Módulo de Integración de la Infraestructura de Clave Pública con el Sistema de Administración de Identidades del Ministerio del Interior.

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

Autores*:* Norbert Said Martinez Machado
Hector Luis Pérez Sánchez

Tutor*:* Ing. Felix Alejandro Prieto Carratala

Ciudad de La Habana, Cuba
Junio de 2010.

DECLARACIÓN DE AUTORÍA

Declaramos que somos los únicos autores del trabajo titulado:
"Módulo de Integración de la Infraestructura de Clave Pública con el Sistema de Administración de Identidades del Ministerio del Interior" y autorizamos a la Universidad de las Ciencias Informáticas los derechos patrimoniales con carácter exclusivo.

Para que así conste firmamos la presente a los ___ días del mes de _____ del año _____.

_____ _____
Norbert Said Martinez Machado Hector Luis Pérez Sánchez

_____
Felix Alejandro Prieto Carratala

OPINIÓN DEL TUTOR DEL TRABAJO DE DIPLOMA

AGRADECIMIENTOS

Norbert:

Hector:

DEDICATORIA

Norbert:

Hector:

RESUMEN

Como parte de las transformaciones que se están realizando para la automatización de los procesos en el Ministerio del Interior (MININT) y de esta manera dar muestra fehaciente del compromiso que esta prestigiosa entidad tiene con el pueblo, surge el Sistema de Administración de Identidades (SAI). El cual tiene como objetivo principal manejar los datos de seguridad (usuarios, contraseñas, roles de usuarios, recursos y permisos sobre los recursos) necesarios para el funcionamiento de un conjunto de aplicaciones de manera individual. El SAI está compuesto por diferentes módulos dentro de los cuales se encuentra el módulo de Integración PKI el cual tiene como objetivo principal integrar la Infraestructura de Clave Pública (PKI) con el SAI lo que permitirá crear, asignar y publicar certificados digitales.

En el presente trabajo se realizó un estudio de las diferentes soluciones existentes en el mercado, así como las herramientas disponibles para implementar el sistema que permita la integración de la PKI con el SAI. Se identificaron y describieron todas las funcionalidades requeridas. Se planteó la propuesta de solución, se definió la arquitectura y se realizaron las pruebas para comprobar la validez de la solución.

PALABRAS CLAVES:

MININT: Ministerio del Interior.

SAI: Sistema de Administración de Identidades.

PKI: Infraestructura de Clave Pública.

ÍNDICE DEL CONTENIDO

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

INTRODUCCIÓN

La automatización de los procesos que en el *Ministerio del Interior* (MININT) se están realizando es muestra fehaciente de las transformaciones que esta institución está llevando a cabo, para conservar el alto sentido revolucionario de esta prestigiosa entidad, y de esta manera dar fe del compromiso con la visión para lo que la misma fue creada, proteger y servir con eficiencia a la población.

Para comenzar el nuevo proceso de cambios era necesario dar los primeros pasos, y romper con los antiguos modos de trabajo. Los formatos duros fueron sustituidos por sus homólogos los digitales, y la antigua máquina de escribir por la eficiente y mucho más cómoda computadora personal, y de este modo se puso todo el desarrollo tecnológico que en ese entonces ocurría en función de la automatización de los procesos que antes eran engorrosos y para nada rápidos. Era el advenimiento de las *Tecnologías de la Información y la Comunicación* (TIC).

No eran pocas las ventajas que las TICs proporcionaba, pero dentro de ese mar de ventajas había peligros hasta entonces desconocidos, que tenían como objetivo el penetrar espacios restringidos, y lograr acceso a recursos no permitidos de dicha entidad. De esta manera, se debía poner fin a estas prácticas y así evitar que estas pudieran poner en tela de juicio los paradigmas que defiende este ministerio.

Así conceptos como Integridad, Confidencialidad y Seguridad, llegaron para ser propiedades fundamentales desde entonces en los procesos que dentro del MININT se han estado llevando a cabo. Ya estos procesos no podían simplemente ejecutarse, sino que cada uno llevaba consigo datos relevantes que determinaban quien podía ejecutarlos, y como lo haría, y así mantener un control total de los recursos que estos manejaban.

Así surge el *Sistema de Administración de Identidades* (SAI) desarrollado por el polo productivo de la Universidad de las Ciencias Informáticas, *Centro de Identificación y Seguridad Digital* (CISED) cuya función principal es la de manejar los datos de seguridad (usuarios, contraseñas, roles de usuarios, recursos y permisos sobre los recursos) necesarios para el funcionamiento de un conjunto de aplicaciones de manera individual.

Por otro lado, era necesario el dotar a cada usuario, o entidad del MININT que el SAI administraba de un certificado digital, siempre y cuando el usuario requiera de este certificado. Un certificado digital no es más que un documento digital con el cual un tercero confiable garantiza la vinculación entre una

entidad y su clave pública, este tercero no es más que una autoridad certificadora. Es por ello surge la necesidad de utilizar una *Infraestructura de Clave Pública* (PKI), la cual permitirá entre otras cosas asignar dichos certificados digitales a los usuarios o entidades del MININT que lo requieran.

Partiendo de lo planteado anteriormente surge la siguiente ***situación problémica***, para la cual es necesario dar una solución: el certificado digital es el documento que identifica a un usuario o entidad, el cual es emitido por la PKI del MININT, estos certificados son solicitados, publicados y asignados a los usuarios de forma manual lo que trae consigo demoras e ineficiencia en la solicitud, publicación, así como en la asignación de dichos certificados solicitados. Es por ello que se hace necesario garantizar la integración de la PKI con el SAI del MININT para poder solicitar, publicar y asignar dichos certificados digitales emitidos a estos usuarios y entidades de forma automatizada. Permitiendo llevar a cabo todos los procesos vinculados a la ejecución de dicha integración con calidad y eficiencia acorde con las intensiones de desarrollo tecnológico del MININT.

De lo planteado anteriormente surge la siguiente interrogante la cual define el ***problema científico:*** ¿Cómo lograr la integración de la infraestructura de Clave Pública con el Sistema de Administración de Identidades del Ministerio del Interior?

Tomando como ***objeto de estudio:*** los procesos asociados a la integración de la PKI con el SAI, donde el ***campo de acción*** estará enmarcado en la solicitud, publicación y asignación de certificados digitales emitidos por la PKI a usuarios o entidades del SAI del MININT.

Para dar solución al problema existente se plantea como ***objetivo general:*** implementar un sistema que permita la integración de la PKI con el SAI del MININT. Para cumplir con el objetivo general se derivan los siguientes ***objetivos específicos:***

- ✓ Definir los requerimientos funcionales solicitados por el cliente con respecto a la gestión de los certificados digitales en el SAI.
- ✓ Evaluar y proponer las herramientas a utilizar en el análisis, diseño e implementación del sistema.
- ✓ Documentar la propuesta de descripción del sistema.
- ✓ Seleccionar la Metodología de Desarrollo a utilizar durante la confección del sistema.
- ✓ Realizar el análisis y diseño de un módulo automatizado para integrar la PKI con el SAI.
- ✓ Implementar el sistema.

Para dar solución a la interrogante planteada en el problema científico, la investigación se sustenta en la siguiente ***Hipótesis:***

La integración de la Infraestructura de Clave Pública con el Sistema de Administración de Identidades del Ministerio del Interior logrará la gestión automatizada de las solicitudes de publicación y asignación de certificados digitales a usuarios o entidades del Sistema de Administración de Identidades.

De la hipótesis antes mencionada se pueden plantear las siguientes ***variables de la investigación:***

Variable independiente: integración de la PKI con el SAI.

*Variable dependiente**:*** gestión de las solicitudes de publicación de certificados digitales y asignación de certificados digitales emitidos a usuarios o entidades del SAI.

Operacionalización de las variables

Permitir la integración de la PKI con el SAI.	Factibilidad	Automaticidad	Alta Mediana Poca
		Seguridad	Alta Mediana Poca
		Tiempo de Ejecución	Rápido Medio Lento
Permitir la gestión de las solicitudes de publicación de certificados digitales.	Factibilidad	Automaticidad	Alta Mediana Poca
		Seguridad	Alta Mediana Poca
		Tiempo de Ejecución	Rápido Medio Lento
Permitir la asignación de certificados digitales emitidos a usuarios o entidades.	Factibilidad	Automaticidad	Alta Mediana Poca
		Seguridad	Alta Mediana Poca
		Tiempo de Ejecución	Rápido Medio Lento

Tabla 1. Operacionalización de las variables

Para dar cumplimiento al trabajo se proponen las siguientes ***tareas de la investigación:***

- ✓ Investigación sobre la arquitectura de seguridad PKI y los estándares que esta define.
- ✓ Investigación sobre la librería criptográfica BouncyCastle C# para el trabajo con criptografía de clave pública, y generación de peticiones de certificación PKCS#10.
- ✓ Realización de un estudio del funcionamiento del servicio de directorio *Lightweight Directory Access Protocol* (LDAP).
- ✓ Investigación sobre la *Application Programming Interface* (API) Novell para el manejo de directorios LDAP.
- ✓ Diseño e implementación de un sistema que permita la integración de la PKI con el SAI.

Métodos de Investigación Científica

Entre los métodos teóricos que se emplearán para nuestra investigación está:

- ✓ ***Analítico-Sintético:*** Se estudiarán las diferentes soluciones que setonraf uu(i)22(c)-32(a)21(d)223(a)

- ✓ ***Observación científica:*** Se utilizará para analizar el curso natural del problema y del objeto de estudio con el fin de extraer mediante la observación las principales características identificativas y fenoménicas.

El presente documento está compuesto de tres capítulos:

En el primer Capítulo: Fundamentación Teórica, este capítulo contiene una base teórica, su objetivo es el de la compresión del problema planteado. Aquí están descritos conceptos, y definiciones con el fin de tener un dominio parcial, además de hacer referencia a las tendencias y tecnologías actuales que se usaron. En el segundo Capítulo: Características de la Solución, se presenta la descripción de la propuesta de solución, se definen las personas y se identifican y describen los escenarios y los requisitos de calidad de servicio. En el tercer Capítulo: Desarrollo, se describe la arquitectura y los patrones utilizados para la implementación del sistema, así como los diferentes diagramas utilizados para presentar la información. Además, se realizan las pruebas unitarias y de caja blanca para comprobar la validez del sistema elaborado.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

1.1 Introducción

Desde los inicios de la criptografía han surgido muchos métodos para lograr transmitir de forma confiable y lo más segura posible información importante. Antes de de existir las computadoras los mensajes se transmitían cara a cara a través de una persona que era considerada confiable, con el desarrollo en la forma de pensar del hombre fueron surgiendo sistemas criptográficos más eficientes basados en claves que solo tenían conocimiento de ellas, las personas que escribían el mensaje y las que lo recibían.

Con el desarrollo de la computación surgieron los algoritmos asimétricos o de clave pública los cuales aumentaron la seguridad en el intercambio de información, entre ellos se destaca el intercambio de claves de Diffie-Hellman en el año 1976, el cual fue el primer sistema asimétrico de clave pública en el cual se intercambiaron secretos a través de una clave de autenticación.

Estos algoritmos asimétricos fueron los antecedentes que permitieron la creación de una infraestructura robusta y confiable la cual se denominó Infraestructura de Clave Pública, la cual fue creada para los usuarios de internet y sitios que deseen comunicaciones seguras y confiables.

1.2 Antecedentes

- ✓ ***VeriSign Managed PKI:*** Es la primera plataforma totalmente integrada de infraestructura de Clave Pública, diseñada para asegurar un conjunto de aplicaciones sobre Intranets, Extranets, redes privadas virtuales y comercio electrónico, posibilitando la máxima flexibilidad, rendimiento, escalabilidad, disponibilidad y seguridad.

Creado por la compañía VeriSign, permite establecer una robusta infraestructura de Clave Pública y una Autoridad Certificante de forma rápida y a bajo coste, permite control total sobre la política de seguridad, la jerarquía de la infraestructura, los modelos de autenticación y el manejo integral del ciclo de vida de los certificados digitales.

- ✓ ***RSA Certificate Manager:*** Creado por la compañía RSA Security, administra y valida certificados digitales. Está construido para la administración. Presenta un potente motor para firmar digitalmente los certificados de usuario final y los eventos del sistema, así como un repositorio de datos integrado para el almacenamiento de los certificados, los datos del sistema y el certificado de información de estado. Ayuda a las organizaciones a definir la libre administración de sus propios procedimientos de seguridad, los formatos de certificado y reglas para el ciclo de vida de un certificado.

- ✓ ***Comodo Certificate Manager:*** Creado por la compañía Comodo, el Administrador de certificados reduce el tiempo de gestión, gastos jurídicos, el desarrollo y las operaciones necesarias para la seguridad y la administración de la PKI. Además, permite la gestión centralizada de certificados digitales, reduce los costos y mejora la seguridad.

- ✓ ***Managed Services PKI:*** Creado por la compañía Entrust, permite establecer y mantener un ambiente de confianza mediante certificados, la firma digital y la autenticación de certificados. Esta solución le permiten controlar el acceso a los recursos, evitar el robo de información y cumplir con las regulaciones de privacidad y la firma digital.

1.3 Conceptos Fundamentales Asociados al Dominio del Problema

1.3.1 Criptografía Asimétrica o de Clave Pública

"A diferencia de los algoritmos de cifrado simétrico, en los que la información se cifra y descifra con la misma clave, los algoritmos asimétricos basan su funcionamiento en un par de claves (matemáticas dependientes) para cada usuario, con la característica de que la información cifrada con una clave, sólo puede descifrarse con la otra del mismo par." [1]

A cada usuario se le asigna un par de claves, una pública y otra privada. La Clave Pública puede ser conocida por cualquiera, mientras que la Clave Privada solo debe ser conocida por el usuario y nunca hacerse pública pues de lo contrario comprometería su seguridad.

Estos algoritmos asimétricos se usan para encriptar datos por lo que han sido diseñados con el objetivo de poder intercambiar información de manera segura sin necesidad de conocer una clave secreta de cifrado acordada anteriormente.

Dentro de los algoritmos asimétricos más utilizados encontramos el algoritmo RSA. Donde sus principales funciones lo podemos encontrar en canales de comunicación segura, con el uso de *Secure Socket Layer* (SSL).

1.3.2 Infraestructura de Clave pública (PKI)

"La PKI es un protocolo que trata de describir los procesos organizativos necesarios para la gestión de certificados digitales de claves públicas para el intercambio seguro de información, que permite firmar digitalmente un documento electrónico (un email, el código de un programa, una transacción bancaria, unos análisis médicos, etc., etc., etc.), o permite identificar a una persona o empresa en Internet, o permite acceder a un recinto o servicio restringido. Los usos son innumerables. Las PKIs, son sistemas mixtos hardware/software, basados en diferentes agentes que permiten dotar a máquinas y usuarios de Certificados Digitales de Identidad (certificados X509v3)." [2]

Los objetivos que persigue una PKI son los siguientes:

- ✓ **Autentificación de usuarios:** Asegura la identidad de un usuario, al acceder a servicios distribuidos en red, ya que sólo el usuario conoce su clave privada, evitando así la suplantación de identidad.
- ✓ **No repudio:** Impide que una vez firmado un documento el usuario niegue haberlo escrito.
- ✓ **Integridad de la información:** Previene la modificación malintencionada o accidental de los datos firmados, durante su transporte, almacenamiento o manipulación.
- ✓ **Acuerdo de claves secretas:** Garantiza la confidencialidad de la información intercambiada.

