Universidad de las Ciencias Informáticas

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

Análisis y Diseño del Componente de Seguridad del Sistema para la Gestión y Análisis de Información Estadística en la Salud Pública Cubana

Autores

Yunelsi Aylén Pompa Benítez

Donni Blas Delgado Consuegra

Tutor

Ing. Alfredo Manuel Guzmán Martínez

Ciudad de La Habana, Junio de 2010

"Año 52 de la Revolución"

Declaramos que somos los únicos autores del presente trabajo y autorizamos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmamos la presente a los 24 días del mes de junio del año 2010.

Donni Blas Delgado Consuegra

Autor

Yunelsi Aylén Pompa Benítez

Autora

Ing. Alfredo Manuel Guzmán Martínez

Tutor

TUTOR: **Ing. Alfredo M. Guzmán Martínez**: Graduado de Ingeniero en Ciencias Informáticas en la Universidad de las Ciencias Informáticas (UCI) en el año 2008. Se desempeña como profesor de la UCI, pertenece al departamento de Ingeniería de Software y Programación de la Facultad 7 y al departamento de Sistemas de Apoyo a la Salud en el Centro de Informática Médica (CESIM) en la rama de la salud. Ha impartido las asignaturas de Preparación para la Prueba de Nivel de Programación (PNP), Gráfico por Computadoras y Programación 3. Actualmente trabaja en el proyecto de Colaboración Médica.

Correo electrónico: amguzman@uci.cu

A Fidel Castro, quien hizo realidad un sueño tan esperado, creando esta maravillosa Universidad.

A mi mamá y mi abuela (mami), por quererme, apoyarme y sacrificarse en todos estos años.

A mi familia, por apoyarme durante la carrera y preocuparse por mí.

A Gualberto, por todo su amor, su entrega y apoyo.

A Donni, mi compañero de tesis, con el cual compartí este sueño tan esperado.

A Alfredo nuestro tutor, a Arieskien nuestro oponente y a todo el tribunal, por toda la ayuda que nos ofrecieron durante estos meses para lograr la realización de este trabajo.

A mis amistades de la UCI que han estado conmigo a lo largo de estos 5 años y han hecho inolvidable mi estancia en la universidad.

A todas esas personas que no mencione pero que fueron parte de este largo camino, convirtiendo los sueños en realidades.

A todos muchas gracias.

Aylén.

Muchas han sido las personas involucradas en el hecho de que yo este realizando este sueño…

En primer lugar quiero agradecerles a mis padres, a los que les debo todo y con los que estaré siempre en deuda, por quererme tanto y guiarme hasta aquí, por darme todo lo que tuvieron a su alcance para que alcanzara mis metas, por servirme siempre de ejemplos, decirles que los admiro mucho y que no solo le agradezco sus consejos y el apoyo constante, sino además, el privilegio de ser su hijo.

A mi hermana Naliny que la quiero mucho, a mi cuñado Arturo y a mis sobrinitos Mariam y Marcos.

A mi familia, en especial a mi tía Olguita, por ayudarme y apoyarme siempre en todo.

A todos mis amigos, por estar conmigo todo este tiempo y compartir tantas cosas juntos.

A Melba, mi novia por su amor y comprensión.

A mi compañera de tesis, Yunelsi por compartir conmigo la realización de este trabajo.

A nuestro tutor Alfredo, a Arieskien nuestro oponente y todo el tribunal por su ayuda y enseñanza durante la realización de este trabajo.

A todos los que de una forma u otra han contribuido a la realización de este trabajo de diploma, de corazón, muchas gracias.

Donni.

Dedicado a las personas más importantes de mi vida, las cuales siempre confiaron que este día llegaría. A ustedes con todo mi amor:

A Niurka y CHela, porque ustedes son lo más grande que tengo, por ser mis guías, por sacrificarse tanto por mí; gracias por apoyarme en cada momento de mi vida y por quererme tanto; las quiero mucho.

A mis hermanos Angélica y José Alejandro, porque ustedes son el regalo más preciado que me ha dado la vida.

A Digna, por apoyarme en todo momento, quererme y estar siempre pendiente de mí.

A Gualberto, por ser mi gran amor, por quererme, darme fuerzas cuando pensaba que no era posible y hacer que mis días fueran cada vez mejores.

A todos los quiero mucho.

Aylén.

A mis padres que siempre confiaron en mí y me impulsaron a seguir adelante, por eso quiero regalarles este momento y honrarlos por tanto amor y dedicación.

A mi hermana Naliny, a mí cuñado Arturo y a mis sobrinos Mariam y Marcos.

A mi familia.

A mis amigos.

Donni.

Resumen

El Sistema para la Gestión y Análisis de la Información de Estadística en la Salud Pública Cubana, depende completamente de la conectividad con el Sistema de Autenticación, Autorización y Auditoría (SAAA) para autenticar los usuarios, además no posee la flexibilidad para poder realizar este proceso de autenticación mediante un Protocolo Ligero de Acceso a Directorios (LDAP), siendo estas las razones principales de la realización del presente trabajo, el cual se centra en el análisis y diseño de un componente de seguridad que garantice de forma adecuada la autenticación en entornos conectados o desconectados.

Se propone como metodología de desarrollo el Proceso Unificado de Desarrollo de Software (RUP), ya que es un marco de trabajo genérico que puede especializarse para una gran variedad de sistemas de software, para diferentes áreas de aplicación, diferentes niveles de aptitud y tamaños de proyecto; y como herramienta de modelado Enterprise Architec 7.1, pues la misma cubre el desarrollo de software desde el paso de los requerimientos a través de las etapas del análisis y modelo de diseño.

El componente contará con opciones de configuración para la conexión con el SAAA o con un LDAP, para el caso de que no exista conectividad brindará la posibilidad a los usuarios de autenticarse de forma local en el sistema, además permitirá guardar un registro de todas las operaciones realizadas por un usuario autenticado en el sistema.

Palabras claves: Autenticación, Autorización, Auditoría, Componente y Seguridad.

INTRODUCCIÓN **1**

CAPÍTULO 1. FUNDAMENTACIÓN TEÓRICA **5**

1.1 INTRODUCCIÓN 5
1.2 CONTROL DE ACCESO 5
1.3 ESTUDIO DEL ARTE 6
1.4 ESTÁNDARES DE SEGURIDAD 8
1.5 MODELOS DE AUTENTICACIÓN 9
1.6 TENDENCIAS Y TECNOLOGÍAS ACTUALES 10
1.6.1 Arquitectura de software 10
1.6.2 Estilos arquitectónicos 11
1.6.4 Lenguaje de marcas 12
1.6.5 Lenguajes de programación 13
1.6.6 CMMI (Capability Maturity Model Integration) 15
1.6.7 Metodología de desarrollo 16
1.6.8 Lenguaje de modelado 17
1.6.9 Herramienta de modelado 18
1.6.10 Base de datos 18
1.6.11 Herramienta de desarrollo 20
1.6.12 Ambiente de desarrollo 20
1.7 CONCLUSIONES 21

CAPÍTULO 2. CARACTERÍSTICAS DEL SISTEMA **22**

2.1 INTRODUCCIÓN 22
2.2 MODELO DE DOMINIO 22
2.3 PROPUESTA DEL SISTEMA 23
2.4 ESPECIFICACIÓN DE LOS REQUISITOS DE SOFTWARE 24
2.5 MODELO DE CASOS DE USO DEL SISTEMA 26
2.6 PROTOTIPOS DE INTERFAZ DE USUARIO 39
2.7 CONCLUSIONES 40

CAPÍTULO 3: ANÁLISIS Y DISEÑO DEL SISTEMA **41**

3.1 INTRODUCCIÓN 41
3.2 MODELO DE ANÁLISIS 41
3.3 PATRÓN DE ARQUITECTURA A UTILIZAR 44
3.4 MODELO DEL DISEÑO 45
3.5 DISEÑO DE LA BASE DE DATOS 49
3.6 CONCLUSIONES 52

CONCLUSIONES 53

RECOMENDACIONES 54

REFERENCIAS BIBLIOGRÁFICAS 55

BIBLIOGRAFÍA 58

ANEXOS 61

ANEXO 1: PROTOTIPOS DE INTERFAZ DE USUARIO 61
ANEXO 2: DIAGRAMAS DE CLASES DEL ANÁLISIS 66
ANEXO 3: DIAGRAMAS DE COLABORACIÓN DEL ANÁLISIS 66
ANEXO 4: DIAGRAMAS DE SECUENCIA 68

GLOSARIO DE TÉRMINOS 72

Introducción

El mundo de hoy, está inmerso en una revolución tecnológica, la cual encuentra su principal impulso en el acceso y en la capacidad de procesamiento de información sobre todos los temas y sectores de la actividad humana. La informática ha contribuido a que culturas y sociedades se desarrollen tanto en la esfera socioeconómica como política, permitiéndoles alcanzar un mejoramiento en la calidad de vida y eficiencia en la prestación de servicios. Cuba, aunque se encuentra fuertemente bloqueada por el imperialismo, hace todo lo posible para estar a la altura de los países desarrollados en esta rama, llevando a cabo la informatización de todos los sectores del país.

A través del programa que se lleva a cabo en el Estado Cubano para la informatización de la sociedad, se ha visto un notable incremento en el uso de las tecnologías de la información, destacándose el Sistema Nacional de Salud (SNS) donde se prioriza el desarrollo integral de soluciones informáticas con el objetivo de mejorar la calidad de la gestión económica, estadística y sanitaria del mismo.

Los antecedentes de la aplicación de dichas soluciones en el SNS se remontan a la década del 70 del siglo pasado, en los albores de estos años se creó la primera Comisión del Ministerio de Salud Pública que se dedicó al estudio de las necesidades de desarrollo de la informática en la Salud Pública Cubana. [1]

En la segunda mitad de la citada década se crea en 1976 el Instituto de Desarrollo de la Salud, fruto ya de una política definida del Ministerio de Salud Pública (MINSAP) para el perfeccionamiento de los recursos humanos y se introduce la informática como herramienta necesaria e imprescindible para el análisis de los problemas sanitarios en las diferentes especialidades. [2]

Después de sucesivos cambios organizativos y estructurales durante los años posteriores en 1998 fue creado el Centro de Desarrollo Informático de Salud Pública con la misión de enfrentar el diseño e implementación de los sistemas informáticos a los diferentes niveles del SNS, encaminados a la esfera económica de control de medicamentos y estadísticas. [3]

Actualmente se trabaja integradamente en el desarrollo de un grupo de aplicaciones para la informatización del sector de la salud. En su desarrollo e implementación participan diferentes empresas

del Ministerio de la Informática y Comunicaciones (MIC) y el MINSAP como son: la Empresa Nacional de Software (DESOFT), la Empresa Especializada en Soluciones Informáticas (SOFTEL), la Universidad de las Ciencias Informáticas (UCI) y el Centro de Desarrollo Informático para la Salud Pública (CEDISAP). [4]

Para ello, el MINSAP ha definido un grupo de premisas que incluyen los últimos adelantos de la informática y las comunicaciones tales como la utilización de XML, Web Service, Linux, MySQL, PHP que facilitan la integración de las aplicaciones, y la compatibilidad de los productos a desarrollar. [5]

La UCI como uno de los pilares fundamentales concebidos por la Revolución Cubana para desarrollar el proceso de informatización de la sociedad cubana, cuenta con 9 facultades, donde cada una desarrolla en conjunto con su plan de estudio un segundo perfil, hacia el cual orientan sus proyectos y productos de software.

La Facultad 7 cuenta con el Centro de Informática Médica (CESIM) que es responsable de crear y dar soporte a los productos que se dedican a la salud pública, el cual se divide a su vez en varios departamentos, siendo uno de estos Sistemas de Apoyo a la Salud (SAS), donde se trabaja actualmente en la implementación de varios sistemas informáticos como son Balance Material, Colaboración Médica y el Sistema de Información Estadístico Complementario de Salud (SIE-C), con los que se pretende mejorar y agilizar los procesos que se realizan en el MINSAP.

Específicamente el SIE-C trabaja en el desarrollo del Sistema para la Gestión y Análisis de Información Estadística en la Salud Pública Cubana con el cual se pretende sustituir un grupo de soluciones de gestión de la información de carácter local existentes en diversos lugares del país que no se ha estandarizado a nivel nacional, lo que ha propiciado que en estos momentos cuenten con pocas funcionalidades y no respondan a las exigencias de las necesidades estadísticas actuales.

Con el fin de mantener la confidencialidad, integridad y confiabilidad de la información que se gestiona en los diferentes software de salud debido a la importancia que la misma posee, se hace imprescindible contar con un sistema de seguridad informática potente que incluya aspectos relacionados con políticas, estándares, valoración de riesgos y otros elementos importantes para lograr una correcta administración de los recursos tecnológicos y de esta forma evitar la destrucción, el uso no autorizado o robo de información.

En este sentido, las empresas que enfocan su trabajo hacia el desarrollo de productos y servicios informáticos para la salud, han creado diferentes sistemas entre los que sobresale el Sistema de Autenticación, Autorización y Auditoría (SAAA), el cual fue puesto en marcha en diciembre del 2003 por SOFTEL, con el objetivo de manejar toda la información referente a la autenticación y autorización del personal de salud con acceso al mismo y lograr la centralización nacional de todo el proceso de gestión de la seguridad en los sistemas sanitarios.

A pesar de todo lo que se ha avanzado en el trabajo para mejorar la funcionalidad y la seguridad de las aplicaciones del SNS, todavía algunas presentan limitaciones, siendo un ejemplo el Sistema para la Gestión y Análisis de Información en la Salud Pública Cubana, el cual depende totalmente de la conectividad con el SAAA para gestionar la autenticación y autorización de sus usuarios, además de que actualmente el sistema sólo gestiona la seguridad a través del SAAA, lo que impide que este proceso se pueda realizar haciendo uso de un Protocolo Ligero de Acceso a Directorios (LDAP).

Tomando en consideración esta problemática se hace necesario realizar el análisis y diseño del componente de seguridad del Sistema para la Gestión y Análisis de Información Estadística en la Salud Pública Cubana que permita su funcionamiento en ambiente desconectado, es decir, cuando no exista conexión entre dicho sistema y el SAAA y que aumente la flexibilidad del sistema, permitiendo realizar el proceso de autenticación mediante un LDAP.

En este sentido el **problema científico** se enfoca en: ¿Cómo garantizar la ejecución adecuada del proceso de autenticación, autorización y auditoría en el Sistema para la Gestión y Análisis de Información Estadística en la Salud Pública Cubana en entornos conectados o desconectados?

El problema planteado tiene como **objeto de estudio**: Proceso de gestión de la autenticación, autorización y auditoría de las aplicaciones del Sistema Nacional de Salud. A partir del objeto de estudio se plantea como **campo de acción**: Proceso de gestión de la autenticación, autorización y auditoría del Sistema para la Gestión y Análisis de Información Estadística en la Salud Pública Cubana.

Como **objetivo general** se define: Realizar análisis y diseño del componente de seguridad que garantice una ejecución adecuada del proceso de gestión de la autenticación, autorización y auditoría del Sistema

para la Gestión y Análisis de Información Estadística en la Salud Pública Cubana en entornos conectados o desconectados.

Las **Tareas de la Investigación** que se llevan a cabo para dar solución al objetivo trazado son:

- Analizar el estado del arte de los sistemas de gestión de seguridad para aplicaciones de software.
- Valorar las tendencias tecnológicas actuales en cuanto a estándares y políticas para la implementación de la seguridad en soluciones informáticas.
- Analizar el funcionamiento del Protocolo Ligero de Acceso a Directorios (LDAP).
- Analizar funcionamiento de sistemas en entorno desconectado.
- Valorar lenguajes, herramientas y la metodología a utilizar.
- Definir los requisitos de software.
- Diseñar la Base de Datos.
- Elaborar el análisis y diseño del componente de seguridad.
- Diseñar el prototipo no funcional del componente de seguridad.

Este trabajo ha sido organizado en tres capítulos, los cuales se estructuran de la siguiente manera:

En el **capítulo 1 Fundamentación Teórica** se aborda el estudio del arte del tema en cuestión y algunos estándares utilizados para la seguridad en las aplicaciones, se detallan modelos para la autenticación. Además, se describen las tecnologías, metodología y herramientas a utilizar para dar solución al problema planteado.

En el **capítulo 2 Características del Sistema** se realiza una descripción detallada de las características que conforman el sistema propuesto, se definen los requerimientos funcionales y no funcionales, y se expone el modelo conceptual de dominio.

En el **capítulo 3 Análisis y Diseño del Sistema** se modelan los diagramas de clases del análisis y del diseño, así como los diagramas de interacción correspondientes. Se hace referencia al patrón de arquitectura a utilizar para la construcción del diseño propuesto.

