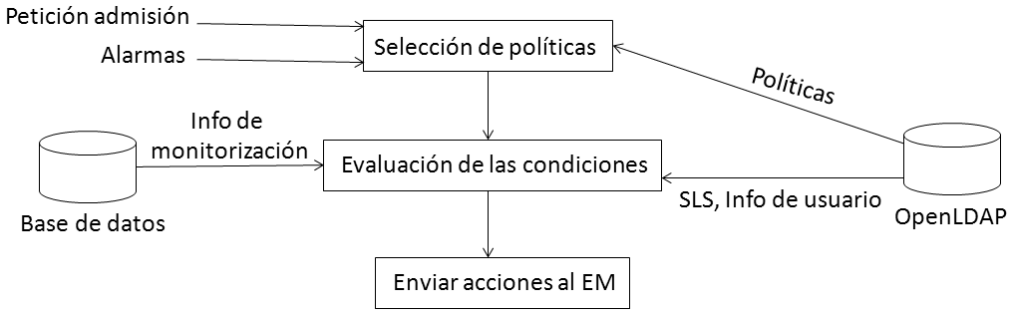


Figura 11. Esquema del sistema Open LDAP para la gestión de políticas (22).

1.7.1.5. Librería BouncyCastle

BouncyCastle (BC) es una implementación de algoritmos criptográficos. Está organizada de forma que sus interfaces de programación de aplicaciones sean adaptables para su uso bajo cualquier entorno de infraestructura adicional.

Un requisito de diseño inicial de *BouncyCastle* es que existieran versiones de la librería para el entorno Java por lo que existen 2 juegos de librerías. La librería compatible JCE se basa en API de bajo nivel, de modo que el código fuente es un ejemplo de implementación de problemas criptográficos comunes con dichas API. Estas están optimizadas para gestionar eficientemente los algoritmos criptográficos, de forma que se puedan usar en entornos de bajos recursos (JavaME) o no es posible usar las librerías JCE (por ejemplo en un *applets*) (23).

Dentro de las funcionalidades que posee la librería *BouncyCastle* para C# podemos citar la gestión de PKCS#12, la creación de los certificados digitales X.509 y las CRL definidas para estos, así como operaciones criptográficas con algoritmos simétricos(AES, TDES), asimétricos (RSA, ECDSA) y el uso de secretos compartidos utilizando DH y ECDH. La última versión estable de esta librería es la 1.7, liberada en abril del 2011.

1.7.1.6. Máquina Virtual de Java

La Máquina Virtual Java (JVM, por sus siglas en inglés) es el entorno en el que se ejecutan los programas Java, su misión principal es la de garantizar la portabilidad de las aplicaciones Java. Define esencialmente un ordenador abstracto y especifica las instrucciones que este ordenador puede ejecutar. Las tareas principales de la JVM son las siguientes:

- Reservar espacio en memoria para los objetos creados.
- Liberar la memoria no usada.
- Asignar variables a registros y pilas.
- Llamar al sistema huésped para ciertas funciones, como los accesos a los dispositivos.
- Vigilar el cumplimiento de las normas de seguridad de las aplicaciones Java (24).

1.7.2. Metodologías

Rumbaugh ha planteado que: “Una metodología de ingeniería de *software* es un proceso para la producción organizada del *software*, empleando para ello una colección de técnicas predefinidas y convencionales en las notaciones. Una metodología se presenta normalmente como una serie de pasos, con técnicas y notaciones asociadas a cada uno de ellos. Los pasos de la producción del *software* se organizan normalmente en un ciclo de vida consistente en varias fases de desarrollo” (25).

Una metodología no es más que un conjunto de procedimientos, técnicas, herramientas y un soporte documental que ayuda en la realización de un *software*. Estas tienen como objetivo primordial lograr que los productos finales sean eficientes y que cumplan con los requeremientos planteados por el usuario. Como los requisitos son tan variables y desiguales, han surgido varias metodologías de desarrollo, las cuales han sido clasificadas en dos grandes grupos de acuerdo con sus características y los objetivos que persiguen: robustas y ágiles.

1.7.2.1. Metodologías robustas

Las metodologías robustas o pesadas están orientadas al control de los procesos, detallan rigurosamente tanto las tareas y actividades del equipo de desarrollo como los artefactos de *software*. Establecen las actividades a desarrollar, herramientas a utilizar y notaciones que se usarán. Son las más tradicionales y altamente recomendadas para proyectos grandes y complejos, donde se requiere de una gran organización.

Dentro de las metodologías pesadas se encuentra el Proceso Unificado de Desarrollo (RUP, por sus siglas en inglés), que por ser la más completa constituye un ejemplo académico cuando se quieren estudiar estas metodologías. RUP cuenta con cuatro fases de trabajo (Inicio, Elaboración, Construcción y Despliegue) divididas en flujos de trabajo, esta estructura hace que el desarrollo con RUP sea dirigido por casos de uso, centrado en la arquitectura e iterativo - incremental (26).

Estas metodologías demandan de un numeroso equipo de proyecto y generan una gran cantidad de documentación, que en ocasiones resulta inconveniente para proyectos sencillos y con poco tiempo para el desarrollo, como es el caso que se plantea. Por estas razones se decide desechar este tipo de metodología y analizar las ágiles.

1.7.2.2. Metodologías ágiles

El Manifiesto Ágil valora al individuo y las interacciones del equipo de desarrollo sobre el proceso y las herramientas. Las personas son el principal factor de éxito de un proyecto *software*. Es más importante construir un buen equipo que construir el entorno. Muchas veces se comete el error de construir primero el entorno y esperar que el equipo se adapte automáticamente. Es mejor crear el equipo y que éste configure su propio entorno de desarrollo en base a sus necesidades (27).

En el Manifiesto se plantea que el desarrollo de *software* funcional es más importante que conseguir una buena documentación. La regla a seguir es no producir documentos a menos que sean necesarios de forma inmediata para tomar una decisión importante. Estos documentos deben ser cortos y centrarse en lo fundamental. En él se define que debe existir la colaboración entre el cliente y el grupo de desarrollo, más que la negociación de un contrato entre ambos. Se propone que exista una interacción constante entre el cliente y el equipo de desarrollo. Esta colaboración entre ambos será la que marque la marcha del proyecto y asegure su éxito. Además plantea que se debe dar respuesta a los cambios más que seguir estrictamente un plan. La habilidad de responder a los cambios que puedan surgir a lo largo del proyecto (cambios en los requisitos, en la tecnología, en el equipo, etc.) determina también el éxito o fracaso del mismo. Por lo tanto, la planificación no debe ser estricta sino flexible y abierta (27).

Las metodologías ágiles permiten dar una rápida respuesta a cambios en los requisitos a lo largo del proyecto e ir entregando de forma continua y en plazos cortos *software* funcional. El trabajo conjunto entre el cliente y el equipo de desarrollo minimiza los costos frente a cambios y eliminar el trabajo innecesario. Estas metodologías prestan gran atención a la excelencia técnica y al buen diseño. Evita malentendidos de requerimientos entre el cliente y el equipo. Además cada componente del producto final será probado para ver si satisface los requerimientos (27).

Características de las principales metodologías utilizadas:

Programación Extrema (XP):
XP es una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en el desarrollo de *software*, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores, y propiciando un buen clima de trabajo. Se basa en la retroalimentación continua entre el cliente y el equipo de desarrollo, la comunicación fluida entre todos los participantes, la simplicidad en las soluciones implementadas y el coraje para enfrentar los cambios. Está concebida para proyectos con poco tiempo de desarrollo y equipos pequeños, con pocos roles, por lo que la programación se realiza en parejas. Propone que el diseño debe ser sencillo, que funcione con todas las pruebas, sin lógica duplicada y con el menor número de clases y métodos posible (28).

Microsoft Solution Framework *(MSF for Agile Software Development):*
MSF se caracteriza por ser una metodología de planificación adaptable a cambios y orientada a las personas. Su proceso introduce ideas importantes del *software* ágil, admite una estrategia que utiliza

múltiples iteraciones y un enfoque para la construcción de aplicaciones que se basa en escenarios. Además esta metodología incorpora prácticas para el manejo de la calidad del servicio (el rendimiento y la seguridad) y facilita la automatización y la orientación que se necesita para apoyar el equipo de trabajo, incluyendo la gestión de configuración y de proyectos.

La definición, desarrollo y prueba del producto se realizan en pequeñas iteraciones provenientes del proceso incremental del proyecto, reduciéndose así el margen de error en las estimaciones y proporcionándose información rápida acerca de la exactitud de los planes del proyecto.

Esta metodología soporta 17 flujos de trabajo básicos, en los cuales se agrupan diferentes actividades, e incluye además cinco fases para el desarrollo y seguimiento del producto, estas son: Visión, Planeación, Desarrollo, Estabilización e Implementación o Despliegue (29). En la Figura 12 se pueden apreciar las fases de esta metodología.

Figura 12. Fases de MSF.

- **Fase Visión:** En esta fase el equipo identifica la visión y el alcance del proyecto, mediante la elaboración de un documento de visión y alcance. Durante esta fase, los objetivos del proyecto se forman y se crea una declaración de visión que define todo el proyecto. Esta visión compartida ayuda al equipo a trabajar hacia un objetivo común.
- **Fase Planificación**: En esta fase se debe concretar cómo va a estar estructurada la solución para lo cual hay que crear el documento de planificación y diseño de la arquitectura y diseñar las pruebas donde se plantean los diferentes escenarios para probar la validez de los criterios utilizados para el diseño.
- **Fase Desarrollo**: En esta etapa de desarrollo se realiza la codificación del sistema y se realizan las configuraciones necesarias para que la solución funcione, es importante hacer

pruebas continuamente para verificar el comportamiento del producto a lo largo del desarrollo y no únicamente al final del proceso.

- **Fase Estabilización**: En esta fase se selecciona el entorno de prueba piloto y lo que se pretende con esto es identificar las deficiencias con un grupo reducido de usuarios para corregirlas, para que en el futuro no hayan problemas cuando se use la solución, se debe crear un plan de gestión de incidencias, realizar la revisión del documento final de la arquitectura y elaborar el plan de despliegue o implementación.
- **Fase Despliegue** o **Implementación**: En esta etapa final se debe liberar la solución y asegurarse de que esté estable y se pueda utilizar. Además se crea un registro de mejoras y sugerencias, se revisan las guías y manuales y entregar el proyecto final.

Este ciclo se puede llevar a cabo de forma iterativa, de manera que cuando se libere la solución se pueda iniciar nuevamente la metodología para darle más funcionalidad.

MSF mantiene siempre el enfoque al usuario. Es decir, ayudándolo a asegurar que la solución implantada realmente es lo que el usuario necesita, que mejorará el desempeño del usuario, y no una que va a ser desechada y olvidada al momento de su liberación. Permite desarrollar siguiendo una filosofía de reutilización de múltiples componentes, lo cual reduce el tiempo necesario para desarrollar nuevas aplicaciones, garantiza la uniformidad e interoperabilidad entre las mismas, y las hace mucho más flexibles para incorporar los cambios que sean necesarios en el futuro.

Esta metodología establece un equipo de trabajo balanceado, con tareas y objetivos claramente definidos, que permitan no solo desarrollar buenos sistemas, sino también saber en todo momento cuál es el grado real de avance del proyecto, y cuáles son los riesgos que se corren si se decide introducir modificaciones al mismo una vez que el desarrollo ya ha sido iniciado (28).

Argumento de metodología a utilizar:

Para elegir la metodología a utilizar en el proyecto de desarrollo de *software*, se tuvo en cuenta las principales características del mismo:

- Complejidad: Alta.
- Tiempo estimado de desarrollo: 5 meses.

- Cantidad de personal: 2.

Una vez que estén bien definidos estos factores, se realiza un análisis preliminar para identificar cuál se ajusta más al proyecto.

Se decide no utilizar una metodología robusta ya que están diseñadas para grupos de trabajos grandes y son utilizadas generalmente en proyectos que tienen larga duración ya que llevan una documentación exhaustiva de todo el proyecto.

Teniendo en cuenta las principales características del proyecto se escoge la metodología MSF Ágil porque es un marco de trabajo de referencia para construir e implantar sistemas distribuidos basados en herramientas y tecnologías de *Microsoft*, por ejemplo la plataforma .NET. Provee una estructura orientada a facilitar el análisis, diseño e implantación de soluciones tecnológicas efectivas. Este marco permite exponer, revelar y manejar riesgos críticos, determinar los criterios de planeación, y establecer las interdependencias necesarias para una ejecución exitosa de los proyectos. Por la cantidad de personal en el proyecto, se decidió tomar la metodología MSF Ágil ya que los miembros del equipo pueden tomar diferentes roles durante el ciclo de vida del proyecto.

1.7.3. Lenguajes

1.7.3.1. Lenguaje de programación

Un lenguaje de programación puede definirse como una construcción mental del ser humano para expresar programas. Está compuesto por un grupo de reglas gramaticales, símbolos utilizables, términos con sentido único y una regla principal que resume las demás. Para que esta construcción mental sea operable en un computador debe existir otro programa que controle la validez o no de lo escrito (30).

***C Sharp* o C#**

C# (en inglés “C Sharp”) es el lenguaje orientado a objetos diseñado por *Microsoft* para su plataforma .NET.Combina los mejores elementos de múltiples lenguajes de amplia difusión como C++, Java, Visual

Basic o Delphi. C#, como parte de la plataforma .NET, está normalizado por *Ecma International*[10] desde diciembre de 2001 (31).

Aunque es posible escribir código para la plataforma .NET en muchos otros lenguajes, C# es el único que ha sido diseñado específicamente para ser utilizado en ella, por lo que programarla usando C# es mucho más sencillo e intuitivo que hacerlo con cualquiera de los otros lenguajes, ya que C# carece de elementos heredados innecesarios en .NET. Por esta razón, se suele decir que C# es el lenguaje nativo de .NET

Se selecciona este lenguaje dado que es nativo de la plataforma .NET, en la cual se desarrolla el sistema. Además este sistema debe de ser integrado con la aplicación ya desarrollada DocSec y se recomienda usar el mismo lenguaje por cuestiones de estandarización y políticas del proyecto.

Java

Java es un lenguaje orientado a objetos y de alto nivel, fue creado por Sun Microsystems. Recoge los elementos típicos en el resto de los lenguajes de programación, agregando variadas ventajas como son la sencillez a la hora de desarrollar en el mismo y la portabilidad de las aplicaciones creadas. Java ha sido pensado para soportar aplicaciones que serán ejecutadas en diversos entornos. Esto es posible ya que el mismo genera *bytecode*, usándolos como lenguaje intermedio y siendo indiferente de la arquitectura, el código pasa al intérprete de Java el cual realiza las acciones para que la aplicación sea compactible con las diversas plataformas *hardware* y *software* (32).

Se utiliza este lenguaje para realizar el pedido necesario para tener actualizados los certificados que intervienen en el proceso de EAC, ya que la librería *BouncyCastle* de C# no cuenta con todas las funcionalidades necesaria para poder realizar esta operación.

1.7.3.2. Lenguaje de Modelado

Lenguaje Unificado de Modelado 2.1 (UML, por sus siglas en inglés)

[10] **Ecma International**: Tiene como objetivos desarrollar en cooperación con las organizaciones de estándares nacionales, europeos e internacionales, estándares y reportes técnicos para facilitar y estandarizar el uso de las Tecnologías de Información y Comunicación y Dispositivos Electrónicos.

UML es un lenguaje gráfico que especifica, construye, visualiza y documenta las partes o artefactos originados durante un proceso de desarrollo de *software*. Este lenguaje de modelado permite establecer conceptos, artefactos ejecutables y modelar sistemas utilizando conceptos orientados a objetos. Además de permitir encaminar el desarrollo del escalamiento en sistemas complejos de misión crítica, establecer el soporte a la planeación y al control de proyectos y permitir alta reutilización y minimización de costos (33). UML puede ser empleado para dar soporte a una metodología de desarrollo de *software*, pero no especifica en sí qué metodología o proceso utilizar.

1.7.4. Herramientas

1.7.4.1. Entorno de desarrollo

Un Entorno de Desarrollo Integrado (IDE, por sus siglas en inglés), es un programa donde es posible escribir código fuente, compilarlo y ejecutarlo. Permiten desarrollar las aplicaciones de forma mucho más rápida, incorporando en muchos casos librerías con componentes previamente implementados.

Visual Studio 2010 Ultimate

Microsoft Visual Studio no es más que un entorno de desarrollo integrado para *Windows*. Este entorno de desarrollo soporta diferentes lenguajes de programación entre los cuales se pueden mencionar: Visual C++, Visual C#, Visual J#, y Visual Basic. *Microsoft Visual Studio 2010 Ultimate* es un potente paquete de herramientas de administración del ciclo de vida de las aplicaciones. Con este paquete se garantiza la calidad de los resultados, desde el diseño hasta la implementación. *Visual Studio* posee disímiles características, entre las principales se encuentran las siguientes (34):

Visual Studio posee disímiles características, entre las principales se encuentran las siguientes:

- Depuración y Diagnóstico: Presenta *IntelliTrace*, que no es más que una valiosa característica de depuración. Los evaluadores pueden archivar errores enriquecidos y modificables para que los desarrolladores puedan reproducir siempre el error del que se informe y el estado en el que se encontró.
- Herramientas de Prueba: *Visual Studio 2010 Ultimate* incorpora todas las herramientas avanzadas de *Microsoft* para pruebas.

- Arquitectura y Modelado: Los diagramas por capas ayudan a garantizar el cumplimiento de la arquitectura y permiten validar artefactos de código con respecto a los diagramas. Además, admite los cinco diagramas de UML más comunes que conviven junto con su código.

Fundamentación del IDE a utilizar

Se seleccionó el IDE *Visual Studio 2010 Ultimate* para cumplir IDE de desarrollo del *software* al cual se integrará el sistema, el mismo utiliza la plataforma .NET y el lenguaje de programación C#. Es un *software* potente, seguro y sus características hacen más fácil el proceso de desarrollo.

NetBeans IDE 7.2

El IDE *NetBeans* es un entorno de desarrollo integrado disponible para Windows, Mac, Linux y Solaris. El proyecto *NetBeans* consiste en un IDE de código abierto y una plataforma de aplicaciones que permiten a los desarrolladores crear rápidamente aplicaciones web, empresariales, de escritorio y aplicaciones móviles utilizando la plataforma Java, así como JavaFX, PHP, JavaScript y Ajax, Ruby y Ruby on Rails, Groovy y Grails, y C/C++. *NetBeans* IDE 7.2 ofrece un rendimiento significativamente mejorado y la experiencia de codificación, con nuevas capacidades de análisis de código estático en el editor de Java y de exploración del proyecto. Esta versión incluye características notables, como es la integración con el generador de escena para crear visuales de JavaFX, soportando los múltiples *frameworks* de PHP y muchas otras mejoras en Java EE, Maven, C/C++ y la Plataforma *NetBeans* (35).