1.3.3 Componentes de una PKI

PKI incluirá una o varias autoridades de registro para certificar la identidad de los usuarios; una o varias autoridades de certificación que emitan los certificados de clave pública; un repositorio de certificados, accesible vía web u otro medio, donde se almacenen los certificados y las listas de

revocación de certificados, donde se listan los certificados suspendidos o revocados y por último los usuarios o entidades finales que son los que utilizan los servicios que ofrece la PKI.

Figura 1. Esquema de los componentes de la PKI

1.3.3.1 Autoridad Certificadora (AC)

"Es la entidad que asegura la identidad de los usuarios de los certificados digitales. Posee su propio par de claves y firma digitalmente los certificados con su clave privada. Confiando en la Firma Digital de la Autoridad Certificadora, puede confiarse en cualquier certificado generado por la misma." [3]

Entre las tareas realizadas por la Autoridad Certificadora se encuentran:

- ✓ Procesar peticiones de Certificado a través de la Autoridad de Registro. Estas solicitudes están compuestas básicamente por los datos identificativos y la clave pública del solicitante.
- ✓ Generar los Certificados y almacenarlos en el repositorio público (ejemplo: LDAP o una Base de Datos).
- ✓ Gestionar la caducidad y renovación de certificados.
- ✓ Gestionar la revocación de certificados.

Toda la veracidad de la Autoridad de Certificación se basa en la inviolabilidad de su propia clave privada, por lo cual proteger dicha clave es de vital importancia. En la aplicación se utilizará como autoridad certificadora de prueba, EJBCA, a continuación de describen las principales características.

EJBCA es definida como una autoridad certificadora multifuncional basada en la tecnología Java, constituye una autoridad certificadora robusta, escalable, de alto rendimiento y basada en componentes. Además, dispone de una herramienta de administración vía web. Ofrece un conjunto de funcionalidades disponibles a través de servicios web.

1.3.3.2 Autoridad de Registro (AR)

"Es la encargada de establecer los mecanismos para que los usuarios soliciten su propio certificado, de tal forma que se asegure la identidad de dicho usuario. A este procedimiento se le denomina "Proceso de Registro" y se realiza a través de la denominada "Autoridad de Registro"." [4]

Existen dos formas principales de registro:

- ✓ **Registro Clásico:** La persona solicita a una oficina de registro un certificado digital, esta luego de acreditar su identidad le proporciona de forma segura su clave privada y su certificado digital.

- ✓ **Registro Remoto:** La persona realiza una solicitud de certificado digital a través de Internet. Para ello empleará un navegador que generará el par de claves y enviará su clave pública a la Autoridad de Registro la cual será firmada por la Autoridad Certificadora y le será devuelto su certificado digital.

1.3.3.3 Lista de Revocación de Certificados (CRL)

La Lista de Revocación de Certificados no es más que una lista de certificados (más concretamente sus números de serie), los cuales han sido revocados o derogados, por lo tanto, ya no son válidos y en los que ninguna autoridad certificadora o usuario debe confiar.

Poseen dos estados de revocación:

- ✓ ***Revoked:*** Este estado es irreversible, un certificado es revocado cuando se descubre que la autoridad de certificación ha expedido indebidamente un certificado, o si una clave privada se cree que ha sido comprometida.

- ✓ ***Hold:*** Este estado reversible puede ser usado para tomar nota de la incapacidad temporal del certificado (ejemplo: si el usuario no está seguro si la clave privada se ha perdido).

1.3.3.4 Repositorios de Certificados

"Es el componente encargado de hacer disponibles las claves públicas de las identidades registradas antes de que puedan utilizar sus certificados. Suelen ser repositorios X.500 o LDAP. Cuando el usuario necesita validar un certificado debe consultar el repositorio de certificados para verificar la firma del firmante del certificado, garantizar la vigencia del certificado comprobando su periodo de validez y que no ha sido revocado por la CA y que además cumple con los requisitos para los que se expidió el certificado; por ejemplo, que el certificado sirve para firmar correo electrónico." [5]

Además, los repositorios son las estructuras encargadas de almacenar la información relativa a la PKI, entiéndase los certificados generados y la Lista de Revocación de Certificados. Es por ello que en una PKI los dos repositorios más importantes son el repositorio de certificados y el repositorio de Listas de Revocación de Certificados.

1.3.4 Certificados Digitales

Los certificados digitales son emitidos por una autoridad de certificación y poseen diferentes datos para poder identificar al titular. Entre los datos se destacan, el nombre del titular (que puede ser una empresa o una persona), un número de serie, la fecha de expiración del certificado y una clave pública que permite encriptar la información. El estándar más utilizado para estos certificados es el X.509v3.

Existen diferentes tipos de certificados digitales, a continuación se presentan los que más se acoplan a nuestra investigación.

"Desde el punto de vista de la finalidad, los certificados electrónicos se dividen en:

- ✓ ***Certificados SSL para cliente:*** *usados para identificar y autentificar a clientes ante servidores en comunicaciones mediante el protocolo Secure Socket Layer, y se expiden normalmente a una persona física, bien un particular, bien un empleado de una empresa.*

- ✓ ***Certificados SSL para servidor:*** *usados para identificar a un servidor ante un cliente en comunicaciones mediante el protocolo Secure Socket Layer, y se expiden generalmente a nombre de la empresa propietaria del servidor seguro o del servicio que éste va a ofrecer, vinculando también el dominio por el que se debe acceder al servidor. La presencia de este certificado es condición imprescindible para establecer comunicaciones seguras SSL.*

- ✓ ***Certificados para AC:*** *que identifican a las propias Autoridades Certificadoras, y es usado por el software cliente para determinar si pueden confiar en un certificado cualquiera, accediendo al certificado de la AC y comprobando que ésta es de confianza."* [6]

1.3.4.1 Certificado Digital X509v3

El certificado X.509v3 es un estándar de la *Unión Internacional de telecomunicaciones* (UIT) para infraestructuras de clave pública donde se especifican formatos de atributos estándar para certificados de claves públicas y un algoritmo de validación de la ruta de certificación. Es la versión más actualizada de los certificados X.509.

En el sistema X.509v3, una autoridad certificadora emite un certificado asociando una clave pública a un Nombre Distinguido particular o a un Nombre Alternativo tal como una dirección de correo electrónico o una entrada de DNS (Sistema de Nombre de Dominios). El certificado X.509v3 es la pieza central de la infraestructura de clave pública y es la estructura de datos que enlaza la clave pública con los datos que permiten identificar al titular.

Los formatos de codificación más comunes son *Distinguished Encoding Rules* (DER) o *Privacy-enhanced Electronic Mail* (PEM). Se definen un conjunto de atributos los cuales nos brindan la posibilidad de estandarizar estos atributos a las necesidades de los usuarios.

Entre las extensiones de archivo de certificados X.509v3 se encuentran:

- ✓ ***.DER*** - Certificado codificado en DER.
- ✓ ***.PEM*** - Certificado codificado en Base64.
- ✓ ***.P12*** – PCKC#12, puede contener certificados públicos y claves privadas protegidas con clave.

EL certificado digital que usará la aplicación sigue el estándar X509v3, el cual es utilizado por los navegadores y los datos que figuran generalmente en un certificado son:

- ✓ ***Versión:*** versión del estándar X.509, generalmente la 3, que es la más actual.
- ✓ ***Número de serie:*** número identificador del certificado, único para cada certificado expedido por una autoridad certificadora determinada.
- ✓ ***Algoritmo de firma:*** algoritmo criptográfico usado para la firma digital.
- ✓ ***Autoridad Certificadora:*** datos sobre la autoridad que expide el certificado.
- ✓ ***Fechas de inicio y de fin de validez del certificado****:* Definen el tiempo de validez del mismo, que generalmente es de un año.
- ✓ ***Propietario:*** persona o entidad vinculada al certificado. Dentro de este apartado se usan una serie de abreviaturas para establecer datos de identidad. Entre ellas:

    - ➢ CN (nombre común del usuario), O (organización), L (ciudad), S (estado o provincia), C (país), E (correo electrónico), UID (ID de usuario).

- ✓ ***Clave pública:*** representación de la clave pública vinculada a la persona o entidad (en hexadecimal), junto con el algoritmo criptográfico para el que es aplicable.
- ✓ **Firma de la Autoridad Certificadora**: asegura la autenticidad del mismo.

1.3.5 Sistema de Administración de Identidades (SAI)

El SAI es el encargo de manejar los datos de seguridad (usuarios, contraseñas, roles de usuarios, recursos y permisos sobre los recursos) necesarios para el funcionamiento de un conjunto de aplicaciones. Es el encargado de asignar a cada usuario, o entidad del MININT que el SAI administra, de un certificado digital, siempre y cuando el usuario requiera de este certificado.

1.4 Necesidad de la Implementación de un sistema para la integración de la PKI con el SAI

La integración de la PKI al SAI es de vital importancia para la culminación de los procesos de automatización que se están realizando en el MININT. La tecnología PKI tiene muchas ventajas lo que la hace hoy en día imprescindible si se quiere obtener un nivel de seguridad elevado. Entre las ventajas pueden mencionarse autentificación de usuarios, el no repudio, la integridad y confidencialidad de la información y todo ello se logra a través de certificados digitales, los cuales serán asignados a cada usuario del sistema que requiera dicho certificado digital, con el cual se vincula la identidad de dicho usuario con su clave, permitiendo lograr todo lo planteado anteriormente y mantener un control eficiente sobre cada usuario.

1.5 Tendencias Tecnologías

1.5.1 Estándares utilizados por la PKI

1.5.1.1 Estándar Criptográfico de Clave Pública (PKCS)

"Se refiere a un grupo de estándares de criptografía de clave pública producidos por los laboratorios de RSA en cooperación con los desarrolladores de sistemas de seguridad en todo el mundo con el fin de acelerar el despliegue de la criptografía de clave pública. Publicado por primera vez en 1991 como resultado de las reuniones con un pequeño grupo de los primeros adoptantes de tecnologías de clave pública, los documentos PKCS se han convertido en referencia y ampliamente aplicados." [7]
Los estándares PKCS van desde el número 1 hasta el 15, a continuación un resumen de los utilizados en el desarrollo de nuestro sistema.

- ✓ ***PKCS 10:*** Esta norma define el formato de los mensajes enviados a una Autoridad de Certificación para solicitar la certificación de una clave pública. Esta norma describe la sintaxis de los pedidos de certificados los cuales consisten en un nombre, una clave pública y opcionalmente un conjunto de atributos, todos firmados por una entidad de pedidos de certificados. Estos pedidos son enviados a una Autoridad Certificadora la cual transforma el pedido en un certificado X509v3.

- ✓ ***PKCS12:*** *"Este estándar especifica un formato portable para almacenar y transportar certificados, claves privadas y secretos varios."* [8]. Es el formato preferido por muchos para realizar operaciones de gestión de certificados y es soportado por la mayoría de los navegadores. Tiene la ventaja de que es capaz de almacenar el certificado con su correspondiente clave, con el certificado raíz de la autoridad certificadora y otros certificados posibles de la cadena en un único fichero.

1.5.1.2 Petición de Comentarios (RFC)

Las Petición de Comentarios o (*Request For Comments, por sus siglas en inglés)* son una serie de notas las cuales se abrevian como RFC. En estos documentos se recogen los estándares de TCP/IP, el cual es el protocolo bajo la cual funciona Internet. Cada RFC tiene un título y un número asignado, que no puede repetirse ni eliminarse aunque el documento quede obsoleto.

A continuación se describen algunos de los RFC más importantes utilizados en la investigación:

- ✓ **RFC 4513 – LDAP: Métodos de Autenticación y Mecanismos de Seguridad**

  - ➢ Este documento describe los métodos de autenticación y los mecanismos de seguridad de la Lightweight Directory Access Protocol (LDAP).
  - ➢ Este documento analiza los estados de autenticación y autorización en una sesión a un servidor LDAP.
  - ➢ Este documento describe los diferentes métodos de autenticación existentes.

- ✓ **RFC 4519 – LDAP: Esquemas para Aplicaciones de Usuarios**

  - ➢ Este documento es una especificación técnica del Lightweight Directory Access Protocol (LDAP).
  - ➢ Proporciona una especificación técnica de los tipos de atributos y clases de objetos destinados a usarse por los clientes del directorio LDAP para muchos servicios del directorio.

- Este documento no cubre los atributos utilizados para la administración de servidores de directorio, ni tampoco incluye objetos de directorio definido para usos específicos en otros documentos.

✓ **RFC 4523 – LDAP: Definiciones de esquemas para Certificados X.509**

- Este documento describe el esquema de representación de los certificados X.509 y los elementos relacionados en los directorios accesibles mediante el Lightweight Directory Access Protocol (LDAP).
- Describe los tipos de atributos de los certificados, los pares de certificados y de las listas de revocación de certificados.
- Describe las definiciones de sintaxis de LDAP, la afirmación de asociados y los valores de atributo.

1.5.2 Librería Criptográfica BouncyCastle C#

Es una librería criptográfica desarrollada por el proyecto Software Libre, la cual ofrece una API que contiene un conjunto de clases que permiten la Generación de claves (claves secretas y pares de claves pública y privada), el Cifrado simétrico, el Cifrado con curva elíptica, el Cifrado asimétrico, Funciones de resumen y Acuerdo de claves, las cuales permiten trabajar con los algoritmos de cifrado más usados en la actualidad.

1.5.3 Interfaces para Aplicaciones de Programas (API) Novell para LDAP.

Es una API creada por la compañía Novell para acceder, manejar y actualizar directorios LDAP. Además esta API ofrece un conjunto de clases que permiten gestionar las entradas y definiciones de esquema en servidores LDAPv3. Proporciona clases para el uso de controles LDAP. Ofrece clases de utilidad para el uso de aplicaciones LDAP. Proporciona los métodos para establecer una conexión autenticada o anónima a un servidor LDAP, así como los métodos para buscar, modificar, comparar, y borrar entradas en el directorio.

1.5.4 Tecnologías de Servicio de Directorio (LDAP)

El Protocolo de Acceso Ligero a Directorio, mejor conocido como LDAP (por sus siglas en inglés), está basado en el estándar X.500, pero significativamente más simple y más realmente adaptado para satisfacer las necesidades del usuario. 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. LDAP también es considerado una base de datos puesto que permite realizar consultas aunque no fue concebido para ello pues no está optimizado para guardar y actualizar información constantemente.

Posee una estructura en forma de árbol de directorio. Habitualmente, almacena la información de autenticación (usuario y contraseña) y es utilizado para autenticarse, aunque es posible almacenar otra información (datos de contacto del usuario, ubicación de diversos recursos de la red, permisos, certificados). A manera de síntesis, LDAP es un protocolo de acceso unificado a un conjunto de información sobre una red.