Capítulo 1. Fundamentación Teórica

1.1 Introducción

La seguridad de una aplicación depende de la atención que le presten los programadores y administradores durante el período de duración de la aplicación. Dado que van surgiendo nuevas amenazas, las aplicaciones deben examinarse con el fin de identificar posibles errores. La autenticación y autorización son factores claves para garantizar la seguridad. La implementación de un sistema que conjuntamente pueda garantizar que estas dos funcionalidades puedan ser autenticas y seguras lleva un conjunto de investigaciones con el fin de garantizar la calidad del producto final a obtener.

En el presente capítulo se realiza un estudio del tema en cuestión y se exponen elementos y estándares de seguridad, tomándolos como punto de partida para la concepción de la solución planteada. Además, se analizan las tecnologías, metodología, herramientas de software y varios estilos arquitectónicos.

1.2 Control de acceso

El control de acceso de los diferentes usuarios a las aplicaciones constituye una poderosa herramienta para proteger la entrada a un sistema completo o sólo a ciertos directorios concretos e incluso a ficheros o programas individuales; este control consta generalmente de dos pasos: [6]

- **Autenticación:** Es el proceso de verificación de la identidad digital de un remitente de una comunicación que hace una petición para conectarse a un sistema. El remitente puede ser una persona que usa un ordenador u otro medio electrónico, un ordenador por sí mismo o un programa. Es un modo de asegurar que los usuarios son realmente quienes dicen ser.
- **Autorización:** Es el proceso por el cual se autoriza al usuario identificado a acceder a determinados recursos del sistema, es decir, se comprueba que los usuarios con identidad válida solo tengan acceso aquellos recursos sobre los cuales tengan privilegios.

Es importante implementar una fuerte política de **Auditoría** a través de la cual quedan registrados todos los accesos y peticiones realizadas por los usuarios, donde siempre quedan almacenados un conjunto de datos como: usuario, servicio que consume, componente y dirección IP desde el cual accede, fecha, hora, tipo de traza que genera. Además, una descripción que permite aumentar el nivel de detalles acerca de las acciones de los usuarios, lo que contribuye a facilitar el proceso de análisis de las trazas.

1.3 Estudio del arte

A lo largo de la historia ha existido la necesidad de controlar el acceso a ciertas áreas, esta necesidad motivada inicialmente por el propio sentimiento de pertenencia, resulta el motor impulsor para que comenzaran a utilizarse mecanismos de control de accesos. Hoy en día se ha evolucionado tecnológicamente, y en este sentido se han desarrollado y automatizado sistemas que se encargan de controlar el acceso sobre áreas o informaciones que se desean proteger.

Ejemplos de sistemas existentes en la UCI:

Centro de Control para el Sistema de Información para la Salud

Versión renovada del componente de seguridad SAAA puesto en marcha por SOFTEL. Se desarrolló en el año 2007 y el mismo heredó todos los requerimientos de su antecesor.

El Centro Control aportó los siguientes beneficios: [7]

- Homologación de datos de los usuarios con el Registro de Ciudadanos debido a necesidades de los nuevos módulos en desarrollo.
- El perfeccionamiento de los procesos de gestión de usuarios y asignación de privilegios.
- Eliminación de los usuarios y sus respectivas trazas por el sistema, es lógica y no física, fortaleciéndose el proceso de auditoría.
- Generación automática de ficheros de configuración y despliegue.

Componente de Seguridad para aplicaciones del Área Temática Sistemas de Apoyo a la Salud

En el año 2008 la dirección del Departamento SAS decidió elaborar un nuevo SAAA-SAS que unificara las fortalezas de sus sistemas predecesores y permitiera gestionar los requerimientos de seguridad de los productos que allí se desarrollan. Este sin dudas logró solventar algunas dificultades pues permitió:

- Gestionar de forma eficiente los requerimientos de seguridad para todos los sistemas informáticos que pertenecen al Departamento de SAS, proporcionando facilidades de mantenimiento a estos sistemas y agilizando los procesos de construcción de nuevas aplicaciones.

- Aumento de los niveles de integración entre los diferentes sistemas del Departamento de SAS, evitando que cada uno posea de manera aislada la administración de sus usuarios, esta información será almacenada y gestionada centralizadamente.
- Perfeccionamiento de los procesos de gestión de usuarios y asignación de privilegios.

Sistema de Autenticación, Autorización y Auditoría para los productos desarrollados en el Área Temática Sistemas de Apoyo a la Salud (SAAA-SAS) v 1.1

En el año 2009 se desarrolló el SAAA-SAS v1.1 el cual aumentó las funcionalidades de sus antecesores. De forma general, la realización del SAAA-SAS aporta beneficios tales como: [8]

- Mayor gestión de los requerimientos de seguridad para los productos desarrollados mediante un servicio web desarrollado que utiliza el patrón proxy que sirve de interfaz a las peticiones de los sistemas clientes y garantiza el acceso seguro a los mismos.
- Mejoramiento del rendimiento del sistema aun cuando el volumen de datos relacionados con la información auditable crezca, debido a la implementación de una política de balanceo de trazas para las tablas donde se almacena la información de esta naturaleza.
- Mejor apariencia visual de los reportes relacionados con la Auditoría, que permite al usuario seleccionar los elementos que le resulten de mayor interés y mostrar la información de forma más intuitiva.

Sistema de Control de Acceso a Comedores (CONTACC)

El sistema consta de un componente principal: el CONTACC, ubicada en cada una de las puertas de los comedores. Esta se conecta a un Servidor Web donde se encuentra el Sistema de Gestión de Comensales, el cual hace uso de un Servidor de Base de Datos, que provee al mismo de información sobre las personas que acceden por las puertas. En el caso de la aplicación cliente, usa bases de datos locales para comprobar la existencia de las personas consultando la información, y para registrar los accesos de las personas si no hay conexión con el Servidor Web. También se conecta a un repositorio de fotos para buscar las fotos de las personas.

1.4 Estándares de seguridad

XACML 2.0 (eXtensible Access Control Markup Language)

XACML es un nuevo lenguaje basado en XML para la protección de datos, ha sido desarrollado por SUN y aceptado por el consorcio OASIS (Organization for the Advancement of Structured Information Standards) como estándar. La implementación de XACML ha sido liberada como software libre.

XACML describe un lenguaje y una política de control de solicitud de acceso/respuesta (ambos escritos en XML). Esta política se utiliza para describir los requisitos generales de control de acceso, y tiene puntos de extensión estándar para la definición de nuevas funciones y tipos de datos, combinando la lógica. La solicitud/respuesta le permite formar una consulta para preguntar si o no, una determinada acción se debería permitir, e interpretar el resultado. La respuesta incluye siempre si en la solicitud se debe permitir el uso de uno de los cuatro valores: Permiso, Denegado, Indeterminado (se produjo un error o algún valor requerido no se encuentra, así que una decisión no puede ser) o No Aplicable (la solicitud no puede ser contestada por este servicio).

Ventajas que presenta XACML:

- Utiliza un lenguaje unificado y portable para expresar sus políticas.
- Facilita el intercambio de políticas con otros sistemas.
- Define sus políticas con un modelo flexible y adaptado al nivel de detalle que sea necesario.

SAML 2.0 (Security Assertion Markup Language)

SAML aprobado por OASIS es un estándar para intercambiar información de autenticación y autorización entre dominios. Está diseñado para ofrecer Single Sign-On (SSO) para interacciones automáticas o manuales entre sistemas. Este permite el intercambio de información de autenticación y autorización sobre usuarios, dispositivos o cualquier entidad identificable llamados sujetos. Usando sintaxis de XML, SAML define el protocolo petición-respuesta por el cual los sistemas aceptan o rechazan sujetos basados en aserciones así como enlaces y perfiles. Además, incluye la información necesaria para identificar y validar el contenido de la aserción, tales como la identidad del emisor de la afirmación, el período de validez y la firma digital de la afirmación.

1.5 Modelos de autenticación

Modelo de autenticación: centralizado y descentralizado

En la autenticación *centralizada,* los usuarios y sus claves se ubican en un repositorio central. Las diferentes aplicaciones se configuran para identificar este lugar y hacer la autenticación contra el repositorio.

En el modelo *descentralizado*, cada servicio de la red maneja sus claves de forma independiente, por ejemplo los usuarios de Oracle, los administradores de un sitio web; cada uno de estas aplicaciones maneja por separado sus claves y las mismas no son compartidas.

LDAP (Lightweight Directory Access Protocol)

Protocolo Ligero de Acceso a Directorios 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 a la que pueden realizarse consultas. Está basado en el estándar X.500. 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, entre otros datos). LDAP es un protocolo de acceso unificado a un conjunto de información sobre una red. [9]

Características de un directorio LDAP: [10]

- 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.
- Usa un sistema jerárquico de almacenamiento de información.
- Permite múltiples directorios independientes.
- Funciona sobre TCP/IP y SSL (Secure Socket Layer).
- La mayoría de las aplicaciones disponen de soporte para LDAP.
- La mayoría de servidores LDAP son fáciles de instalar, mantener y optimizar.

Dadas las características de un LDAP sus usos más comunes son: [11]

- Directorios de información.

- Sistemas de autenticación/autorización centralizada.
- Sistemas de correo electrónico.
- Servidores de certificados públicos y llaves de seguridad.
- Autenticación única o SSO para la personalización de aplicaciones.
- Perfiles de usuarios centralizados.
- Libretas de direcciones compartidas.

Sistema de Autenticación, Autorización y Auditoría (SAAA)

El SAAA es un sistema desarrollado por SOFTEL para garantizar y centralizar la seguridad de las aplicaciones de salud en Cuba, pero esta empresa no está autorizada para compartir ningún tipo de información relacionada con el mismo, por lo que no pueden dar a conocer los servicios que brinda el software. Debido a informaciones dadas por el Departamento SAS, se conoce que el SAAA cuenta con un limitado nivel en cuanto a la gestión para la administración de usuarios y no permite validar y homologar los datos personales de los usuarios con un registro externo que contenga los datos de los ciudadanos. El SAAA define roles de administrador, editor y visualizador en todos los niveles del Sistema Nacional de Salud como son: unidades de salud, municipal, provincial y nacional.

1.6 Tendencias y tecnologías actuales

En este punto se trata los conceptos fundamentales relacionados con la metodología, tecnologías, y herramientas que se proponen para el desarrollo del componente de seguridad.

1.6.1 Arquitectura de software

En la actualidad existen un sin número de definiciones de Arquitectura de Software, pero ninguna que sea adoptada completamente por la totalidad de los arquitectos del mundo. Aunque es válido aclarar que ninguna definición reprocha a la otra, sino que es otro modo de ver las cosas.

Entre las definiciones más reconocidas se encuentra la de Paul Clements: “La Arquitectura de Software es, a grandes rasgos, una vista del sistema que incluye los componentes principales del mismo, la conducta de esos componentes según se la percibe desde el resto del sistema y las formas en que los componentes interactúan y se coordinan para alcanzar la misión del sistema. La vista arquitectónica es

una vista abstracta, aportando el más alto nivel de comprensión y la supresión o diferimiento del detalle inherente a la mayor parte de las abstracciones". [12]

Por otra parte, la definición más reconocida internacionalmente por muchos arquitectos tributa a la industria y pertenece a la IEEE 1471, la cual cita así: "La Arquitectura de Software es la organización fundamental de un sistema encarnada en sus componentes, las relaciones entre ellos y el ambiente y los principios que orientan su diseño y evolución." [13]

1.6.2 Estilos arquitectónicos

Los estilos y patrones permiten reutilizar experiencias anteriores en la resolución de problemas bien definidos de manera que se estandariza el comportamiento a seguir.

- Un estilo contempla las cuatro "C" de la Arquitectura de Software (elementos, conectores, configuraciones y restricciones), por sus siglas en inglés.
- Sirve para sintetizar estructuras de soluciones que luego serán refinadas a través del diseño.
- Definen los patrones posibles de las aplicaciones, evitando errores arquitectónicos.
- Permiten evaluar arquitecturas alternativas con ventajas y desventajas conocidas ante diferentes conjuntos de requerimientos no funcionales.

Modelo-Vista-Controlador

Este patrón se utiliza cuando es necesario modularizar la interfaz de usuario, las reglas de negocios y el control de eventos. El modelo administra el comportamiento y los datos del dominio de aplicación, responde a requerimientos de información sobre su estado (usualmente formulados desde la vista) y responde a instrucciones de cambiar el estado (habitualmente desde el controlador). Mantiene el conocimiento del sistema. No depende de ninguna vista y de ningún controlador. La vista maneja la visualización de la información mientras que el controlador interpreta las acciones del ratón y el teclado, informando al modelo y/o a la vista para que cambien según resulte apropiado. [14]

Arquitectura en Capas

Garlan y Shaw la definen como una organización jerárquica donde cada capa proporciona servicios a la capa inmediatamente superior y se sirve de las prestaciones que le brinda la inmediatamente inferior. Los componentes de cada capa consisten en conjuntos de procedimientos; las interacciones entre las mismas,

usualmente proceden por invocación de dichos procedimientos y por definición, los niveles de abajo no pueden usar funcionalidad ofrecida por los de arriba. [15]

1.6.4 Lenguaje de marcas

Con el desarrollo de los programas que procesan texto surgen los primeros lenguajes informáticos especializados en tareas de descripción y estructuración de información: los lenguajes de marcas. La base de estos lenguajes es el uso de marcas o etiquetas con las cuales se crean las estructuras de datos que luego se procesarán. Estos lenguajes almacenan sus datos y programas en formato de texto, lo que trae consigo una mayor portabilidad de los programas para ser implementados en cualquier arquitectura de hardware o software; facilidad de mantenimiento debido a que los archivos son más entendibles por humanos; y además porque los datos y programas pueden ser creados y modificados por cualquier editor de textos. [16]

Existen tres clases de lenguajes de marcas: [17]

- **Marcado de presentación:** Indica el formato del texto. Este tipo de marcado es útil para maquetar la presentación de un documento para su lectura, pero resulta insuficiente para el procesamiento automático de la información. El marcado de presentación resulta más fácil de elaborar, sobre todo para cantidades pequeñas de información, sin embargo, resulta complicado de mantener o modificar, por lo que su uso se ha ido reduciendo en proyectos grandes en favor de otros tipos de marcado más estructurados.
- **Marcado procedimental:** El mismo está enfocado hacia la presentación del texto, sin embargo, también es visible para el usuario que edita el texto. El programa que representa el documento debe interpretar el código en el mismo orden en que aparece. Por ejemplo, para formatear un título, debe haber una serie de directivas inmediatamente antes del texto en cuestión, indicándole al software instrucciones tales como centrar, aumentar el tamaño de la fuente, o cambiar a negrita. Inmediatamente después del título deberá haber etiquetas inversas que reviertan estos efectos. En sistemas más avanzados se utilizan macros o pilas que facilitan el trabajo.
- **Marcado descriptivo o semántico:** Utiliza etiquetas para describir los fragmentos de texto, pero sin especificar cómo deben ser representados o en qué orden, lo cual le da al mismo una gran

flexibilidad. El marcado descriptivo también simplifica la tarea de reformatear un texto, debido a que la información del formato está separada del propio contenido.

XML (Extensible Markup Language)

El estándar Lenguaje de Marcas Ampliable, es un metalenguaje extensible de etiquetas. Se diseñó para evitar la complejidad del Lenguaje de Marcado Generalizado (SGML). La particularidad más importante del XML es que no posee etiquetas prefijadas con anterioridad, ya que es el propio diseñador quien las crea, dependiendo del contenido del documento.

Tiene como ventajas: [18]

- Los programadores pueden diseñar sus propios tipos de documentos usando XML.
- XML puede dar más y mejores facilidades para la representación en los visualizadores.
- Elimina muchas de las complejidades de SGLM en favor de la flexibilidad del modelo, con lo que la escritura de programas para manejar XML será más sencilla.
- La información será más accesible y reutilizable, porque la flexibilidad de las etiquetas de XML pueden utilizarse sin tener que amoldarse a reglas específicas de un fabricante.
- Es extensible, ya que después de diseñado y puesto en producción, es posible extender XML con la adición de nuevas etiquetas, de modo que se pueda continuar utilizando sin complicación alguna.
- La información contenida puede ser fácil de usar, porque las habilidades hipertextuales de XML son mayores que las de HTML.
- Es multiplataforma.

1.6.5 Lenguajes de programación

Un lenguaje de programación es un conjunto de símbolos y reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos y expresiones. Son herramientas que permiten crear programas y software. Además, facilitan la tarea de programación, ya que disponen de formas adecuadas que permiten ser leídas y escritas por personas, a su vez resultan independientes del modelo de computador a utilizar.

C++

C++ es un lenguaje de programación diseñado a mediados de los años 1980 por Bjarne Stroustrup. La intención de su creación fue extender al exitoso lenguaje de programación C con mecanismos que permitan la manipulación de objetos. El nombre actual, C++, fue adoptado en el año 1983 con el objetivo de indicar que el lenguaje era un "C mejorado". [19]

El lenguaje C++ soporta varios estilos de programación, por ejemplo: procedural, orientado a objetos; intenta ser tan eficiente y portable como lo es el lenguaje C. Además, permite la sobrecarga de los operadores, la inclusión de directivas del preprocesador y los tipos genéricos, entre otras funcionalidades. Soporta plenamente el polimorfismo y la herencia, destacándose la existencia de la herencia múltiple. [20]

Este lenguaje ha sido utilizado para el desarrollo de disímiles aplicaciones, que van desde sistemas de gestión y videojuegos, hasta sistemas operativos. Entre ellos se encuentran los Sistemas Operativos Unix, Linux y Windows.