Fundamentación del IDE a utilizar

Se escogió en *NetBeans* IDE 7.2 para poder generar los pedidos de certificado utilizando el lenguaje de programación Java.

1.7.4.2. Herramienta de modelado

Para poder definir la herramienta de modelado a utilizar para desarrollar el sistema se realizó una comparación entre dos de estas herramientas. En la Tabla 1 se puede apreciar la comparación realizada.

Propiedades	Visual Paradigm	Rational Rose
Observaciones generales.	Apoya el ciclo vital completo del desarrollo, la última notación de	Provee productos de UML para los lenguajes comunes de la industria para especificación,

	UML para modelar la representación visual y la generación del código.	visualización, construcción y documentación de los artefactos de los sistemas de *software*.
Ámbito de utilización.	Análisis y diseño orientados a objetos, construcción, pruebas y despliegue.	Se enmarca dentro del desarrollo de modelado para fines académicos, investigativos y comerciales.
Soporte completo UML.	UML 2.1.	UML 1.3.
Robustez	Herramienta con gran robustez.	No presenta auto guardado. Depende de la acción propia de guardado ejecutada por el usuario.
Actualización.	Posee servicio de actualización.	Posee servicio de actualización.
Ingeniería inversa	Código a modelo, código a diagrama Java, C++, esquemas XML, XML, .NETexe/dll, CORBAIDL.	C++, VB, COM, código ADA, J2EE,Corba / IDL, MIDL.

Tabla 1. Comparación entre herramientas de modelado (36).

Fundamentación de la herramienta de modelado a utilizar

Se decide utilizar Visual Paradigm para UM 8.0 ya que es una herramienta que soporta el ciclo de vida completo del desarrollo de *software*: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. Permite construir diagramas de diversos tipos, código inverso, generar código desde diagramas y generar documentación. Además es una herramienta que presenta versiones con licencias libres, mientras que Rational Rose es una herramienta propietaria.

1.8. Conclusiones

- El estudio de los conceptos relacionados con el proceso de Control de Acceso Extendido y el análisis de los sistemas similares existentes, demostró que existen diferentes tipos de implementación de esta medida de seguridad, y se hace necesario poder realizar la lectura de los datos que se encuentren protegidos.
- El análisis de los sistemas *Golden Reader*, JMRTD y D Scan Master evidenció la necesidad de desarrollar un sistema que permita la ejecución del EAC-UE en su versión 1, del EAC-UE en su

versión 2, del EAC-Singapury además permita la gestión de los certificados y de los sistemas de inspección.

- El análisis de las diferentes metodologías, tecnologías y lenguajes según las necesidades de la solución, determinó la utilización de la metodología MSF Ágil, UML como lenguaje de modelado y la herramienta Visual Paradigm para el modelado. Como tecnología la plataforma .NET y el lenguaje de programación C#, usando el IDE de desarrollo Visual Studio 2010 Ultimate. Además se utilizó para el trabajo con los pedidos de certificados el lenguaje de programación Java y el IDE de desarrollo NetBeans.

Capítulo 2: Propuesta de Solución.

2.1. Introducción

De acuerdo con la metodología MSF para el Desarrollo de *Software* Ágil, guía para el proceso de desarrollo del presente sistema, se deben puntualizar un conjunto de conceptos y requerimientos utilizando un lenguaje entendible por todos los involucrados en el proceso en cuestión. Dicha metodología plantea que se debe adquirir una visión clara de lo que se desea desarrollar y además propone la realización de una planificación que indique al equipo de trabajo hacia la exitosa construcción del sistema, así como una descripción de los escenarios y requisitos del sistema. Además se presenta la arquitectura propuesta en conjunto con la descripción de sus aspectos más importantes. En esta sección se realizan las actividades presentes dentro de los flujos de trabajo que sugiere la metodología para iniciar el desarrollo de un proyecto.

2.2. Fase Visión

En este epígrafe se realiza la documentación que corresponde a la fase Visión. Se elabora y aprueba el documento de alcance, en el que quedan reflejadas las funcionalidades y los servicios que debe ofrecer la solución a implantar. Se crea el equipo de trabajo y se distribuyen las competencias y responsabilidades. Se elabora el plan de trabajo, en el que se marcan fechas y contenidos para esta fase y las siguientes.

2.2.1. Propuesta de solución

El sistema está compuesto por dos servicios de WCF, dos aplicaciones de ventanas para la administración de los servicios y una librería de clases, además se implementará una aplicación de pruebas que simule las operaciones de un chip de pasaporte-e. Contará con dos módulos fundamentales: Ejecución del EAC y Repositorio. El primero realiza las operaciones pertinentes para ejecutar el EAC que tienen implementados los pasaportes-e, mientras que el segundo realiza el proceso de la gestión de los certificados digitales y de los sistemas de inspección.

El proceso comienza cuando llega un viajero a un punto de inspección y entrega su pasaporte-e para que sea verificado por el Sistema de Control Migratorio del Servicio de Administración de Identificación, Migración y Extranjería (SAIME), y este posee la medida de seguridad EAC implementada. El SAIME

intenta acceder a los datos sensibles que se encuentran en el chip y de no poder, solicita la ejecución del EAC a la librería de clases que sirve de interfaz al sistema DocSec (EACIS). Primeramente se identifica el tipo de EAC con el que están protegidos los datos, para luego llevar a cabo los procesos de ejecución de estos.

De ser la versión implementada por Singapur se obtiene la llave simétrica que se encuentra cifrada en el grupo de datos 13 con la llave pública del sistema de inspección autorizado, esta es enviada desde la librería EACIS hasta el servicio del sistema DocSec (Servicio General) donde se encuentra la llave privada necesaria para desencriptarla. Como resultado se obtiene la llave simétrica necesaria para la ejecución del comando *EXTERNAL AUTHENTICATE* el cual posibilita la lectura de los datos protegidos.

De la ejecución del EAC de la Unión Europea se han identificado dos versiones, estas se basan en dos mecanismos fundamentales: la autenticación del *chip* y la autenticación del terminal. Para la ejecución de la autenticación del terminal de ambas versiones es necesario presentarle al *chip* una cadena de certificados: el certificado del sistema de inspección en el cual están los permisos sobre los grupos de datos y el certificado del verificador de documento. Estos se encuentran almacenados en un repositorio LDAP, al cual se accede a través de la interfaz del servicio del sistema DocSec. La principal diferencia entre las versiones de la Unión Europea es el orden de ejecución de los procesos, además, en la versión 2 se incluyen nuevos elementos que incrementan la seguridad.

La versión 2 inicia con la autenticación del terminal, para lo cual el sistema de inspección genera un par de llaves Diffie-Hellman efímeras con los parámetros del dominio obtenidos durante la ejecución del BAC. Le envía al *chip* el comprimido de la llave pública generada, así como un valor auxiliar aleatorio. El *chip* envía un valor aleatorio al terminal para la construcción y firma de un *token* que luego verifica. En la autenticación del *chip*, este le envía su llave pública *Diffie-Hellman* y los parámetros del dominio al terminal, como respuesta, el terminal le entrega la llave pública efímera generada durante su autenticación. Ambos calculan el secreto compartido común y el *chip* escoge un valor aleatorio el cual utiliza para la confección de las llaves de sección con las que genera un *token*. El valor aleatorio y el *token* son enviados al terminal, el primero se utiliza para generar las nuevas llaves de sección del terminal y con estas realizar la comprobación del segundo. Una vez ejecutados estos dos mecanismos de forma satisfactoria, el *chip* permite el acceso a los datos protegidos por el EAC.

Capítulo 2: Propuesta de Solución

La solución debe de ser capaz de gestionar los sistemas de inspección desde los cuales se podrán acceder a los pasaportes-e en la Aplicación de Administración de SG. Para adicionar un sistema se debe conocer previamente la dirección IP de la computadora cliente, su dirección física y se genera un serial que se incluye en la configuración de su servicio. Estos datos son almacenados en el directorio LDAP. Para enviar cualquier información entre el servidor y los sistemas de inspección se genera un *token* que es adicionado a cada uno de los mensajes. Además, la solución realizará la gestión de los certificados digitales necesarios para la ejecución del EAC-UE, así como posibilitará de obtener nuevos pedidos de certificados en los sistemas de inspección. Toda la comunicación entre los servicios del DocSec y los del sistema de inspección (Servicio Sistema Inspección) se encuentra cifrada a través del algoritmo criptográfico RSA utilizando llaves de 1024 bytes. Cuando un sistema de inspección desea enviar algún dato al servicio del DocSec, este obtiene la llave pública y cifra el contenido, solo el destinatario con su llave privada tendrá acceso a dicha información. En la Figura 13 se puede apreciar todos los componentes del sistema.

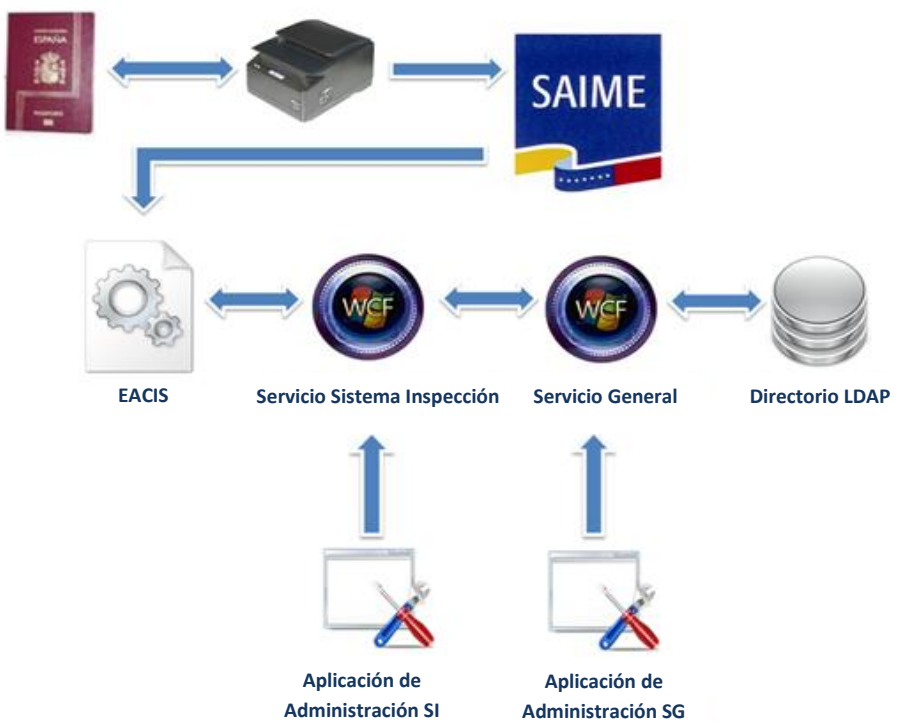


Figura 13. Propuesta de solución (Elaboración Propia).

La Tabla 2 muestra la descripción de los módulos en los que se dividió la implementación del sistema.

Módulos	Descripción
Ejecución del EAC.	Realiza todas las operaciones pertinentes para comprobar que el sistema de inspección tenga permiso para leer los datos sensibles contenidos en el *chip* del pasaporte electrónico.
Repositorio.	Realiza el proceso de gestión de los certificados digitales y las listas de revocación.

Tabla 2. Módulos del sistema.

La Tabla 3 describe quien interactuará con el sistema.

Personas	Descripción
Sistema de control migratorio.	Es el encargado realizar las verificaciones, la lectura del documento y las operaciones necesarias luego de obtener los datos biométricos adicionales. Es el sistema de inspección.
Administrador del sistema.	Es el encargado de gestionar lo relacionado con los certificados así como mantener actualizados los repositorios de certificados.

Tabla 3. Definición de personas.

2.3. Fase Planificación

Luego culminada la fase Visión del sistema, se continúa con la fase Planificación. En esta etapa del proyecto el equipo de trabajo analiza, identifica y además prioriza los requerimientos que describen la solución, conjuntamente con ello se generan algunos artefactos como son: la lista de escenarios, así como la lista de requisitos de calidad del servicio, los cuales son utilizados para especificar los requisitos del *software* que sirven de guía para el proceso de desarrollo. Seguidamente se abordarán los artefactos generados en esta fase de acuerdo a la metodología MSF para el Desarrollo de *Software* Ágil.

2.3.1. Listado de escenarios

Los escenarios definen la interacción entre las personas y el sistema, estos registran los pasos específicos a seguir para lograr el cumplimiento efectivo de una funcionalidad. Una descripción detallada

de los escenarios es de gran ayuda a la hora de implementar las funcionalidades, es por ello que se llevará a cabo en este momento su definición.

Módulo: Ejecución del EAC.

ESC: 1. Ejecutar el EAC-UE versión 1.

ESC: 2. Autenticar *chip* versión 1.

ESC: 3. Autenticar terminal versión 1.

ESC: 4. Ejecutar el EAC-UE versión 2.

ESC: 5. Autenticar terminal versión 2.

ESC: 6. Autenticar *chip* versión 2.

ESC: 7. Ejecutar el EAC-Singapur.

ESC: 8. Adicionar sistemas de inspección.

ESC: 9. Eliminar sistemas de inspección.

ESC: 10. Modificar sistemas de inspección.

ESC: 11. Listar sistemas de inspección.

Módulo: Repositorio.

ESC: 12. Autenticar usuario.

ESC: 13. Adicionar certificados.

ESC: 14. Modificar certificados.

ESC: 15. Eliminar certificados.

ESC: 16. Mostrar certificados.

ESC: 17. Listar certificados.

ESC: 18. Generar cadena de certificados.

ESC: 19. Enviar cadena de certificados.

ESC: 20. Generar pedidos de certificados.

ESC: 21. Generar el par de llaves (Llave pública y llave privada).

Priorizar la lista de los escenarios

Es de gran importancia la priorización de los escenarios porque permite identificar los escenarios más relevantes para darle un tratamiento diferenciado durante su implementación, por ejemplo los escenarios que sean de prioridad alta se implementarán en las primeras iteraciones. Los escenarios que se clasificarán en prioridad baja se simbolizarán con el número 3, los de prioridad media con 4 y los de prioridad alta con 5. Ver en el Anexo 1 el listado de los escenarios con sus correspondientes prioridades.

A continuación se presenta la Figura 14, en ella se pueden apreciar los escenarios por módulos que se definieron para la solución del problema y sus respectivas prioridades, éstas basadas según el peso que tienen los escenarios en el sistema.

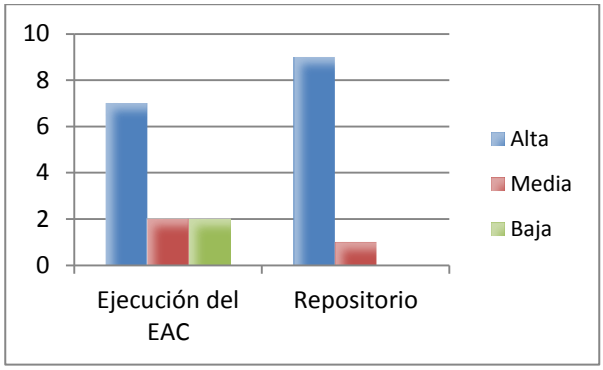


Figura 14. Prioridad de escenarios por módulos.

Plan de iteraciones

Para lograr la implementación en tiempo y forma del sistema de ejecución del EAC en pasaportes-e, es necesario hacer una estimación del tiempo que tomará ejecutar la codificación de cada uno de los

escenarios. En relación con la prioridad que tenga cada escenario se decide cuáles de ellos se desarrollarán en las primeras iteraciones, pues las funcionalidades críticas del sistema deben ser codificadas en las iteraciones más tempranas de su ciclo de vida. El listado de los escenarios con sus correspondientes iteraciones se encuentra en el Anexo 2.

La implementación de los escenarios estará dividida en las siguientes iteraciones:

Iteración 1: Se propone codificar los escenarios que proveen la prioridad alta de los módulos Ejecución del EAC y Repositorio.

Iteración 2: Se codificarán los escenarios que proveen la prioridad media y baja de los módulos Ejecución del EAC y Repositorio.

2.3.2. Requisitos de calidad del servicio

Los requisitos de calidad del servicio, como son conocidos por la metodología MSF, son las especificaciones que el sistema debe cumplir, se clasifican en diferentes tipos como de usabilidad, disponibilidad, eficiencia, entre otros. A continuación se mostrará la lista de los requerimientos que se identificaron para el desarrollo del sistema.

- Usabilidad:

RCS: 1. Será aplicable a cualquier sistema de inspección que precise la verificación del EAC.

RCS: 2. El sistema será distribuido en idioma español.

- Disponibilidad:

RCS: 3. El sistema será accesible las 24 horas del día los 365 días del año.

RCS: 4. No se afectará el funcionamiento del sistema cuando se esté actualizado el repositorio.

- Eficiencia:

RCS: 5. El sistema, en el módulo de verificación del EAC debe emitir respuestas en el orden menor de 30 segundos.

- Diseño e implementación:

RCS: 6. Se utilizará lenguaje de programación C# para la implementación del sistema.

RCS: 7. Será implementado el pedido de los certificados en el lenguaje de programación Java, por no contar el *framework* de .NET con las funcionalidades necesarias para trabajar con los certificados que utilizan los pasaportes-e.

- Interfaces de usuario:

RCS: 8. Se utilizarán menús para acceder a las funcionalidades que brinda el sistema.

RCS: 9. Cuenta con notificaciones que informen al usuario sobre el correcto funcionamiento de los procesos o de los errores que pudiesen ocurrir.

- Seguridad:

RCS: 10. Solo se atenderán en el servidor las peticiones que provengan de un sistema de inspección cuyo *token* de seguridad sea válido.

RCS: 11. El canal de comunicación tiene que estar cifrado para el intercambio de datos utilizando el algoritmo criptográfico RSA.

RCS: 12. Las llaves se guardarán en almacenes de llaves PKCS #12 protegidas por contraseña.

2.4. Descripción de los escenarios

En la Tabla 4 se presenta una descripción de escenario, la descripción completa de los escenarios se encuentra en el Anexo 3.

Ejecutar el EAC-UE versión 1.