"Un directorio LDAP destaca sobre los demás tipos de bases de datos por las siguientes características:

- ✓ *Es muy rápido en la lectura de registros*
- ✓ *Permite replicar el servidor de forma muy sencilla y económica*
- ✓ *Muchas aplicaciones de todo tipo tienen interfaces de conexión a LDAP y se pueden integrar fácilmente*
- ✓ *Dispone de un modelo de nombres globales que asegura que todas las entradas son únicas*
- ✓ *Usa un sistema jerárquico de almacenamiento de información.*
- ✓ *Funciona sobre TCP/IP y SSL*
- ✓ *La mayoría de servidores LDAP son fáciles de instalar, mantener y optimizar."* [9]

Existen varias implementaciones y aplicaciones reales del protocolo LDAP entre las que se destacan:

- ✓ Active Directory
- ✓ Novell Directory Services
- ✓ OpenLDAP
- ✓ Apache Directory Server
- ✓ Red Hat Directory Server

En la aplicación se usará OpenLDAP que es el servicio de directorio que brinda el SAI, a continuación de describen las principales características de este servicio de directorio.

"OpenLDAP es una versión libre de LDAP que es un protocolo a nivel de aplicación que soporta un servicio de directorio, se parece a una libreta de direcciones. OpenLDAP se basa en el estándar de servicio de directorio ISO X.500 y en su protocolo DAP (Directory Access Protocol). Se diseñó para ser un protocolo simple y eficiente para acceder al directorio DAP, por eso lo de lightweight, implementa un subconjunto de operaciones del X.500. La base de datos se diferencia de una relacional principalmente en la terminología, por ejemplo un registro es una entrada y campo un atributo, aparte de que la base de datos del OpenLDAP está sólo optimizada en las lecturas, de ahí su rápido acceso a sus datos. El directorio nos va a permitir, entre otras cosas, que los usuarios puedan buscar de manera rápida información sobre otros usuarios (e-mail, teléfonos, entre otras opciones) o bien autenticarse, aunque este servicio no siempre nos va a permitir el cambio de esa información, eso dependerá de nuestros permisos." [10].

Organiza la información de manera jerárquica en forma de árbol. Tiene un conjunto de ventajas que convierte a OpenLDAP en uno de los sistemas de directorio más utilizados en la actualidad como son: su flexibilidad y escalabilidad. Además, está optimizado en lectura de registros, en realizar criterios de búsqueda complejos, permitir la réplica de la base de datos y los sistemas operativos disponen de soporte para OpenLDAP.

1.5.5 Tecnología de Sistema de Gestión de Base de Datos Oracle 10gR2

En la aplicación se utilizará Oracle 10gR2 que es la base de datos que usa el SAI y la que le brinda a la aplicación. Oracle 10gR2 es uno de los gestores de base de datos más usados en el mundo, por sus características únicas es considerado como uno de los más completos. A pesar de tener un alto precio posee ventajas como son: una gran potencia y rapidez haciendo posible el manejo de un gran volumen de datos con un alto rendimiento. Es escalable y permite grandes demandas de usuarios y es adaptable a cambios bruscos. La tecnología de Oracle, optimiza el tiempo de ejecución de sus consultas ya que hace uso de consultas en paralelo en diferentes nodos. Posee una gran capacidad para la réplica de datos, máxima seguridad, administración simplificada, soporte de transacciones y facilidades en las tareas de recuperación y respaldo.

1.5.6 Tecnologías de Desarrollo

1.5.6.1 Lenguaje Unificado de Modelado (UML)

El Lenguaje Unificado de Modelado (UML, por sus siglas en inglés, *Unified Modeling Language*) es el lenguaje de modelado de sistemas de *software* más conocido y utilizado en la actualidad y el que se utilizará en la aplicación. Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema de *software*. Incluye aspectos conceptuales tales como procesos de negocios y funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes de *software* reutilizables.

"UML es una especificación de notación orientada a objetos. Se basa en las anteriores especificaciones BOOCH, RUMBAUGH y COAD-YOURDON. Divide cada proyecto en un número de diagramas que representan las diferentes vistas del proyecto. Estos diagramas juntos son los que representa la arquitectura del proyecto." [11]

UML introduce nuevos diagramas que permiten tener una visión dinámica del sistema lo que facilita detectar problemas de la estructura en la fase de diseño. UML es un estándar, es el único lenguaje de diseño orientado a objetos existentes. Su utilización es independiente del lenguaje de programación y de las características de los proyectos, gracias a que UML puede modelar cualquier tipo de proyectos.

1.5.6.2 Altova UModel

Altova UModel posibilita el desarrollo de un *software* exitoso y es utilizado en la aplicación puesto que permite crear e interpretar diseños de *software* mediante la potencia del estándar UML 2.1. Dibuja el diseño de la aplicación y genera código Java o C# a partir de sus planos. Permite realizar ingeniería inversa de programas existentes a diagramas UML claros y precisos para abarcar rápidamente su arquitectura de *software*.

Puede corregir código generado o los modelos y producir automáticamente nuevos diagramas o regenerar el código. Soporta todos los tipos de diagramas UML existentes. Soporta hiperenlaces entre diagramas UML y desde diagramas a ficheros externos o sitios web. Interoperabilidad multi-plataforma.

Es eficiente y fácil de usar. Posee una rica interfaz visual y usabilidad superior que ayudan a disminuir la curva de aprendizaje de UML.

1.5.6.3 MSF para Metodologías de Desarrollo Ágil como metodología de desarrollo

"MSF para Metodologías de Desarrollo Ágil es un proceso de desarrollo manejado por escenarios, basado en contexto, que utiliza muchas de las ideas incorporadas en Team System (herramientas de Microsoft). Este proceso incorpora las prácticas probadas desarrolladas en Microsoft con respecto a los requerimientos, diseño, seguridades, rendimiento y pruebas. MSF para metodologías de desarrollo ágil presenta una guía muy recomendable a los Desarrolladores y Gestores de proyectos de software que pueden adaptarla a la metodología de su empresa, en la que incluye documentos de ejemplo, plantillas, archivos en blanco de Project, Excel y Word para la administración de proyectos, requerimientos, seguridad y pruebas." [12]

Promueve el desarrollo de un *software* basado en el trabajo en grupo. Este modelo además posee una flexibilidad que hace posible que sea usado tanto en grandes proyectos como en pequeños proyectos. Está compuesto de ciclos e iteraciones, posee un Modelo de Equipo, el cual describe el papel de los diversos miembros del equipo en un proyecto de desarrollo de *software* y el Modelo de Gobierno, el cual establece el orden de las actividades del proyecto, representando completamente el ciclo de vida de éste. Esta metodología en equipo establece ocho roles, entre los que se encuentran: Analista de Negocio, Jefe de Proyecto, Arquitecto, Desarrollador, Probador y Administrador de Despliegue. Además, esta metodología consta de cinco fases, las cuales son: Visión, Planificación, Desarrollo, Estabilización y Despliegue.

1.5.7 Plataforma .NET

"La plataforma .NET es la propuesta de Microsoft para competir con la plataforma Java. La plataforma .NET de Microsoft está diseñada para que se puedan desarrollar componentes de software utilizando casi cualquier lenguaje de programación, de forma que lo que escribamos en un lenguaje pueda utilizarse desde cualquier otro de la manera más transparente posible." [13]

Existen compiladores de múltiples lenguajes para la plataforma .NET entre ellos: *Visual Basic*, C#, C++, *ASP*, *Python*, entre otros. La plataforma .NET ofrece el kit de desarrollo de *software* .NET Framework para crear Servicios Web o aplicaciones, el cual contiene el *Common Language Runtime* (CLR), que se considera el núcleo de esta plataforma ya que es el encargado de gestionar la ejecución de código compilado para esta plataforma. Además ofrece ADO.NET para manejar el acceso a las bases de datos, ASP.NET para crear páginas activas y WinForms para construir aplicaciones Windows. La Plataforma .NET posee un conjunto de ventajas que lo hacen muy potente con respecto a otras plataformas de desarrollo.

"A continuación se resumen las ventajas más importantes que proporciona .Net Framework:

- ✓ ***Código administrado:*** *El CLR realiza un control automático del código para que este sea seguro, es decir, controla los recursos del sistema para que la aplicación se ejecute correctamente.*

- ✓ ***Interoperabilidad multilenguaje:*** *El código puede ser escrito en cualquier lenguaje compatible con .Net ya que siempre se compila en código intermedio (MSIL).*

- ✓ ***Compilación just-in-time:*** *El compilador JIT incluido en el Framework compila el código intermedio (MSIL) generando el código máquina, propio de la plataforma. Se aumenta así el rendimiento de la aplicación al ser específico para cada plataforma.*

- ✓ ***Garbage collector:*** *El CLR proporciona un sistema automático de administración de memoria denominado recolector de basura. El CLR detecta cuándo el programa deja de utilizar la memoria y la libera automáticamente. De esta forma, el programador no tiene por qué liberar la memoria de forma explícita aunque también sea posible hacerlo manualmente, podemos liberar el objeto para que el recolector de basura lo elimine de la memoria.*

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

1.5.7.1 Microsoft Visual Studio.NET

Microsoft Visual Studio.NET es el *Entorno de Desarrollo Integrado* (IDE) para sistemas operativos Windows que utiliza la plataforma.NET y el que se utilizará en la aplicación. Soporta varios lenguajes de programación tales como Visual C++, Visual C#, Visual J#, ASP.NET, entre otros. Permite a los desarrolladores crear aplicaciones de escritorio, sitios, aplicaciones web y servicios web en cualquier entorno que soporte la plataforma .NET. A continuación se describen algunas de las características y ventajas de este IDE.

- ✓ Permite mejorar la interoperabilidad entre código nativo y código manejado por .NET. Esta integración más profunda simplificará el trabajo de diseño y codificación.
- ✓ Permite la creación de soluciones multiplataforma adaptadas para funcionar con las diferentes versiones de .Net Framework.
- ✓ Posee una interfaz gráfica agradable y muy fácil de utilizar.
- ✓ Utiliza el .NET Framework 3.5 que incluye biblioteca ASP.NET y AJAX para desarrollar aplicaciones web más eficientes, interactivas y altamente personalizadas que funcionen para todos los navegadores más populares.
- ✓ Mejora las capacidades de las Pruebas Unitarias al ejecutarlas de forma más rápida independientemente de si lo hacen en el entorno IDE o desde la línea de comandos.

1.5.7.2 Lenguaje de Programación C#

"C# es un lenguaje orientado a objetos, elegante y con seguridad de tipos que permite a los desarrolladores crear una amplia gama de aplicaciones sólidas y seguras que se ejecutan en .NET Framework. Puede utilizar este lenguaje para crear aplicaciones cliente para Windows tradicionales, servicios Web XML, componentes distribuidos, aplicaciones cliente-servidor, aplicaciones de base de datos, y muchas tareas más." [15]

Posee un conjunto de ventajas que lo hacen muy potente con respecto a otros lenguajes de programación, lo que posibilitó su elección, entre las que se destacan:

- ✓ La sintaxis de C# es muy expresiva, sencilla y fácil de aprender.

- ✓ El lenguaje C# es una evolución de los lenguajes C y C++. Utiliza muchas de las características de C++ en las áreas de instrucciones, expresiones y operadores.
- ✓ C# también admite métodos y tipos genéricos, que proporcionan mayor rendimiento y seguridad.
- ✓ Lenguaje orientado a objetos que admite los conceptos de encapsulación, herencia y polimorfismo.

1.6 Propuesta y Selección de Herramientas

Luego de un estudio y de varias investigaciones sobre los temas relacionados con el problema a resolver, para dar solución a la implementación del sistema que permita la integración de la Infraestructura de Clave Pública y el Sistema de Administración de Identidades del Ministerio de Interior se elegirá para el trabajo con los servidores LDAP y los algoritmos de cifrado a la librería BouncyCastle C# y la API Novell para LDAP debido a las facilidades de trabajo que brindan.

Para la gestión de las solicitudes de publicación de certificados digitales, se usará como repositorio para guardar los certificados de los usuarios y por correspondiente los datos de los usuarios, servidores OpenLDAP por permitir la autenticación y acceso a datos de una forma rápida y segura y tener la ventaja de ser multiplataforma.

Para la implementación del sistema se utilizará la plataforma.NET con el IDE Microsoft Visual Studio.NET y el lenguaje C#, el cual forma parte de la familia de los lenguajes de .NET lo que permitirá desarrollar el sistema por su facilidad de trabajo, la potencia, robustez y al mismo tiempo sencillez y elegancia que posibilitarán un alto grado de productividad.

1.7 Conclusiones

El desarrollo de este capítulo sirvió para investigar sobre la Infraestructura de Clave Pública (PKI) y los diferentes productos que existen en el mundo que ofrecen soluciones utilizando una PKI, lo que permitió identificar las características y ventajas de esta tecnología, lo que la hace imprescindible para dar solución a los problemas identificados. Se identificaron un conjunto de estándares, herramientas y tecnologías que permiten realizar el modelado de la solución, así como se estudiaron las diferentes librerías que brindan las potencialidades necesarias para lograr la implementación de dicha solución. Todo ello posibilitó plantear la solución propuesta y abordar el por qué de la necesidad de la integración de la PKI del MININT con el SAI del MININT.

CAPÍTULO 2: CARACTERÍSTICAS DE LA SOLUCIÓN

2.1 Introducción

En el presente capítulo se describen las primeras fases de la metodología utilizada y se describen las principales relaciones entre conceptos en el Diagrama de vista global del sistema. Se propone la propuesta de solución, así como la descripción tanto de los escenarios de sistema como de los requisitos de calidad de servicios obtenidos del proceso de levantamiento de información, donde se pretende lograr la comprensión de las partes implicadas y que sirva de base para posteriores artefactos.

2.2 Diagrama de Vista Global del Sistema

La metodología utilizada en nuestra aplicación no utiliza procesos de negocio por lo que se llegó a la conclusión de utilizar un Diagrama de Vista Global del Sistema el cual representa mediante conceptos relacionados entre sí una visión del sistema. Este modelo muestra cómo funciona el entorno en el cual está enmarcada la aplicación.

Sistema: Representa el sistema a desarrollar, en este caso el módulo de Integración PKI.

Autoridad Certificadora: Es la encargada de realizar la gestión de los certificados digitales y de las entidades finales.

Certificado: Representa los certificados digitales X509v3, los cuales serán utilizados por las entidades finales.

Entidad: Es la encargada de almacenar los datos de las entidades finales.

Solicitudes: Es la encargada de obtener las solicitudes de peticiones de publicación de certificados digitales.

Reporte: Es la encargada de generar los reportes de actualización de los perfiles de las entidades finales.

Conexión: Es la encargada de gestionar todas las conexiones que se establecen en el sistema, entiéndase por ello, conexiones a los repositorios de certificados y de identidades respectivamente y de la conexión la base de datos donde se encuentran las solicitudes.