CSharp (C#)

C# facilita la integración del componente de seguridad con el Sistema de Gestión y Análisis de Información Estadística en la Salud Pública Cubana. C# es orientado a objetos y está estandarizado por Microsoft como parte de su plataforma .NET. El lenguaje ha sido diseñado para ser robusto, moderno y sencillo, funciona bajo un entorno de recolección automática de la memoria, lo que aumenta la productividad del desarrollador. Con C# se han desarrollado diversos sistemas, sobresalen las aplicaciones de escritorio, las aplicaciones web y los componentes reutilizables.

Características principales de C#:

- **Sencillez de uso:** Elimina elementos añadidos por otros lenguajes y que facilitan su uso y comprensión. Además, no se incorporan elementos poco útiles.
- **Modernidad:** Al ser un lenguaje de última generación, incorpora elementos que son muy útiles para el programador.
- **Orientado a objetos:** No permite la inclusión de funciones ni variables globales que no estén incluidos en una definición de tipos, por lo que la orientación a objetos es más pura y clara. Además soporta todas las características del paradigma de programación orientada a objetos, como son la encapsulación, la herencia y el polimorfismo.

- **Orientado a componentes:** La propia sintaxis de C# incluye elementos propios del diseño de componentes que otros lenguajes tienen que simular. La sintaxis de C# incluye por ejemplo formas de definir propiedades, eventos o atributos.
- **Seguridad de tipos:** Incluye mecanismos de control de acceso a tipos de datos, lo que garantiza que no se produzcan errores difíciles de detectar como un acceso a memoria de ningún objeto. Para ello, el lenguaje provee de una serie de normas de sintaxis, como por ejemplo no realizar conversiones entre tipos que no sean compatibles y se puede controlar los desbordamientos en operaciones aritméticas, produciéndose excepciones cuando se produzcan. Además, no se pueden usar variables no inicializadas previamente.

1.6.6 CMMI (Capability Maturity Model Integration)

CMMI es un modelo de referencia para el crecimiento de capacidades y madurez, que se enfoca tanto en procesos de Administración como de Ingeniería de Sistemas y Software. Describe áreas de procesos y prácticas asociadas a cada nivel ayudando a las organizaciones a elevar la madurez y capacidad de su proceso de software. El modelo de CMMI consta de dos representaciones continúo y escalonado, además define 6 niveles para medir la capacidad de los procesos. En la universidad se utiliza el escalonado, donde el proceso está encaminado a alcanzar una certificación internacional del nivel 2 de madurez, y así convertirse en la primera empresa de Cuba certificada en este modelo. Dentro del nivel 2 se encuentran las siguientes áreas de procesos:

- MA: Medición y Análisis.
- PP: Planeación del Proyecto.
- REQM: Administración de Requisitos.
- PMC: Monitoreo y Control del Proyecto.
- CM: Administración de la Configuración.
- SAM: Administración de Acuerdos con Proveedores.
- PPQM: Aseguramiento de la Calidad a Procesos y Productos.

REQM

Esta área de proceso tiene como propósito gestionar los requisitos de los productos del proyecto y componentes del producto e identificar inconsistencias entre dichos requisitos y la planificación del proyecto. Además, gestiona todos los requisitos recibidos y generados por el proyecto, incluyendo los técnicos, no técnicos, así como aquellos añadidos al proyecto por la organización. Una parte de la gestión de requisitos será documentar los cambios a los requisitos y mantener la trazabilidad bidireccional entre la fuente de los requisitos y los productos.

Debido a que en la universidad se tiene experiencia con la metodología del Proceso Unificado de Desarrollo Software (RUP), se decidió utilizarla en el modelo CMMI. Además, dado que el enfoque de RUP es orientado al modelo se utilizará un lenguaje concreto y bien definido (UML 2.0).

1.6.7 Metodología de desarrollo

Las metodologías de desarrollo de software son un marco de trabajo usado para estructurar, planificar y controlar el proceso de desarrollo en sistemas de información. La finalidad de una metodología de desarrollo es garantizar la eficiencia en el proceso de generación de software.

RUP

El Proceso Unificado de Desarrollo Software es un marco de trabajo genérico que puede especializarse para una gran variedad de sistemas de software, para diferentes áreas de aplicación, diferentes tipos de organización, diferentes niveles de aptitud y tamaños de proyecto. RUP divide el proceso de desarrollo en ciclos, teniendo un producto final al culminar cada una de ellos, estos a la vez se dividen en fases que finalizan con un hito, en donde se pone en práctica la toma de decisiones. Se caracteriza por:

- **Ser iterativo e incremental:** Todo sistema informático complejo supone un gran esfuerzo que puede durar desde varios meses hasta años. Por lo tanto, lo más práctico es dividir un proyecto en varias fases. Cada recorrido por las fases se denomina iteración en el proyecto en la que se realizan varios flujos de trabajo. Además, cada iteración parte de la anterior incrementado o revisando la funcionalidad implementada.
- **Dirigido por casos de uso:** Basándose en los casos de uso, los desarrolladores crean una serie de modelos de diseño e implementación que los llevan a cabo. Estos modelos se validan para que

sean conformes a los casos de uso. Finalmente, los casos de uso también sirven para realizar las pruebas sobre los componentes desarrollados.

- **Centrado en la arquitectura:** Cada aspecto está representado por un gráfico con su notación correspondiente. El concepto de arquitectura de software incluye los aspectos estáticos y dinámicos más significativos del sistema.

Incluye artefactos, que son los productos tangibles que conforman el producto final, y roles, papel que desempeña una persona dentro del proceso. Estas características han hecho que junto con el UML, RUP constituya la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos.

1.6.8 Lenguaje de modelado

La construcción de cualquier proyecto de ingeniería requiere de etapas de modelación que permitan experimentar y visualizar el sistema que se construirá. Uniendo varios conceptos y teorías, se puede conceptualizar un lenguaje de modelado como una estandarización de notaciones y reglas, que permitan graficar un sistema; o parte de él.

UML 2.0

Lenguaje Unificado de Modelado es el más conocido y utilizado en la actualidad; está respaldado por el OMG (Object Management Group). Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema. UML ofrece un estándar para describir un modelo del sistema, incluyendo aspectos conceptuales tales como procesos de negocio y funciones del sistema. Proporciona un vocabulario y reglas para permitir una comunicación. Está compuesto por elementos que no son más que abstracciones que constituyen los bloques básicos de construcción, los cuales pueden unirse mediante relaciones para conformar los diagramas. [21]

Características que lo hacen ideal:

- Sirve para el modelado completo de sistemas complejos, tanto en el diseño de los sistemas de software como para la arquitectura de hardware donde se ejecuten.
- Es completamente independiente del lenguaje de implementación, de tal forma que los diseños realizados usando UML, se pueda implementar en cualquier lenguaje que soporte las posibilidades de UML (principalmente lenguajes orientados a objetos).

- Se pueden automatizar determinados procesos y permite generar código a partir de los modelos y a la inversa (a partir del código fuente generar los modelos). Esto permite que el modelo y el código estén actualizados, por lo que siempre se puede mantener la visión en el diseño, de la estructura del proyecto.

1.6.9 Herramienta de modelado

La herramienta CASE (Computer Aided Software Engineering) Enterprise Architect 7.1, soporta la especificación de UML 2.0, describe un lenguaje visual por el cual se pueden definir mapas o modelos de un proyecto. Es una herramienta progresiva que cubre todos los aspectos del ciclo de desarrollo, proporcionando una trazabilidad completa desde la fase inicial del diseño a través del despliegue y mantenimiento. También provee soporte para pruebas, mantenimiento y control de cambio.

Algunas de las características claves son:

- Crear elementos del modelo UML para un amplio alcance de objetivos.
- Ubicar esos elementos en diagramas y paquetes.
- Crear conectores entre elementos.
- Documentar los elementos que se han creado.
- Generar código para el software que se esté construyendo.
- Realizar ingeniería inversa a un proyecto ya creado, independientemente del lenguaje en que haya sido implementado.

1.6.10 Base de datos

Una base de datos es una colección de elementos de datos interrelacionados que pueden procesarse por uno o más sistemas de aplicación. Entre las principales características de los sistemas de base de datos podemos mencionar: [22]

- Independencia lógica y física de los datos.
- Redundancia mínima.
- Acceso concurrente por parte de múltiples usuarios.
- Integridad de los datos.
- Consultas complejas optimizadas.
- Seguridad de acceso y auditoría.

- Respaldo y recuperación.
- Acceso a través de lenguajes de programación estándar.

Sistemas Gestores de Bases de Datos

PostgreSQL 8.3

PostgreSQL 8.3 es un sistema de base de datos de código abierto muy potente. Soporta gran parte del estándar SQL, y en algunos aspectos, está diseñado para que sea extensible por los usuarios. Se caracteriza por posibilitar transacciones ACID (Atomicidad, Consistencia, Aislamiento y Durabilidad), claves foráneas, vistas, secuencias, sub-peticiones, lanzadores, tipos y funciones definidos por el usuario, reunión externa, control de concurrencia multiversión. También posee interfaces gráficas de usuario y enlazadores para algunos lenguajes de programación. [23]

SQLite 3.5

Es una pequeña librería programada en lenguaje C que implementa un completo motor de base de datos multiplataforma que no precisa configuración. Se distribuye bajo licencia de dominio público y es muy rápido.

SQLite combina el motor y la interfaz de la base de datos en una única biblioteca, y almacena los datos en un único archivo de texto plano. Esto hace que cada usuario pueda crear tantas bases de datos como desee sin la necesidad de la intervención de un administrador de bases de datos que gestione los espacios de trabajo, usuarios y permisos de acceso. El hecho de almacenar toda la base de datos en un único archivo facilita la portabilidad de los datos.

- Es multiplataforma (Funciona en Linux, Windows, MacOs).
- Soporta operaciones en memoria.
- Realiza operaciones de manera eficiente, utilizando llamadas simples a subrutinas y funciones. Esto reduce la latencia en el acceso a la base de datos, debido a que las llamadas a funciones son más eficientes que la comunicación entre procesos.
- No asigna una cantidad fija de espacio en disco para cada fila en los campos de una tabla, todo lo contrario, utiliza la cantidad de espacio en disco necesaria para almacenar la información real del campo.

Por lo anteriormente planteado se decide utilizar como gestor de base de datos PostgreSQL 8.3 para la realización del componente de seguridad, aunque es recomendable utilizar SQLite.

1.6.11 Herramienta de desarrollo

Visual Studio 2008

El entorno de desarrollo integrado (IDE) Visual Studio 2008, soporta varios lenguajes de programación tales como Visual C++, Visual C#, Visual J#, ASP.Net y Visual Basic.Net, aunque actualmente se han desarrollado las extensiones necesarias para muchos otros. Permite a los desarrolladores crear aplicaciones, sitios y aplicaciones web, así como servicios web en cualquier entorno que soporte la plataforma .NET. Así se pueden crear aplicaciones que se intercomuniquen entre estaciones de trabajo, páginas web y dispositivos móviles. [24]

Ventajas que presenta:
A las mejoras de desempeño, escalabilidad y seguridad con respecto a la versión anterior, se agregan entre otras: [25]

- Permite incorporar características del nuevo Windows Presentation Foundation sin dificultad tanto en los formularios de Windows existentes como en los nuevos.
- Mejora 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: 2.0. (Incluido con Visual Studio 2005), 3.0 (incluido en Windows Vista) y 3.5 (incluido con Visual Studio 2008).

1.6.12 Ambiente de desarrollo

Después de un análisis realizado de las diferentes tecnologías y herramientas, siguiendo los lineamientos arquitectónicos del Departamento SAS y teniendo en cuenta las características de las mismas que permitan realizar un mejor desarrollo de la propuesta de solución, se decide utilizar como herramienta de modelado Enterprise Architect 7.1, la cual utiliza como lenguaje UML 2.0 y permite generar todos los artefactos necesarios durante el proceso de desarrollo, además se hará uso del modelo de referencia CMMI utilizado por la Universidad para alcanzar una certificación internacional del nivel 2 de madurez.

Como IDE se propone Visual Studio 2008 y como lenguaje de programación C#. Para el almacenamiento de los datos, el Departamento SAS define que se utilice en sus productos de software PostgreSQL 8.3, pero partiendo que el sistema propuesto es pequeño, que no va a contar con gran cantidad de información y no va a necesitar mucha concurrencia, se recomienda utilizar por las ventajas que ofrece SQLite 3.5 y así agilizar los procesos y evitar consumir recursos innecesarios del ordenador. Para guiar el diseño del componente de seguridad se utilizará la Arquitectura en 3 Capas.

1.7 Conclusiones

En este capítulo se realizó un estudio de los procesos y sistemas de autenticación de las aplicaciones sanitarias en el país. Además, se definieron las tecnologías y herramientas para el desarrollo de la propuesta de solución.

Capítulo 2. Características del Sistema

2.1 Introducción

En este capítulo se brinda una descripción de las principales características del sistema propuesto. Debido a que los procesos no están visibles y no se puede describir el negocio por casos de usos se realiza el modelo de dominio, el cual incluye conceptos que se relacionan entre sí para una mejor comprensión del funcionamiento del sistema. Además se definen los requerimientos funcionales y no funcionales que regirán el desarrollo del componente de seguridad y la trazabilidad de estos con los casos de usos.

2.2 Modelo de dominio

El Modelo de Dominio se realiza si no se determinan los procesos de negocio con fronteras bien establecidas, donde se logren ver claramente quienes son las personas que realizan cada proceso de negocio, quienes son los beneficiados con cada uno de estos procesos y quienes son las personas que desarrollan las actividades en cada uno de estos procesos. Permite de manera visual mostrar al usuario los conceptos fundamentales que se manejan en el dominio del sistema en desarrollo. Se representa en UML con un Diagrama de Clases donde se muestra las clases conceptuales, asociaciones y los atributos.

Definición de los conceptos principales

Para el entendimiento del diagrama de Modelo de Dominio que se expondrá a continuación, se hace necesaria la realización previa de la definición de los conceptos involucrados en dicho modelo.

Usuario: Representa a las personas que interactúan con el sistema de estadística.

Privilegios: Definen el nivel de acceso del usuario en el sistema.

Operaciones: Acciones realizadas por un usuario autenticado en el sistema.

Componente: Es una unidad ejecutable que representa el núcleo de la aplicación, la cual puede ser implantada independientemente y ser a la vez sujeto de composición de terceras partes, es decir, se puede tomar el componente y agregarlo a otro componente en desarrollo o simplemente consumir algunos de los servicios que brinda. [26]

Registro de Trazas: Historial donde se almacenan los eventos realizados por el usuario al interactuar con el sistema.

Diagrama del Modelo de Dominio

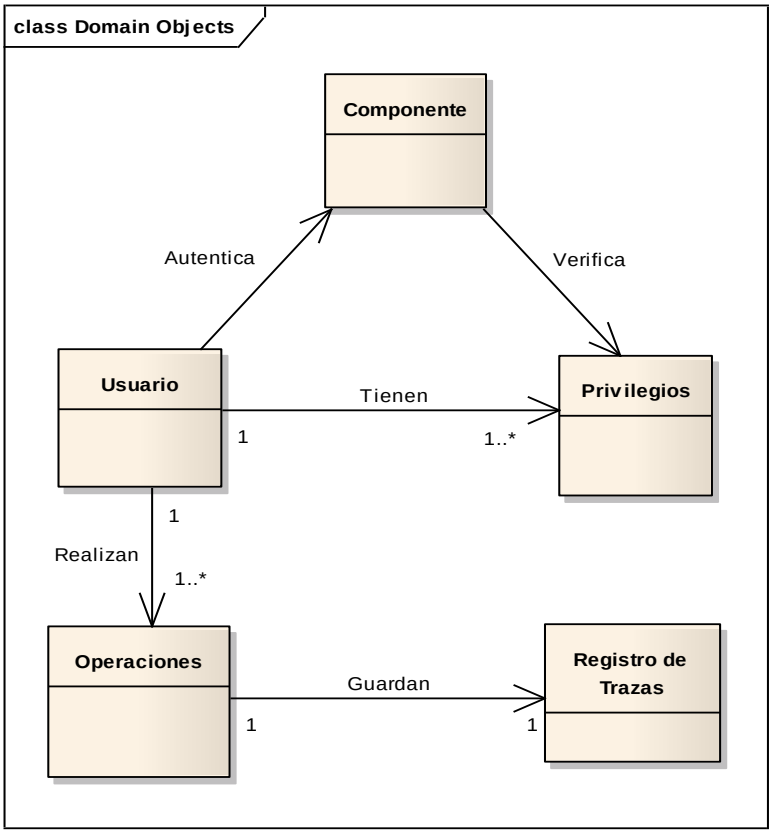


Figura 1: Modelo de Dominio.