Nombre del escenario: Ejecutar el EAC-UE versión 1.		**Identificador:** ESC 1
Objetivo del escenario: Realizar el EAC-UE versión 1 a los pasaportes electrónicos.		
Persona: Sistema de control migratorio.		
Iteración: 1	**Prioridad:** Crítico	**Complejidad:** 3
Precondiciones: Debe haberse identificado que el pasaporte tenga implementado el EAC-UE versión 1.		
Descripción: El escenario comienza cuando se invoca la funcionalidad de obtener los datos sensibles almacenados en el pasaporte. Para realizar el mismo es necesario realizar la autenticación del *chip* y luego la autenticación del terminal, para verificar si ese sistema de inspección tiene los permisos para leer los datos sensibles almacenados en el *chip* del pasaporte.		
Validaciones: Verificar que se haya ejecutado correctamente el EAC-UE versión 1.		

Tabla 4. Descripción del escenario: Ejecutar el EAC-UE versión 1.

2.5. Fase Desarrollo

El primer paso importante en la fase Desarrollo es la especificación de la arquitectura de *software*, la cual a manera de concepto es la organización fundamental de un sistema representada en sus componentes, las relaciones entre ellos, el ambiente y los principios que orientan su diseño y evolución. La misma involucra un conjunto de decisiones significativas acerca de la organización del sistema, selecciona sus elementos estructurales y sus interfaces, así como su comportamiento. También involucra funcionalidad, usabilidad, tolerancia a cambios, rendimiento, reutilización y aspectos estéticos (37).
Además es una vista estructural de alto nivel, que ocurre tempranamente en el ciclo de vida y define los estilos o grupos de estilos adecuados para cumplir con los requerimientos de calidad del servicio.

2.5.1. Especificación de la arquitectura

El primer paso importante en la fase de construcción es la especificación de la Arquitectura de Software, la cual a manera de concepto es la organización fundamental de un sistema representada en sus componentes, las relaciones entre ellos, el ambiente y los principios que orientan su diseño y evolución. La misma involucra un conjunto de decisiones significativas acerca de la organización del sistema, selecciona sus elementos estructurales y sus interfaces, así como su comportamiento. También involucra funcionalidad, usabilidad, tolerancia a cambios, rendimiento, reutilización y aspectos estéticos (38).

Después de realizar un análisis se selecciona para el desarrollo del sistema el estilo arquitectónico basado en componentes ya que este describe un acercamiento al diseño de sistemas como un conjunto de componentes que exponen interfaces bien definidas y que colaboran entre sí para resolver el problema existente. Este estilo tiene como principios que: los componentes son diseñados de forma que puedan ser reutilizados en distintos escenarios aunque algunos son diseñados para una tarea específica, los componentes exponen interfaces que permiten al código usar sus funcionalidades y no revelan detalles internos de los procesos que realizan o de su estado, además los componentes están diseñados para ser lo más independientes posibles de otros componentes, por lo que pueden ser desplegados sin afectar a otros componentes o sistemas (38).

El sistema está compuesto por cinco componentes, de los cuales 3 utilizan el estilo arquitectónico N-capas para su desarrollo. Se decidió hacer uso de este estilo debido a que entre sus principales características se pueden mencionar su gran flexibilidad a la hora de confeccionar las estructuras de las

clases y las relaciones entre ellas, ya que no es un estilo en el que las capas sean de estricto cumplimiento, mayormente se usan 3 capas presentación, controlador, y acceso a datos pero la cantidad de capas varían de acuerdo a la necesidad. El flujo de datos en este estilo está concebido unidireccionalmente, es decir los datos viajarán en una dirección y desde la capa presentación no se puede acceder a la base de datos, y viceversa, lo que asegura la integridad de los datos. Posibilita realizar cambios en una de las capas y las demás no se verán afectadas, permitiendo además trabajar de manera transparente una vez establecidas las conexiones entre las capas. El componente IS_EAC.dll es la librería de clases donde se ejecuta el EAC-UE; IS_Service y DocSec_Service son servicios de WCF encargados de la comunicación segura entre los componentes y la ejecución del EAC-Singapur; DocSec_WindowApp e IS_WindowApp son aplicaciones de administración; la primera además posee funcionalidades para la administración de los certificados digitales y los sistemas de inspección. En la Figura 15 muestra la arquitectura definida para el desarrollo del sistema.

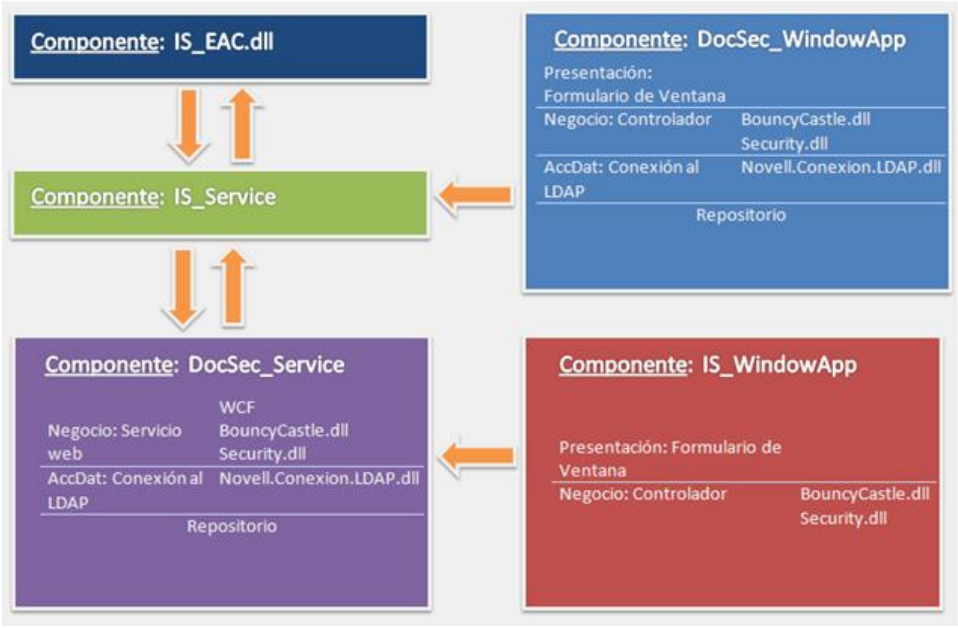


Figura 15. Arquitectura del sistema (Elaboración Propia).

2.6. Diagrama de clases

El diagrama de clases es un diagrama de tipo estático, que describe la estructura de un sistema representando las clases que serán utilizadas, sus atributos y las relaciones que existen entre ellas. Las clases del sistema están organizadas de una forma estructural adecuada según lo necesario en cada una de ellas, lo que las hace fáciles de manipular y permite que se interrelacionen siempre que se necesite (39).

En la Figura 16 se representan las clases del componente DocSec_Service con sus atributos, métodos y relaciones. Los diagramas de clases de los restantes componentes se encuentran en el Anexo 4.

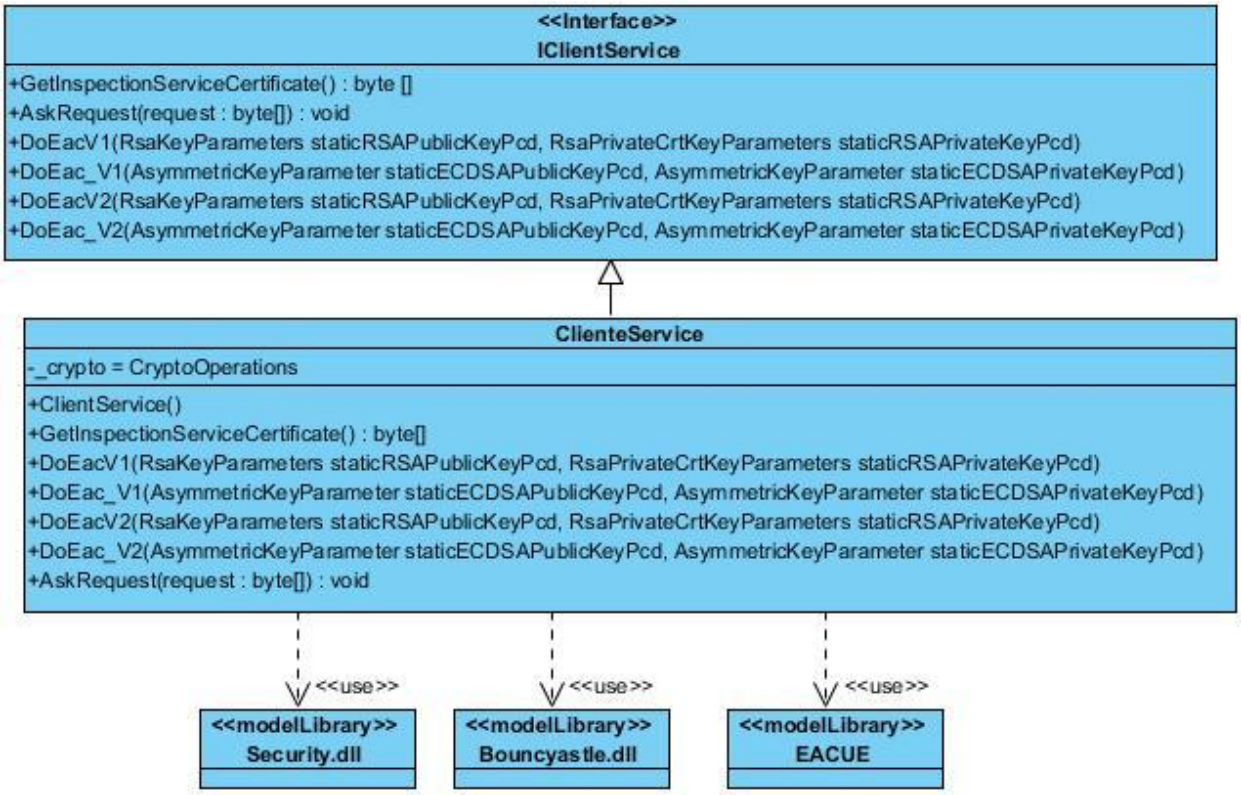


Figura 16.Diagrama de clases (Elaboración Propia).

2.7. Patrones de diseño

El desarrollo de un sistema acarrea, en la mayoría de las ocasiones, darle solución a problemas muy complejos que ya alguien más ha resuelto. Por esta razón uno de los pasos a tener en cuenta cuando se

decide desarrollar un proyecto de *software* es identificar qué patrones pueden ser utilizados. Entiéndase por patrón como una solución estándar para un problema común de programación. Dentro de los patrones de diseño se encuentran los conocidos Patrones Generales de Asignación de Responsabilidades de *Software* (GRASP, por sus siglas en inglés) y Banda de los Cuatro (GOF, por sus siglas en inglés). A continuación se mencionan los patrones de diseño GRASP que se utilizarán en el desarrollo del sistema.

- Patrón Experto.

El Patrón Experto perteneciente al grupo de patrones GRASP consiste en asignar una responsabilidad al experto en información: la clase que cuenta con la información necesaria para cumplir la responsabilidad (40). En la Figura 17 se puede apreciar el uso de este patrón con el que se pretende que los objetos realicen las acciones relacionadas con la información que poseen.

```
public class DH
{
    public DHPublicKeyParameters PublicKey { get; private set; }
    public DHPrivateKeyParameters PrivateKey { get; private set; }
    public DHParameters DHParameters { get; private set; }

    public DH(DHParameters dhParametersPicc)
    {
        try
        {
            IAsymmetricCipherKeyPairGenerator keyGen = GeneratorUtilities.GetKeyPairGenerator("DH");

            KeyGenerationParameters kgp = new DHKeyGenerationParameters(new SecureRandom(), dhParametersPicc);
            keyGen.Init(kgp);
            AsymmetricCipherKeyPair dhTerminalKeyPair = keyGen.GenerateKeyPair();

            PublicKey = (DHPublicKeyParameters)dhTerminalKeyPair.Public;
            PrivateKey = (DHPrivateKeyParameters)dhTerminalKeyPair.Private;
        }
        catch (Exception)
        {

            throw new Exception("Parámetros Diffie-Hellman incorrectos");;
        }
```

Figura 17. Ejemplo patrón Experto.

- Patrón Creador.

El Patrón Creador perteneciente al grupo de patrones GRASP se basa en asignarle a la clase B la responsabilidad de crear una instancia de la clase A (40). Este patrón guía la asignación de responsabilidades relacionadas con la creación de objetos y tiene como propósito fundamental encontrar

un creador que se debe conectar con el objeto producido en cualquier evento, como se muestra en la Figura 18.

```
public class LDAPController
{
    private LDAPOperations _ldap;

    public LDAPController()
    {
        _ldap = new LDAPOperations();
    }
```

Figura 18. Ejemplo patrón Creador.

- Patrón Controlador.

El Patrón Controlador perteneciente al grupo de patrones GRASP se basa en asignar la responsabilidad del manejo de un mensaje de los eventos de un sistema a una clase. Un evento del sistema es un evento de alto nivel generado por un actor externo; es un evento de entrada externa (40). En la Figura 19 se aprecia un ejemplo de este patrón, el cual propone el diseño de clases con la responsabilidad de controlar el flujo de eventos del sistema a clases específicas.

```
public class FormsController
{
    private LDAPService _ldapService;

    #region Certificate

    public string [] ShowLDAPCertificate(string uid)...

    public void AddLDAPCertificate(string [] value)...

    public void DeleteLDAPCertificate(string uid)...

    public void ModifyLDAPCertificate(Dictionary<object, object> dictionary)...

    public List<string[]> ListLDAPCertificate()...

    #endregion
```

Figura 19. Ejemplo patrón Controlador.

- Patrón Bajo Acoplamiento.

El Patrón Bajo Acoplamiento perteneciente al grupo de patrones GRASP consiste en asignar una responsabilidad para mantener bajo acoplamiento. El acoplamiento es una medida de la fuerza con que

una clase está conectada a otras, con las que conoce y con que recurre a ellas (40). El patrón propone el diseño de clases más independientes, lo que reduce el impacto del cambio y facilita la reutilización en otros sistemas, en la Figura 20 se puede apreciar un ejemplo.

```
public class OpenLdapConnection : ResourceConnection
{
    connection attrs

    public OpenLdapConnection()...

    public LdapConnection LdapConnection { get; set; }

    public override void InitConnection()...

    public override void Open()
    {
        if (LdapConnection != null)
        {
            try
            {
                LdapConnection.Connect(Host, Port);
                LdapConnection.Bind(User, Password);
            }
            catch (LdapException ldapExc)
            {
                throw new ResourceConnectionException("Can't Open the connection", ldapExc);
            }
        }
        else
        {
            throw new NullReferenceException();
        }
    }

    public override void Close()...
}
```

Figura 20. Ejemplo patrón Bajo Acoplamiento.

2.8. Conclusiones

- La especificación de los requisitos que debe cumplir el sistema ha sentado las bases para la definición de la arquitectura del sistema, a partir del patrón arquitectónico basado en componentes para describir la estructura general y el patrón n-capas para 3 de los 5 componentes que conforman el sistema.
- Mediante el diseño de los diagramas de clases pertenecientes a cada uno de los componentes del sistema, apoyándose en la utilización de los patrones GRASP, se ha representado de manera estática el conjunto de clases, interfaces y colaboraciones del sistema para su implementación.

Capítulo 3: Implementación y pruebas

3.1. Introducción

En este capítulo se describirán y modelarán las clases y funcionalidades que darán solución al problema de la investigación. Además se realizarán las pruebas unitarias y las de caja negra para verificar el cumplimiento de los requisitos establecidos.

3.2. Pautas de codificación

Un estándar de codificación completo comprende todos los aspectos de la generación de código. Si bien los programadores deben implementar un estándar de forma prudente, éste debe tender siempre a lo práctico. Un código fuente completo debe reflejar un estilo armonioso, como si un único programador hubiera escrito todo el código de una sola vez. Al comenzar un proyecto de *software*, se debe establecer un estándar de codificación para asegurarse de que todos los programadores del proyecto trabajen de forma coordinada. Cuando el proyecto de *software* incorpore código fuente previo, o bien cuando realice el mantenimiento de un sistema de *software* creado anteriormente, el estándar de codificación debería establecer cómo operar con la base de código existente. La legibilidad del código fuente repercute directamente en lo bien que un programador comprende un sistema de *software*. La mantenibilidad del código es la facilidad con que el sistema de *software* puede modificarse para añadirle nuevas características, modificar las ya existentes, depurar errores, o mejorar el rendimiento. Aunque la legibilidad y la mantenibilidad son el resultado de muchos factores, una faceta del desarrollo de *software* en la que todos los programadores influyen especialmente es en la técnica de codificación. El mejor método para asegurarse de que un equipo de programadores mantenga un código de calidad es establecer un estándar de codificación sobre el que se efectuarán luego revisiones del código de rutinas (41).

A continuación se definen las pautas de codificación utilizadas en la implementación del sistema.

- El código fuente debe ser escrito en inglés.
- Cada línea debe contener cuando más una sentencia.

- El nombre de los métodos y de las clases debe ser lo más descriptivo posible comenzando siempre con mayúscula, si es un nombre compuesto por más de una palabra cada una debe comenzar con mayúscula y sin espacio entre ellas. Ejemplo del nombre de un método: *EACChipAuthenticationVersion1*. Ejemplo del nombre de clase: *MRTDChipSimulator*.
- El nombre de las interfaces deben comenzar con la letra I. Ejemplo: *IService1*.
- El nombre de las variables locales deben comenzar con minúscula, si es un nombre compuesto por más de una palabra cada una debe comenzar con mayúscula exceptuando la primera y sin espacio entre ellas. Ejemplo: *curveGenerator.*
- El nombre de las variables privadas debe comenzar con *underscore* (_). Ejemplo: *_randomNoncePicc*.
- Las variables de una sola letra, como i o j sólo se utilizarán para índices cortos.
- Las constantes deben escribirse con mayúscula, si es un nombre compuesto las palabras serán separadas por *underscore* (_).
- Se minimizará el uso de abreviaturas; pero si se emplean, serán de forma coherente. Una abreviatura sólo debe tener un significado, y del mismo modo a cada palabra abreviada sólo debe corresponder una abreviatura. Ejemplo: *DH* corresponde a *Diffie- Hellman*.
- Al principio de cada método se harán comentarios estándar, que indiquen el propósito del mismo. Un comentario podría consistir en una breve introducción que explicará por qué existe y qué puede hacer el método.
- Cuando se modifique el código, se mantendrán actualizados los comentarios circundantes.
- Se usarán frases completas cuando se escriban comentarios. Los comentarios deben aclarar el código, no añadirle ambigüedad.
- Se establecerá un tamaño estándar de sangría y se alinearán las secciones de código mediante la sangría predeterminada.
- Se alinearán verticalmente las llaves de apertura y cierre donde los pares de llaves se alinean.
- Se utilizarán espacios antes y después de los operadores siempre que eso no altere la sangría aplicada al código.