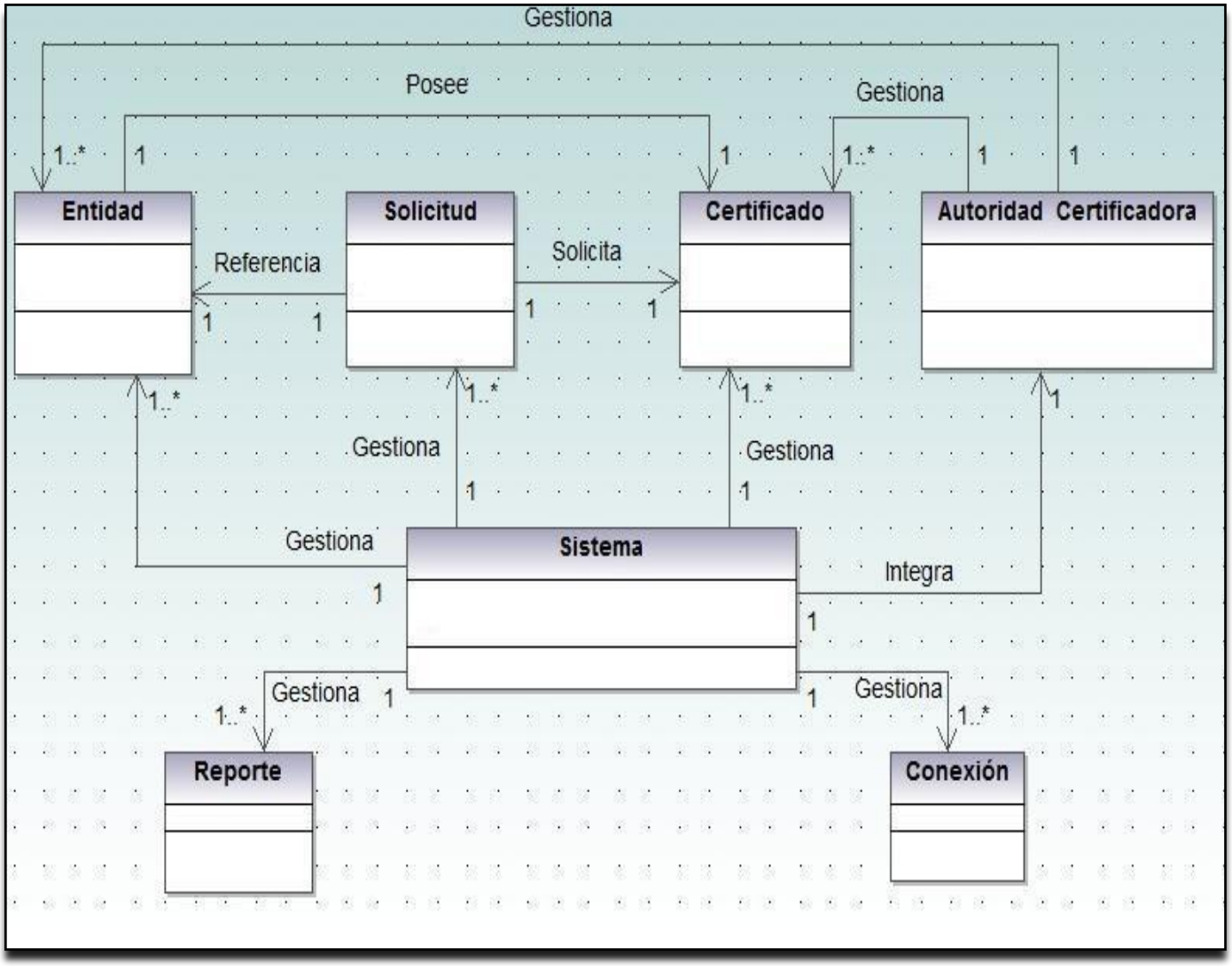


Figura 2. Diagrama de Vista Global del Sistema

2.3 Propuesta de Solución

A partir de los escenarios y los requisitos de calidad de servicios identificados se tiene como objetivo darle cumplimiento a la siguiente propuesta de solución. Para asegurar la integración de la PKI con el SAI, el sistema debe permitir brindar de forma eficiente y rápida, respuestas a las peticiones de

publicación de certificados digitales. Para ello el sistema debe conectarse a la base de datos del SAI para obtener estas solicitudes, y conectarse al repositorio de identidades del SAI donde se encuentran los datos de las entidades finales, los cuales son necesarios para que el sistema genere las peticiones de solicitud de publicación de certificados digitales y las envíe a la autoridad certificadora competente (EJBCA) para que genere dicho certificado. El certificado generado por la EJBCA le será entregado al sistema y este será el encargado de publicar dicho certificado en el repositorio de certificados del SAI por lo que se debe asegurar la conexión con este repositorio. Luego de culminar las operaciones anteriores el sistema se encargará de asignar los certificados a los usuarios o entidades finales que lo necesiten. Además será capaz de revocarlos en caso de que no cumplan con los requisitos que lo caracterizan como válido.

2.4 Visión

En esta fase se obtiene una visión del proyecto compartida, comunicada, entendida y alineada con los objetivos del negocio. Además, el equipo y el cliente integran el proyecto y definen los escenarios, sus alcances y restricciones.

2.4.1 Personas

Son los usuarios o sistemas externos que interactúan con la aplicación.

Persona	Descripción
Sistema	Podrá realizar todas las operaciones que existan en el sistema.

Tabla 2. Descripción de Persona

2.4.2 Documento Visión

En este mundo en constante evolución, las técnicas informáticas modernas brindan soluciones informáticas a organizaciones gubernamentales para cumplir los paradigmas que estas defienden. La seguridad, concepto que en otros tiempos de avance informático no ocupaba un sitio prioritario como en estos tiempos de crímenes informáticos, tales como la suplantación de identidad, apoderamiento de recursos confidenciales, llegó para ser un baluarte a tener en cuenta en todo sitio donde aplicaciones

informáticas estén presentes, y de esta manera brindar mecanismos de manejo de datos de seguridad tales como (usuarios, contraseñas, roles de usuarios, recursos y permisos sobre los recursos) en aplicaciones que se ejecutan de forma individual. Así surge el SAI solución informática para la administración de estos datos en el MININT. Un sistema que permite autenticar, autorizar y administrar de manera centralizada procesos de administración de identidades, creación de cuentas para usuarios y entidades finales.

Son variadas las aplicaciones de este tipo en el mercado mundial, que administran usuarios y roles, gestionan contraseñas, y aplican políticas de control de acceso. Pero presentan un detalle que las hace inaccesibles a países en vías de desarrollo como el nuestro, y es que están accesibles solo a organizaciones con grandes fondos e ingresos económicos que pueden costear los exuberantes precios. No solo para su adquisición sino también, todo el proceso de soporte que estas requieren, tales como: capacitación del personal y adaptación a las condiciones de la empresa. Sin embargo, esto no cambia la realidad de que, en el MININT las tareas de administración son lentas y víctimas de errores que aunque humanos dan al traste a la ineficiencia y falta de confiabilidad en esos procesos.

El SAI cuenta con un área dedicada al aprovisionamiento de usuarios llamada PROVISIONing. Aplicación en proceso de desarrollo, la cual cuenta con varios módulos para su funcionamiento. Ellos son: Portal de Usuario, Administración, Políticas de Acceso, Gestor de Conectores, Reportes, Motor de Tareas, Alertas y Notificaciones, e Integración PKI. El presente trabajo de diploma se enfoca en el desarrollo del módulo Integración PKI.

El desarrollo del módulo Integración PKI llega con la necesidad de integrar el sistema con la Infraestructura de clave pública, infraestructura que ofrece mecanismos de seguridad basados en criptografía de clave pública, en procesos de creación y asignación de certificados digitales a usuarios y entidades los cuales el sistema de aprovisionamiento maneja.

Los usuarios que obtienen valor del sistema son:

- Administrador

El módulo de Integración PKI estará disponible a usuarios de tipo Administrador. Este módulo permitirá atender las solicitudes de publicación y asignación de certificados digitales a usuarios y entidades finales.

Conceptos fundamentales:

PROVISIONing: Representa la aplicación de aprovisionamiento a la cual pertenece el módulo propuesto.

Administrador: Representa un usuario del sistema encargado de administrar todos los módulos con que cuenta la aplicación de aprovisionamiento.

2.5 Planificación

Durante esta fase se listan y se describen los escenarios y los requisitos de calidad de servicio, así como se establece el plan te iteración para el control de las funcionalidades del sistema.

2.5.1 Requisitos de Calidad de Servicios

A continuación se describen las especificaciones identificadas relativas a requisitos de calidad de servicios los cuales documentan las características del sistema y aseguran como debe funcionar el sistema. Entre los requisitos de calidad de servicios encontrados están:

Rendimiento:

- Garantizar el funcionamiento estable de la aplicación integrada al sistema PROVISIONing.

Confiabilidad:

- El sistema se mantendrá disponible todo el tiempo requerido.
- Se garantizará la consistencia de los datos, se realizarán comprobaciones y validaciones automáticas en todos los casos posibles.

Seguridad:

- Mantener la seguridad y control de usuarios, garantizando el acceso de cada uno de ellos sólo a los niveles establecidos según el rol que ejecute.

✓ **Confidencialidad**

- La información manejada por el sistema estará protegida de acceso no autorizado y divulgación mediante los mecanismos de seguridad definidos por PKI.

✓ **Integridad**

- La información manejada por el sistema será objeto de cuidadosa protección contra la corrupción y estados inconsistentes.

✓ **Disponibilidad**

- A los usuarios autorizados se les garantizará el acceso a la información.

2.5.2 Escenarios del Sistema

A continuación se listan los escenarios identificados en nuestra solución, los cuales garantizarán las condiciones que el sistema debe tener y por ende el resultado exitoso de nuestra aplicación. Estos son los encargados de registrar una sola trayectoria de la interacción entre el usuario y el sistema. Además, un escenario registra las medidas específicas que se trazarán para que el usuario logre alcanzar una meta determinada.

Listado de Escenarios:

ES1. Conectar a repositorio de identidades.
ES2. Conectar a repositorio de certificados digitales.
ES3. Obtener solicitudes de publicación de certificados digitales.
ES4. Generar peticiones PKCS#10 de certificados digitales.
ES5. Enviar peticiones a la autoridad certificadora.
ES6. Chequear respuestas de las peticiones PKCS#10.
ES7. Publicar certificados digitales.
ES8. Asignar certificados a entidades finales.
ES9. Emitir notificación de actualización.
ES10. Revocar certificados digitales.

2.5.3 Plan de Iteraciones

Un plan de iteración detalla el trabajo que debe realizarse dentro de una iteración. El plan de iteración termina en la reunión de planificación de iteración, justo antes del inicio de otra iteración. Dicho plan especifica la prioridad de los escenarios a realizar en cada iteración proponiendo ejecutar en las primeras fases los escenarios de mayor prioridad que representan funcionalidades críticas del sistema. Además, se estiman los días que puede demorar implementar los escenarios identificados y se clasifican según el riesgo.

Iteración #1: Se implementarán los escenarios que proveen las funcionalidades más críticas del sistema con prioridad alta.

Iteración #2: Se implementarán los escenarios que proveen otras funcionalidades del sistema con prioridad media.

Iteración #3: Última iteración, se implementarán los escenarios restantes.

No	Escenarios	Prioridad	Riesgo	Esfuerzo (días)	Iteración
1	Conectar a repositorio de identidades	Alta	Medio	9	1
2	Conectar a repositorio de certificados digitales	Alta	Medio	9	1
3	Obtener solicitudes de publicación de certificados digitales	Alta	Medio	10	1
4	Generar peticiones PKCS#10 de certificados digitales	Media	Alto	6	2
5	Enviar peticiones a la autoridad certificadora	Media	Alto	5	2
6	Publicar certificados digitales	Media	Alto	6	2
7	Asignar certificados a entidades finales	Media	Alto	5	2
8	Chequear respuestas de las peticiones PKCS#10	Baja	Medio	3	3
9	Emitir notificación de actualización	Baja	Bajo	3	3
10	Revocar certificados digitales	Baja	Alto	2	3

Tabla 3. Plan de Iteraciones

2.5.4 Descripción de los Escenarios

A continuación se describen los escenarios identificados durante el proceso de levantamiento de información, los cuales representan una guía a seguir para lograr el desarrollo exitoso del sistema y juegan un importante papel puesto que posibilita evitar errores innecesarios durante el proceso de desarrollo del sistema.

Descripción del Escenario: Conectar a repositorio de identidades

Título:	Conectar a repositorio de identidades
Iteración:	1
Estado:	Cerrado
Razón de estado:	Proceso completado
Persona:	Sistema
Descripción:	El escenario comienza cuando el sistema se dispone a atender las solicitudes de publicación de certificados digitales. Para ello se conecta al repositorio de identidades, acción necesaria para la obtención de datos referentes a las solicitudes de certificación. El escenario culmina luego de conectar o no al repositorio de identidades.
Historial:	1. Inicio de descripción del escenario
Prioridad:	Alta
Identificador:	ES1
Orden de Magnitud	2

Tabla 4. Descripción del Escenario Conectar a repositorio de identidades

Descripción del Escenario: Conectar a repositorio de certificados digitales

Título:	Conectar a repositorio de certificados digitales
Iteración:	1
Estado:	Cerrado
Razón de estado:	Proceso completado
Persona:	Sistema
Descripción:	El escenario comienza cuando el sistema se dispone a atender las solicitudes de publicación de certificados digitales. Para ello se conecta al repositorio de certificados digitales necesario para la publicación de los certificados digitales emitidos a entidades finales. El escenario culmina luego de conectar o no al repositorio de certificados digitales.
Historial:	1. Inicio de descripción del escenario
Prioridad:	Alta
Identificador:	ES2
Orden de Magnitud	2

Tabla 5. Descripción del Escenario Conectar a repositorio de certificados digitales

Descripción del Escenario: Obtener solicitudes de publicación de certificados digitales

Título:	Obtener solicitudes de publicación de certificados digitales
Iteración:	1
Estado:	Cerrado

Razón de estado:	Proceso completado
Persona:	Sistema
Descripción:	El escenario comienza una vez el sistema conectado a los repositorios de identidades y certificados digitales, este obtiene las solicitudes de certificación donde se encuentran los datos necesarios para el desarrollo de los restantes escenarios. El escenario culmina una vez obtenidas las solicitudes.
Historial:	1. Inicio de descripción del escenario
Prioridad:	Alta
Identificador:	ES3
Orden de Magnitud	2

Tabla 6. Descripción del Escenario Obtener solicitudes de publicación de certificados digitales

Descripción del Escenario: Generar peticiones PKCS#10 de certificados digitales

Título:	Generar peticiones PKCS#10 de certificados digitales
Iteración:	2
Estado:	Cerrado
Razón de estado:	Proceso completado
Persona:	Sistema
Descripción:	El escenario comienza una vez obtenidas las solicitudes de publicación de certificados digitales, el sistema genera una petición PKCS#10 para cada solicitud de certificación, para esto se apoya de datos necesarios presentes en dicha solicitud. El escenario culmina una vez generadas las peticiones PKCS#10 para cada una de las solicitudes de certificación.