2.3 Propuesta del sistema

A partir de los conocimientos obtenidos durante el proceso de investigación se propone realizar el análisis y diseño del componente de seguridad del Sistema para la Gestión y Análisis de Información Estadística en la Salud Pública Cubana, que permita la autenticación y autorización de los usuarios, así como la asignación de privilegios a estos, aún cuando no exista conectividad con el SAAA. Además, el componente debe permitir que el proceso de autenticación se realice a través de un LDAP.

- El componente de seguridad contará con opciones de configuración que permitan la autenticación y autorización mediante el SAAA o un LDAP.

- Si el Sistema para la Gestión y Análisis de Información Estadística en la Salud Pública Cubana (aplicación de escritorio) está configurado para conectarse al SAAA y no tiene conectividad con el mismo, el componente debe permitir a los usuarios autenticarse en ambiente desconectado, por lo que toda la información de los usuarios que tienen acceso a los módulos de estadística deben estar guardados de forma segura en una base de datos en cada ordenador donde esté funcionando el sistema.
- El componente deberá gestionar la información de autenticación y autorización referente a los usuarios en caso que el Sistema para la Gestión y Análisis de Información Estadística en la Salud Pública Cubana esté configurado para conectarse a un LDAP.
- El sistema controlará las acciones de los usuarios autenticados guardando cada operación realizada en el sistema en un registro de trazas.

De esta forma, el sistema informático queda comprendido por varios elementos como: software, hardware y personas. Generalmente este proceso comienza tomando la visión definida en el Modelo de Dominio, realizándose un análisis del mismo con el propósito de establecer los requerimientos. Una vez que los requisitos hayan sido identificados el modelo del sistema puede ser realizado.

2.4 Especificación de los requisitos de software

La captura de requisitos para un sistema constituye uno de los procesos más importantes en la etapa de desarrollo de software pues es donde se definen las funcionalidades del mismo. Los requerimientos de software son condiciones o capacidades que tiene que ser alcanzada por un sistema o componente para satisfacer un contrato, estándar u otro documento impuesto formalmente. Estos se clasifican en dos grupos: los requisitos funcionales, los cuales son las capacidades o condiciones que el sistema debe cumplir y los requisitos no funcionales que son las propiedades o cualidades que el producto debe tener.

Para dar solución al problema planteado se identificaron los siguientes requisitos que debe cumplir el componente de seguridad:

Requerimientos funcionales

RF1. Configurar Sistema.

RF1.1 Configurar SAAA.

RF1.2 Configurar LDAP.

RF2. Autenticar usuario.

RF2.1 Autenticar SAAA.

RF2.2 Autenticar LDAP.

RF2.3 Autorizar el acceso al usuario.

RF2.4 Obtener privilegios.

RF2.5 Registrar Traza.

RF2.6 Verificar conexión.

RF2.7 Guardar Información XML.

RF2.8 Encriptar Información XML.

RF2.9 Desencriptar Información XML.

RF3. Gestionar usuario.

RF3.1 Adicionar usuario.

RF3.2 Modificar usuario.

RF3.3 Eliminar usuario.

RF3.4 Buscar usuario.

RF3.5 Listar usuario.

RF4. Gestionar roles.

RF4.1 Adicionar rol.

RF4.2 Modificar rol.

RF4.3 Eliminar rol.

RF4.4 Buscar rol.

RF4.5 Listar rol.

RF5. Buscar Traza.

RF6. Listar Traza.

RF7. Editar Perfil.

Requerimientos no funcionales

RNF1. Apariencia o interfaz externa.

- La interfaz que se presentará al usuario debe ofrecer facilidades de entendimiento y sencillez al realizar las operaciones que en ella se presentan.

RNF2. Software.

- Permitir que el sistema se ejecute sobre Sistema Operativo Windows XP Profesional.

RNF3. Hardware.

- Ordenador Pentium o superior.
- Teclado y mouse.
- Monitor VGA o superior.
- 128 MB de memoria RAM o superior.
- Disco Duro de 40 GB.

RNF4. Seguridad.

- Confiabilidad: La información manejada por el sistema está protegida contra acceso no autorizado.
- Disponibilidad: Los usuarios autorizados tendrán acceso a la información en todo momento.
- Integridad: La información puede ser creada, modificada y eliminada solo por las personas autorizadas.

RNF5. Usabilidad.

- El sistema podrá ser utilizado por cualquier usuario autorizado con conocimientos informáticos básicos.
- Debe garantizar un acceso fácil y rápido.

2.5 Modelo de casos de uso del sistema

El Modelo de Casos de Uso del Sistema permite que los desarrolladores y los clientes lleguen a una comprensión sobre las condiciones y posibilidades que debe cumplir un sistema. Este modelo contiene actores, casos de usos y sus relaciones.

Definición de los actores

Un actor del sistema es una persona o la abstracción de software que interactúa con el sistema: puede intercambiar información con él, representar el rol que juegan una o varias personas y ser un recipiente pasivo de información.

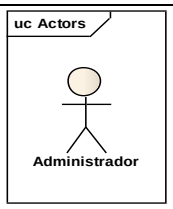
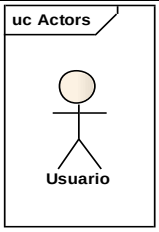
Actores del Sistema	Descripción
	Actor encargado de realizar las tareas de administración. Además, controla la trazabilidad de los usuarios en el sistema, llevando a cabo procesos de auditorías sobre las transacciones de un determinado usuario.
	Representa a las personas que interactúan con el Sistema para la Gestión y Análisis de Información Estadística en la Salud Pública Cubana.

Diagrama de Casos de Uso del Sistema

Representa gráficamente los procesos y su interacción con los actores. A continuación se muestra el diagrama correspondiente:

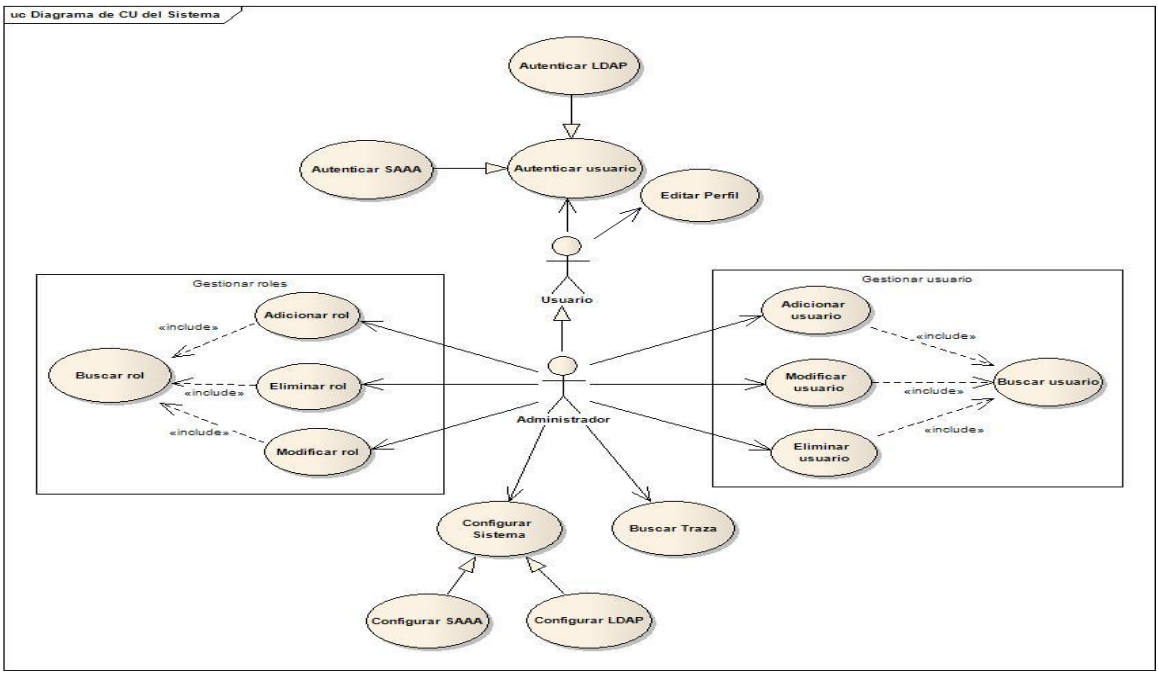


Figura 2: Diagrama de Casos de Uso del Sistema.

Descripción de los Casos de Uso

Para comprender la funcionalidad asociada a cada caso de uso no es suficiente con la representación gráfica del Diagrama de Caso de Uso del Sistema, por lo que es necesaria la descripción de cada uno, ya que representa los procesos que son objetos de automatización del mismo.

Tabla 1 CU_Configurar Sistema.

Objetivo	Configurar el sistema para realizar el proceso de autenticación, ya sea mediante el SAAA o un LDAP.
Actores	Administrador (inicia).
Resumen	El caso de uso inicia cuando el actor desea configurar el sistema. El sistema notifica si la configuración es a través del SAAA o un LDAP, terminando así el caso de uso.
Complejidad	Media.
Prioridad	Crítico.
Precondiciones	El actor debe estar autenticado.
Postcondiciones	La configuración se realizó mediante el SAAA o un LDAP. Si no se realiza la operación, el sistema muestra un mensaje al actor.

Tabla 2 CU_Configurar SAAA.

Objetivo	Configurar el sistema para realizar el proceso de autenticación mediante el SAAA.
Actores	Administrador (inicia).
Resumen	El caso de uso inicia cuando el actor desea configurar el sistema a través del SAAA, el mismo debe introducir la dirección del SAAA (URL). El sistema notifica que la configuración se realizó, terminando así el caso de uso.
Complejidad	Media.
Prioridad	Crítico.

Precondiciones	El actor debe estar autenticado.
Postcondiciones	La configuración del sistema se realizó mediante el SAAA. En caso contrario, el sistema muestra un mensaje al actor.

Tabla 3 CU_Configurar LDAP.

Objetivo	Configurar el sistema para realizar el proceso de autenticación mediante un LDAP.
Actores	Administrador (inicia).
Resumen	El caso de uso inicia cuando el actor desea configurar el sistema a través de un LDAP, el mismo debe introducir los parámetros necesarios para su configuración. El sistema notifica que la configuración se realizó, terminando así el caso de uso.
Complejidad	Media.
Prioridad	Crítico.
Precondiciones	El actor debe estar autenticado.
Postcondiciones	La configuración se realizó mediante un LDAP. En caso contrario, el sistema muestra un mensaje al actor.

Tabla 4 CU_Autenticar usuario.

Objetivo	Permitir la entrada al sistema.
Actores	Usuario (inicia).
Resumen	El caso de uso inicia cuando el usuario intenta acceder al sistema introduciendo usuario y contraseña, por lo que se debe verificar la conexión entre el Sistema de Estadística y el SAAA o un LDAP, en caso de no existir conexión se debe autenticar de forma local en el sistema. El caso de uso finaliza cuando el actor se haya autenticado correctamente y el sistema muestre el nivel de acceso del mismo.

Complejidad	Media.
Prioridad	Crítico.
Precondiciones	El actor debe introducir usuario y contraseña.
Postcondiciones	Permite la entrada al usuario según el nivel de acceso. En caso de error, envía un mensaje al usuario.

Tabla 5 CU_Autenticar SAAA.

Objetivo	Permitir la entrada al sistema.	
Actores	Usuario (inicia).	
Resumen	El caso de uso inicia cuando el usuario intenta acceder al sistema; debe introducir usuario y contraseña. El caso de uso finaliza cuando el actor se haya autenticado correctamente y el sistema muestre el nivel de acceso del mismo.	
Complejidad	Alta.	
Prioridad	Crítico.	
Precondiciones	El actor debe introducir usuario y contraseña. El sistema se encuentra configurado a través del SAAA.	
Postcondiciones	Permite la entrada al usuario según el nivel de acceso. En caso de error, envía un mensaje al usuario.	
Flujo de eventos		
Flujo básico <Autenticar SAAA**>**		
	Actor	**Sistema**
1.	El actor introduce usuario y contraseña.	2. Verifica la conexión con el SAAA. En caso de no existir conexión, ver Nº Evento 1: “No hay conexión”.
		3. Verifica los datos introducidos por

		el usuario. Si hay campos incorrectos, ver Nº Evento 2: "Existen campos incorrectos."
		4. Permite la entrada al usuario, con la asignación de los privilegios.
		5. Obtiene los privilegios otorgados al usuario para actualizar la caché.
		6. Termina el caso de uso.
Flujos alternos		
Nº Evento1<No hay conexión**>**		
	Actor	**Sistema**
		1. Muestra mensaje al actor: "La operación no se pudo completar por fallos en la conexión. Por favor, si desea acceder al sistema introduzca su contraseña para trabajar en ambiente desconectado".
2.	El actor introduce su contraseña.	3. Verifica la contraseña introducida en su perfil. Si hay campos incorrectos, ver Nº Evento 2: "Existen campos incorrectos."
		4. Permite la entrada al usuario, con la asignación de los privilegios obtenidos de la caché.
Flujos alternos		
Nº Evento2<Existen campos incorrectos**>**		
		1. Muestra un mensaje: "Se introdujeron datos incorrectos, por

		favor rectifíquelos".
		2. Muestra un identificador con los campos incorrectos.
Requisitos no funcionales	RNF1, RNF4, RNF5.	

Tabla 6 CU_Autenticar LDAP.

Objetivo	Permitir la entrada al sistema.
Actores	Usuario (inicia).
Resumen	El caso de uso inicia cuando el usuario intenta acceder al sistema; debe introducir usuario y contraseña. El caso de uso finaliza cuando el actor se haya autenticado correctamente y el sistema muestre el nivel de acceso del mismo.
Complejidad	Alta.
Prioridad	Crítico.
Precondiciones	El actor debe introducir usuario y contraseña. El sistema se encuentra configurado a través de un LDAP.
Postcondiciones	Permite la entrada al usuario según el nivel de acceso. En caso de error, envía un mensaje al usuario.

Tabla 7 CU_Adicionar usuario.

Objetivo	Adicionar un usuario en el sistema.
Actores	Administrador (inicia).
Resumen	El caso de uso inicia cuando el actor accede a la opción Adicionar usuario. El sistema brinda la posibilidad de introducir los datos del usuario. El caso de uso culmina cuando se adiciona el nuevo usuario.
Complejidad	Alta.

<table>
<tr><td>Prioridad</td><td colspan="2">Útiles.</td></tr>
<tr><td>Precondiciones</td><td colspan="2">Realizar una búsqueda para verificar que el usuario no este adicionado. Se realiza cuando el sistema se encuentre configurado mediante un LDAP.</td></tr>
<tr><td>Postcondiciones</td><td colspan="2">Se adicionó el usuario en el sistema. En caso de error, muestra un mensaje al actor.</td></tr>
<tr><td colspan="3">Flujo de eventos</td></tr>
<tr><td colspan="3">Flujo básico <Adicionar usuario></td></tr>
<tr><td></td><td>Actor</td><td>Sistema</td></tr>
<tr><td>1.</td><td>El caso de uso inicia cuando el administrador accede a la opción Adicionar usuario.</td><td>2. Muestra los datos y brinda la posibilidad de introducirlos:
Usuario.
Rol.
Nivel.
Y permite:
Adicionar.
Cancelar. Ver Nº Evento1: “Cancelar”.</td></tr>
<tr><td>3.</td><td>Introduce los datos para adicionar un usuario:
Usuario.
Rol.
Nivel.</td><td></td></tr>
<tr><td>4.</td><td>Selecciona la opción Adicionar.</td><td>5. Valida los datos introducidos por el actor. Si hay datos incorrectos, ver Nº Evento2: “Datos incorrectos”.</td></tr>
<tr><td></td><td></td><td>6. Adiciona el usuario.</td></tr>
</table>

		7. Muestra un mensaje: "Se ha adicionado el usuario".
		8. Termina el caso de uso.
Flujos alternos		
Nº Evento1 <Cancelar**>**		
	Actor	**Sistema**
1.	El actor selecciona la opción Cancelar operación.	2. Regresa a la vista anterior.
		3. Termina el caso de uso.
Nº Evento2 <Datos incorrectos**>**		
	Actor	**Sistema**
		1. Muestra un mensaje: "Se introdujeron datos incorrectos, por favor rectifíquelos".
		2. Muestra un indicador con los campos incorrectos.
Relación	**CU Incluido**	Buscar usuario.
Requisitos no funcionales	RFN1, RNF3.	

Tabla 8 CU_Modificar usuario.

Objetivo	Modificar los datos de un usuario.
Actores	Administrador (inicia).
Resumen	El caso de uso inicia cuando el actor selecciona un usuario y accede a la opción Modificar usuario, el sistema brinda la posibilidad de modificar sus datos, ya sea introduciéndolos o seleccionándolos. El sistema

	actualiza las modificaciones realizadas por el actor. Termina el caso de uso.
Complejidad	Alta.
Prioridad	Útiles.
Precondiciones	Seleccionar usuario para modificar sus datos. Se realiza cuando el sistema se encuentre configurado mediante un LDAP.
Postcondiciones	Se modificó los datos del usuario seleccionado. En caso de error, el sistema muestra un mensaje al actor.