Capítulo 3: Implementación y pruebas

3.3. Diagrama de componentes

Un diagrama de componentes permite visualizar con más facilidad la estructura general del sistema y el comportamiento del servicio que estos componentes proporcionan y utilizan a través de las interfaces (42).

En la Figura 21 se muestra el diagrama de componentes del sistema, el cual está compuesto por los 5 componentes que lo conforman y las relaciones entre ellos.

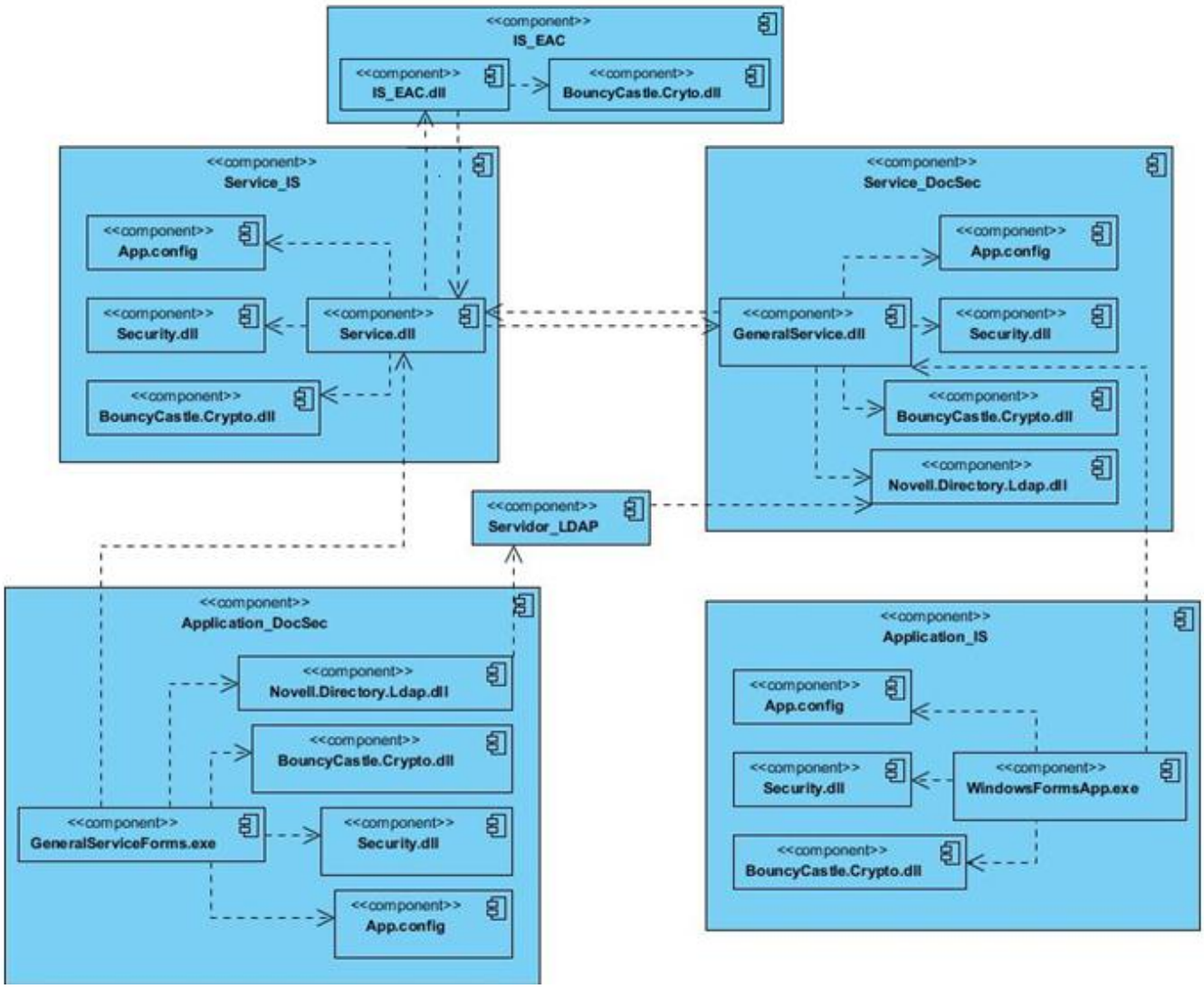


Figura 21. Diagrama de componentes (Elaboración Propia).

Capítulo 3: Implementación y pruebas

A continuación se describen los componentes que conforman el sistema:

IS_EAC.dll es una librería de clases la cual posee las funcionalidades necesarias para la ejecución del EAC-UE en sus dos versiones, se comunica con el servicio del sistema de inspección para obtener las cadenas de certificados necesarias para la comunicación con el *chip*. Además sirve de interfaz para la ejecución del EAC-Singapur, recibiendo los parámetros necesarios para su ejecución y enviándolos hacia el servicio del sistema DocSec.

IS_Service es un servicio implementado en WCF en el que se exponen las funcionalidades para la comunicación segura con el sistema DocSec, así como las operaciones para la creación de los pedidos de certificados y las cadenas de certificados utilizadas por el componente IS_EAC. Para proveer la protección de los datos de IS_Service se utiliza el componente IS_WindowApp, el cual en su capa de presentación expone a los usuarios funcionalidades para la generación de las llaves RSA utilizadas en la comunicación segura entre los servicios, las cuales son ejecutadas en la capa de negocio. DocSec_Service es un servicio implementado en WCF el cual es el encargado, en su capa de negocio, de la ejecución del EAC-Singapur, la gestión de las cadenas de certificados digitales y de los pedidos de certificados a los sistemas de inspección. Para la persistencia de la información en el directorio LDAP, el componente posee una capa de acceso a datos. DocSec_WindowApp es una aplicación de escritorio la cual es la encargada de gestionar la seguridad del DocSec_Service. Posee funcionalidades en su capa de negocio para la gestión de los certificados digitales, de los sistemas de inspección, así como la generación de los pedidos de certificados. Estas funcionalidades son expuestas en la capa de interfaz y con la capa de acceso a datos realiza la gestión de la información en el directorio LDAP. La librería Security.dll ofrece las funcionalidades para el trabajo con los certificados digitales, operaciones criptográficas de firma y verificación con RSA, cifrado y descifrado usando TDES y la generación de los *token* de seguridad para el envío de los datos entre los servicios.

3.4. Diagrama de despliegue

Un diagrama de despliegue es un modelo de objetos que describe la distribución física del sistema en términos de cómo se distribuyen las funcionalidades entre los nodos de cómputo. Cada nodo representa un recurso de cómputo, normalmente un procesador o un dispositivo similar. En la Figura 22 se muestra el diagrama de despliegue del sistema.

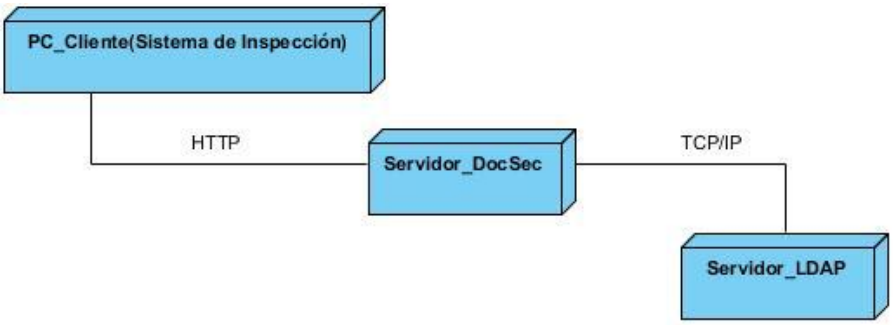


Figura 22. Diagrama de despliegue (Elaboración Propia).

La PC_Cliente (Sistema de Inspección) representa el punto de control migratorio donde estará instado el servicio para la ejecución del EAC.

El Sevidor_DocSec es el encargado de la gestión de los certificados para la ejecución del EAC-UE, la ejecución del EAC-Singapur y la gestión de los sistemas de inspección.

El Servidor_LDAP es el encargado de almacenar los certificados y la información de los sistemas de inspección autorizados.

3.5. Interfaz gráfica

La interfaz gráfica que se muestra en la Figura 23 permite gestionar los certificados, cumpliendo con las funcionalidades de adicionar certificado, eliminar certificado, modificar certificado, exportar certificado y listar los certificados existentes. Se puede apreciar que garantiza la gestión de los sistemas de inspección, permitiendo adicionarlos, eliminarlos, modificarlos y listarlos. Además posibilita generar y cargar las llaves utilizadas para el intercambio de datos.

Capítulo 3: Implementación y pruebas

Figura 23. Interfaz gráfica de administración (Elaboración Propia).

3.6. Fase Estabilización

En esta fase se le aplican las pruebas al *software* desarrollado, comprobando así si este cumple con los requisitos necesarios para llevar a cabo un despliegue satisfactorio del sistema. Las pruebas son un conjunto de actividades que se llevan a cabo sistemáticamente, estas pueden planificarse por adelantado y ejecutarse una vez construido el código. Dentro de cada una de las etapas de desarrollo de un *software* las pruebas son fundamentales ya que a partir de ellas es posible controlar que los productos cumplan requisitos mínimos de operatividad además de garantizar la calidad de estos productos.

Las pruebas constituyen una actividad en la cual un sistema o componente es ejecutado bajo condiciones específicas, se observan o almacenan los resultados y se realiza una evaluación de algún aspecto del sistema o componente (43).

3.6.1. Pruebas unitarias

Las pruebas unitarias tienen como objetivo fundamental, comprobar los caminos lógicos del *software* proponiendo casos de prueba que se ejerciten en conjuntos específicos de condiciones y/o bucles. Estas pruebas son realizadas al código implementado para examinar el estado del programa en varios puntos, y determinar si el estado real coincide con el esperado, llamando directamente métodos que pasándole los parámetros apropiados simulen el proceso. Estas pruebas se le realizaron a las funcionalidades más críticas, utilizando la herramienta *Visual Studio Team System* 2010.

Capítulo 3: Implementación y pruebas

A continuación se muestra en la Tabla 5 una de las pruebas que se le realizaron al sistema, y su resultado, el cual indica que las funcionalidades probadas están correctamente implementadas. El resto de las pruebas unitarias se encuentran en el Anexo 5.

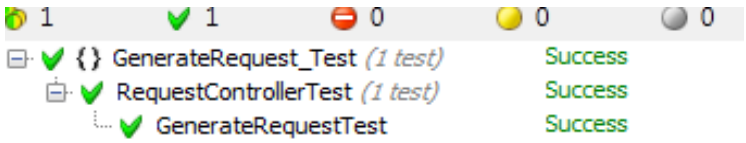
Prueba de Unidad		
Nombre Prueba: GenerateRequest_Test.		
Estado: Satisfactoria.	*Tipo:* Caja Blanca.	*Última Ejecución:* 13/05/2013
Ejecutado por: Daynelis Valdes Monrabal.		*Verificado por:* Alina Surós Vicente.
Descripción: Para que la ejecución de esta prueba tenga resultados satisfactorios y genere un pedido se deben proporcionar los parámetros necesarios.		
Entrada: *List<string> parameters*		
Criterio de aceptación: Genera un pedido.		
Resultado: 		

Tabla 5. Descripción de la prueba unitaria: GenerateRequest_Test.

Resultados de las pruebas

Fueron aplicadas 9 pruebas unitarias a funcionalidades del sistema, arrojando los resultados que se muestran en la Tabla 6.

Iteración	No	Funcionalidad	Resultado
1	1	GenerateRequest (List<string> parameters)	No satisfactorio
1	2	Encrypt (byte[] data)	Satisfactorio
1	3	Decrypt (byte[] data)	Satisfactorio
1	4	ListLDAPCertificate ()	Satisfactorio
1	5	AddLDAPCertificate (string[] value)	Satisfactorio

2	6	DeleteLDAPCertificate (string uid)	Satisfactorio
2	7	ModifyLDAPCertificate (Dictionary<object, object> dictionary)	No satisfactorio
2	8	GenerateRequest (List<string> parameters)	Satisfactorio
3	9	ModifyLDAPCertificate (Dictionary<object, object> dictionary)	Satisfactorio

Tabla 6. No conformidades identificadas en las pruebas unitarias.

A continuación en la Figura 24 se muestran las no conformidades que fueron identificadas en cada una de las iteraciones de implementación del sistema.

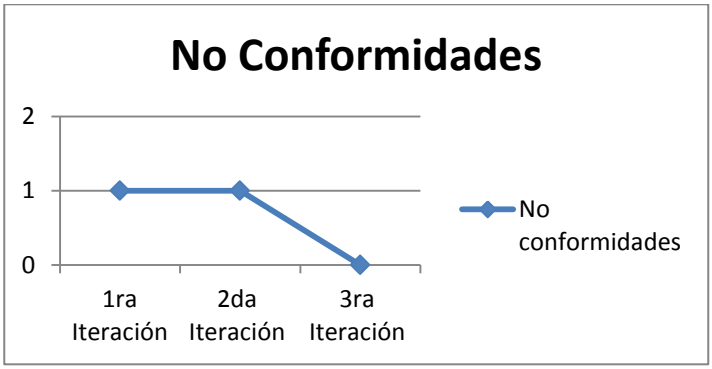


Figura 24. Representación de las no conformidades por iteraciones de las pruebas unitarias.

3.6.2. Pruebas de caja negra o validación del sistema

Dentro de las pruebas más conocidas están las de caja negra o de validación del sistema que son aplicadas a las interfaces del sistema. Se refieren a pruebas que se llevan a cabo mediante casos de pruebas que pretenden demostrar que las funciones del producto son operativas, que la entrada de los datos se realiza de manera adecuada, que se produce un resultado correcto y que la integridad de la información externa se mantiene.

Las pruebas de caja negra se centran principalmente en los requisitos funcionales del *software* y permiten obtener un conjunto de condiciones de entrada que ejerciten completamente estos requisitos.

A continuación se presenta en la Tabla 7 la descripción de las variables y en Tabla 8 el caso de prueba del escenario Eliminar certificado, los restantes se podrán ver en el Anexo 6.

Eliminar certificado:

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	id	campo de texto	No	Admite caracteres alfanuméricos.

Tabla 7. Descripción de las variables: Eliminar certificado.

Escenario	Descripción	id	Respuesta del sistema	Flujo central
EC 15: Eliminar correctamente un certificado.	Se procede a eliminar un certificado.	Válido Test 2	El sistema muestra un mensaje de confirmación y elimina el certificado.	Ir en el menú **Inicio** a: **Gestionar Certificado**. Ir a la opción: **Eliminar Certificado**. Oprimir el botón: **Eliminar.**
		Inválido $fghh	El sistema muestra un mensaje de error.	Ir en el menú **Inicio** a: **Gestionar Certificado**. Ir a la opción: **Eliminar Certificado**. Oprimir el botón: **Eliminar.**

Tabla 8. Caso de prueba del escenario: Eliminar Certificado.

Resultado de las pruebas

Fueron aplicados 15 casos de pruebas de caja negra al *software* que arrojaron las no conformidades que se muestran en el Anexo 7. En cada iteración de prueba se genera un resumen de todas las no conformidades existentes, para entregárselas a los desarrolladores que son los encargados de erradicarlas.

En la Figura 25 se muestran las no conformidades que fueron identificadas al probar los casos de pruebas diseñados.

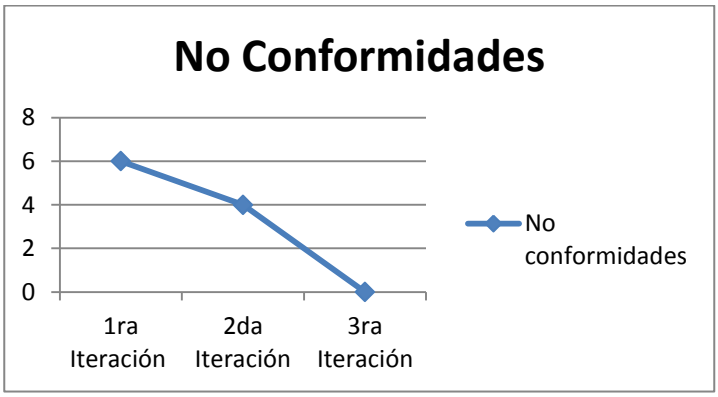


Figura 25. Representación de las no conformidades por iteraciones.

3.7. Conclusiones

- La definición de pautas de codificación ayudó en la organización del código.
- La realización de los diferentes diagramas permitió obtener una visión detallada del sistema. El diagrama de despliegue permitió una mejor visibilidad de la distribución de los nodos físicos que serán necesarios para el despliegue del sistema.
- Las pruebas unitarias posibilitaron evaluar las respuestas de funcionalidades implementadas, mientras que las pruebas de caja negra brindaron la opción de comprobar la correcta ejecución de los requerimientos funcionales previamente definidos, en cada una de las iteraciones realizadas. Las pruebas realizadas al sistema tuvieron un resultado satisfactorio.

Conclusiones

Después de haber completado el trabajo de diploma “Módulo de Control de Acceso Extendido para el sistema de verificación de seguridad de documentos de viajes de lectura mecánica”, se concluye que:

- El análisis de los principales sistemas que implementan medidas de seguridad en pasaporte-e, ha posibilitado conocer el funcionamiento del EAC ayudando en la definición de los requisitos del sistema.
- El análisis de diferentes herramientas, tecnologías, lenguajes y metodologías ha permitido definir cuáles utilizar en el desarrollo del sistema.
- La especificación de los requisitos que debe cumplir el sistema han posibilitado aplicar el patrón arquitectónico basado en componentes para describir la estructura general del sistema y el patrón n-capas para la estructura de 3 de los 5 componentes que conforman la solución.
- El desarrollo de módulo de EAC para el sistema DocSec ha permitido brindar las funcionalidades criptográficas necesarias para acceder a los datos biométricos adicionales almacenados en los pasaportes-e, permitiendo mejorar la identificación de los portadores de dichos documentos.
- La ejecución de las pruebas han posibilitado validar el correcto funcionamiento del sistema.