Historial:	1. Inicio de descripción del escenario
Prioridad:	Alta
Identificador:	ES4
Orden de Magnitud	1

Tabla 7. Descripción del Escenario Generar peticiones PKCS#10 de certificados digitales

Descripción del Escenario: Enviar peticiones a la autoridad certificadora

Título:	Enviar peticiones a la autoridad certificadora
Iteración:	2
Estado:	Cerrado
Razón de estado:	Proceso completado
Persona:	Sistema
Descripción:	El escenario comienza una vez generadas todas las peticiones PKCS#10. A continuación, el sistema envía cada una de las peticiones a la autoridad certificadora responsable de atender dichas peticiones de certificación. El escenario culmina una vez enviadas las peticiones PKCS#10 a la autoridad certificadora.
Historial:	1. Inicio de descripción del escenario
Prioridad:	Alta
Identificador:	ES5
Orden de Magnitud	1

Tabla 8. Descripción del Escenario Enviar peticiones a la autoridad certificadora

Descripción del Escenario: Chequear respuestas de las peticiones PKCS#10

Título:	Chequear respuestas de las peticiones PKCS#10
Iteración:	3
Estado:	Cerrado
Razón de estado:	Proceso completado
Persona:	Sistema
Descripción:	El escenario comienza cuando una vez enviadas las peticiones de certificación a la autoridad certificadora, el sistema obtiene las respuestas de certificación, las cuales son traducidas como certificados digitales según el estándar X509v3. El escenario culmina una vez que son obtenidas todas las respuestas y traducidas, correspondientes a cada una de las peticiones enviadas.
Historial:	1. Inicio de descripción del escenario
Prioridad:	Alta
Identificador:	ES6
Orden de Magnitud	1

Tabla 9. Descripción del Escenario Chequear respuestas de las peticiones PKCS#10

Descripción del Escenario: Publicar certificados digitales

Título:	Publicar certificados digitales
Iteración:	2
Estado:	Cerrado
Razón de estado:	Proceso completado

Persona:	Sistema
Descripción:	El escenario comienza una vez obtenidos los certificados digitales según el estándar X509v3. A continuación, el sistema publica en el repositorio de certificados digitales los certificados emitidos a entidades finales. El escenario culmina una vez publicado cada uno de estos certificados digitales.
Historial:	1. Inicio de descripción del escenario
Prioridad:	Alta
Identificador:	ES8
Orden de Magnitud	1

Tabla 10. Descripción del Escenario Publicar certificados digitales

Descripción del Escenario: Asignar certificados a entidades finales

Título:	Asignar certificados a entidades finales
Iteración:	2
Estado:	Cerrado
Razón de estado:	Proceso completado
Persona:	Sistema
Descripción:	El escenario comienza al acceder al repositorio de identidades para obtener las entidades finales para las que fueron solicitados certificados digitales, luego obtiene los certificados digitales emitidos a estas entidades y los asigna. El escenario culmina una vez asignados todos los certificados.
Historial:	1. Inicio de descripción del escenario

Prioridad:	Alta
Identificador:	ES9
Orden de Magnitud	1

Tabla 11. Descripción del Escenario Asignar certificados a entidades finales

Descripción del Escenario: Emitir notificación de actualización

Título:	Emitir notificación de actualización
Iteración:	3
Estado:	Cerrado
Razón de estado:	Proceso completado
Persona:	Sistema
Descripción:	El escenario comienza una vez atendidas todas las solicitudes de certificación, generando una notificación con los detalles de cada actualización que fue realizada según los datos de su solicitud. El escenario culmina una vez emitidas las notificaciones de actualización.
Historial:	1. Inicio de descripción del escenario
Prioridad:	Alta
Identificador:	ES10
Orden de Magnitud	1

Tabla 12. Descripción del Escenario Emitir notificación de actualización

Descripción del Escenario: Revocar certificados digitales

Título:	Revocar certificados digitales
Iteración:	3
Estado:	Cerrado
Razón de estado:	Proceso completado
Persona:	Sistema
Descripción:	El escenario comienza cuando es solicitada la revocación de un certificado digital. Para esto el sistema obtiene los datos necesarios para la selección del certificado a revocar y la razón de revocación. El escenario culmina una vez revocado el certificado solicitado.
Historial:	1. Inicio de descripción del escenario
Prioridad:	Alta
Identificador:	ES11
Orden de Magnitud	1

Tabla 13. Descripción del Escenario Revocar certificados digitales

2.6 Conclusiones

Este capítulo posibilita tener una visión parcial de la aplicación, así como obtener todas las funcionalidades y las cualidades que se deben tener en cuenta para lograr un sistema exitoso. Se determinó la persona que será la encargada de interactuar con la solución. Además, se describieron todos los escenarios previstos y se elaboró el plan a seguir para la implementación del sistema, lo que posibilitó elaborar la propuesta de solución, por lo que quedan planteadas las características del sistema propuesto.

CAPÍTULO 3: DESARROLLO, ESTABILIZACIÓN Y DESPLIEGUE

3.1 Introducción

En el presente capítulo se describe la fase más importante, la fase de Desarrollo, así como la fase de Estabilización y la fase de Despliegue. Durante estas fases se abordarán temas como la arquitectura y los patrones usados para la elaboración de la propuesta de solución. Se realizarán los diferentes diagramas lo que posibilitará tener una información visual de la solución, entiéndase por ellos, el diagrama de clases, el diagrama de aplicación y el diagrama de centro de datos lógico. Se realizarán pruebas unitarias y de caja blanca al sistema, las cuales permitirán comprobar la validez del *software*, y se elaborará el plan de despliegue.

3.2 Desarrollo

Esta fase inicia la implementación del sistema, se obtienen entregas parciales del producto, desarrollados por partes para medir su progreso y así asegurarse que puedan integrarse para obtener el producto final.

3.2.1 Arquitectura

Se refiere a un grupo de abstracciones y patrones que nos brindan un marco de referencia útil para guiarnos en el desarrollo y construcción del *software*. Permite a los programadores, diseñadores, ingenieros y analistas poder trabajar bajo una línea común que les posibilite la compatibilidad necesaria para lograr el objetivo deseado.

3.2.2 Patrones

Un patrón es una solución a un problema particular en un contexto determinado, codifica conocimiento específico acumulado por la experiencia en un dominio y describe los componentes, con sus responsabilidades y relaciones.

3.2.3 Patrones de Arquitectura

Son aquellos que expresan un esquema organizativo estructural fundamental para sistemas de *software*.

El patrón empleado en la aplicación es el patrón de Arquitectura en Capas.

✓ **Arquitectura en Capas**

Es un estilo en la que el objetivo primordial es la separación de la lógica de negocios de la lógica de diseño, un ejemplo básico de esto es separar la capa de datos de la capa de presentación al usuario.

La arquitectura en capas tiene como ventaja principal que el desarrollo se realiza en diversos niveles y en caso de tener necesidad de cambiar algo solo se ejecuta en el nivel requerido. También distribuye el trabajo del desarrollo de la aplicación por niveles para que cada equipo se centre en el trabajo que le fue asignado y se abstraiga del resto. Además, las tareas están bien definidas por niveles en esta arquitectura.

El sistema presentado se elaboró de la siguiente forma:

Capa de Aplicación: Esta capa contiene todas las clases controladoras que representan el comportamiento de la aplicación. Además, depende de la capa de lógica del negocio y de la capa de acceso a datos.

Capa de Lógica del Negocio: Esta capa contiene todas las clases encargadas del negocio del sistema y todas las interfaces encargadas de comunicarse con la capa de acceso a datos por lo que puede afirmarse que esta capa depende de la capa de acceso a datos y representa la frontera del cliente con la capa de acceso a datos.

Capa de Acceso a Datos: Esta capa se utiliza para el acceso al Sistema Gestor de Base de Datos, en la aplicación se utiliza (Oracle10gR2); de la cual se obtienen las solicitudes de publicación de certificados. Además, se utiliza para obtener datos de los repositorios de identidades y para guardar datos en el repositorio de certificados.

3.2.4 Patrones de Diseño

Un patrón de diseño es una descripción de clases y objetos comunicándose entre sí adaptados para resolver un problema de diseño general en un contexto particular.

Entre de los patrones empleados en la aplicación se encuentran:

✓ **Fachada**

Es un patrón utilizado en el diseño de la aplicación puesto que provee una interfaz unificada sencilla que sirve de intermediaria entre un cliente y una interfaz o grupo de interfaces más complejas y sus interacciones. Facade puede hacer una biblioteca de *software* más entendible y fácil de utilizar, al hacer su código más legible, inclusive permite que a una API mal diseñada pueda ser diseñada correctamente.

En el ámbito de la aplicación fue utilizado en las clases controladoras, las cuales gestionan las principales operaciones sobre los repositorios de identidad y certificados, así como las ofrecidas por el servicio web del EJBCA, con el fin de encapsular el comportamiento de alguna API o conjunto de clases que intervienen en estas operaciones.

Entre las ventajas que ofrece este patrón se encuentran:

- Los clientes se olvidan de toda la complejidad del negocio, sólo les interesa los resultados obtenidos.
- El mantenimiento es más fácil.
- Reduce las dependencias del código externo.
- Aísla a los clientes de cambios sustanciales en los sistemas, ya sea un cambio de requisitos o de diseño de base de datos.

3.2.5 Patrones GRASP

Describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en forma de patrones. Son una serie de "buenas prácticas" de aplicación recomendable en el diseño de *software*.

Entre de los patrones empleados en la aplicación se encuentran:

✓ **Bajo Acoplamiento**

El patrón bajo acoplamiento utilizado para el diseño de la aplicación brinda una solución a la modularidad necesitada en el mismo, manteniendo lo menos posibles ligada las clases que forman la solución, evitando así que una modificación en alguna de ellas suponga una gran repercusión en las restantes, potenciando así la reutilización, y disminuyendo la dependencia entre las clases.

✓ **Experto**

El patrón experto utilizado en el diseño de la aplicación define como asignar de forma adecuada las responsabilidades en un modelo de clases. Este indica que la responsabilidad de la creación de un objeto o la implementación de un método, debe recaer en la clase que conoce toda la información necesaria para crearlo, obteniendo un diseño con mayor cohesión y manteniendo la información encapsulada.

✓ **Controlador**

El patrón controlador utilizado en el diseño de la aplicación es un patrón que sirve como intermediario entre una determinada interfaz y el algoritmo que la implementa, de tal forma que es la que recibe los datos del usuario y la que los envía a las distintas clases según el método llamado. Este patrón sugiere que la lógica de negocios debe estar separada de la capa de presentación lo que posibilita aumentar la reutilización de código y a la vez tener un mayor control.

3.2.6 Diagrama de Aplicación

En el diagrama de aplicación se muestra el ámbito de la solución, algunos elementos de despliegue, así como referencias a bases de datos externas y servicios web externos. El módulo de integración presenta tres sub-proyectos internos principales: el sub-proyecto <Integration.IM>, donde se encuentran las clases que brindan una interfaz a las operaciones sobre los principales recursos con que se trabajan (los repositorios de identidades, certificados, así como la autoridad certificadora), algunos datos de configuración y la clase que brinda la interfaz a las operaciones que representan los escenarios del sistema.

También está incluido el sub-proyecto <Integration.Business>, donde se encuentran las clases del negocio, entidades, conectores a los recursos externos, manejadores, enumeradores, colecciones de datos. Y finalmente el sub-proyecto <Integration.CA> que contiene un conjunto de clases que ofrecen una capa de abstracción a las operaciones brindadas por el servicio web de la autoridad certificadora.

Además, se muestran API y recursos externos utilizados por estos sub-proyectos, entre las API utilizadas se encuentran: la <API_Novell> para las operaciones con los repositorios bajo el protocolo LDAP y la <API_BouncyCastle> para el trabajo con criptografía de clave pública. Entre los recursos externos se encuentran: el <EjbcaWSService> servicio web de la autoridad certificadora, la <CertificationRequestDB> base de datos externa donde se encuentran las solicitudes de certificación, y los repositorios <IdentityLDAP> y <CertificateLDAP> de identidades y certificados respectivamente.