Tabla 9 CU_Eliminar usuario.

Objetivo	Eliminar un usuario del sistema.
Actores	Administrador (inicia).
Resumen	El caso de uso inicia cuando el actor selecciona un usuario y accede a la opción Eliminar usuario. El sistema elimina el usuario y termina el caso de uso.
Complejidad	Media.
Prioridad	Útiles.
Precondiciones	Para eliminar un usuario, este debe haber sido seleccionado. Se realiza cuando el sistema se encuentre configurado mediante un LDAP.
Postcondiciones	Se eliminó el usuario seleccionado. Si no se realiza la operación, el sistema muestra un mensaje al actor.

Tabla 10 CU_Buscar usuario.

Objetivo	Buscar un usuario.
Actores	Administrador (inicia).
Resumen	El caso de uso inicia cuando el actor desea adicionar, modificar o eliminar un usuario. El sistema realiza una búsqueda y muestra un listado de usuarios, terminado así el caso de uso.

Complejidad	Media.
Prioridad	Útiles.
Precondiciones	Se realiza cuando el sistema se encuentre configurado mediante un LDAP.
Postcondiciones	Se realizó una búsqueda en el sistema. Si no se realiza la operación, el sistema muestra un mensaje al actor.

Tabla 11 CU_Adicionar rol.

Objetivo	Adicionar un rol en el sistema.
Actores	Administrador (inicia).
Resumen	El caso de uso inicia cuando el actor selecciona adicionar un rol. El sistema le da la posibilidad de introducir los datos y adicionar un nuevo rol. Termina el caso de uso.
Complejidad	Alta.
Prioridad	Útiles.
Precondiciones	Realizar una búsqueda para verificar que el rol no este adicionado. Se realiza cuando el sistema se encuentre configurado mediante un LDAP.
Postcondiciones	Se adicionó el rol. Si no se realiza la operación, el sistema muestra un mensaje al actor.

Tabla 12 CU_Eliminar rol.

Objetivo	Eliminar un rol del sistema.
Actores	Administrador (inicia).
Resumen	El caso de uso inicia cuando el actor selecciona un rol y accede a la opción Eliminar rol. El sistema elimina el rol y termina el caso de uso.
Complejidad	Media.
Prioridad	Útiles.
Precondiciones	Para eliminar un rol, este debe haber sido seleccionado. Se realiza cuando el sistema se encuentre configurado mediante un LDAP.

Postcondiciones	Se eliminó el rol por el administrador. Si no se realiza la operación, el sistema muestra un mensaje al actor.

Tabla 13 CU_Modificar rol.

Objetivo	Modificar la información de un rol.
Actores	Administrador (inicia).
Resumen	El caso de uso inicia cuando el actor selecciona un rol y accede a la opción Modificar rol, el sistema muestra los datos del rol y brinda la posibilidad de cambiar su información. El actor modifica los datos y el sistema los actualiza, terminando así el caso de uso.
Complejidad	Alta.
Prioridad	Útiles.
Precondiciones	Seleccionar rol para modificar sus datos. Se realiza cuando el sistema se encuentre configurado mediante un LDAP.
Postcondiciones	Se modificó la información del rol. Si no se realiza la operación, el sistema muestra un mensaje al actor.

Tabla 14 CU_Buscar rol.

Objetivo	Realizar búsqueda de un rol.
Actores	Administrador (inicia).
Resumen	El caso de uso inicia cuando el actor desea adicionar, modificar o eliminar un rol. El sistema realiza una búsqueda y muestra un listado de roles, terminado así el caso de uso.
Complejidad	Media.
Prioridad	Útiles.
Precondiciones	Se realiza cuando el sistema se encuentre configurado mediante un LDAP.

Postcondiciones	Se realizó una búsqueda en el sistema. Si no se realiza la operación, el sistema muestra un mensaje al actor.

Tabla 15 CU_Buscar Traza.

Objetivo	Buscar las trazas registradas en el sistema.
Actores	Administrador (inicia).
Resumen	El caso de uso inicia cuando el actor hace una búsqueda de las trazas registradas en el sistema. Esta búsqueda se puede realizar por diferentes parámetros, el sistema mostrará las trazas según los criterios de búsqueda.
Complejidad	Alta.
Prioridad	Crítico.
Precondiciones	Seleccionar criterios de búsqueda.
Postcondiciones	Se encontraron las trazas dado los parámetros de búsqueda. Si no se realiza la operación, el sistema muestra un mensaje al actor.

Tabla 16 CU_Editar Perfil.

Objetivo	Editar el perfil de un usuario.
Actores	Usuario (inicia).
Resumen	El caso de uso inicia cuando el actor selecciona Editar Perfil, en el mismo deberá escoger o introducir una pregunta y darle respuesta a la misma, siendo esta su contraseña para acceder al sistema en ambiente desconectado. El caso de uso finaliza cuando se haya editado el perfil.
Complejidad	Alta.
Prioridad	Crítico.
Precondiciones	El usuario debe estar autenticado.

Postcondiciones	Se editó el perfil del usuario. Si no se realiza la operación, el sistema muestra un mensaje al actor.

2.6 Prototipos de interfaz de usuario

A continuación se muestran algunos prototipos no funcionales para el componente de seguridad. Para consultar los prototipos restantes ver **Anexo1**:

Interfaz: Autenticar.

Interfaz: Presentación del Componente de Seguridad.

Interfaz: Configurar Sistema.

Interfaz: Configurar LDAP.

2.7 Conclusiones

En este capítulo se expusieron los conceptos relacionados en el Modelo de Dominio para lograr comprender el funcionamiento del sistema. Se brindó una panorámica de la propuesta del sistema definiendo sus requisitos funcionales y no funcionales. Además, se hizo una descripción detallada de los casos de uso del sistema.

Capítulo 3: Análisis y Diseño del Sistema

3.1 Introducción

En este capítulo se expone el análisis y diseño para dar solución al problema planteado, modelándose los artefactos necesarios que contribuyen a la implementación del componente de seguridad. Se realiza una vista al Modelo de Análisis, donde se muestra los diagramas de clases del análisis. También se realiza una vista del Modelo de Diseño, así como los diagramas de clases del diseño y los diagramas de interacción.

3.2 Modelo de análisis

El Modelo de Análisis es la primera representación técnica de un sistema. El Modelo de Análisis debe lograr tres objetivos primarios: describir lo que requiere el cliente, establecer una base para la creación de un diseño de software y definir un conjunto de requisitos que se pueda validar una vez que se construye el software.

El propósito del análisis es definir todas las clases que son relevantes al problema que se va a resolver, las operaciones y atributos asociados, las relaciones y comportamientos asociadas con ellas. Este modelo es usado para representar la estructura global del sistema, describe la realización de casos de uso y sirve como una abstracción del Modelo de Diseño.

Clases del Análisis

Representan abstracciones de una o varias clases y/o subsistemas del diseño del sistema, las cuales se caracterizan por centrarse en el tratamiento de los requisitos funcionales. Las clases del análisis se clasifican en:

- **Clase Interfaz:** Modela la interacción entre el sistema y sus actores.
- **Clase Entidad:** Modela información que posee una larga vida.
- **Clase Control:** Representa coordinación, secuencia, transacciones, control de otros objetos y a menudo encapsula a un caso de uso en concreto.

A continuación se muestran algunos diagramas de clases del análisis. Para consultar los diagramas de clases correspondientes a otros casos de uso ver **Anexo 2**:

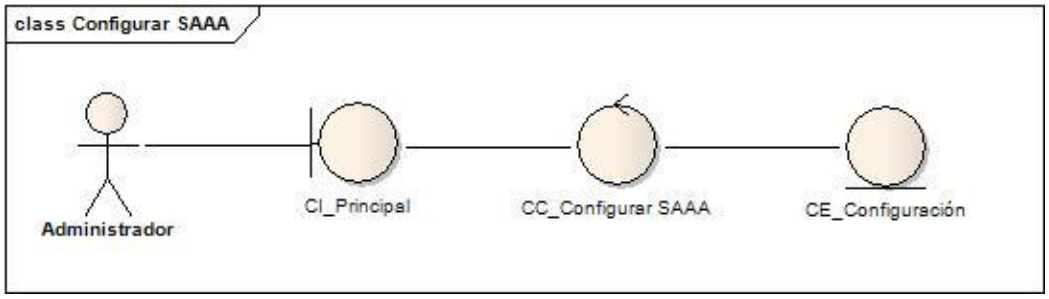


Figura 3: Diagrama de Clase del Análisis: CU_Configurar SAAA.

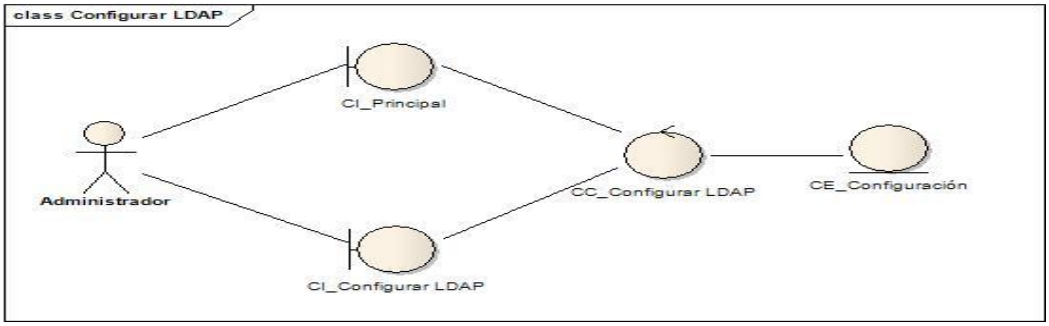


Figura 4: Diagrama de Clase del Análisis: CU_Configurar LDAP.

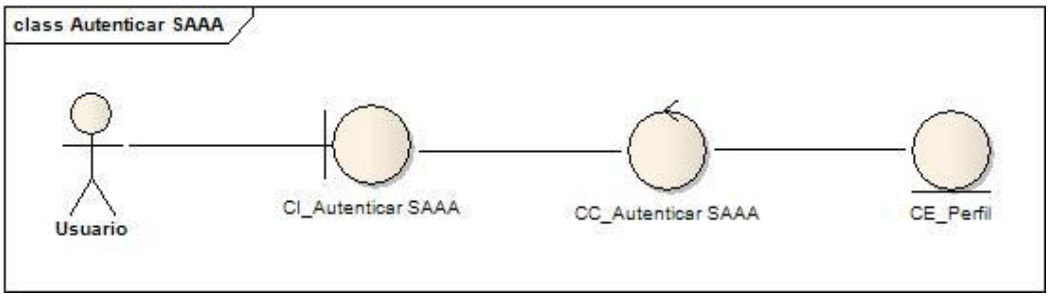


Figura 5: Diagrama de Clase del Análisis: CU_Autenticar SAAA.

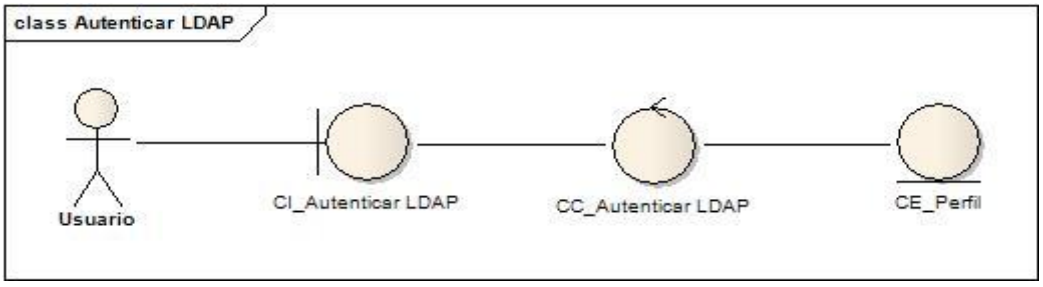


Figura 6: Diagrama de Clase del Análisis: CU_Autenticar LDAP.

Diagramas de Interacción

Los diagramas de interacción se utilizan para modelar los aspectos dinámicos de un sistema, lo que conlleva a modelar instancias concretas o prototípicas de clases interfaces, componentes y nodos, junto con los mensajes enviados entre ellos, todo en el contexto de un escenario que ilustra un comportamiento. [27]

UML define dos tipos de diagramas de interacción, los cuales sirven para expresar interacciones semejantes o idénticas del mensaje:

- **Diagrama de secuencia:** Es un diagrama que destaca la ordenación temporal de los mensajes. Muestran las interacciones expresadas en función de secuencias temporales.
- **Diagrama de colaboración:** Es un diagrama que destaca la organización estructural de los objetos que envían y reciben mensajes. Muestran las relaciones entre los objetos y los mensajes que intercambian.

A continuación se muestran algunos diagramas de colaboración del análisis. Para consultar los diagramas de colaboración correspondientes a otros casos de usos ver **Anexo 3**:

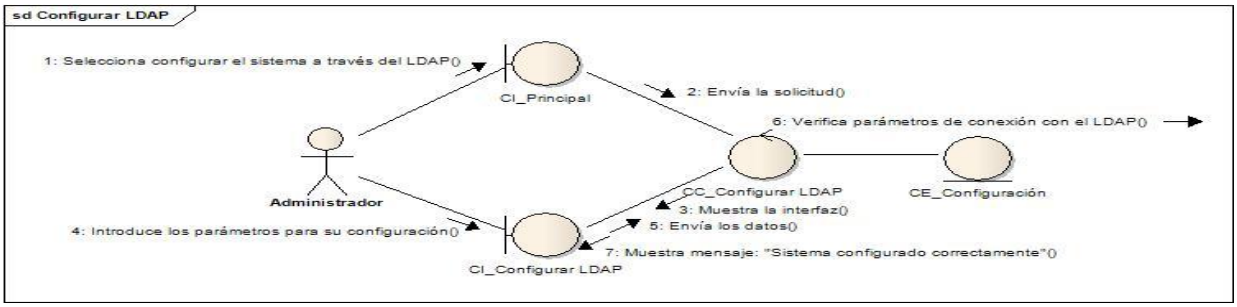


Figura 7: Diagrama de Colaboración del Análisis: CU_Configurar LDAP.

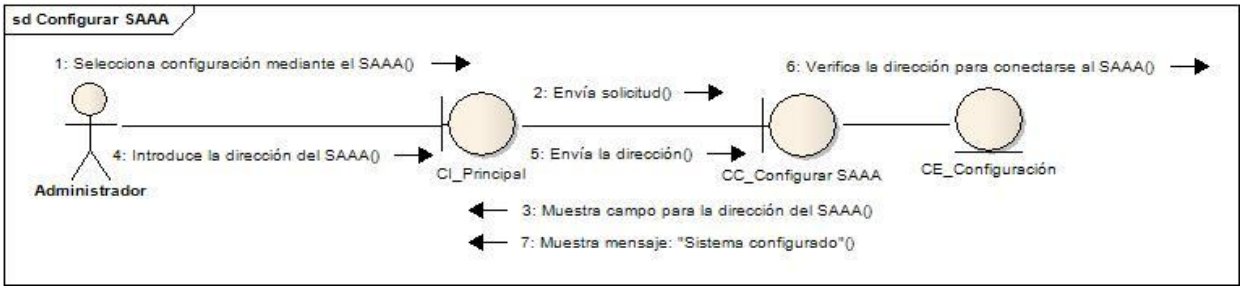


Figura 8: Diagrama de Colaboración del Análisis: CU_Configurar SAAA.

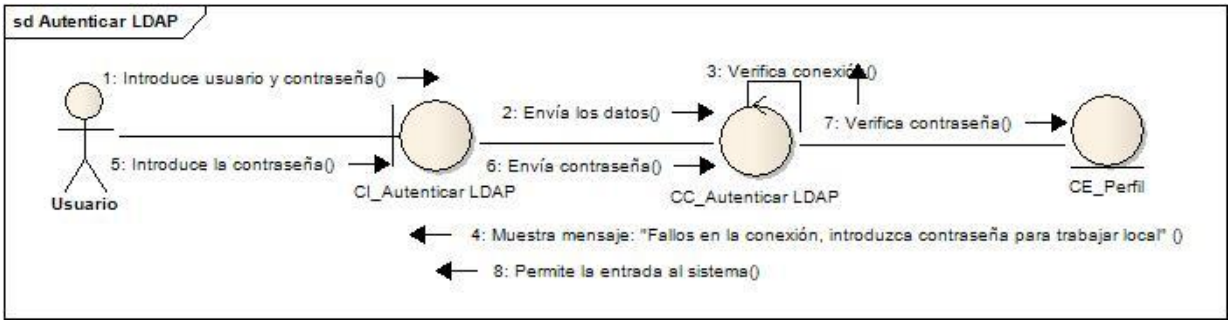


Figura 9: Diagrama de Colaboración del Análisis: CU_Autenticar LDAP.