Recomendaciones

- Implementar el acceso a los datos que se encuentran en el *chip*, y construir los objetos necesarios para realizar las operaciones criptográficas.
- Implementar un mecanismo de almacenamiento de llaves de mayor seguridad, por ejemplo: *Smart Cards.*
- Implementar las funcionalidades para la obtener los grupos de datos 3 y 4 una vez ejecutado el EAC de forma satisfactoria.

Trabajos citados

1. **OACI.** *Documento 9303. Parte 1 Pasaportes de lectura mecánica. Volumen 1 Pasaportes con datos de lectura mecánica almacenados en formato óptico de reconocimiento de caracteres.* s.l. : OACI/ICAO., 2006. 92-9194-871-3.

2. —. *Documento 9303. Parte 1 Pasaportes de lectura mecánica. Volumen 2 Especificaciones para pasaportes electrónicos con capacidad de identificación biométrica.* 2007. 92-9194-896-9.

3. **Security, Federal Office for Information.** *Advanced Security Mechanisms for Machine Readable Travel Documents - Extended Access Control (EAC). TR-03110.* 2004.

4. **ISO/IEC.** *ISO/IEC 7501-1:2008 - Identification cards -- Machine readable travel documents -- Part 1: Machine readable passport.* [En línea] 2008. http://www.iso.org/iso/home/store/catalogue_ics/catalogue_detail_ics.htm?csnumber=45562.

5. **SPRING.** *Singapore Standard SS 529: 2006, Specifications for SmartCard ID.* 2006.

6. **Perdomo Cuevas, Yadier.** *Proyecto técnico: Puntos de Control Migratorio de la República Bolivariana de Venezuela.* 2008.

7. **Surós Vicente, Alina.** Serie Científica. [En línea] 2008. [Citado el: Diciembre 10, 2012.] http://publicaciones.uci.cu/index.php/SC/article/view/48/49.

8. **TAG-MRTD.** *EXTENDED ACCESS CONTROL.* Montreal : ICAO/OACI, 2006.

9. **OACI.** *TECHNICAL ADVISORY GROUP ON MACHINE READABLE TRAVEL DOCUMENTS.* Montréal : s.n., 2007.

10. **Regula.** *Radio Frequency Identification Chip, Software Development Kit.* 2009.

11. **REAL ACADEMIA ESPAÑOLA.** REAL ACADEMIA ESPAÑOLA. [En línea] 2013. [Citado el: Mayo 24, 2013.] http://buscon.rae.es/draeI/SrvltConsulta?TIPO_BUS=3&LEMA=criptograf%C3%ADa.

12. **López, Manuel José Lucena.** *Criptografía y seguridad en computadores.* 1999.

13. **Master, Death.** *Criptosistemas Informáticos.* 2004.

14. **López, Manuel José Lucena .** *Criptografía y Seguridad en Computadores.* Tercera. 2004.

15. Golden Reader Tool. [En línea] Febrero 14, 2013. https://www.bsi.bund.de/EN/Topics/ElectrIDDocuments/Projects/projectsGRT/GRT_node.html.

Trabajos citados

16. D SCAN Master. [En línea] [Citado el: Enero 20, 2013.] http://www.crossmatch.com/product_assets/brochures/D_SCAN_Master_Spanish.pdf.

17. JMRTD. [En línea] JMRTD team, 2013. [Citado el: Enero 10, 2013.] http://jmrtd.org/about.shtml.

18. **MorónRuano, David.** Ventajas de .Net. [En línea] http://www.desarrolloweb.com/articulos/1329.php.

19. **Hernández Sánchez, Sergio y Blázquez Román, Fernando.** *.NET Framework.* 2005.

20. **Microsoft.** ¿Qué es Windows Communication Foundation? [En línea] 2013. [Citado el: Febrero 6, 2013.] http://msdn.microsoft.com/es-es/library/ms731082.aspx.

21. **RSA .** *PKCS #12: Personal Information Exchange Syntax Standard.* s.l. : RSA Laboratories, 1999.

22. **Morón Ruano, David.** *LDAP.* s.l. : A.B.M, 2010. pág. 4.

23. **Bouncycastle.** bouncycastle.org. [En línea] 2013. [Citado el: Febrero 6, 2013.] http://www.bouncycastle.org/.

24. La Máquina Virtual Java. [En línea] [Citado el: Mayo 3, 2013.] http://www.sc.ehu.es/sbweb/fisica/cursoJava/fundamentos/introduccion/virtual.htm.

25. **Colombia., Microsoft.** http://www.microsoft.com/colombia/portafolio/msf.htm. [En línea] Octubre 29, 2009.

26. **Jacobson, Ivar , Booch , Grady y Rumbaugh , James.** *El Proceso Unificado de Desarrollo de Software.* s.l. : España.

27. **Canós, José H., Letelier, Patricio y Penadés, Mª Carmen.** [En línea] noqualityinside.com.ar/nqi/nqifiles/XP_Agil.pdf.

28. **EcuRed.** Programación Extrema o XP. [En línea] 2012. http://www.ecured.cu/index.php/Programación_Extrema_o_XP.

29. **Microsoft.** Chapter 1 - Introduction to the Microsoft Solutions Framework. [En línea] Microsoft, 2013. [Citado el: Febrero 13, 2013.] http://technet.microsoft.com/en-us/library/bb497060.aspx.

30. HI-04: Glosarios Buscar. [En línea] http://aprendeenlinea.udea.edu.co/lms/moodle/mod/glossary/showentry.php?courseid=297&concept=lenguaje+de+programación.

31. **ECHEVERRY JHON, NATALIA.** *LENGUAJE DE PROGRAMACIÓN C# .* s.l. : UNIVERSIDAD DE ANTIOQUIA, 2011.

32. **Oracle.** Java. [En línea] [Citado el: Mayo 2, 2013.] http://www.java.com/es/download/faq/whatis_java.xml.

Trabajos citados

33. ¿Qué es UML? [En línea] [Citado el: Febrero 5, 2013.] http://profesores.fi-b.unam.mx/carlos/aydoo/uml.html.

34. **Correa Bautista, José Carlos y Machado García, Yendry.** *SUBSISTEMA DE APROVISIONAMIENTO DE USUARIOS PARA EL SISTEMA DE ADMINISTRACIÓN DE IDENTIDADES.* La Habana : s.n., 2012. pág. 24.

35. **Oracle.** NetBeans. [En línea] [Citado el: Mayo 3, 2013.] https://netbeans.org/community/releases/72/.

36. Csribd. [En línea] [Citado el: Mayo 8, 2013.] http://es.scribd.com/doc/36990887/Tabla-Herramientas-CASE.

37. **Etcheverry, Lorena.** Pedeciba. [En línea] Marzo 2010. [Citado el: Marzo 20, 2013.] www.pedeciba.edu.uy/bioinformatica/sibdyw/Clase_3.pdf.

38. **de la Torre Llorente, César.** *Guía de Arquitectura N-Capas orientada al Dominio con .NET 4.0.* 2010.

39. **López Torres, Deyanira y Mayet Sánchez, Wilver.** *Sistema para la ejecución de la autenticación pasiva de los pasaportes electrónicos.* La Habana : s.n., 2012. pág. 46.

40. **CRAIG, L.** *UML y Patrones Introducción al análisis y diseño orientado a objetos.* 2004.

41. **Microsoft.** Revisiones de código y estándares de codificación. [En línea] 2013. [Citado el: Febrero 10, 2013.] http://msdn.microsoft.com/es-es/library/aa291591%28v=vs.71%29.aspx.

42. Diagramas de componentes de UML: Referencia. [En línea] [Citado el: 04 03, 2013.] http://msdn.microsoft.com/es-es/library/dd409390.aspx.

43. **López Torres, Deyanira y Mayet Sánchez, Wilver .** *Sistema para la ejecución de la autenticación pasiva de los pasaportes electrónicos.* La Habana : s.n., 2012. pág. 56.

44. **Corporation, Microsoft. Process Guidance. 2006.**

45. **Framework, Microsoft Solutions.** Microsoft Solutions Framework . [En línea] http://www.microsoft.com/MSF.

46. **Colombia., Microsoft.** [En línea] http://www.microsoft.com/colombia/portafolio/msf.htm.

47. **Aguilar Baquero, Gabriela Rebeca.** *Análisis e implementación de un sistema automatizado de digitalización de documentos (SADO) para soluciones inteligentes, en CIENCIAS DE LA COMPUTACIÓN,ESCUELA POLITÉCNICA DEL EJÉRCITO. Active Directory.* 2011.

48. **ERICH, G. y H., RICHARD.** *Design Patterns: Elements of Reusable Object-Oriented Software.* 1995.

49. **Bustos, Guillermo.** *Guía de Uso de la Herramienta CASE Visual Paradigm Standard Edition Versión 8.0.* 2010.

Trabajos citados

50. **López Torres, Deyanira y Mayet Sánchez, Wilver .** *Sistema para la ejecución de la autenticación pasiva de los pasaportes electrónicos.* La Habana : s.n., 2012. pág. 44.

51. Archivos Mensuales. [En línea] [Citado el: Enero 20, 2013.] http://chsos20112906045.wordpress.com/2011/09/.

52. Diagramas de componentes de UML: Referencia. [En línea] [Citado el: Abril 3, 2013.] http://msdn.microsoft.com/es-es/library/dd409390.aspx.

53. *AltovaUModel2010 .*

54. Scribd. [En línea] [Citado el: Mayo 8, 2013.] http://es.scribd.com/doc/36990887/Tabla-Herramientas-CASE.

55. **OACI.** *Machine Readable Travel Documents.* 2006. Vol. Specifications for Electronically Enabled Passports with Biometric Identification Capability.

Glosario de términos

A

API (Application Programming Interface): interfaz de programación de aplicaciones, conjunto de convenciones internacionales que definen cómo debe invocarse una determinada función de un programa desde una aplicación.

Arquitectura: indica la estructura, funcionamiento e interacción entre las partes del *software*.

B

BouncyCastle (BC): está organizada de forma que sus API sean adaptables para su uso bajo cualquier entorno de infraestructura adicional. El paquete se distribuye bajo licencia privativa.

C

CA (Autoridad de Certificación): entidad responsable de emitir y revocar certificados digitales.

CLR (Common Language Runtime): entorno de ejecución para los códigos de los programas que corren sobre la plataforma *Microsoft* .NET. El CLR es el encargado de compilar una forma de código intermedio llamada *Common Intermediate Language*, al código de máquina nativo, mediante un compilador en tiempo de ejecución.

D

DLL (Dynamic Link Library): la Biblioteca de Vínculos Dinámicos es un archivo que contiene funciones que se pueden llamar desde aplicaciones u otras DLL. Los desarrolladores utilizan las DLL para poder reciclar el código y aislar las diferentes tareas.

E

Escenario: define la interacción entre las personas y el sistema, estos registran los pasos específicos a seguir para lograr el cumplimiento efectivo de una funcionalidad.

F

Glosario de términos

Framework: estructura de soporte definida en la cual otro proyecto de *software* puede ser organizado y desarrollado. Típicamente, puede incluir soporte de programas, bibliotecas y un lenguaje interpretado entre otros *software* para ayudar a desarrollar y unir los diferentes componentes de un proyecto.

G

Golden Reader Tool (TRB): aplicación para la lectura de documentos electrónicos de identificación.

I

IDE: *software* compuesto por un conjunto de herramientas de programación. Es un entorno de programación que ha sido empaquetado como un programa de aplicación, es decir, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica (GUI).

J

JMRTD: es una implementación de código abierto en Java del MRTD del *chip*, que cumple con los estándares de la OACI.

L

LDAP: es un protocolo a nivel de aplicación que permite el acceso a un servicio de directorio ordenado y distribuido para buscar diversa información en un entorno de red.

M

Módulo: componente auto controlado de un sistema que posee una interfaz bien definida hacia otros componentes.

Metodología de desarrollo de *software*: proceso para la producción organizada del *software*, empleando para ello una colección de técnicas predefinidas y convencionales en las notaciones. Una metodología se presenta normalmente como una serie de pasos, con técnicas y notaciones asociadas a cada uno de ellos. Los pasos de la producción del *software* se organizan normalmente en un ciclo de vida consistente en varias fases de desarrollo.

O

Glosario de términos

OACI (Organización de la Aviación Civil Internacional): organización encargada de establecer normas y regulaciones internacionales necesarias para garantizar la seguridad, eficiencia y regularidad del transporte aéreo.

P

PKI (Infraestructura de Clave Pública): es una combinación de *hardware* y *software*, políticas y procedimientos de seguridad que permiten la ejecución con garantías de operaciones criptográficas como el cifrado, la firma digital o el no repudio de transacciones electrónicas.

R

Requisito de calidad de servicio: material resultante esperado del producto final. El mismo puede ser un resultado, un problema resuelto o una característica, entre otros.

S

Sistema: Un sistema informático como todo sistema, es el conjunto de partes interrelacionadas, *hardware*, *software* y de recurso humano que permite almacenar y procesar información.

T

Tecnología: conjunto de conocimientos técnicos, ordenados científicamente, que permiten diseñar y crear bienes y servicios que facilitan la adaptación al medio ambiente y satisfacer tanto las necesidades esenciales como los deseos de la humanidad.

W

Windows Communication Foundation (WCF): es un marco de trabajo para la creación de aplicaciones orientadas a servicios.

X

XML (Extensible Markup Language*)*: lenguaje de marcado extensible, es un metalenguaje extensible de etiquetas desarrollado por el World Wide Web Consortium (W3C).

Anexo 1

Listado de los escenarios con sus correspondientes prioridades:

Escenarios	**Prioridad**
Ejecutar el EAC-UE versión 1.	5
Autenticar *chip*.	5
Autenticar terminal.	5
Ejecutar el EAC-UE versión 2.	5
Autenticar terminal.	5
Autenticar *chip*.	5
Ejecutar el EAC-Singapur.	5
Adicionar sistemas de inspección.	4
Eliminar sistemas de inspección.	3
Modificar sistemas de inspección.	4
Listar sistemas de inspección.	3
Autenticar usuario.	5
Adicionar certificados.	5
Modificar certificados.	5
Eliminar certificados.	4
Mostrar certificados.	5

Listar certificados.	3
Generar cadena de certificados.	5
Enviar cadena de certificados.	4
Generar pedidos de certificados.	5
Generar el par de llaves	5

Tabla 9. Prioridad de escenarios.

Anexo 2

Plan de iteraciones:

Iteración	Escenario	Prioridad	Riesgo	Esfuerzo(Días)
1	Ejecutar el EAC-UE versión 1.	5	Alto	6
1	Autenticar *chip*.	5	Alto	3
1	Autenticar terminal.	5	Alto	3
1	Ejecutar el EAC-UE versión 2.	5	Alto	6
1	Autenticar terminal.	5	Alto	3
1	Autenticar *chip*.	5	Alto	3
1	Ejecutar el EAC-Singapur.	5	Alto	2
2	Adicionar sistemas de inspección.	4	Alto	3
2	Eliminar sistemas de inspección.	3	Medio	3
2	Modificar sistemas de inspección.	4	Alto	3
2	Listar sistemas de inspección.	3	Alto	3
1	Autenticar usuario.	5	Alto	5
1	Adicionar certificados.	5	Alto	3
1	Modificar certificados.	5	Alto	3
2	Eliminar certificados.	4	Alto	3
1	Mostrar certificados.	5	Alto	3

2	Listar certificados.	3	Medio	3
1	Generar cadena de certificados.	5	Alto	3
2	Enviar cadena de certificados.	4	Alto	4
1	Generar pedidos de certificados.	5	Alto	5
1	Generar el par de llaves	5	Alto	5

Tabla 10. Plan de iteraciones.

Anexo 3

Descripción de escenarios.

Módulo: Ejecución de EAC.

Autenticar *chip* versión 1.

Nombre del escenario: Autenticar *chip* versión 1.		**Identificador:** ESC 2
Objetivo del escenario: Realizar autenticación del *chip* para la versión 1 del EAC-UE. Es necesario para que el *chip* pueda comprobarle al terminal que quiere realizar la lectura de los datos que el *chip* es auténtico.		
Persona: Sistema de control migratorio.		
Iteración: 1	**Prioridad:** Crítico	**Complejidad:** 3
Precondiciones: Debe haberse identificado que el pasaporte tenga implementado el EAC-UE versión 1.		

Descripción: Este escenario comienza cuando se ha identificado que es la versión 1 del EAC-UE la que tiene implementada el pasaporte.
Para la realización de este mecanismo es necesario realizar los siguientes procedimientos:
• El *chip* le envía su llave pública DH y los parámetros del dominio al terminal.
• El terminal genera un par de llaves efímeras con los parámetros recibidos y le envía la llave pública al *chip*.
• El terminal y el *chip* calculan:
  ○ El secreto compartido utilizando las llaves DH.
  ○ Las llaves de sección derivadas del secreto compartido.
• El terminal calcula el comprimido de la llave pública para su posterior autenticación.		

Validaciones: Verificar que comenzó correctamente la ejecución del EAC-UE versión 1.		

Tabla 11. Descripción del escenario: Autenticar *chip*.

Autenticar terminal versión 1.

Nombre del escenario: Autenticar terminal.		**Identificador:** ESC 3
Objetivo del escenario: Realizar autenticación del terminal para la versión 1 del EAC-UE. Es necesario para que el		

terminal pueda comprobarle al *chip* que tiene los permisos requeridos para poder realizar la lectura de los datos.		
Persona: Sistema de control migratorio.		
Iteración: 1	**Prioridad:** Crítico	**Complejidad:** 3
Precondiciones: Debe haberse identificado que el pasaporte tenga implementado el EAC-UE versión 1. Debe haberse realizado la autenticación del *chip*.		
Descripción: Este escenario comienza cuando se ha identificado que es la versión 1 del EAC-UE la que tiene implementada el pasaporte y después que se comienza a realizar la autenticación del *chip*. Para la realización de este mecanismo es necesario realizar los siguientes procedimientos: • El terminal le envía una cadena de certificados al *chip*. • El *chip* comprueba la validez de esta cadena y extrae la llave pública del terminal. • El *chip* escoge un valor aleatorio y se lo envía al terminal. • El terminal firma el valor recibido y se lo envía al *chip*. • El *chip* comprueba la validez de la firma.		
Validaciones: Verificar que se establece la comunicación entre el *chip* y el terminal.		

Tabla 12. Descripción del escenario: Autenticar terminal.

Ejecutar el EAC-UE versión 2.