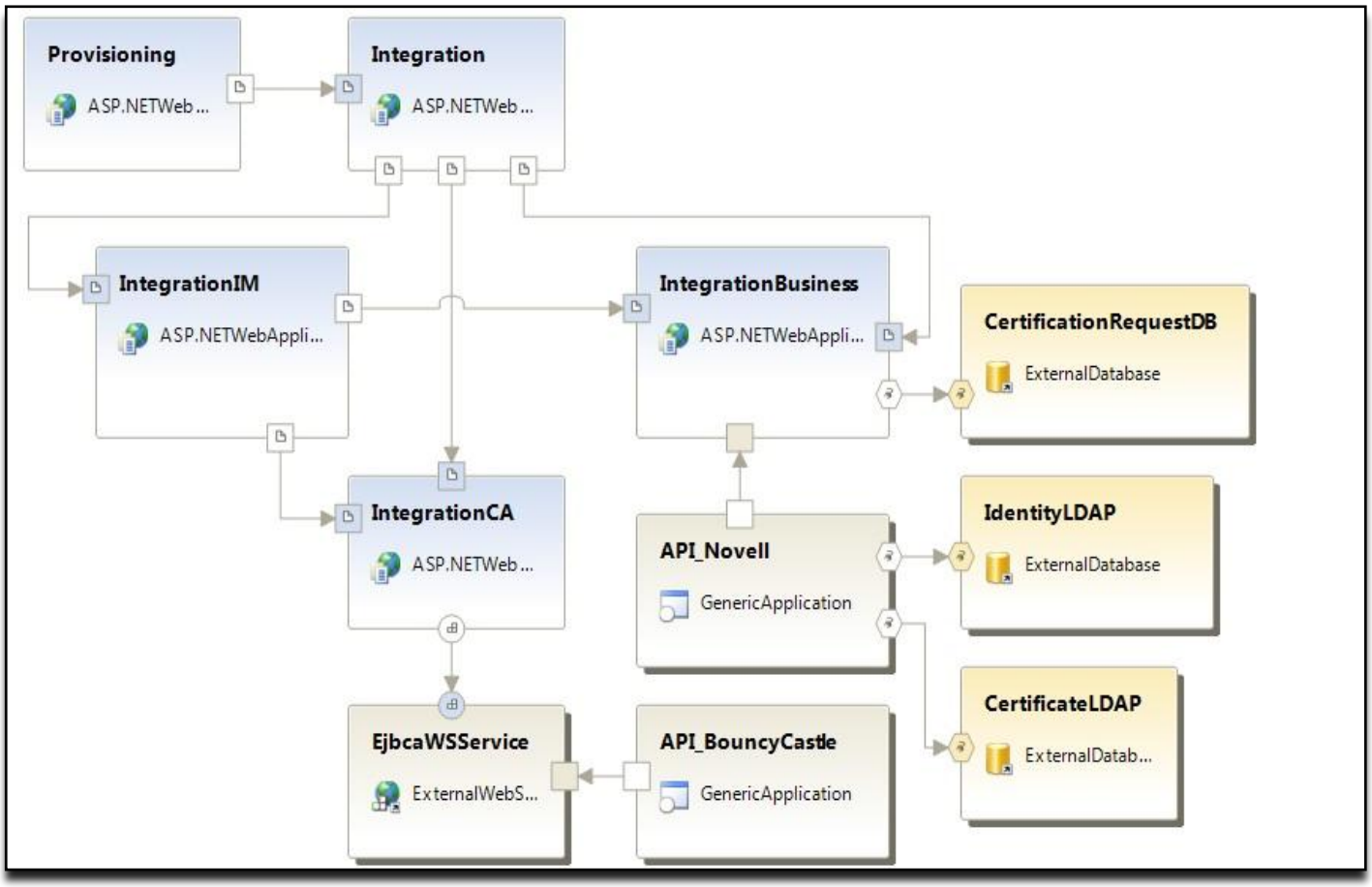
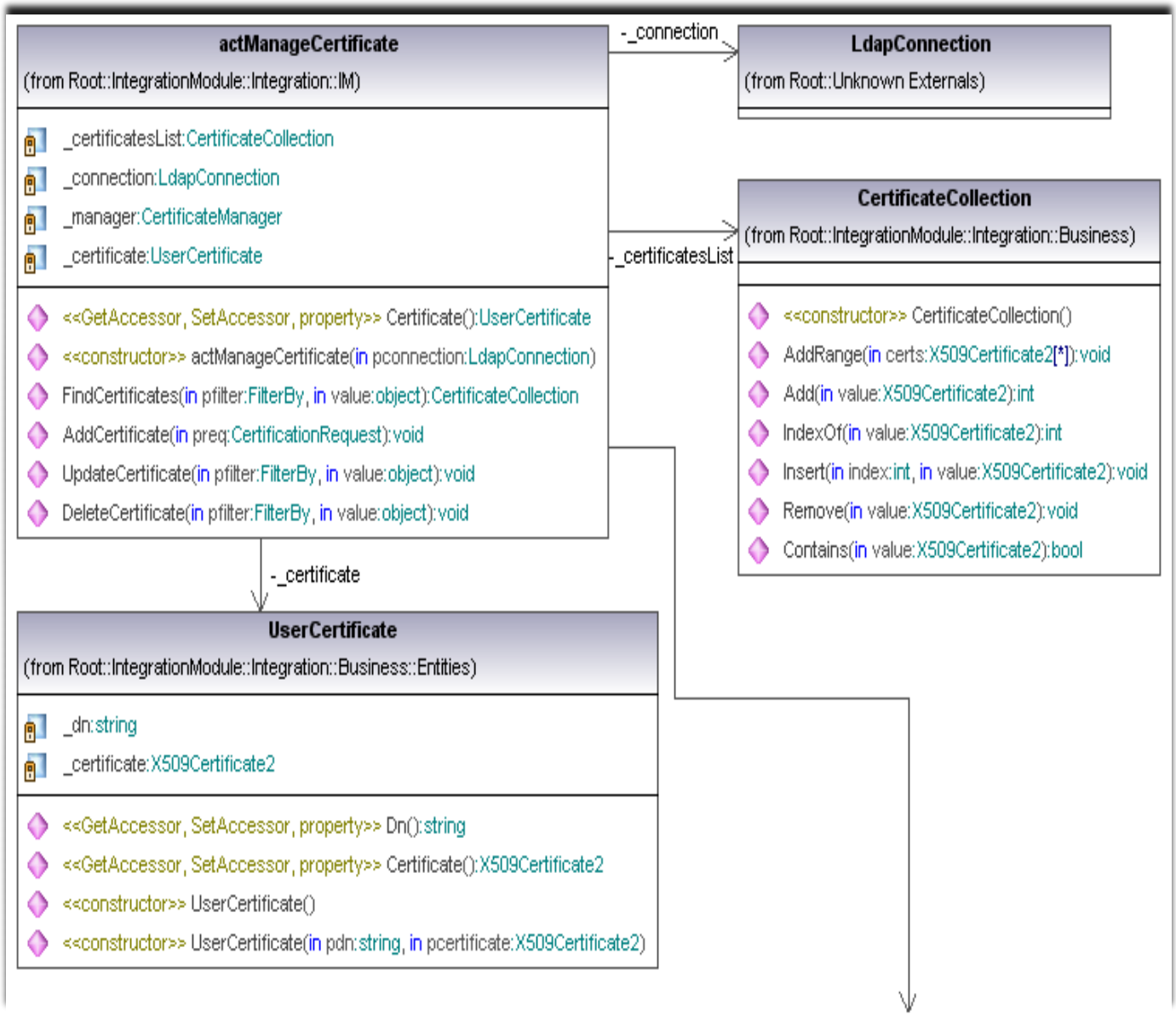


Figura 3. Diagrama de Aplicación

3.2.7 Diagrama de Clases

Un diagrama de clases es un diagrama que representa una agrupación lógica o física de un grupo de clases con sus métodos y atributos y las relaciones que se establecen entre ellas. Además, los diagramas de clases son importantes porque sirven para visualizar, especificar y documentar modelos estructurales y para construir sistemas ejecutables, aplicando ingeniería directa o inversa. A continuación se muestran los diagramas de clases para algunas de las acciones manejadas por el sistema, el resto se encuentra en el **ANEXO 1.**



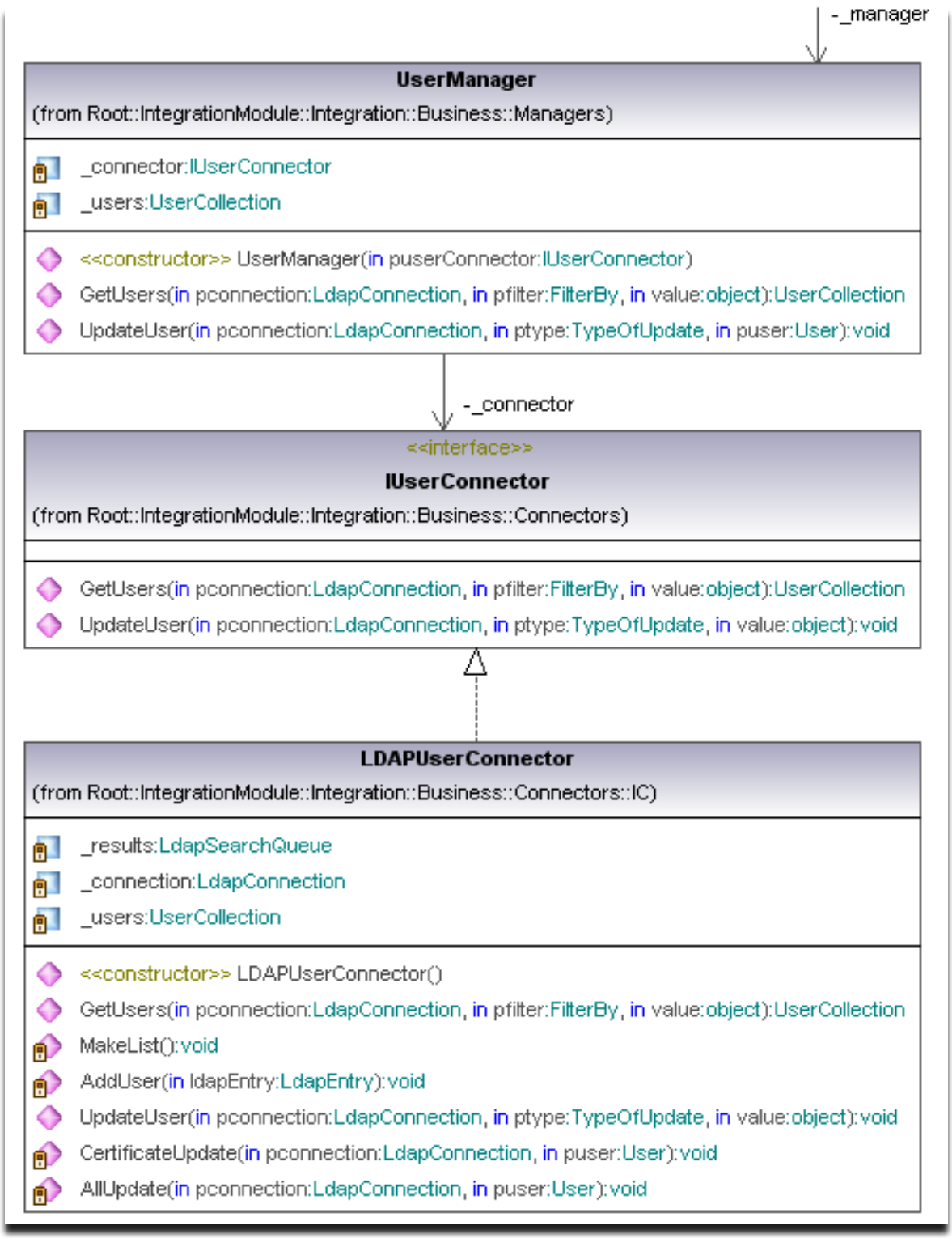


Figura 4: Diagrama de Clases de la acción Manejar Certificado

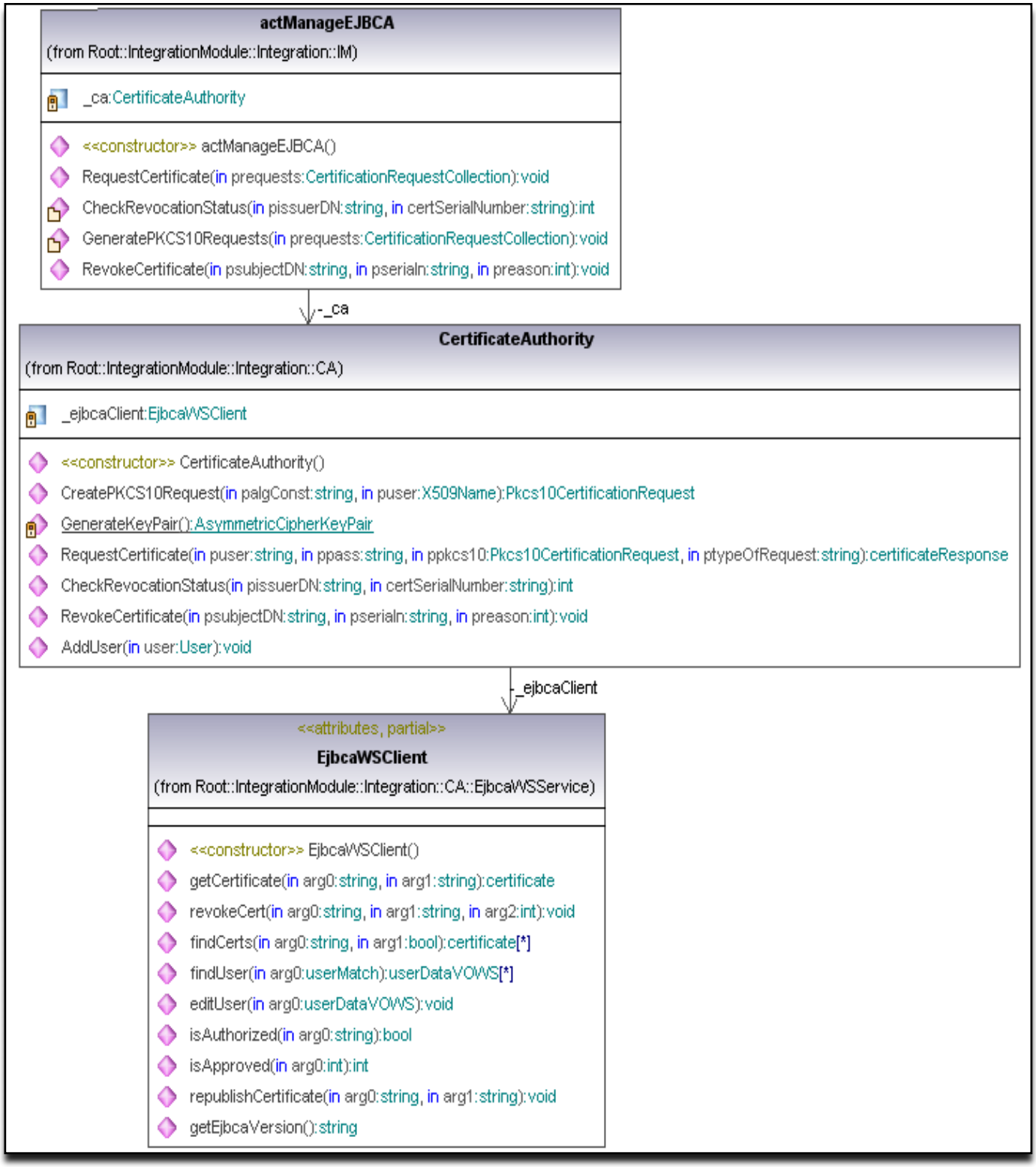


Figura 5: Diagrama de Clases de la acción Manejar Autoridad Certificadora

3.2.8 Diagrama de Centro de Datos Lógico

Un diagrama de centros de datos lógicos define o documenta configuraciones específicas de *software* del servidor de aplicaciones que tienen un propósito específico como: un servidor web seguro para el usuario. En este diagrama se muestra cómo estas configuraciones lógicas de servidores están interconectadas entre sí. Estos servidores lógicos se pueden agrupar dentro de las zonas que definen los límites lógicos de comunicación. Este diagrama muestra además los recursos externos a la zona, con los cuales se interconecta el módulo de integración tales como: el servidor de bases de datos, donde se encuentran las solicitudes de certificación, el repositorio de identidades, el repositorio de certificados, así como el servidor que brinda el servicio web de la autoridad certificadora.