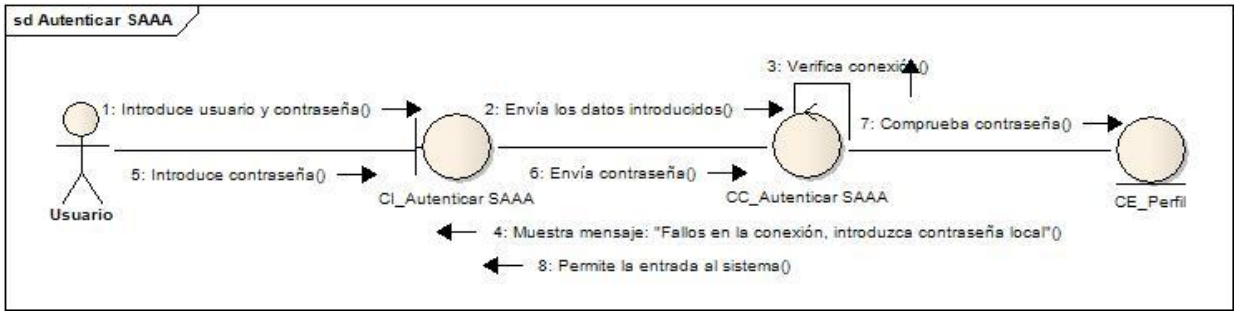


Figura 10: Diagrama de Colaboración del Análisis: CU_Autenticar SAAA.

3.3 Patrón de arquitectura a utilizar

Entre los estilos arquitectónicos más utilizados en la actualidad fue seleccionada la Arquitectura en 3 Capas para guiar el diseño del Componente de Seguridad. La Arquitectura en 3 Capas consiste en separar un proyecto en Capa de Presentación, Capa de Negocio y Capa de Datos.

- **Capa de presentación:** Pantallas que se le muestran al usuario para que interaccione con el programa, comunicándole y recolectando la información suministrada por el usuario en un mínimo de proceso (realiza validaciones para comprobar que no hay errores de formato).
- **Capa de negocio:** Es donde residen los programas que se ejecutan, se reciben las peticiones del usuario y se envían las respuestas tras el proceso. Se denomina capa de negocio porque es aquí donde se establecen todos los procesos que deben realizarse.
- **Capa de datos:** Es donde residen los datos y es la encargada de acceder a los mismos. Está formada por uno o más gestores de bases de datos que realizan todo el almacenamiento de datos, reciben solicitudes de almacenamiento o recuperación de información desde la capa de negocio.

3.4 Modelo del diseño

El Modelo del Diseño es un modelo de objetos que describe la realización física de los casos de uso centrándose en cómo los requisitos funcionales y no funcionales, junto con otras restricciones relacionadas con el entorno de implementación, tienen impacto en el sistema a considerar. Además, el Modelo de Diseño sirve de abstracción de la implementación del sistema y es, de ese modo, utilizada como una entrada fundamental a las actividades de implementación. [28]

Diagramas de clases del diseño

A través del flujo de diseño, uno de los artefactos más importantes a obtener son los diagramas de clases de diseño, donde se exponen las clases que intervienen en las realizaciones de los casos de uso del sistema. En este tipo de diagrama se representa un nivel de detalle más alto que los diagramas de clases del análisis, relacionándose con el lenguaje de programación del cual se hará uso en la implementación del sistema. [29]

Los diagramas de clases del diseño son los encargados de representar las relaciones entre clases, interfaces así como la colaboración entre ellos. Constituyen la vista del diseño estático de un sistema. Contienen las definiciones de las entidades de software y además de visualizar, estructurar y documentar los modelos, ayudan a construir el sistema a través de la ingeniería directa e inversa de código.

A continuación se muestran algunos diagramas de clases del diseño:

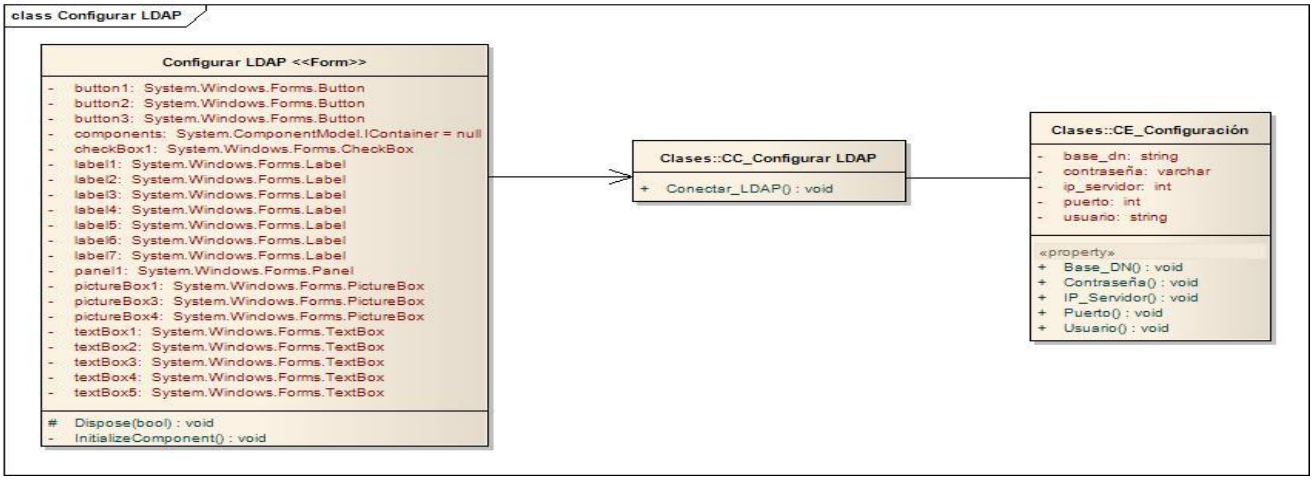


Figura 11: Diagrama de Clase del Diseño: CU_Configurar LDAP.

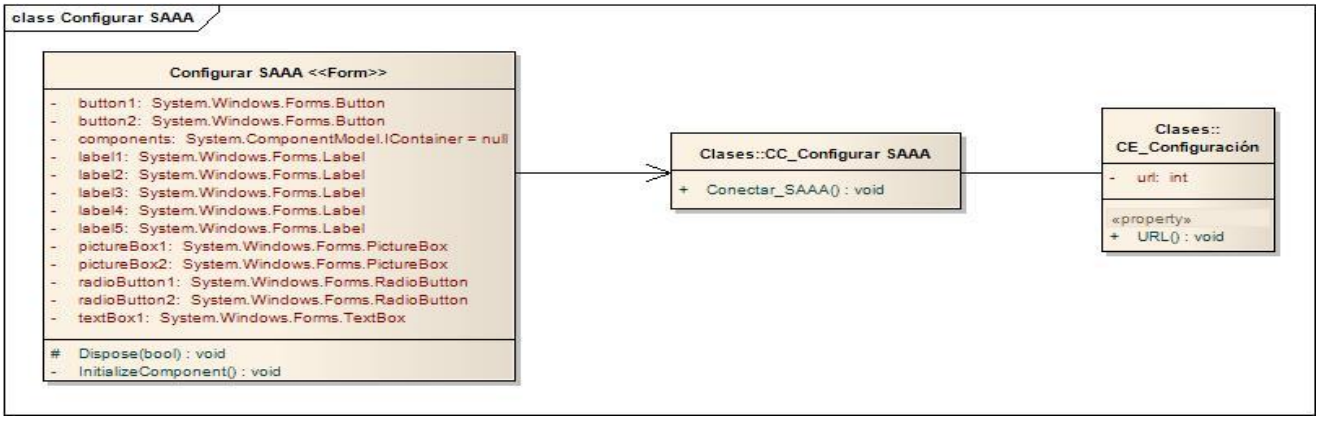


Figura 12: Diagrama de Clase del Diseño: CU_Configurar SAAA.

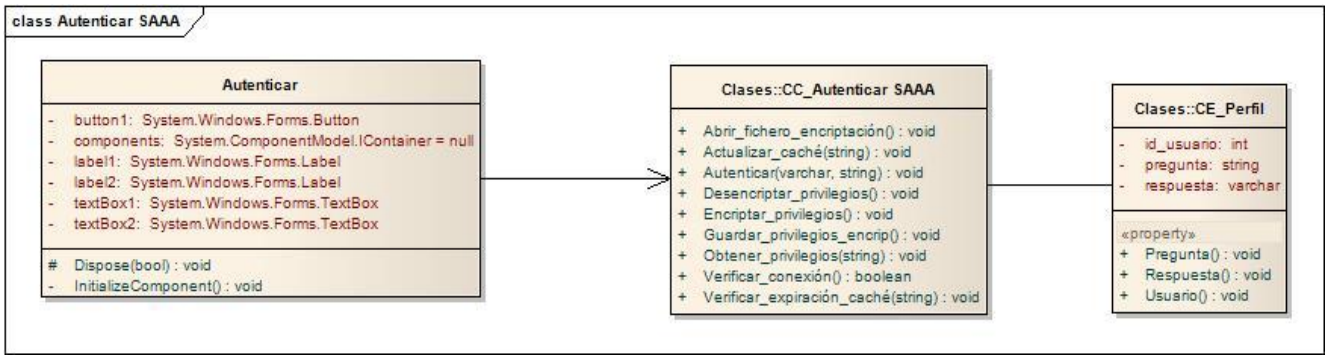


Figura 13: Diagrama de Clase del Diseño: CU_Autenticar SAAA.

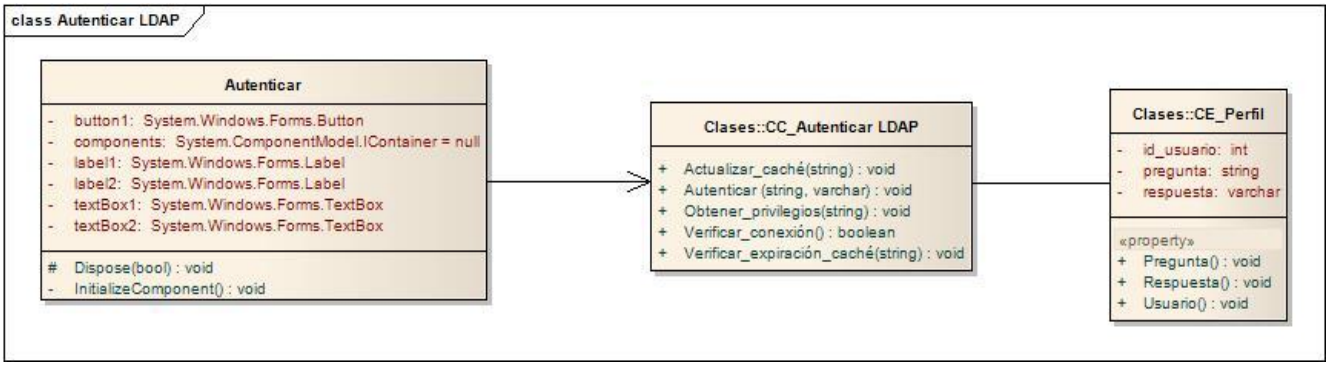


Figura 14: Diagrama de Clase del Diseño: CU_Autenticar LDAP.

Descripción del Diagrama de Clase del Diseño del CU_Configurar LDAP

Nombre: Formulario "Configurar LDAP"
Descripción: Esta es la interfaz encargada de visualizar al usuario los parámetros que debe insertar para realizar la configuración del sistema mediante un LDAP.

Descripción de la clase formulario Configurar LDAP.

Nombre: Controladora "Configurar LDAP"
Descripción: Esta es la clase encargada de lograr la conexión entre el sistema de estadística y el LDAP, para poder autenticar a los usuarios que tienen acceso al sistema.

Descripción de la clase controladora Configurar LDAP.

Nombre: Entidad "Configuración"
Descripción: Esta es la clase que guardará los parámetros de configuración del sistema a través de un LDAP.

Descripción de la clase entidad Configuración.

A continuación se muestran algunos diagramas de secuencia. Para consultar los diagramas de secuencia correspondientes a los casos de uso restantes ver **Anexo 4**:

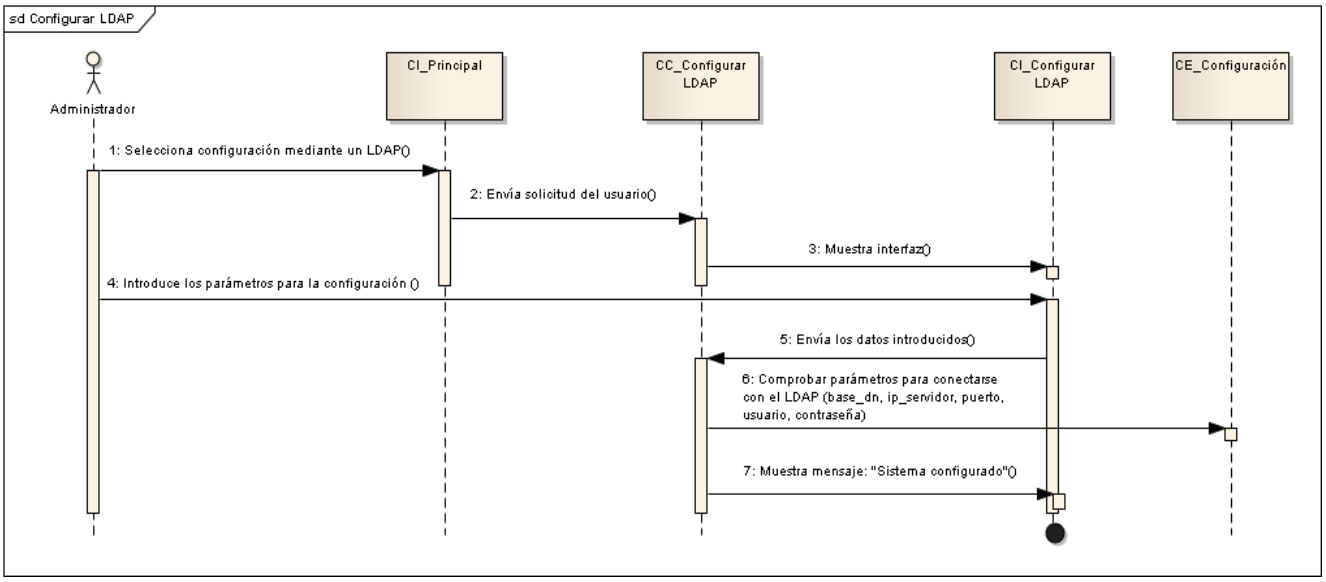


Figura 15: Diagrama de Secuencia: CU_Configurar LDAP.

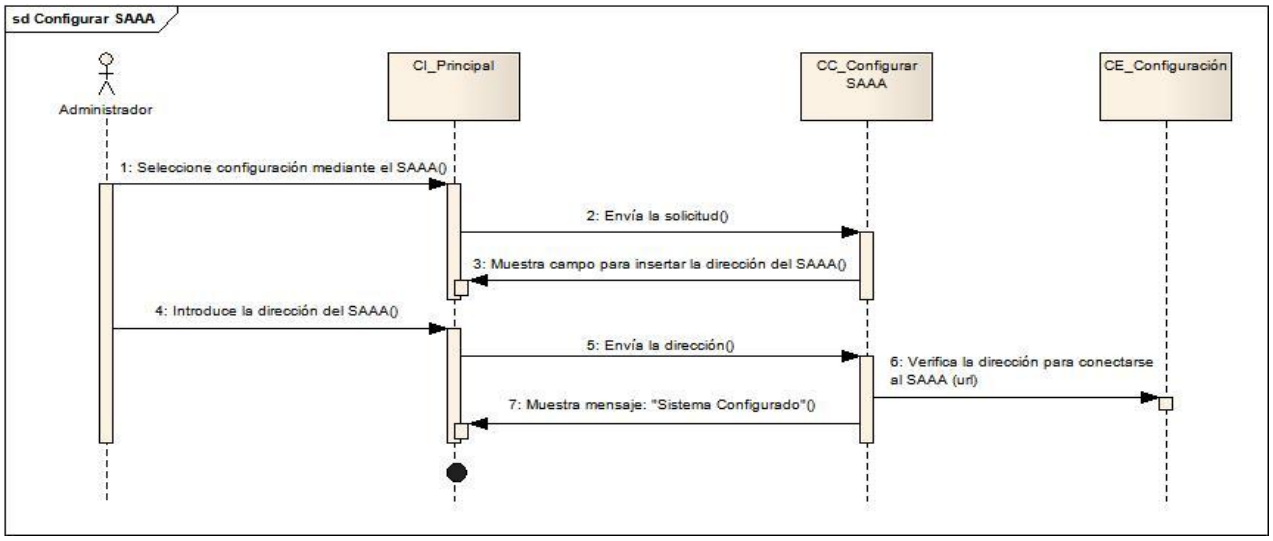


Figura 16: Diagrama de Secuencia: CU_Configurar SAAA.