Nombre del escenario: Ejecutar el EAC-UE versión 2.		**Identificador:** ESC 4
Objetivo del escenario: Realizar el EAC-UE versión 2 a los pasaportes electrónicos.		
Persona: Sistema de control migratorio.		
Iteración: 1	**Prioridad:** Crítico	**Complejidad:** 3
Precondiciones: Debe haberse identificado que el pasaporte tenga implementado el EAC-UE versión 2. Debe haberse realizado previamente los mecanismos de seguridad del BAC.		
Descripción: El escenario comienza cuando se invoca la funcionalidad de obtener los datos sensibles almacenados en el pasaporte. Para realizar el mismo es necesario realizar la autenticación del terminal y luego la autenticación del *chip*, para verificar si ese sistema de inspección tiene los permisos para leer los datos sensibles almacenados en el *chip* del pasaporte.		

Validaciones: Verificar que se haya ejecutado correctamente el EAC-UE versión 2.

Tabla 13. Descripción del escenario: Ejecutar el EAC-UE versión 2.

Autenticar terminal versión 2.

Nombre del escenario: Autenticar terminal versión 2.		**Identificador:** ESC 5
Objetivo del escenario: Realizar autenticación del terminal para la versión 2 del EAC-UE. Es necesario para que el terminal pueda comprobarle al *chip* que tiene los permisos requeridos para realizar la lectura de los datos.		
Persona: Sistema de control migratorio.		
Iteración: 1	**Prioridad:** Crítico	**Complejidad:** 3
Precondiciones: Debe haberse identificado que el pasaporte tiene implementado el EAC-UE versión 2. Debe haberse realizado previamente los mecanismos de seguridad del BAC.		
Descripción: Este escenario comienza cuando se ha identificado que es la versión 2 del EAC-UE la que tiene implementada el pasaporte. Para la realización de este mecanismo es necesario realizar los siguientes procedimientos: • El terminal le envía una cadena de certificados al *chip*. • El *chip* comprueba la validez de esta cadena y extrae la llave pública del terminal. • El terminal genera un par de llaves DH efímeras. Le envía el comprimido de la llave pública al *chip* y puede enviarle un valor auxiliar aleatorio. • El *chip* escoge un valor aleatorio y se lo envía al terminal. • El terminal firma el valor recibido y se lo envía al *chip*. • El *chip* comprueba la validez de la firma.		
Validaciones: Verificar que comenzó correctamente la ejecución del EAC-UE versión 2.		

Tabla 14. Descripción del escenario: Autenticar terminal.

Autenticar *chip* versión 2.

Nombre del escenario: Autenticar *chip* versión 2.	**Identificador:** ESC 6
Objetivo del escenario: Realizar autenticación del *chip* para la versión 2 del EAC-UE. Es necesario para que el *chip* pueda comprobarle al terminal su autenticidad.	

Persona: Sistema de control migratorio.		
Iteración: 1	**Prioridad:** Crítico	**Complejidad:** 3
Precondiciones: Debe haberse identificado que el pasaporte tiene implementado el EAC-UE versión 2. Debe de haberse realizado la autenticación del terminal.		
Descripción: Este escenario comienza cuando se ha identificado que es la versión 2 del EAC-UE la que tiene implementada el pasaporte y después de la ejecución de la autenticación del terminal. Para la realización de este mecanismo es necesario realizar los siguientes procedimientos: • El *chip* envía la llave DH pública y los parámetros del dominio al terminal. • El terminal envía la llave DH pública efímera al *chip*. • El *chip* calcula el comprimido de la llave recibida y la compara con el comprimido recibido en la autenticación del terminal. • El *chip* y el terminal calculan el secreto compartido utilizando las llaves DH. • El *chip* escoge un valor aleatorio y lo utiliza para generar las llaves de sección. Genera un *token* con el cifrado del comprimido de la llave pública del terminal utilizando las llaves generadas y envía el valor aleatorio y el *token* al terminal. • El terminal deriva las nuevas llaves de sección utilizando el valor aleatorio y con estas verifica el token recibido.		
Validaciones: Verificar que se establece la comunicación entre el *chip* y el terminal.		

Tabla 15. Descripción del escenario: Autenticar *chip*.

Ejecutar el EAC-Singapur.

Nombre del escenario: Ejecutar el EAC-Singapur.		**Identificador:** ESC 7
Objetivo del escenario: Realizar el EAC-Singapur a los pasaportes electrónicos.		
Persona: Sistema de control migratorio.		
Iteración:1	**Prioridad:** Crítico	**Complejidad:** 3
Precondiciones: Debe haberse identificado que el pasaporte tiene implementado el EAC-Singapur.		
Descripción: Este escenario comienza cuando se ha identificado que es el EAC-Singapur el que tiene implementado el pasaporte. Es necesario para poder obtener los datos sensibles que están almacenados en el		

pasaporte. Para realizar el mismo es necesario obtener la llave simétrica encriptada que esta almacenada en el grupo de dato 13 del *chip*. Se descifra esta llave y es devuelta para la posterior ejecución del comando *EXTERNAL AUTHENTICATE*.
Validaciones: Verificar que se haya ejecutado correctamente el EAC-Singapur.

Tabla 16. Descripción del escenario: Ejecutar el EAC-Singapur

Adicionar sistema de inspección.

Nombre del escenario: Adicionar sistemas de inspección.		**Identificador:** ESC 8
Objetivo del escenario: Adicionar sistemas de inspección.		
Persona: Administrador del sistema.		
Iteración: 2	**Prioridad:** Crítico	**Complejidad:** 3
Precondiciones: -.		
Descripción: Este escenario comienza cuando se adicionan nuevos sistemas de inspección. Es necesario para tener un control sobre los sistemas de inspección.		
Validaciones: Verificar que se haya adicionado correctamente el sistema de inspección.		

Tabla 17. Descripción del escenario: Configurar nuevos lectores al sistema.

Eliminar sistemas de inspección.

Nombre del escenario: Eliminar sistemas de inspección.		**Identificador:** ESC 9
Objetivo del escenario: Eliminar sistemas de inspección.		
Persona: Administrador del sistema.		
Iteración: 2	**Prioridad:** Crítico	**Complejidad:** 2
Precondiciones: El sistema de inspección debe estar añadido.		
Descripción: Este escenario comienza cuando un sistema de inspección no es utilizado. Es necesario cuando ya no se utilice un sistema de inspección, este no pueda realizar ninguna operación.		

Validaciones: Verificar que el sistema de inspección no pueda tener acceso a los datos de los pasaportes.

Tabla 18. Descripción del escenario: Eliminar lectores del sistema.

Modificar sistemas de inspección.

Nombre del escenario: Modificar sistemas de inspección.		**Identificador:** ESC 10
Objetivo del escenario: Modificar sistemas de inspección.		
Persona: Administrador del sistema.		
Iteración: 2	**Prioridad:** Crítico	**Complejidad:** 2
Precondiciones: El sistema de inspección debe estar configurado para tener acceso a los datos del pasaporte.		
Descripción: Este escenario comienza cuando se desea cambiar los datos de un sistema de inspección. Es necesario cuando haya que cambiar algún dato al sistema de inspección.		
Validaciones: Verificar que los datos del sistema de inspección fueron actualizados.		

Tabla 19. Descripción del escenario: Eliminar lectores del sistema.

Listar sistemas de inspección.

Nombre del escenario: Listar sistemas de inspección.		**Identificador:** ESC 11
Objetivo del escenario: Listar sistemas de inspección.		
Persona: Administrador del sistema.		
Iteración: 2	**Prioridad:** Crítico	**Complejidad:** 2
Precondiciones: Los sistemas de inspección deben estar configurados para tener acceso a los datos del pasaporte.		
Descripción: Este escenario comienza cuando se desean conocer los datos de los sistemas de inspección. Es necesario para tener un control de todos los sistemas de inspección.		
Validaciones: Verificar que todos los datos del sistema de inspección sean mostrados.		

Tabla 20. Descripción del escenario: Eliminar lectores del sistema.

Anexo 3

Módulo: Repositorio.

Autenticar usuario.

Nombre del escenario: Autenticar usuario.		**Identificador:** ESC 12
Objetivo del escenario: Que los usuarios autorizados puedan acceder a las aplicaciones de administración de los servicios.		
Persona: Administrador del sistema.		
Iteración: 1	**Prioridad:** Crítico	**Complejidad:** 3
Precondiciones: -		
Descripción: El usuario introduce su usuario y contraseña en los campos de formulario, se comprueban los mismos y de ser correctos le permite el acceso a la aplicación de administración, de ser incorrectos muestra un mensaje de alerta informando el error.		
Validaciones:		

Tabla 21. Descripción del escenario: Autenticar usuario.

Adicionar certificados.

Nombre del escenario: Adicionar certificados.		**Identificador:** ESC 13
Objetivo del escenario: Adicionar certificados.		
Persona: Administrador del sistema.		
Iteración: 1	**Prioridad:** Crítico	**Complejidad:** 3
Precondiciones: El certificado no puede estar almacenado. El administrador debe estar autenticado.		
Descripción: El escenario comienza cuando se ha adquirido un certificado. Es necesario para tener guardados todos los certificados que son necesarios para realizar las operaciones del sistema.		
Validaciones: Verificar que se haya guardado correctamente el certificado.		

Tabla 22. Descripción del escenario: Almacenar certificados.

Modificar certificados.

Nombre del escenario: Actualizar certificados.		**Identificador:** ESC 14
Objetivo del escenario: Actualizar certificados.		
Persona: Administrador del sistema.		
Iteración: 1	**Prioridad:** Crítico	**Complejidad:** 3
Precondiciones: El certificado debe estar almacenado. El administrador debe estar autenticado.		
Descripción: Este escenario comienza cuando se desea cambiar los datos de un certificado. Es necesario cuando haya que cambiar algún dato a un certificado.		
Validaciones: Verificar que sea válido el certificado.		

Tabla 23. Descripción del escenario: Actualizar certificados.

Eliminar certificados.

Nombre del escenario: Eliminar certificados.		**Identificador:** ESC 15
Objetivo del escenario: Eliminar certificados.		
Persona: Administrador del sistema.		
Iteración: 2	**Prioridad:** Crítico	**Complejidad:** 3
Precondiciones: El certificado debe estar almacenado. El administrador debe estar autenticado.		
Descripción: Este escenario comienza cuando un certificado no es utilizado, o no es válido. Es necesario cuando ya no se pueda utilizar un certificado.		
Validaciones: Verificar que sea haya eliminado correctamente el certificado.		

Tabla 24. Descripción del escenario: Actualizar certificados.

Mostrar certificados.

Nombre del escenario: Mostrar certificados.	**Identificador:** ESC 16
Objetivo del escenario: Mostrar certificados.	
Persona: Administrador del sistema.	

Iteración: 1	**Prioridad:** Crítico	**Complejidad:** 3
Precondiciones: El certificado debe estar almacenado. El administrador debe estar autenticado.		
Descripción: Este escenario comienza cuando se desee conocer un certificado. Es necesario cuando se necesita adquirir los datos de algún certificado.		
Validaciones: Verificar que sean correctos los datos mostrados del certificado.		

Tabla 25. Descripción del escenario: Actualizar certificados.

Listar certificados.

Nombre del escenario: Listar certificados.		**Identificador:** ESC 17
Objetivo del escenario: Listar certificados.		
Persona: Administrador del sistema.		
Iteración: 2	**Prioridad:** Crítico	**Complejidad:** 2
Precondiciones: El certificado debe estar almacenado. El administrador debe estar autenticado.		
Descripción: Este escenario comienza cuando se desean conocer los datos de los certificados. Es necesario para tener un control de todos los certificados.		
Validaciones: Verificar que todos los datos de los certificados sean mostrados.		

Tabla 26. Descripción del escenario: Actualizar certificados.

Generar cadena de certificados.

Nombre del escenario: Generar cadena de certificados.		**Identificador:** ESC 18
Objetivo del escenario: Generar cadena de certificados.		
Persona: Administrador del sistema.		
Iteración: 1	**Prioridad:** Crítico	**Complejidad:** 3
Precondiciones: Tienen que estar actualizados todos los certificados que componen la cadena de certificados. El administrador debe estar autenticado.		
Descripción: Este escenario comienza cuando es necesario autenticar el terminal. Se define el país para el cual se		

necesita la cadena de certificados, se realiza una búsqueda en el directorio LDAP de todos los certificados asociados al país especificado y se verifican su fecha de expiración. Como resultado se obtienen dos certificados, el del sistema de inspección y el del verificador de documento.
Validaciones: Verificar que sea válida la cadena de certificados.

Tabla 27. Descripción del escenario: Generar cadena de certificados.

Enviar cadena de certificados.

Nombre del escenario: Enviar cadena de certificados.		**Identificador:** ESC 19
Objetivo del escenario: Enviar cadena de certificados.		
Persona: Administrador del sistema.		
Iteración: 2	**Prioridad:** Crítico	**Complejidad:** 3
Precondiciones: Tienen que estar actualizados todos los certificados que componen la cadena de certificados y la cadena debe estar generada. El administrador debe estar autenticado.		
Descripción: Este escenario comienza después que se ha creado la cadena de certificados. Es necesario para realizar la autenticación del terminal.		
Validaciones: Verificar que ha sido enviada correctamente la cadena de certificados.		

Tabla 28. Descripción del escenario: Enviar cadena de certificados.

Generar pedidos de certificados.

Nombre del escenario: Generar pedidos de certificados.		**Identificador:** ESC 20
Objetivo del escenario: Generar pedidos de certificados.		
Persona: Administrador del sistema.		
Iteración: 1	**Prioridad:** Crítico	**Complejidad:** 3
Precondiciones: Tienen que haber caducado algunos de los certificados del sistema de inspección. El administrador debe estar autenticado.		
Descripción: Este escenario comienza cuando ha expirado un certificado o cuando sus llaves se han visto		

comprometidas. El administrador del sistema define los parámetros del nuevo pedido de certificado, estos son enviados al sistema de inspección para su confección y el almacenamiento de las llaves. Como resultado el sistema de inspección retorna un pedido de certificado al sistema DocSec.
Validaciones: Verificar que ha sido correcta la generación del pedido de certificados.

Tabla 29. Descripción del escenario: Generar pedidos de certificados.

Generar el par de llaves (Llave pública y llave privada).

Nombre del escenario: Generar el par de llaves (Llave pública y llave privada).		**Identificador:** ESC 21
Objetivo del escenario: Generar el par de llaves (Llave pública y llave privada) con el RSA y con ECDSA.		
Persona: Sistema de inspección.		
Iteración: 1	**Prioridad:** Crítico	**Complejidad:** 3
Precondiciones: -.		
Descripción: Este escenario comienza cuando se va comenzar a realizar la autenticación del *chip* y la autenticación del terminal. Es necesario para realizar el proceso de EAC.		
Validaciones: Verificar que son correctas las llaves generadas.		

Tabla 30. Descripción del escenario: Generar el par de llaves (Llave pública y llave privada).

Anexo 4

Diagrama de clases: Aplicación del Sistema de Inspección.

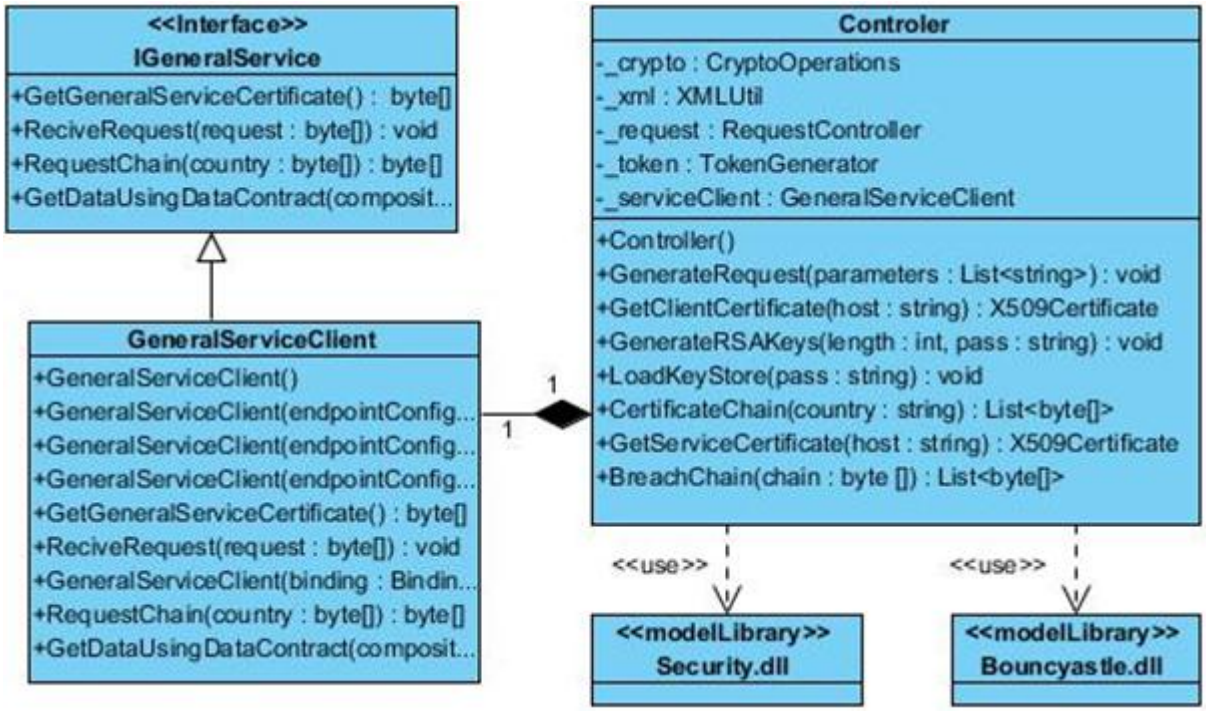


Figura 26. Diagrama de clase: Aplicación del Sistema de Inspección.

Anexo 4

Diagrama de clases: Aplicación del Servicio General.