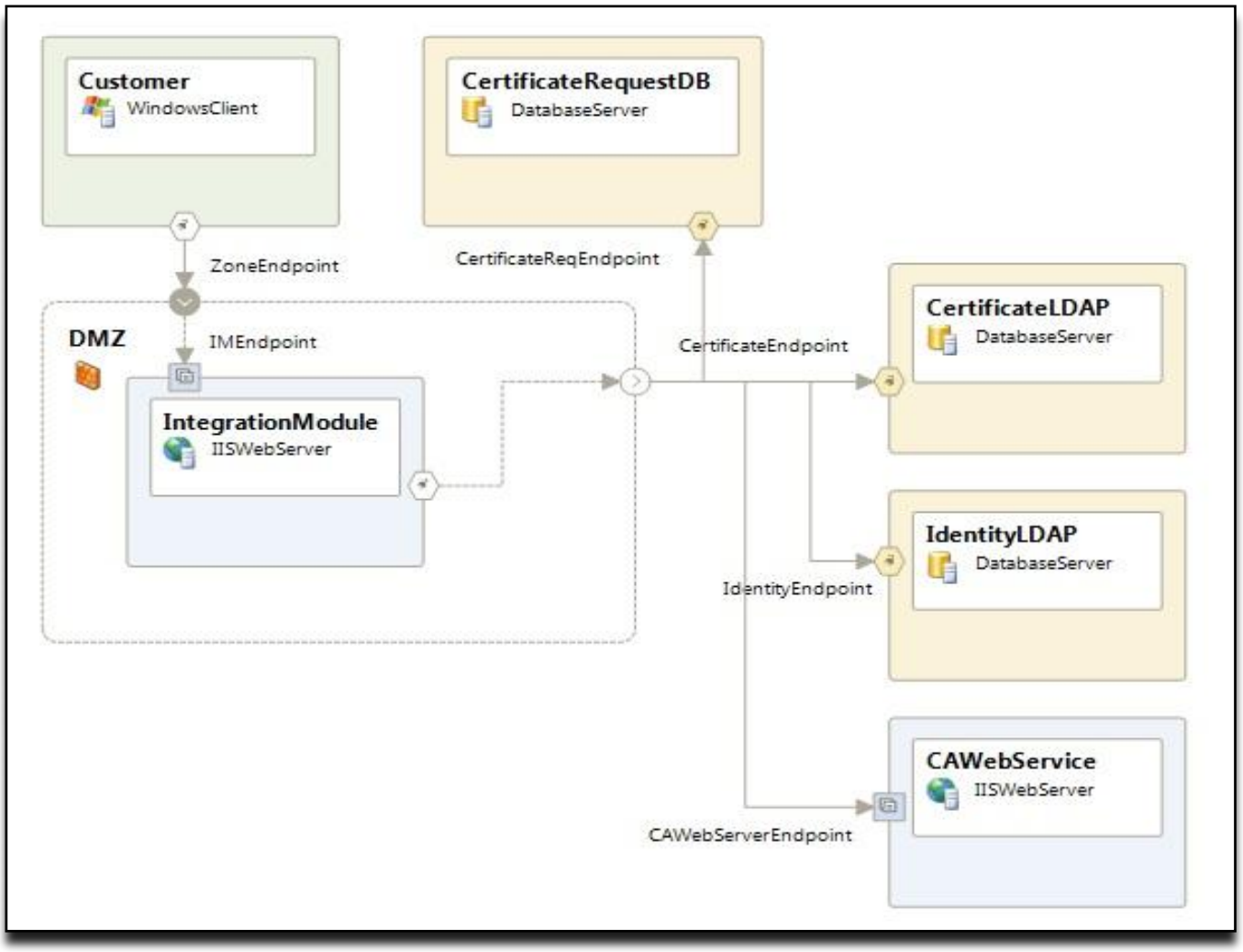


Figura 6: Diagrama de Centro de Datos Lógico

3.3 Estabilización

Esta fase se centra en probar el producto. El proceso de prueba hace énfasis en el uso y el funcionamiento del producto en las condiciones del ambiente real. Es por ello, que esta fase permite validar la propuesta de solución a través de las pruebas a realizar.

3.3.1 Pruebas

El objetivo principal de las pruebas es la de evaluar la calidad del producto que se está desarrollando, mediante la aplicación de pruebas concretas para validar que las suposiciones hechas durante las diferentes fases se están cumpliendo satisfactoriamente, esto quiere decir que se verifica que el producto funcione como se diseñó.

Como forma de verificar los objetivos trazados, se llevó a cabo un proceso de pruebas que validará y le dará un nivel de calidad a la solución implementada. La metodología utilizada propone un conjunto de pruebas, de ellas se utilizarán las pruebas unitarias y las pruebas de caja blanca que son las que más se acopla al sistema elaborado, puesto que permiten comprobar el correcto funcionamiento de secuencias de código.

3.3.2 Prueba Unitaria

Las pruebas unitarias son una de las formas que tenemos de probar pequeñas e individuales porciones de código. A través de ellas se verifica que cierto módulo o funcionalidad se ejecuta dentro de los parámetros especificados. Las pruebas unitarias típicamente son automatizadas aunque pueden hacerse de forma manual, lo que permite detectar los errores más fácilmente y así facilita que el programador cambie el código para mejorar su estructura. Estas pruebas son la base para posteriores pruebas más complejas. Además, son una referencia muy importante para los desarrolladores, pues al ir probando pequeñas porciones de códigos posibilitan que luego cuando se integren todos los componentes en un solo sistema, esta tarea se realice de la forma exitosa y con el menor número de errores posibles.

A continuación se presenta el listado de los escenarios de prueba realizados.

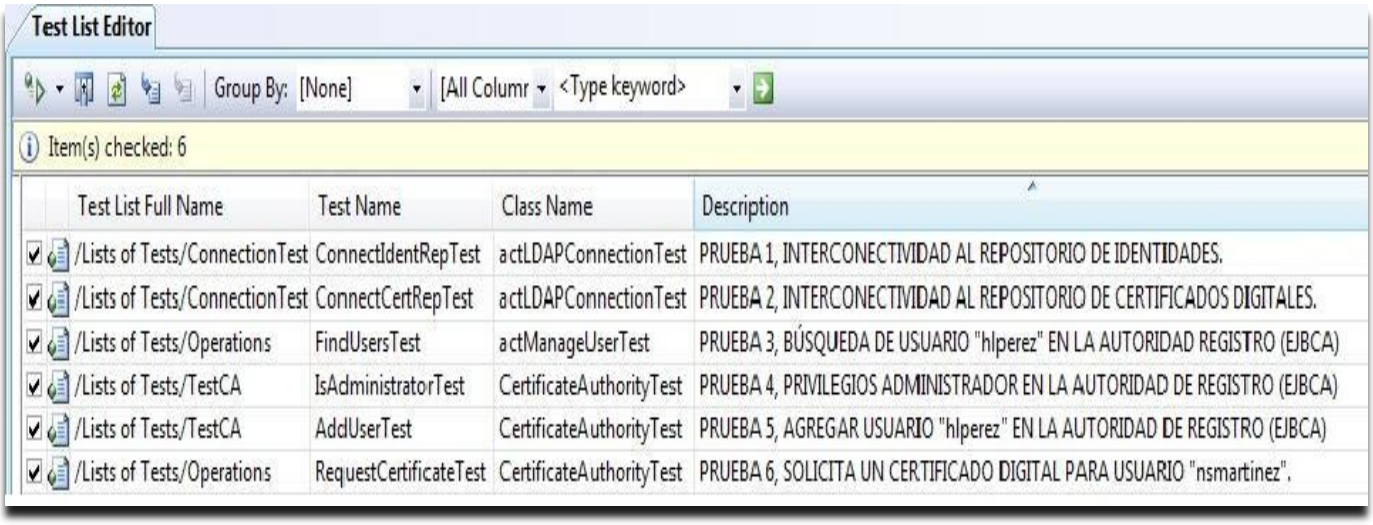


Figura 7: Listado de Métodos de Pruebas Unitarias

Seguidamente se muestra el código de los métodos de prueba realizados, los cuales se listaron anteriormente y pueden verse en la figura 7.

```
[Description("PRUEBA 1, INTERCONECTIVIDAD AL REPOSITORIO DE IDENTIDADES."), TestMethod()]
public void ConnectIdentRepTest()
{
    actLDAPConnection target = new actLDAPConnection();
    TypeOfServer ptype = TypeOfServer.IDENTITY;
    bool expected = true;
    LdapConnection actual;
    actual = target.Connect(ptype);
    Assert.AreEqual(expected, actual.Connected);
}
```

Figura 8: Código correspondiente a la Prueba ConnectIdentRepTest

```
[Description("PRUEBA 2, INTERCONECTIVIDAD AL REPOSITORIO DE CERTIFICADOS DIGITALES."), TestMethod()]
public void ConnectCertRepTest()
{
    actLDAPConnection target = new actLDAPConnection();
    TypeOfServer ptype = TypeOfServer.CERTIFICATE;
    bool expected = true;
    LdapConnection actual;
    actual = target.Connect(ptype);
    Assert.AreEqual(expected, actual.Connected);
}
```

Figura 9: Código correspondiente a la Prueba ConnectCertRepTest

```
[Description("PRUEBA 3, BÚSQUEDA DE USUARIO \"hlperez\" EN LA AUTORIDAD REGISTRO (EJBCA)"), TestMethod()]
public void FindUsersTest()
{
    LdapConnection pconnection = new actLDAPConnection().Connect(TypeOfServer.IDENTITY);
    actManageUser target = new actManageUser(pconnection);
    FilterBy pfilter = FilterBy.ByDN;

    object value = new User(
                        "cn=Hector Luis Perez,ou=identidades,dc=proiden,dc=cu",
                        "Hector Luis", "Hector Luis", "hlperez", "contrasenna").DN;
    string expected = "Hector Luis";

    UserCollection actual;
    actual = target.FindUsers(pfilter, value);
    Assert.AreEqual(expected, actual[0].Surname);
}
```

Figura 10: Código correspondiente a la Prueba FindUsersTest

```
[Description("PRUEBA 4, PRIVILEGIOS ADMINISTRADOR EN LA AUTORIDAD DE REGISTRO (EJBCA)"), TestMethod()]
public void IsAdministratorTest()
{
    CertificateAuthority target = new CertificateAuthority();
    bool expected = true;
    bool actual;
    actual = target.IsAdministrator();

    Assert.AreEqual(expected, actual);
}
```

Figura 11: Código correspondiente a la Prueba IsAdministratorTest

```
[Description("PRUEBA 5, AGREGAR USUARIO \"hlperez\" EN LA AUTORIDAD DE REGISTRO (EJBCA)"), TestMethod()]
public void AddUserTest()
{
    CertificateAuthority target = new CertificateAuthority();
    User user = new User(
                    "cn=Hector Luis Perez,ou=identidades,dc=proiden,dc=cu",
                    "Hector Luis",
                    "Hector Luis",
                    "hlperez",
                    "contrasenna"
                    );
    target.AddUser(user);
}
```

Figura 12: Código correspondiente a la Prueba AddUserTest

```
[Description("PRUEBA 6, SOLICITA UN CERTIFICADO DIGITAL PARA USUARIO \"nsmartinez\"."), TestMethod()]
public void RequestCertificateTest()
{
    CertificateAuthority target = new CertificateAuthority();

    string puser = "nsmartinez";
    string ppass = "contrasenna";
    Pkcs10CertificationRequest ppkcs10 = target.CreatePKCS10Request
                                        (
                                         AlgorithmConstants.SIGALG_SHA1_WITH_RSA,
                                         new X509Name("cn=Hector Luis")
                                         );

    string expected = "C=CU, O=ProIden, CN=AdminCA" ;
    X509Certificate2 actual;
    actual = new X509Certificate2(target.RequestCertificate
                                        (
                                         puser, ppass, ppkcs10,
                                         CertificateHelper.RESPONSETYPE_CERTIFICATE).data
                                         );
    Assert.AreEqual(expected, actual.Issuer);
}
```

Figura 13: Código correspondiente a la Prueba RequestCertificateTest

Seguidamente se muestran los resultados obtenidos de las pruebas realizadas en el Visual Studio Test Editor, los cuales demuestran el correcto funcionamiento de algunas de las funcionalidades de la aplicación a las cuales se le realizaron pruebas unitarias.

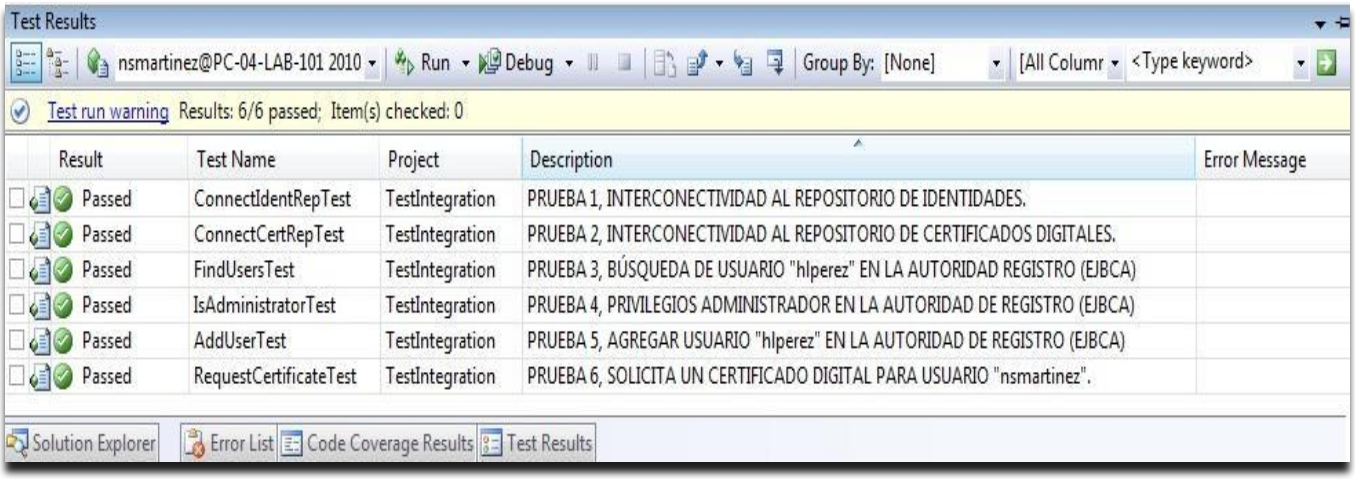


Figura 14: Resultado de las Pruebas Unitarias

3.3.3 Prueba de Caja Blanca

La puesta en práctica de este método requiere del conocimiento de la estructura interna del programa y son derivadas a partir de las especificaciones del diseño o el código. Se basa en la comprobación de los caminos lógicos del *software* dado un código específico. Se puede examinar el estado del programa en varios puntos para determinar si el estado real coincide con el esperado o mencionado. A continuación se realiza la prueba de caja blanca al código, Conectar a recurso repositorio de identidades o certificados.

```
public LdapConnection Connect(TypeOfServer ptype) 1
    {                                                1
       try                                           1
       {
          if (ptype == TypeOfServer.IDENTITY) 2
          {
             _connection.Connect(LDAPsSERVER.IDENTITY_DIRECTORY,
LdapConnection.DEFAULT_PORT);3
             _connection.Bind(LDAPsBind.USER, LDAPsBind.PASS); 3
          }
          else
          {
             _connection.Connect(LDAPsSERVER.CERTIFICATE_DIRECTORY,
LdapConnection.DEFAULT_PORT);4
             _connection.Bind(LDAPsBind.USER, LDAPsBind.PASS);  4
          }
          return _connection; 5
       }
       catch (LdapException ex) 6
       {
          throw (new Exception(string.Format("Error conectando al repositorio de {0}",
ptype == TypeOfServer.IDENTITY ? "identidades" : "certificados"), ex)); 6
       }

    } 7
```

Figura 15: Prueba Conectar a recurso repositorio de identidades o certificados

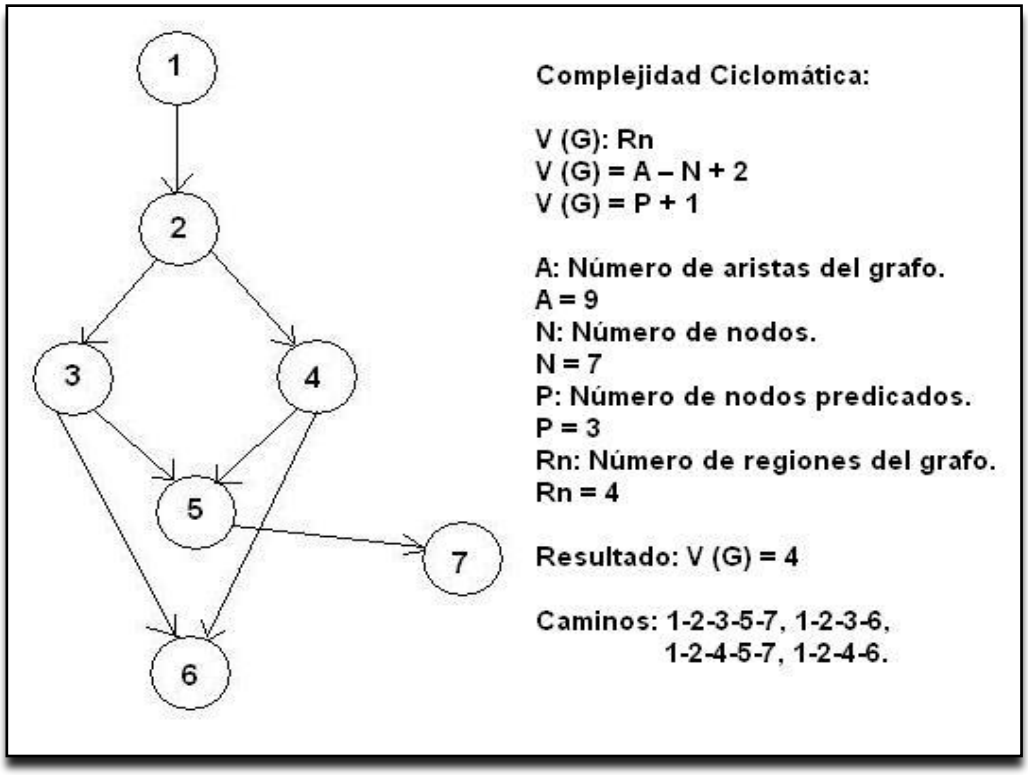


Figura 16: Grafo de Complejidad Ciclomática

Camino: 1-2-3-5-7.