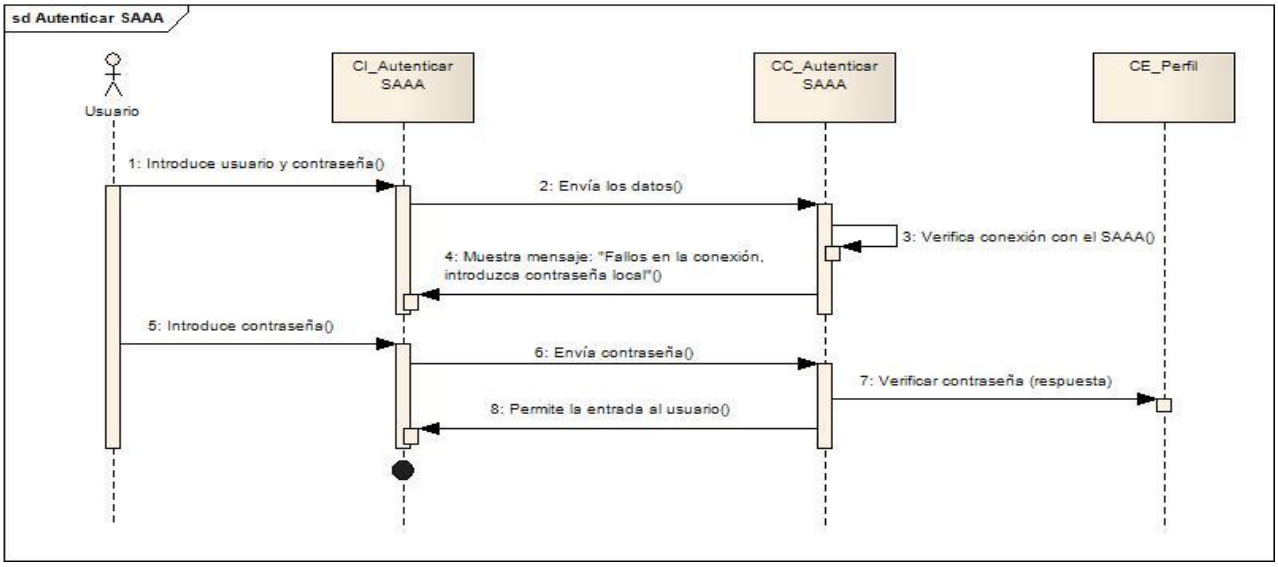


Figura 17: Diagrama de Secuencia: CU_Autenticar SAAA.

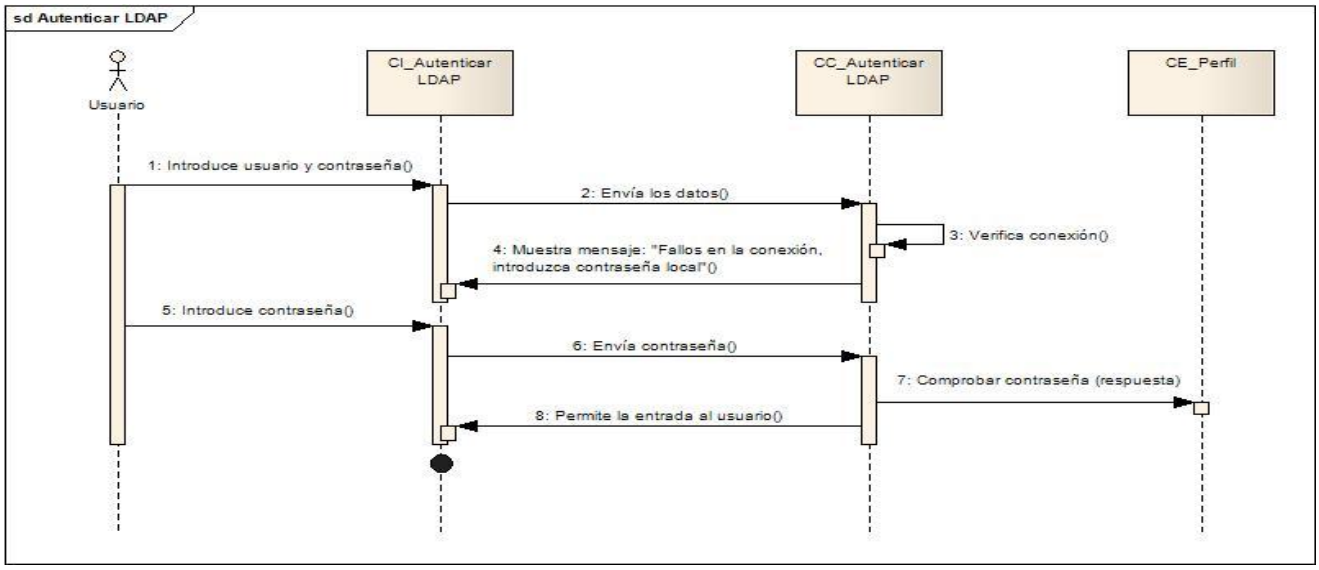


Figura 18: Diagrama de Secuencia: CU_Autenticar LDAP.

3.5 Diseño de la base de datos

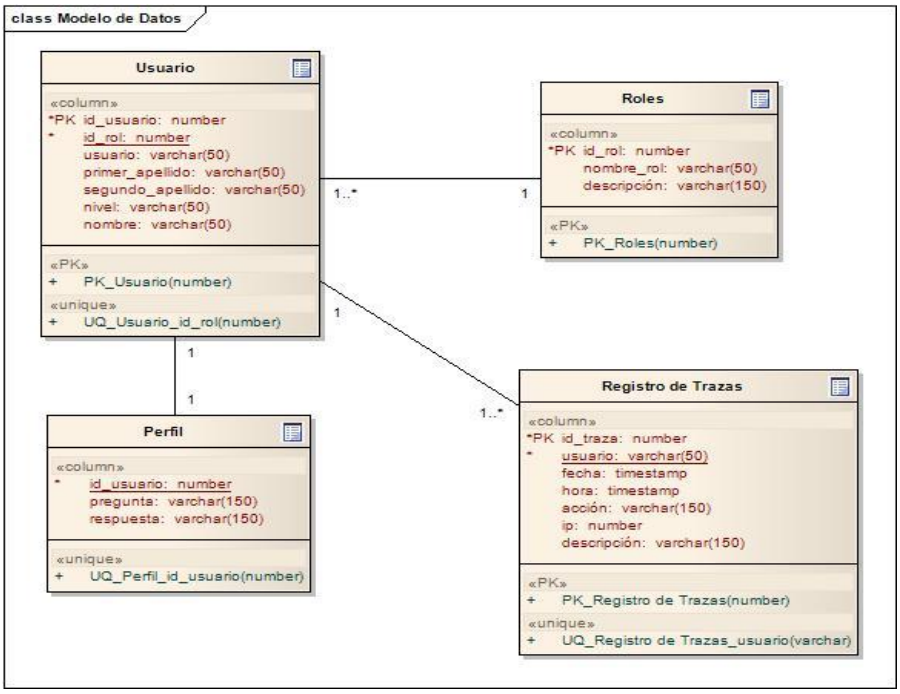


Figura 19: Modelo de Datos.

Descripción de las Tablas de la Base de Datos

Nombre: Usuario		
Descripción: Esta tabla posee la información de los usuarios que tienen acceso al sistema de estadística.		
Atributo	**Tipo**	**Descripción**
id_usuario (llave primaria)	number	Identificador del usuario.
id_rol (llave foránea)	number	Identificador del rol.
usuario	varchar (50)	Usuario que tiene acceso al sistema de estadística.
primer_apellido	varchar (50)	Primer apellido del usuario.
segundo_apellido	varchar (50)	Segundo apellido del usuario.
nivel	varchar (50)	Representa el nivel al que trabaja el usuario: nacional, provincial, municipal y unidades de salud.
nombre	varchar (50)	Nombre del usuario.

Nombre: Roles		
Descripción: Contiene la información de los roles definidos.		
Atributo	**Tipo**	**Descripción**
id_rol (llave primaria)	number	Identificador del rol.
nombre_rol	varchar (50)	Nombre del rol.
descripción	varchar (150)	Breve descripción de las características del rol.

Nombre: Perfil		
Descripción: Contiene la información del perfil de cada usuario.		
Atributo	**Tipo**	**Descripción**
id_usuario (llave foránea)	number	Representa al identificador del usuario que ha editado el perfil.
pregunta	varchar (150)	Pregunta que el usuario debe seleccionar o introducir en su perfil.
respuesta	varchar (150)	Respuesta del usuario.

Nombre: Registro de Trazas		
Descripción: Contiene las trazas registradas por un usuario.		
Atributo	**Tipo**	**Descripción**
id_traza (llave primaria)	number	Identificador de la traza.
usuario (llave foránea)	varchar (50)	Representa al usuario que registra las trazas.
fecha	timestanp	Fecha de registro de la traza.
hora	timestanp	Hora en que se registró la traza.
acción	varchar (150)	Acción realizada por el usuario.
ip	number	IP de la máquina en la cual se registró la traza.
descripción	varchar (150)	Breve descripción de la acción realizada por el usuario.

3.6 Conclusiones

El análisis y diseño de un software brinda la visión de lo que pudiera ser la solución en el desarrollo del mismo. En este capítulo se ha mostrado la idea de las funcionalidades que debe cumplir el componente de seguridad. Se definieron las clases del análisis y el diseño mostrando sus relaciones, así como la colaboración entre ellas mediante diferentes diagramas.

Conclusiones

Con el estudio de las tendencias actuales en cuanto a la seguridad en aplicaciones de software, tanto en el ámbito internacional como nacional, se definieron las tecnologías y herramientas adecuadas, que permitieron obtener de forma eficiente los artefactos en los diferentes flujos de trabajo, como son los diagramas y los prototipos no funcionales del componente de seguridad.

A partir del análisis del funcionamiento de sistemas en entorno desconectado y la correcta identificación de los requisitos funcionales y no funcionales, se obtuvo como resultado el análisis y diseño de un componente de seguridad que permitirá a los usuarios autenticarse en el sistema aún cuando no exista conectividad con el SAAA, solucionándose de esta forma la limitación presentada anteriormente por el Sistema de Gestión y Análisis de Información Estadística en la Salud Pública Cubana.

El componente de seguridad, aumentará la flexibilidad y la funcionalidad del sistema, posibilitando que el proceso de autenticación pueda realizarse mediante un LDAP. Además contará con una base de datos en la cual se almacenará la información referente a los usuarios y guardará un registro de trazas por cada acción que realicen los mismos.

Recomendaciones

El Componente de Seguridad será capaz de mantener la seguridad de la información contenida en el Sistema para la Gestión y Análisis de Información Estadística en la Salud Pública Cubana a través de la autenticación, brindando al usuario autenticado solamente las opciones que son permitidas para el rol que desempeñe, por lo que se recomienda:

- A partir de los artefactos generados implementar el Componente de Seguridad según las funcionalidades propuestas.
- Una vez implementado, el mismo deberá ser sometido a un proceso de pruebas por el Grupo de Calidad.
- Integrar el Componente de Seguridad al Sistema para la Gestión y Análisis de Información Estadística en la Salud Pública Cubana para que sea utilizado como sistema que gestione la Autenticación, Autorización y Auditoría.

Referencias Bibliográficas

[1] Revista Cubana de Salud Pública [En línea] Julio-Septiembre 2006. [Citado el: 10 diciembre de 2009] http://www.imbiomed.com.mx/1/1/articulos.php?id_revista=79&id_ejemplar=4003.

[2] Ídem a la referencia [1].

[3] Ídem a la referencia [1].

[4] Ídem a la referencia [1].

[5] Ídem a la referencia [1].

[6] Marañón, Gonzalo Álvarez. CSIC. Autenticación y Autorización. [En línea] 2001. [Citado el: 25 de enero de 2010] http://www.iec.csic.es/CRIPTONOMICON/autenticacion/control.html.

[7] Centeno Karina, Rojas Danisbel, Solis Héctor. Componente de Seguridad para aplicaciones del Área Temática Sistemas de Apoyo a la Salud. Trabajo de Diploma para optar por el Título de Ingeniero en Ciencias Informáticas. Ciudad de La Habana. Junio de 2008. [Citado el: 8 de abril de 2010].

[8] Ferrera Yudmila, Soulary Aramis. Sistema de Autenticación, Autorización y Auditoría para los Productos desarrollados en el Área Temática Sistemas de Apoyo a la Salud (SAAA-SAS) v1.1. Trabajo de Diploma para optar por el Título de Ingeniero en Ciencias Informáticas. Ciudad de La Habana. Junio de 2009. [Citado el: 8 de abril de 2010]

[9] ¿Qué es LDAP? [En línea] 10 de diciembre de 2004. [Citado el: 25 de enero de 2010] http://www.ldap-es.org/contenido/04/12/1.-%C2%BFque-es-ldap%3F.

[10] Ventajas en el uso de LDAP. [En línea] 19 de noviembre de 2004. [Citado el: 26 de enero de 2010] http://www.ldap-es.org/contenido/04/11/1.2.-ventajas-en-el-uso-de-ldap.

[11] Usos prácticos del LDAP. [En línea] 10 de diciembre de 2004. [Citado el: 26 de enero de 2010] http://www.ldap-es.org/contenido/04/12/1.3.-usos-pr%C3%A1cticos-de-ldap.

[12] Reynoso, Carlos Billy. 2006. MSDN Introducción a la Arquitectura de Software. [Online] 26, 2006. [Citado el: 10 de febrero de 2010] http://www.microsoft.com/spanish/msdn/arquitectura/roadmap_arq/intro.mspx.

[13] Ídem a la referencia [12].

[14] Reynoso, Billy. 2004. Profundizando en Estilos de Arquitectura de Software. 2004. [Citado el: 15 de febrero de 2010].

[15] Ídem a la referencia [14].

[16] Landa, Yuri. Lenguajes de Marca. [En línea] [Citado el: 17 de febrero de 2010] http://www.iiisci.org/Journal/CV$/risci/pdfs/X158QF.pdf.

[17] Ferrera Yudmila, Soulary Aramis. Sistema de Autenticación, Autorización y Auditoría para los Productos desarrollados en el Área Temática Sistemas de Apoyo a la Salud. Trabajo de Diploma para optar por el Título de Ingeniero en Ciencias Informáticas. Ciudad de La Habana. Junio de 2009. [Citado el: 17 de febrero de 2010].

[18] Qué es XML. [En línea] 2002. [Citado el: 20 de febrero de 2010] http://www.edinet.com/sabia2.asp.

[19] Stroustrup, Bjarne. 1998. El lenguaje de programación C++. [Citado el: 14 de junio de 2010] Madrid: Addison-Wesley Pub Co, 1998. ISBN 84-7829-019-2.

[20] Ídem a la referencia [19].

[21] Fowler, Scott. UML gota a gota. S.l.: Prentice Hall, 1997. [Citado el: 20 de febrero de 2010].

[22] ¿Qué son las bases de datos? [En línea] 26 de octubre de 2007. [Citado: 20 de febrero de 2010] http://www.maestrosdelweb.com/principiantes/%C2%BFque-son-las-bases-de-datos.

[23] PostqreSQL. [En línea] [Citado el: 20 de febrero de 2010] http://www.postgresql.org/about/press/presskit82.html.es.

[24] Microsoft Visual Studio 2008 [citado: 25 de febrero de 2010]

http://es.wikipedia.org/wiki/Microsoft_Visual_Studio.

[25] ídem a la referencia [24].

[26] Ferrera Yudmila, Soulary Aramis. Sistema de Autenticación, Autorización y Auditoría para los Productos desarrollados en el Área Temática Sistemas de Apoyo a la Salud (SAAA-SAS) v1.1. Trabajo de Diploma para optar por el Título de Ingeniero en Ciencias Informáticas. Ciudad de La Habana. Junio de 2009. [Citado el: 25 de febrero de 2010].

[27] Diagramas de Interacción. [Citado el: 13 de abril de 2010] http://tvdi.det.uvigo.es/~avilas/UML/node41.html.

[28] Carvajal Erlys. Análisis y diseño del subsistema de Análisis de Resultados de un Simulador de Procesos Químicos. Trabajo de Diploma para optar por el Título de Ingeniero en Ciencias Informáticas. Ciudad de la Habana. 2009. [Citado el: 13 de abril de 2010].

[29] Ídem a la referencia [28].