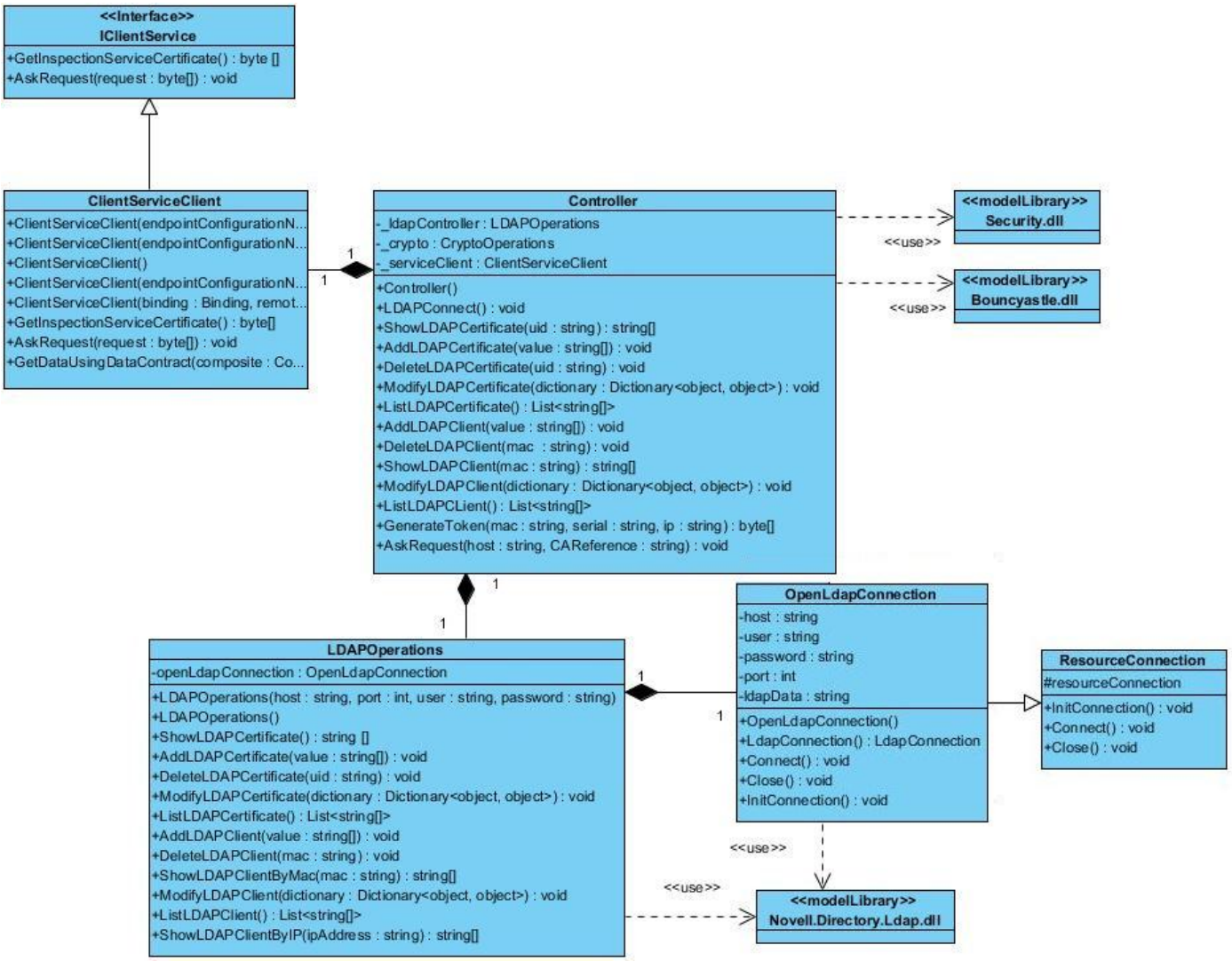


Figura 27. Diagrama de clase: Aplicación del Servicio General.

Anexo 4

Diagrama de clases: Servicio del Sistema General.

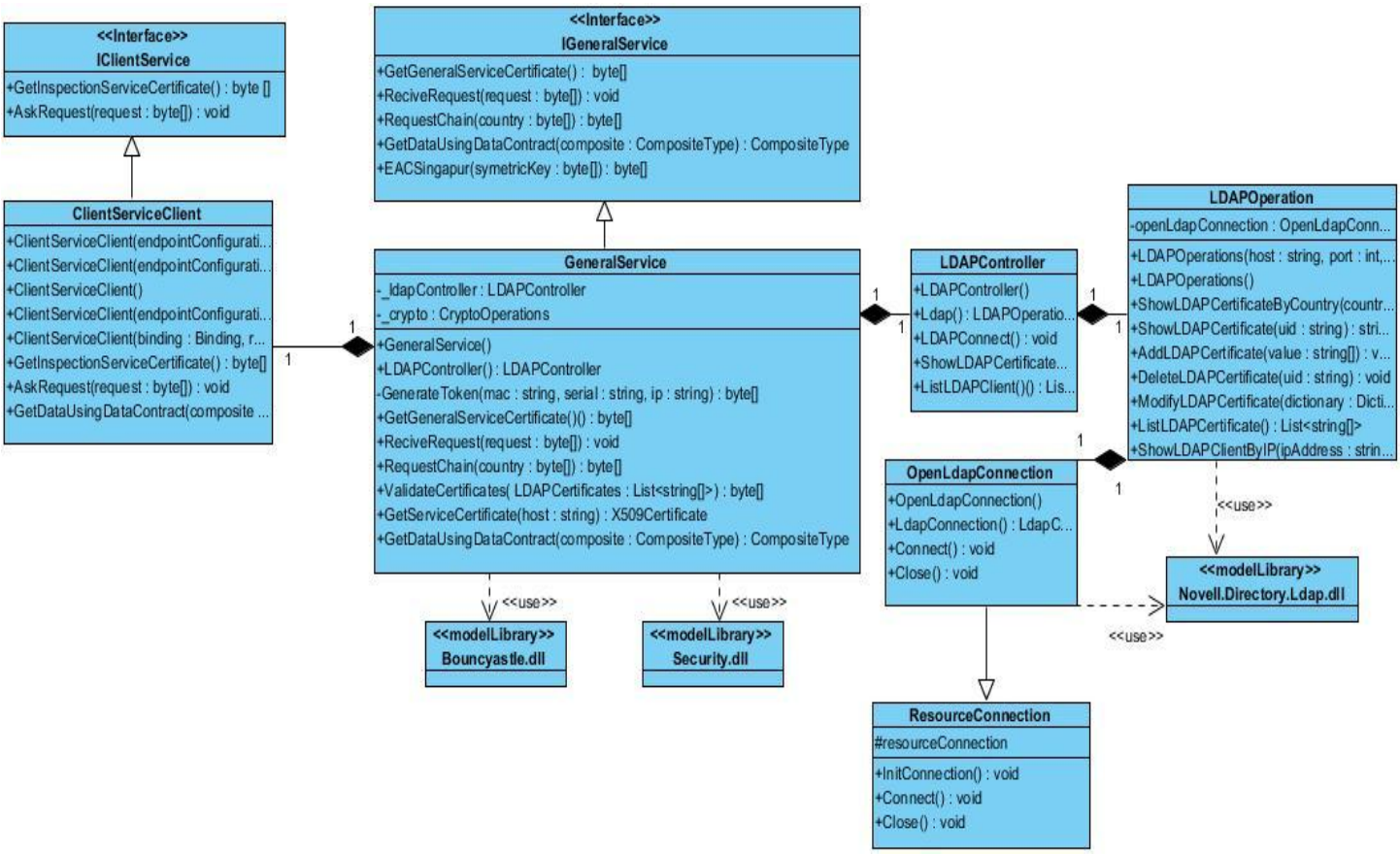


Figura 28. Diagrama de clase: Servicio del Sistema General.

Anexo 5

Prueba unitaria: Encrypt_Test.

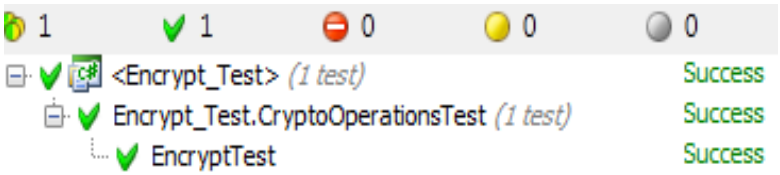
Prueba de Unidad		
Nombre Prueba: Encrypt_Test.		
Estado: Satisfactoria.	*Tipo:* Caja Blanca.	*Última Ejecución:* 13/05/2013
Ejecutado por: Daynelis Valdes Monrabal.		*Verificado por:* Alina Surós Vicente.
Descripción: Para que la ejecución de esta prueba tenga resultados satisfactorios y pueda encriptar correctamente se debe proporcionar los datos a encriptar en un arreglo de byte.		
Entrada: byte[] data.		
Criterio de aceptación: Encripta datos.		
Resultado: 		

Tabla 31. Descripción de la prueba unitaria: Encrypt_Test.

Prueba unitaria: Decrypt _Test.

Prueba de Unidad		
Nombre Prueba: Decrypt_Test.		
Estado: Satisfactoria.	*Tipo:* Caja Blanca.	*Última Ejecución:* 13/05/2013
Ejecutado por: Daynelis Valdes Monrabal.		*Verificado por:* Alina Surós Vicente.
Descripción: Para que la ejecución de esta prueba tenga resultados satisfactorios y pueda desencriptar correctamente se debe proporcionar los datos a desencriptar en un arreglo de byte.		
Entrada: byte[] data.		
Criterio de aceptación:		

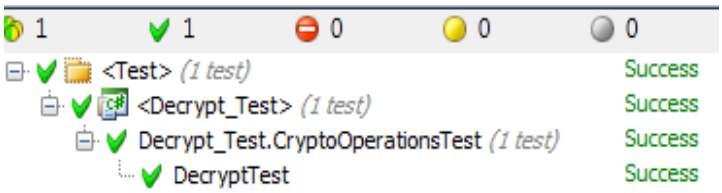
Resultado: 

Tabla 32. Descripción de la prueba unitaria: Decrypt_Test.

Prueba unitaria: ListLDAPCertificate_Test.

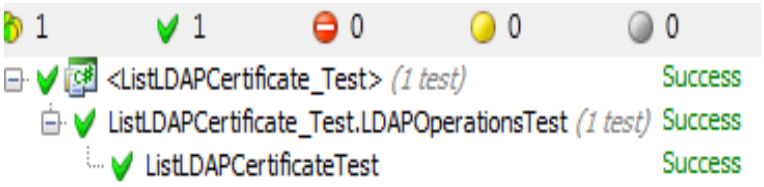
Prueba de Unidad		
Nombre Prueba: ListLDAPCertificate_Test.		
Estado: Satisfactoria.	*Tipo:* Caja Blanca.	*Última Ejecución:* 13/05/2013
Ejecutado por: Daynelis Valdes Monrabal.		*Verificado por:* Alina Surós Vicente.
Descripción: La correcta ejecución de esta prueba permitirá listar los certificados almacenados en el LDAP.		
Entrada:		
Criterio de aceptación: Lista los certificados.		
Resultado: 		

Tabla 33. Descripción de la prueba unitaria: ListLDAPCertificate_Test.

Prueba unitaria: AddLDAPCertificate_Test.

Prueba de Unidad		
Nombre Prueba: AddLDAPCertificate_Test.		
Estado: Satisfactoria.	*Tipo:* Caja Blanca.	*Última Ejecución:* 13/05/2013
Ejecutado por: Daynelis Valdes Monrabal.		*Verificado por:* Alina Surós Vicente.
Descripción: Para que la ejecución de esta prueba tenga resultados satisfactorios y pueda adicionar correctamente un certificado al LDAP se debe proporcionar los valores que componen el certificado.		

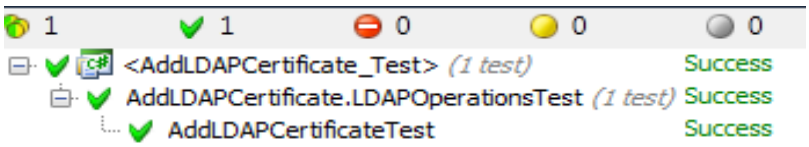
Entrada: string[] value.
Criterio de aceptación: Adiciona certificados.
Resultado: 

Tabla 34. Descripción de la prueba unitaria: AddLDAPCertificate_Test.

Prueba unitaria: DeleteLDAPCertificate_Test.

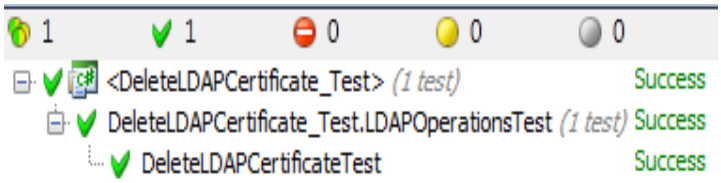
Prueba de Unidad		
Nombre Prueba: DeleteLDAPCertificate_Test.		
Estado: Satisfactoria.	*Tipo:* Caja Blanca.	*Última Ejecución:* 13/05/2013
Ejecutado por: Daynelis Valdes Monrabal.		*Verificado por:* Alina Surós Vicente.
Descripción: Para que la ejecución de esta prueba tenga resultados satisfactorios y pueda eliminar correctamente un certificado al LDAP se debe proporcionar el identificador del certificado.		
Entrada: string uid.		
Criterio de aceptación: Elimina certificados.		
Resultado: 		

Tabla 35. Descripción de la prueba unitaria: DeleteLDAPCertificate_Test.

Prueba unitaria: ModifyLDAPCertificate_Test.

Prueba de Unidad		
Nombre Prueba: ModifyLDAPCertificate_Test.		
Estado: Satisfactoria.	*Tipo:* Caja Blanca.	*Última Ejecución:* 13/05/2013

Anexo 5

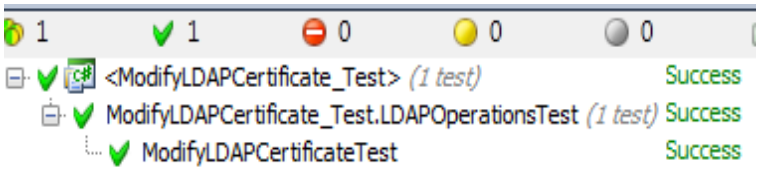
Ejecutado por: Daynelis Valdes Monrabal.	*Verificado por:* Alina Surós Vicente.
Descripción: Para que la ejecución de esta prueba tenga resultados satisfactorios y pueda modificar correctamente un certificado al LDAP se debe proporcionar los valores que se desean modificar del certificado.	
Entrada: Dictionary<object, object> dictionary.	
Criterio de aceptación: Modifica certificados.	
Resultado: 	

Tabla 36. Descripción de la prueba unitaria: ModifyLDAPCertificate_Test.

Anexo 6

Caso de prueba: Ejecutar el EAC-UE versión 1.

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	public_key	campo de texto	No	Admite dígitos hexadecimales.
2	private_key	campo de texto	No	Admite dígitos hexadecimales.

Tabla 37. Descripción de las variables: Ejecutar el EAC-UE versión 1.

Escenario	Descripción	public_key	private_key	Respuesta del sistema	Flujo central
EC 1: Ejecutar correctamente el EAC-UE versión 1.	Se procede a ejecutar correctamente el EAC-UE versión 1.	V	V	El sistema ejecuta correctamente el EAC-UE versión 1.	
		X: 6ea640632fc7dcadf69abd093908899f8de62639e8c5b0a9 Y: b03ec2b4ada8ff15e45cdad31109d94634c126a84f836d36	S:cddb426316d729944548a35880b3fe65c7c9926b5e92174b		
		I	I	El sistema muestra un mensaje de error.	
		5345mE$$	755/hjj-		
		I	V		
		#gfg@nghj	S:9c5d1e2f8617eb874dd42a6cf657e687e0e02abcaf5aa9c4		

Tabla 38. Caso de prueba: Ejecutar el EAC-UE versión 1.

Caso de prueba: Ejecutar el EAC-UE versión 2.

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	public_key	campo de texto	No	Admite dígitos hexadecimales.
2	private_key	campo de texto	No	Admite dígitos hexadecimales.

Tabla 39. Descripción de las variables: Ejecutar el EAC-UE versión 2.

Anexo 6

Escenario	Descripción	public_key	private_key	Respuesta del sistema	Flujo central
EC 4: Ejecutar correctamente el EAC-UE versión 1.	Se procede a ejecutar correctamente el EAC-UE versión 1.	V	V	El sistema ejecuta correctamente el EAC-UE versión 1.	
		X: 6ea640632fc7dcadf69abd093908899f8de62639e8c5b0a9 Y: b03ec2b4ada8ff15e45cdad31109d94634c126a84f836d36	S: cddb426316d729944548a35880b3fe65c7c9926b5e92174b		
		I	I	El sistema muestra un mensaje de error.	
		5345mE$$	755/hjj-		
		I	V		
		#gfg@nghj	S: 9c5d1e2f8617eb874dd42a6cf657e687e0e02abcaf5aa9c4		

Tabla 40. Caso de prueba: Ejecutar el EAC-UE versión 2.

Caso de prueba: Ejecutar el EAC-Singapur.

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	DG3	campo de texto	No	Admite dígitos hexadecimales.
2	key_encrypted	campo de texto	No	Admite dígitos hexadecimales.

Tabla 41. Descripción de las variables: Ejecutar el EAC Singapur.

Escenario	Descripción	key_encrypted	Respuesta del sistema	Flujo central
EC 7: Ejecutar correctamente el EAC Singapur. .	Se procede a ejecutar correctamente el EAC Singapur.	V	El sistema retorna la llave descifrada.	
		3b 23 03 dc 0c 3a 6a 89 75 d0 7f ee cd bc 85 8a eb 29 51 14 e6 2b 81 ff e6 07 30 a0 57 96 dc 79 9f 48 2f bc f9 9d f6 4f b4 ad e2 92 8 51 f4 d8 e0 54 6d 27 2b 41 38 4e 34 55 dc 99 98 a5 f4 11 66 4f a8		

		8a 6c 8d 03 43 95 e9 b8 5b 5a b4 64 e0 02 02 cd 82 c2 3b 1b 1f b4 e2 7f a9 16 2c 3b a2 fc 34 aa f6 c6 cf 00 58 a0 d7 cb 53 9d a7 7f 99 fa ee bf c8 44 a5 b6 3c f1 91 a4 85 75 30 43 be		
		I 589/+5m,	El sistema muestra un mensaje de error.	

Tabla 42. Caso de prueba: Ejecutar el EAC-Singapur.

Caso de prueba: Adicionar sistemas de inspección.

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	dir_ip	campo de texto	No	Admite dígitos entre 0 y 255, además permite (.). Ejemplo: 192.168.101.12.
2	dir_física	campo de texto	No	Admite dígitos y (-), con la siguiente estructura: 5 números-3 números-7 números-5números. Ejemplo: 00426-292-0000007-85527
3	serial	campo de texto	No	Admite dígitos hexadecimales.

Tabla 43. Descripción de las variables: Adicionar sistemas de inspección.