Caso de prueba: Conectar a recurso repositorio de identidades o certificados.

Entrada: Se recibe como tipo de recurso al que se quiere conectar repositorio de identidades.

Resultados: Se verifica que el recurso sea de tipo repositorio de identidades, se conecta al recurso repositorio de identidades, y se retorna la conexión al recurso.

Condiciones: Seleccionar el tipo de recurso al que se quiere conectar.

Camino: 1-2-3-6.

Caso de prueba: Conectar al recurso repositorio de identidades o certificado.

Entrada: Se recibe como tipo de recurso al que se quiere conectar repositorio de identidades.

Resultados: Se verifica que el recurso sea de tipo repositorio de identidades, se lanza una excepción al intentar conectar al recurso repositorio de identidades.

Condiciones: Seleccionar el tipo de recurso al que se quiere conectar.

Camino: 1-2-4-5-7.
Caso de prueba: Conectar a recurso repositorio de identidades o certificados.
Entrada: Se recibe como tipo de recurso al que se quiere conectar repositorio de certificados.
Resultados: Se verifica que el recurso sea de tipo repositorio de certificados, se conecta al recurso repositorio de certificados, y se retorna la conexión al recurso.
Condiciones: Seleccionar el tipo de recurso al que se quiere conectar.

Camino: 1-2-4-6.
Caso de prueba: Conectar al recurso repositorio de identidades o certificado.
Entrada: Se recibe como tipo de recurso al que se quiere conectar repositorio de certificados.
Resultados: Se verifica que el recurso sea de tipo repositorio de certificados, se lanza una excepción al intentar conectar al recurso repositorio de certificados.
Condiciones: Seleccionar el tipo de recurso al que se quiere conectar.

3.4 Despliegue

Durante esta fase, el equipo despliega los componentes en el ambiente de producción, estabiliza el despliegue, apoya a los usuarios, y obtiene la aprobación final del cliente del proyecto.

3.4.1 Plan de Despliegue

Para el despliegue del sistema se realizarán un conjunto de actividades dirigidas fundamentalmente al control de la aplicación por parte de expertos para verificar el correcto funcionamiento del sistema. Entras las actividades a controlar se encuentran la correcta conexión a los repositorios de certificados y de identidades respectivamente; así como la creación, asignación y publicación de los certificados digitales. Se recopilarán las inquietudes y opiniones de los clientes las cuales se tendrán en cuenta para posibles mejoras del sistema. Además de recogerá y analizará el grado de satisfacción de los clientes con el producto elaborado lo que permitirá acumular experiencia para la elaboración de futuros sistemas.

3.5 Conclusiones

En este capítulo se describió la propuesta de solución a partir de los diferentes diagramas utilizados, los cuales permiten comprender de forma más específica los diferentes elementos del diseño empleado para la modelación de la solución. Se describió la arquitectura y los patrones utilizados, lo que permitió organizar de forma eficiente el trabajo para tener mayor facilidad en la implementación del sistema, así como brindar las características necesarias para hacer al sistema lo más óptimo posible, escalable, flexible y de fácil mantenimiento. Además, se validó la propuesta de solución a través de las diferentes pruebas unitarias y de caja blanca realizadas a diferentes porciones de código lo que demostró el correcto funcionamiento del sistema.

CONCLUSIONES GENERALES

- ✓ Con el desarrollo del presente trabajo de diploma se logró diseñar e implementar un sistema que permite la integración de la Infraestructura de Clave Pública con el Sistema de Administración de Identidades.

- ✓ El sistema desarrollado logró darle cumplimiento a los procesos asociados a la gestión de los certificados digitales, puesto que permite la creación, publicación y asignación de dichos certificados a los usuarios o entidades del SAI.

- ✓ El sistema desarrollado logró la conexión necesaria y obligatoria a la base de datos Oracle 10gR2 para obtener solicitudes de publicación de certificados digitales, así como a los repositorios de identidades y certificados respectivamente, para obtener, modificar o actualizar los datos que maneja el SAI.

- ✓ El sistema desarrollado brinda una elevada seguridad en las operaciones que se realizarán puesto que fue desarrollado siguiendo las normas y estándares internacionales aprobados por la *Internet Engineering Task Force* (IETF) para el manejo con Infraestructuras de Clave Pública.

RECOMENDACIONES

Para garantizar el correcto funcionamiento de la solución propuesta en el presente trabajo de diploma se exponen como recomendaciones:

- ✓ Realizar el despliegue del sistema en las instituciones del Ministerio del Interior donde se utilizará para comprobar el correcto funcionamiento del sistema en un entorno de trabajo no controlado.

- ✓ Una vez desplegado el sistema realizar repetidos controles de calidad que permitan medir la eficiencia real del sistema para poder establecer mejoras en posteriores actualizaciones del sistema.

BIBLIOGRAFÍA

Altova. *UModel – UML tool for software Modeling and application Development*. [En línea] [Consultado el: 27 de Enero de 2010.] [Disponible en: http://www.altova.com/umodel.html.]

Baigorri, Extremo Unai y Borja, Sotomayor Basilio. *La plataforma .NET: ¿el futuro de la Web?*

Banco de España. *La infraestructura de clave pública del Banco de España.* [En línea] [Consultado el: 11 de Dicimbre de 2009.] [Disponible en: http://pki.bde.es/index.htm].

Certificadora. *X.509.* [En línea] [Consultado el: 27 de Enero de 2010.] [Disponible en: http://certificadora.com/certificados_digitales/x509.htm].

Comodo. *Certificate Manager for Enterprises.* [En línea] [Consultado el: 3 de Diciembre de 2009.] [Disponible en: http://www.comodo.com/business-security/digital-certificates/certificate-manager.php].

Departamento de Tratamiento de la Información y Codificación. *PKI o los cimientos de una criptografía de clave pública.* [En línea] [Consultado el: 8 de Diciembre de 2009.] [Disponible en: http://www.iec.csic.es/CRIPTonOMICon/susurros/susurros11.html].

Entrust Authority. *Managed Services PKI.* [En línea] [Consultado el: 4 de Diciembre de 2009.] [Disponible en: http://www.entrust.com/managed_services/index.htm].

Garbage Collector. RFC ("Request for Comments"). [En línea] [Consultado el: 20 de Enero de 2010.] [Disponible en: http://www.error500.net/garbagecollector/redes/rfc_request_for_comments.html].

García, Chenlo Bernardo; Sesma, Garbayo Marta y Troyano, Sanz Elisa. 2008. *Investigación y Desarrollo de Aplicaciones Criptográficas en Symbian.* Facultad de Informática.

Information Security Services. *VeriSign Managed PKI.* [En línea] [Consultado el: 7 de Diciembre de 2009.][Disponible en: http://www.iss.com.py/organizaciones.html].

LDAP. *Descripción de LDAP.* [En línea] [Consultado el: 22 de Enero de 2010.] [Disponible en: http://www.ldap-es.org/contenido/04/11/1.1.-descripción-de-ldap].

Network Working Group. 2006. *PKCS #10 v1.7: Certification Request Syntax Standard.*

Network Working Group. The RFC Archive. *Lightweight Directory Access Protocol (LDAP): Authentication Methods and Security Mechanisms.* [En línea] [Consultado el: 20 de Enero de 2010.] [Disponible en: http://www.rfc-archive.org/getrfc.php?rfc=4513].

Network Working Group. The RFC Archive. *Lightweight Directory Access Protocol (LDAP): Authentication Methods and Security Mechanisms.* [En línea] [Consultado el: 20 de Enero de 2010.] [Disponible en: http://www.rfc-archive.org/getrfc.php?rfc=4513]

Network Working Group. The RFC Archive. *Lightweight Directory Access Protocol (LDAP): Schema Definitions for X.509 Certificates.* [En línea] [Consultado el: 20 de Enero de 2010.] [Disponible en: http://www.rfc-archive.org/getrfc.php?rfc=4523].

Network Working Group. The RFC Archive. *Lightweight Directory Access Protocol (LDAP): Schema for User Applications.* [En línea] [Consultado el: 20 de Enero de 2010.] [Disponible en: http://www.rfc-archive.org/getrfc.php?rfc=4519].

Novell. *Using.NET C# LDAP Library*. [En línea] [Consultado el: 26 de Enero de 2010.] [Disponible en: http://www.novell.com/coolsolutions/feature/11204.html].

Osiris LMS. *Análisis y Diseño Orientado a Objetos*. [En línea] [Consultado el: 27 de Enero de 2010.] [Disponible en: http://login.osirislms.com/offline/uml/].

RSA Security Inc. *RSA Certificate Manager Scaling to new heights for secure e-business.* [En línea] [Consultado el: 9 de Diciembre de 2009.] [Disponible en: http://www.rsa.com/products/keon/datasheets/KCA_DS_0508-lowres.pdf.]

Web Taller. *La arquitectura PKI.* [En línea] [Consultado el: 11 de Diciembre de 2009.] [Disponible en: http://www.webtaller.com/maletin/articulos/arquitectura_pki.php].

REFERENCIAS BIBLIOGRÁFICAS

[1]. EuroLogic. *Infraestructura de Clave Pública – PKI.* Criptografía asimétrica. [En línea] [Citado el: 15 de Enero de 2010.] [Disponible en: http://www.eurologic.es/soluciones/que-es-pki.htm].

[2]. **Arnao, David Navarro. 2001.** *PKI: Claves para Entenderla.* ¿Qué es la Infraestructura de Clave Pública? [Citado el: 16 de Enero de 2010.]

[3]. EuroLogic. *Infraestructura de Clave Pública – PKI.* La Autoridad Certificadora. [En línea] [Citado el: 15 de Enero de 2010.] [Disponible en: http://www.eurologic.es/soluciones/que-es-pki.htm].

[4]. EuroLogic. *Infraestructura de Clave Pública – PKI.* La Autoridad de Registro. [En línea] [Citado el: 15 de Enero de 2010.] [Disponible en: http://www.eurologic.es/soluciones/que-es-pki.htm].

[5]. Mailxmail. *Red de área local. Administración y gestión (cuarta parte).* Capítulo 4: Red de área local. Autenticación: Componentes de una PKI (primera parte). [En línea] [Citado el: 19 de Enero de 2010.] [Disponible en: http://www.mailxmail.com/curso-red-administracion/red-area-local-autenticacion-componentes-pki-primera-parte].

[6]. EuroLogic. *Certificados Digitales.* Tipos de Certificados. [En línea] [Citado el: 16 de Enero de 2010.] [Disponible en: http://www.eurologic.es/conceptos/certificados%20digitales.htm].

[7]. RSA Laboratories. *Public-Key Cryptography Standards (PKCS).* [En línea] [Citado el: 18 de Enero de 2010.] [Disponible en: http://www.rsa.com/rsalabs/node.asp?id=2124].

[8]. **Blog de Manuel Gil**. *Formato Certificados X.509.* X.509 o claves RSA ---> Certificados PKCS#12. [En línea] [Citado el: 17 de Enero de 2010.] [Disponible en: http://blogs.sun.com/mgil/entry/font_id_wszh_size_5].

[9]. LDAP. *Ventajas en el uso de LDAP.* [En línea] [Citado el: 23 de Enero de 2010.] [Disponible en: http://www.ldap-es.org/contenido/04/11/1.2.-ventajas-en-el-uso-de-ldap].

[10]. **Capó, Capó Tomeu.** *Servicio de directorio OpenLDAP.* Introducción. [En línea] [Citado: 22 de Enero de 2010.] [Disponible en: http://mundopc.net/articulos/servicio-de-directorio-openldap/].

[11]. Programación en Castellano. *Introducción a UML .*Introducción. [En línea] [Citado el: 25 de Enero de 2010.] [Disponible en: http://www.programacion.com/tutorial/uml/1/].

[12]. **Cabrera, Armando; Solano, Raquel y Montalván, Mayra.** *PROCESOS DE INGENIERÍA DEL SOFTWARE.* MSF para Metodologías de Desarrollo Ágil (MSF4ASD). [Citado el: 23 de Enero de 2010.]

[13]. Elvex. *La plataforma .NET.* Introducción. [En línea] [Citado el: 29 de Enero de 2010.] [Disponible en: http://elvex.ugr.es/decsai/csharp/dotnet/index.xml].

[14]. **Recio, Francisco y Provencio, David.** *Algunas de las ventajas e inconvenientes de la plataforma .Net.* [En línea] [Citado el: 27 de Enero de 2010.] [Disponible en: http://www.desarrolloweb.com/articulos/1329.php].

[15]. MSDN**.** *Introducción al lenguaje C# y .NET Framework.* [En línea] [Citado el: 30 de Enero de 2010.] [Disponible en: http://msdn.microsoft.com/es-es/library/z1zx9t92 (VS.80).aspx].

Glosario

AC: Autoridad Certificadora.

API: Conjunto de funciones que ofrece una librería para ser usado por un sistema.

AR: Autoridad de Registro.

C#: Lenguaje de programación.

CISED: Centro de Identificación y Seguridad Digital.

IETF: Organismo internacional encargado de aprobar estándares o protocolos de internet.

.DER: Tipo de formato de certificado digital.

EJBCA: Es un tipo de autoridad certificadora.

LDAP: Protocolo utilizado para servicios de directorios.

MININT: Ministerio del Interior.

.P12: Tipo de formato de certificado digital.

.PEN: Tipo de formato de certificado digital.

PKCS: Estándar Criptográfico de Clave Pública.

PKI: Infraestructura de Clave Pública.

PROVISIONing: Es el sistema encargado de aprovisionar o manejar las cuentas de los usuarios.

RFC: Peticiones de Comentarios para creación de estándares o protocolos de Internet.

RSA: Algoritmo criptográfico utilizado para cifrar documentos.

SAI: Sistema de Administración de Identidades.

TIC: Tecnología de la Información.