Escenario	Descripción	dir_ip	dir_física	serial	Respuesta del sistema	Flujo central
EC 8: Adicionar correctamente nuevos sistemas de inspección.	Se procede a adicionar nuevos sistemas de inspección	V 192.168.103.10	V 00426-292-0000007-85527	V F65C41AADE0B32183213BE66D3DB55542F6E89CD372A3E9A3028256E108A1F9DBC7CC76C9CD107671DFC917BF7FF54ED9C6A931BD3F1DD5372EEBF44477991A	El sistema adiciona nuevos sistemas de inspección correctamente.	Ir en el menú **Inicio** a: **Gestionar Sistemas de Inspección**. Ir a la opción: **Adicionar SI.** Llenar los datos. Oprimir el botón: **Guardar.**
		I hj.12.36.58	I 256-68-6598-35	V 42C0E62EBFB878E76089FA31FC10	El sistema muestra un mensaje de	Ir en el menú **Inicio** a: **Gestionar Sistemas de Inspección**.

				C790E02F498C8A 18CB0D20E61621 56A0A95374362A9 3AF412D830A48F 07B2939354E37B5 EDF76A1E9B815D 8C36EFB089E2DC	error.	Ir a la opción: **Adicionar SI.** Llenar los datos. Oprimir el botón: **Guardar.**
		V	I	I		
		192.168.103.29	25698-gfd-66985	/hgh5ghg		

Tabla 44. Caso de prueba: Adicionar sistemas de inspección.

Caso de prueba: Eliminar sistemas de inspección.

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	dir_física	campo de texto	No	Admite dígitos y (-), con la siguiente estructura: 5 números-3 números-7 números-5números. Ejemplo: 00426-292-0000007-85527

Tabla 45. Descripción de las variables: Eliminar sistemas de inspección.

Escenario	Descripción	dir_física	Respuesta del sistema	Flujo central
EC 9: Eliminar correctamente un sistema de inspección.	Se procede a eliminar sistemas de inspección.	V 00426-292-0000007-85527	El sistema elimina sistemas de inspección correctamente.	Ir en el menú **Inicio** a: **Gestionar Sistemas de Inspección**. Ir a la opción: **Eliminar SI.** Llenar dato. Oprimir el botón: **Eliminar.**
		I 2568-689-hfjfg-jhjgj	El sistema muestra un mensaje de error.	Ir en el menú **Inicio** a: **Gestionar Sistemas de Inspección**. Ir a la opción: **Eliminar SI** Llenar dato. Oprimir el botón: **Eliminar.**

Tabla 46. Caso de prueba: Eliminar sistemas de inspección.

Caso de prueba: Modificar sistemas de inspección.

Anexo 6

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	dir_ip	campo de texto	No	Admite dígitos entre 0 y 255, además permite (.). Ejemplo: 192.168.101.12.
2	dir_física	campo de texto	No	Admite dígitos y (-), con la siguiente estructura: 5 números-3 números-7 números-5números. Ejemplo: 00426-292-0000007-85527
3	serial	campo de texto	No	Admite dígitos hexadecimales.

Tabla 47. Descripción de las variables: Modificar sistemas de inspección.

Escenario	Descripción	dir_ip	dir_física	serial	Respuesta del sistema	Flujo central
EC 10: Modificar correctamente nuevos sistemas de inspección.	Se procede a modificar nuevos sistemas de inspección.	V	V	V	El sistema modifica correctamente los datos del sistema de inspección.	Ir en el menú **Inicio** a: **Gestionar Sistemas de Inspección**. Ir a la opción: **Modificar SI.** Cambiar datos. Oprimir el botón: **Modificar**.
		192.168.102.15	55274-640-0263172-23423	A5AF5A45CC04B1131C6BE44C47E304FE36A7F2ADD62888F4E908D4A6CBFCE666B18FAA222B297E267D9DD655EE2254DCA8CF06BD61A6C891C6F5B5FF6ECD9E4		
		I	I	V	El sistema muestra un mensaje de error.	Ir en el menú **Inicio** a: **Gestionar Sistemas de Inspección**. Ir a la opción: **Modificar SI.** Cambiar datos. Oprimir el botón: **Modificar**.
		hj.12.36.58	256-68-6598-35	25D39F249B8D863BBCC612948D0A55E5B0BDC98F79072D79CD092EF34C85E2A4DBA38C7DFEC903B9B3FAA2713C1349D601B1F85A768CB212AEDAFE0004C9A1E9		
		V	I	I		
		192.168.103.29	25698-gfd-66985	V*-kjhkkhkkk+-		

Tabla 48. Caso de prueba: Modificar sistema de inspección.

Caso de prueba: Listar sistemas de inspección.

Anexo 6

Escenario	Descripción	Respuesta del sistema	Flujo central
EC 11: Listar los sistemas de inspección.	Se procede a listar todos los sistemas de inspección.	El sistema muestra una lista con todos los sistemas de inspección.	Ir en el menú **Inicio** a: **Gestionar Sistemas de Inspección**. Ir a la opción: **Listar SI.**

Tabla 49. Caso de prueba: Listar sistema de inspección.

Caso de prueba: Adicionar certificado.

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	id	campo de texto	No	Admite caracteres alfanuméricos.
2	país	campo de texto	No	Admite letras. El nombre del país tiene que tener longitud dos.
3	nivel_PKI	combobox	No	Permite seleccionar el nivel de PKI, los cuales están predeterminados.
4	entrar_certificado	radiobutton	No	Permite seleccionar como será cargado el certificado. Esto puede ser cargándolo de una dirección o entrándolo manualmente.
5	certificado	campo de texto	No	Admite caracteres hexadecimales.

Tabla 50. Descripción de las variables: Adicionar certificado.

Escenario	Descripción	id	país	nivel_ PKI	entrar_ cer	certificado	Respuesta del sistema	Flujo central
EC 13: Adicionar correctamente un certificado.	Se procede a adicionar un certificado.	V	V	V	V	V	El sistema adiciona el certificado correctamente.	Ir en el menú **Inicio** a: **Gestionar Sistemas de Inspección**. Ir a la opción: **Adicionar Certificado**. Llenar los datos. Oprimir el botón: **Adicionar**.
		Test 1	CU	CVCA	Manual	7f2181bb7f4e81845f290100420b4355435643413030303031 7f493f060a04007f000702020202018631046 52c7152a454b34cd91c7cd760143a402584864823b3c683954980398c4a26770c90848f609b0d1385c40bd506ef1c975f200b43554356434130303030317f4c0e060904007f0007030102015301035f2		

						50601030004020 55f240601030007 02055f37 3068ca953f229d2 4e3e5448207f9ec d904c425ae5ef54 3cf22923095e2d0 9db6801aa7beda 509c651fabeed46 29af812c8		
		I	I	I	V	I	El sistema muestra un mensaje de error.	Ir en el menú **Inicio** a: **Gestionar Certificados.** Ir a la opción: **Adicionar Certificado.** Llenar los datos. Oprimir el botón: **Adicionar.**
			Cuba	256	Manual	#jdfkgjdkg		
		V	*I*	*I*	*V*	*I*		
		1	125	Sistema de inspección	Archivo	.fiynm		

Tabla 51. Caso de prueba: Adicionar certificado.

Caso de prueba: Modificar certificado.

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	id	campo de texto	No	Admite caracteres alfanuméricos.
2	país	campo de texto	No	Admite letras. El nombre del país tiene que tener longitud dos.
3	nivel_PKI	combobox	No	Permite seleccionar el nivel de PKI, los cuales están predeterminados.
4	entrar_certificado	radiobutton	No	Permite seleccionar como será cargado el certificado. Esto puede ser cargándolo de una dirección o entrándolo manualmente.
5	certificado	campo de texto	No	Admite caracteres hexadecimales.

Tabla 52. Descripción de las variables: Modificar certificado.

Escenario	Descripción	id	país	nivel PKI	certificado	Respuesta del sistema	Flujo central
EC 14: Modificar	Se procede a modificar un	V	V	V	V	El sistema modifica los	Ir en el menú **Inicio** a: **Gestionar Certificados.**
		VNCVC	VN	IS	7F218201857F4E8 201455F29010042		

correctamente un certificado.	certificado.	A01			0C4E4C43564341 4130303030317F4 981FD060A04007 F00070202020202 811CD7C134AA26 4366862A1830257 5D1D787B09F075 797DA89F57EC8C 0FF821C68A5E62 CA9CE6C1C2998 03A6C1530B514E 182AD8B0042A59 CAD29F43831C25 80F63CCFE44138 870713B1A92369E 33E2135D266DBB 372386C400B8439 040D9029AD2C7E 5CF4340823B2A87 DC68C9E4CE3174 C1E6EFDEE12C07 D58AA56F772C072 6F24C6B89E4ECD AC24354B9E99CA A3F6D3761402CD 851CD7C134AA26 4366862A18302575 D0FB98D116BC4B 6DDEBCA3A5A793 9F8639045877C68F B456ED75841DD74 627A36C004E08D45 50275471DFA38641 B3259A03DF1FDA8 FBAD8312FC16F4F 6AE9DB49F941A98 F020DC8F2F7F8701 015F200C4E4C4356 43414130303030327 F4C0E060904007F0 007030102015301C 15F2506000801010 2025F24060101010 102025F37389A8 0B23E3ECE3EC A96CA2F2EBB58 75895D9D1FA4E 1374BFDC1F5E BC711F9EAC68	datos del certificado seleccionado correctamente.	Ir a la opción: **Modificar Certificado**. Cambiar los datos. Oprimir el botón: **Modificar.**

					B18AE1CAA0C4D3D0C0C8644130AD385740C36AE26280D60		
		I	I	V	I	El sistema muestra un mensaje de error.	Ir en el menú **Inicio** a: **Gestionar Certificados.** Ir a la opción: **Modificar Certificado**. Cambiar los datos. Oprimir el botón: **Modificar.**
		,,,yyhh	12	CVCA	gh(hjj)		
		I	I	I	I		
		nfjg#ghgn	Colombia	Verificador1	*h;-hb;h		

Tabla 53. Caso de prueba: Modificar certificado.

Caso de prueba: Exportar certificado.

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	identificador	campo de texto	No	Admite caracteres alfanuméricos.

Tabla 54. Descripción de las variables: Exportar certificado.

Escenario	Descripción	identificador	Respuesta del sistema	Flujo central
EC 16: Exportar correctamente un certificado. .	Se procede a exportar un certificado.	V Test 1	El sistema exporta el certificado correctamente.	Ir en el menú **Inicio** a: **Gestionar Certificados.** Ir a la opción: **Exportar Certificado.** Seleccionar el identificador del certificado que se desea exportar. Oprimir el botón: **Exportar.**
		I $fghh	El sistema muestra un mensaje de error.	Ir en el menú **Inicio** a: **Gestionar Certificados.** Ir a la opción: **Exportar Certificado.** Seleccionar el identificador del certificado que se desea exportar. Oprimir el botón: **Exportar.**

Tabla 55. Caso de prueba: Exportar certificado.

Caso de prueba: Listar certificado.

Anexo 6

Escenario	Descripción	Respuesta del sistema	Flujo central
EC 17: Listar los certificado.	Se procede a listar todos los certificados.	El sistema muestra una lista con todos los certificados.	Ir en el menú **Inicio** a: **Gestionar Certificados.** Ir a la opción: **Listar Certificados.**

Tabla 56. Caso de prueba: Listar certificado.

Caso de prueba: Generar cadena de certificados.

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	país	campo de texto	No	Admite solo dos letras mayúsculas.

Tabla 57. Descripción de las variables: Generar cadena de certificado.

Escenario	Descripción	país	Respuesta del sistema	Flujo central
EC 18: Generar cadenas de certificados correctamente.	Se procede a generar una cadena de certificados.	V CU	El sistema genera cadenas de certificados correctamente.	Ir en el menú **Inicio** a: **Cadena de Certificados.** Ir a la opción: **Solicitar Cadena.** Llena el campo del país. Oprimir el botón: **Solicitar.**
		I hf787	El sistema muestra un mensaje de error.	Ir en el menú **Inicio** a: **Cadena de Certificados.** Ir a la opción: **Solicitar Cadena.** Llena el campo del país. Oprimir el botón: **Solicitar.**

Tabla 58. Caso de prueba: Generar cadena de certificado.

Caso de prueba: Enviar cadena de certificados.

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	país	campo de texto	No	Admite solo dos letras mayúsculas.

Tabla 59. Descripción de las variables: Enviar cadena de certificado.

Escenario	Descripción	país	Respuesta del sistema	Flujo central
EC 19: Enviar cadena de certificados	Se procede a enviar una cadena de certificados.	V FR	El sistema envía cadena de certificados correctamente.	Ir en el menú **Inicio** a: **Cadena de Certificados.** Ir a la opción: **Solicitar Cadena.**

correctamente.				Llena el campo del país. Oprimir el botón: **Solicitar.**
		I ·hf787	El sistema muestra un mensaje de error.	Ir en el menú **Inicio** a: **Cadena de Certificados.** Ir a la opción: **Solicitar Cadena.** Llena el campo del país. Oprimir el botón: **Solicitar.**

Tabla 60. Caso de prueba: Enviar cadena de certificado.

Caso de prueba: Generar pedidos de certificados.

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	SI	campo de texto	No	Admite caracteres alfanuméricos.
2	tipo_llave	combobox	No	Admite solo los valores: RSA y ECDSA.
3	log_llave	combobox	No	Admite solo números.
4	ref_CA	combobox	No	Admite caracteres alfanuméricos.
5	país_HR	campo de texto	No	Admite solo dos letras.
6	código	campo de texto	No	Admite caracteres alfanuméricos.
7	secuencia_HR	campo de texto	No	Admite solo cinco números.
8	Alg_firma	combobox	No	Admite caracteres alfanuméricos.

Tabla 61. Descripción de las variables: Generar pedido de certificado.

Escenario	Descripción	SI	tipo_llave	log_llave	ref_CA	país_HR	código	secuencia_HR	Alg_firma	Respuesta del sistema	Flujo central
EC 20: Generar pedidos de	Se procede a generar pedidos de	V 192.168.	V RSA	V 1024	V Rafael	V CU	V hol	V 00002	V SHA 256	El sistema crea un pedido de certificado	Ir en el menú **Pedido** a: **Crear Pedido.** Llenar los

certificado.	certificado.	103.10					a		WITHRSA	correctamente.	datos. Oprimir el botón: **Generar.**
		I	I	I	I	I	I	I	I	El sistema muestra un mensaje de error.	Ir en el menú **Pedido** a: **Crear Pedido.** Llenar los datos. Oprimir el botón: **Generar.**
		41m/j	ghj	1f@g	n;j-ggj	235/jhj	(kkllk)	15-65l	$2;';		
		V	I	I	I	V	I	I	I		
		Se2+9	RSA	b23+	*h12/	CU	2fef	^^jfkj	)jkj+		

Tabla 62. Caso de prueba: Generar pedido de certificado.

Caso de prueba: Generar el par de llaves.

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	longitud	combobox	No	Admite solo números.
2	contraseña	campo de texto	No	Admite cualquier carácter.
3	confir_contraseña	campo de texto	No	Admite cualquier carácter.

Tabla 63. Descripción de las variables: Generar par de llaves.

Escenario	Descripción	longitud	contraseña	confir_contraseña	Respuesta del sistema	Flujo central
EC 21: Generar par de llaves correctamente.	Se procede a generar par de llaves.	V 1024	V Contra145	V Contra145	El sistema generar par de llaves.	Ir en el menú **Inicio** a: **Llaves RSA.** Ir a la opción: **Crear Llaves.** Llenar los datos. Oprimir el botón:

						Generar.
		I	V	V	El sistema muestra un mensaje de error.	Ir en el menú **Inicio** a: **Llaves RSA.** Ir a la opción: **Crear Llaves.** Llenar los datos. Oprimir el botón: **Generar.**
		ghhh	8*gh-hhgh/*fg	8*gh-hhgh/*fg		
		I	V	*V*		
		123xcvft	-lk+jkg+jj	-lk+jkg+jj		

Tabla 64. Caso de prueba: Generar par de llaves.

Anexo 7

No conformidades detectadas en la realización de las pruebas de caja negra:

Iteración	No	Tipo de NC	Descripción	Impacto	Identificada por	Fecha	Producto
1	1	Adherencia a Producto.	Al aplicarse el CPR Adicionar Certificado, el sistema no valida correctamente la entrada de datos.	Alto.	Daynelis Valdes Monrabal.	06/05/2013	Sistema
1	2	Adherencia a Producto.	Al aplicarse el CPR Eliminar Certificado, el sistema muestra automáticamente un identificador.	Medio.	Daynelis Valdes Monrabal.	06/05/2013	Sistema.
1	3	Adherencia a Producto.	Al aplicarse el CPR Modificar Certificado, el sistema no valida correctamente la entrada de datos.	Alto.	Daynelis Valdes Monrabal.	06/05/2013	Sistema.
1	4	Adherencia a Producto.	Al aplicarse el CPR Exportar Certificado, se detectó que el nombre del menú está incorrecto cambiar Mostrar Certificado por Exportar Certificado.	Medio.	Daynelis Valdes Monrabal.	06/05/2013	Sistema.
1	5	Adherencia a Producto.	Al aplicarse el CPR Exportar Certificado, el sistema permite exportar con datos vacíos.	Alto.	Daynelis Valdes Monrabal.	06/05/2013	Sistema.

Anexo 7

1	6	Adherencia a Producto.	Al aplicarse el CPR Eliminar sistema de inspección, el sistema no muestra mensaje de error cuando el identificador es incorrecto.	Medio.	Daynelis Valdes Monrabal.	06/05/2013	Sistema.
2	7	Adherencia a Producto.	Al aplicarse el CPR Adicionar nuevos sistemas de inspección, el sistema permite entrar datos vacíos.	Alto.	Daynelis Valdes Monrabal.	10/05/2013	Sistema.
2	8	Adherencia a Producto.	Al aplicarse el CPR Modificar sistemas de inspección, el sistema no modifica los datos.	Alto.	Daynelis Valdes Monrabal.	10/05/2013	Sistema.
2	9	Adherencia a Producto.	Al aplicarse el CPR Modificar sistemas de inspección, el sistema no valida correctamente la entrada de datos.	Alto.	Daynelis Valdes Monrabal.	10/05/2013	Sistema.
2	10	Adherencia a Producto.	Al aplicarse el CPR Eliminar sistemas de inspección, el botón Cancelar no funciona.	Medio.	Daynelis Valdes Monrabal.	10/05/2013	Sistema.

Tabla 65. No conformidades identificadas en las pruebas de caja negra.