Bibliografía

1. Arquitectura en 3 Capas. Disponible en: http://kernelerror.net/programacion/php/arquitectura-3-capas/
2. Autenticación y Autorización. 2001. Disponible en: http://www.iec.csic.es/CRIPTONOMICON/autenticacion/control.html.
3. Características de C#. Disponible en: http://www.clikear.com/manuales/csharp/c10.aspx.
4. Carvajal Erlys. Análisis y diseño del subsistema de Análisis de Resultados de un Simulador de Procesos Químicos. Trabajo de Diploma para optar por el Título de Ingeniero en Ciencias Informáticas. Ciudad de la Habana. 2009.
5. Centeno Karina, Rojas Danisbel, Solis Héctor. Componente de Seguridad para aplicaciones del Área Temática Sistemas de Apoyo a la Salud. Trabajo de Diploma para optar por el Título de Ingeniero en Ciencias Informáticas. Ciudad de La Habana. Junio de 2008.
6. CMMI. Disponible en: http://calisoft.uci.cu.
7. CONTACC. Flores Adisley, Rovira Imirys. Sistema de Control de Acceso a Comedores en la Universidad de las Ciencias Informáticas. Trabajo de Diploma para optar por el Título de Ingeniero en Ciencias Informáticas. Ciudad de La Habana, Cuba. Junio 2007.
8. Diagramas de Interacción. Disponible en: http://tvdi.det.uvigo.es/~avilas/UML/node41.html.
9. Documentos Programa de Mejora. Disponible en: http://calisoft.uci.cu/index.php?option=com_content&view=article&id=46&Itemid=27.
10. El Lenguaje Unificado de Modelado (UML). Disponible en: http://www.disca.upv.es/enheror/pdf/ActaUML.PDF.
11. El Proceso Unificado de Desarrollo (RUP). Disponible en: http://www.disca.upv.es/enheror/pdf/ActaUML.PDF.
12. Extensible Access Control Markup Language (XACML). Disponible en: http://www.oasis-open.org/specs/index.php#xacml.
13. Ferrera Yudmila, Soulary Aramis. Sistema de Autenticación, Autorización y Auditoría para los Productos desarrollados en el Área Temática Sistemas de Apoyo a la Salud (SAAA-SAS) v1.1.

Trabajo de Diploma para optar por el Título de Ingeniero en Ciencias Informáticas. Ciudad de La Habana. Junio de 2009.

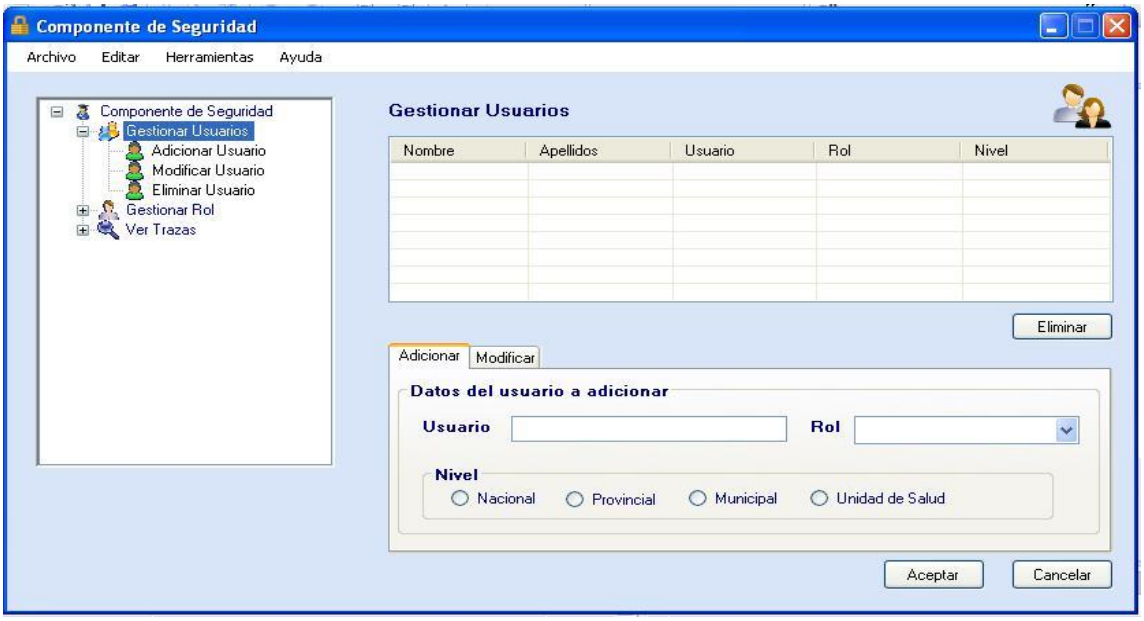
14. Fowler, Scott. UML gota a gota. S.l.: Prentice Hall, 1997.
15. Guía de Usuario de Enterprise Architect. Disponible en: http://www.sparxsystems.com.ar/EAUserGuide/ea.html.
16. Jacobson, Rumbaugh y otros. El Proceso Unificado de desarrollo de Software. s.l.: Addison-Wesley, 2000.
17. Landa, Yuri. Lenguajes de Marca. Disponible en: http://www.iiisci.org/Journal/CV$/risci/pdfs/X158QF.pdf.
18. Lenguajes de programación. 2009. Disponible en: http://www.lenguajes-de-programacion.com./lenguajes-de-programacion.shtml.
19. Microsoft Visual Studio 2008. Disponible en: http://es.wikipedia.org/wiki/Microsoft_Visual_Studio.
20. Modelo de Análisis. PRESSMAN, R. Ingeniería de Software “Un enfoque práctico” 5ta ed. Madrid: Graw Hill, 2002.
21. Modelo de dominio. 2002. Disponible en: http://adimen.si.ehu.es/~rigau/teaching/EHU/ISHAS/Curs2008-2009/Apunts/IS.4.pdf.
22. PostqreSQL. Disponible en: http://www.postgresql.org/about/press/presskit82.html.es.
23. Protocolo LDAP. Disponible en: http://www.ldap-es.org/contenido/04/12/1.-%C2%BFque-es-ldap%3F.
24. ¿Qué son las bases de datos? Disponible en: http://www.maestrosdelweb.com/principiantes/%C2%BFque-son-las-bases-de-datos.
25. Revista Cubana de Salud Pública. Disponible en: http://www.imbiomed.com.mx/1/1/articulos.php?id_revista=79&id_ejemplar=4003.
26. Reynoso, Billy. 2004. Profundizando en Estilos de Arquitectura de Software. 2004.
27. Reynoso, Carlos Billy. 2006. MSDN Introducción a la Arquitectura de Software. http://www.microsoft.com/spanish/msdn/arquitectura/roadmap_arq/intro.mspx.
28. Security Assertion Markup Language (SAML). Disponible en: http://docs.oasis-open.org/security/saml/v2.0.

29. Sistema de Autenticación, Autorización y Auditoria (SAAA). Ricardo Miguel Aluisco. Componente de Seguridad para la Arquitectura de la Informatización de la UCI. Trabajo de Diploma para optar por el Título de Ingeniero en Ciencias Informáticas. Ciudad Habana, Cuba. Junio de 2009.
30. SQLite. Disponible en: http://usuarios.pntic.mec.es/sqlite.php.
31. SQLite. Disponible en: http://www.aplicacionesempresariales.com/sqlite-el-motor-de-base-de-datos-agil-y-robusto.html.
32. Stroustrup, Bjarne. 1998. El lenguaje de programación C++. Madrid: Addison-Wesley Pub Co, 1998. ISBN 84-7829-019-2.
33. Usos prácticos del LDAP. Disponible en: http://www.ldap-es.org/contenido/04/12/1.3.-usos-pr%C3%A1cticos-de-ldap.
34. Ventajas en el uso de LDAP. Disponible en: http://www.ldap-es.org/contenido/04/11/1.2.-ventajas-en-el-uso-de-ldap.

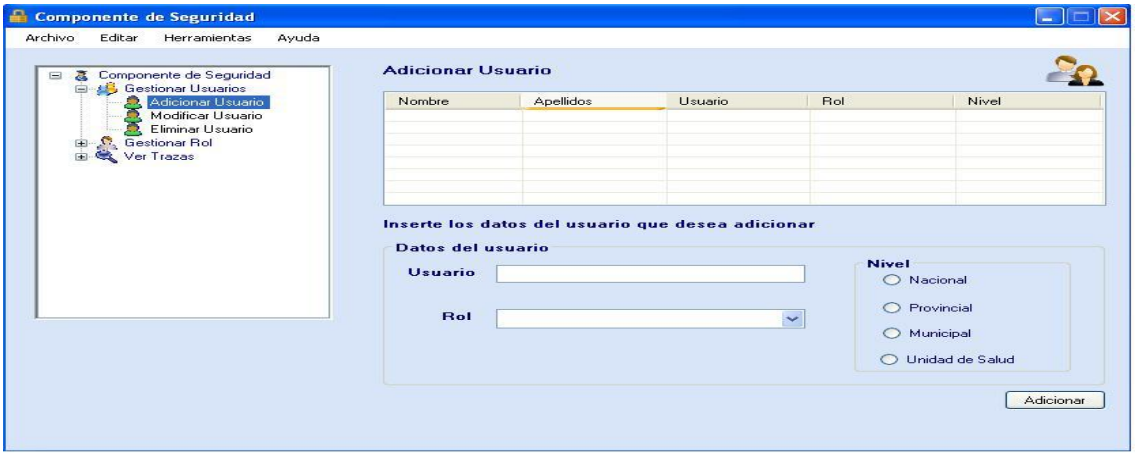
Anexos

Anexo 1: Prototipos de Interfaz de Usuario

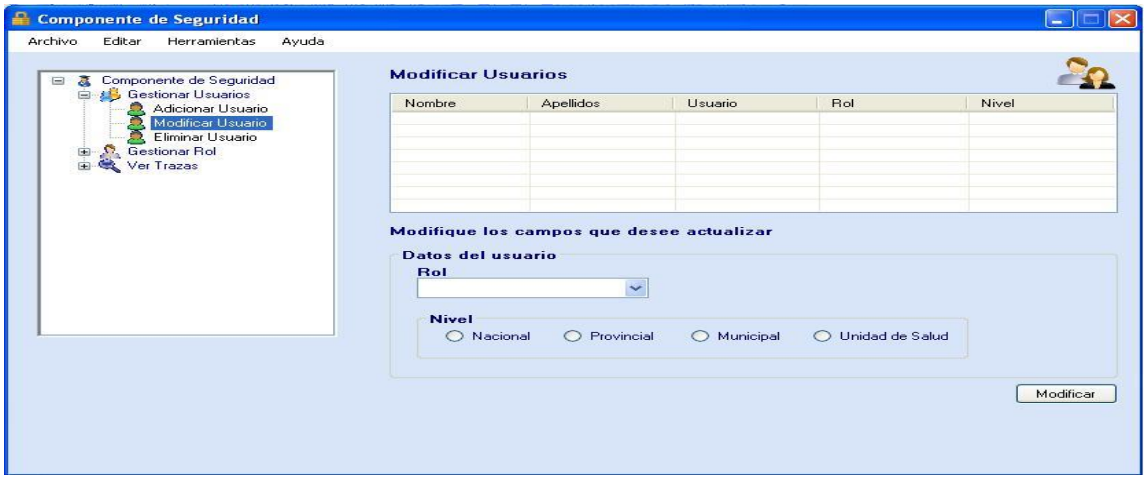
Interfaz: Autenticar (Local)



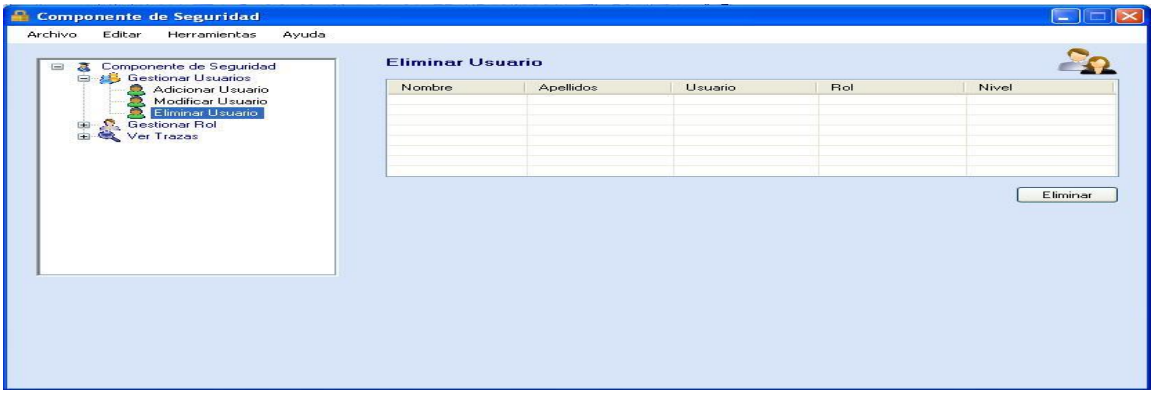
Interfaz: Gestionar Usuarios.



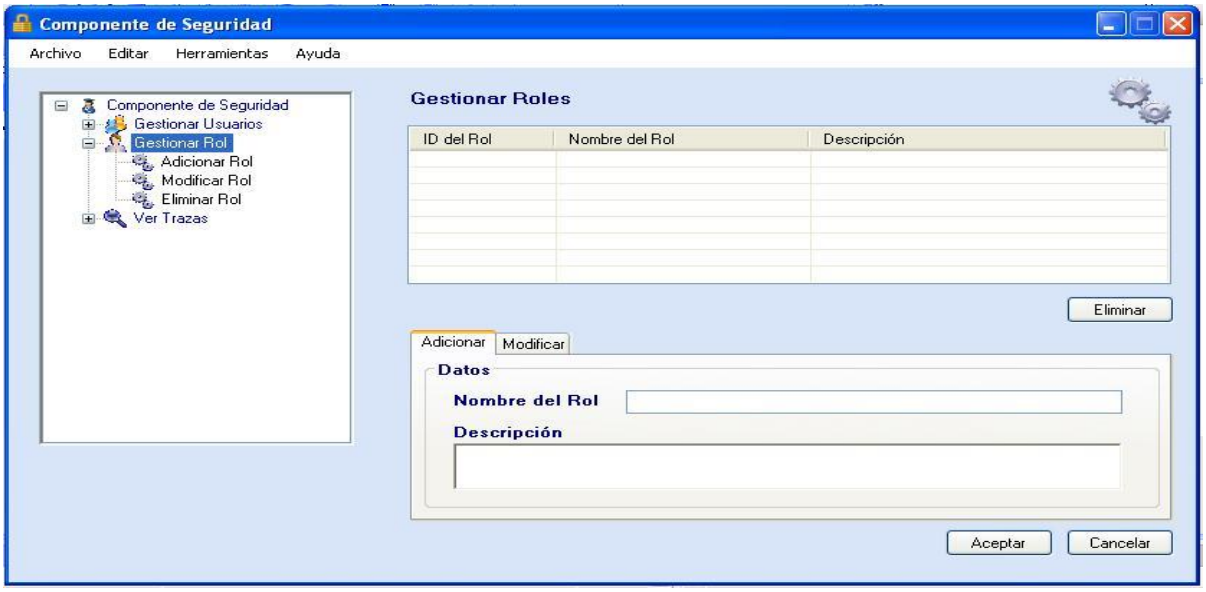
Interfaz: Adicionar usuario.



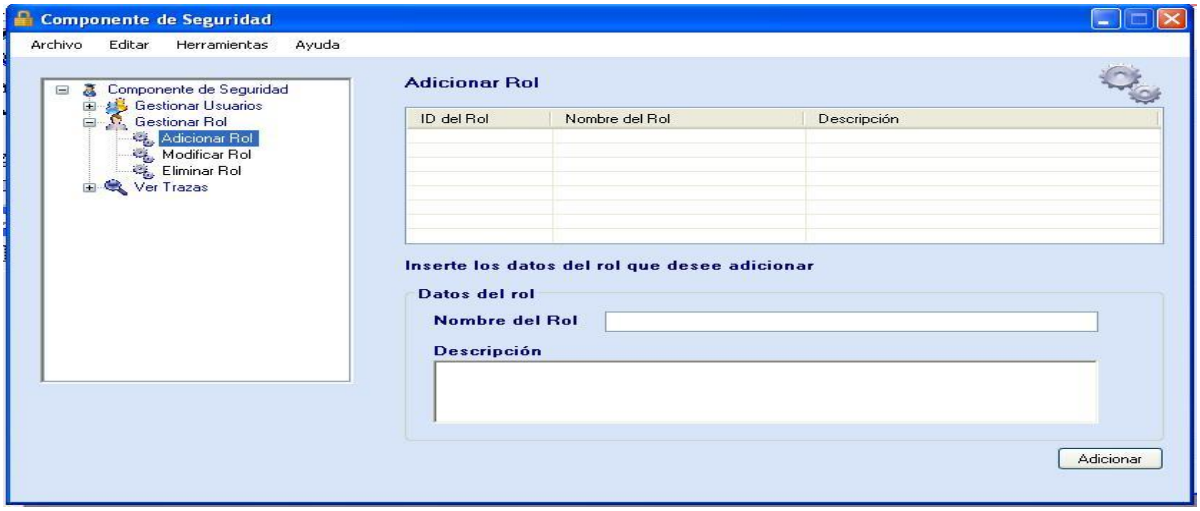
Interfaz: Modificar usuario.



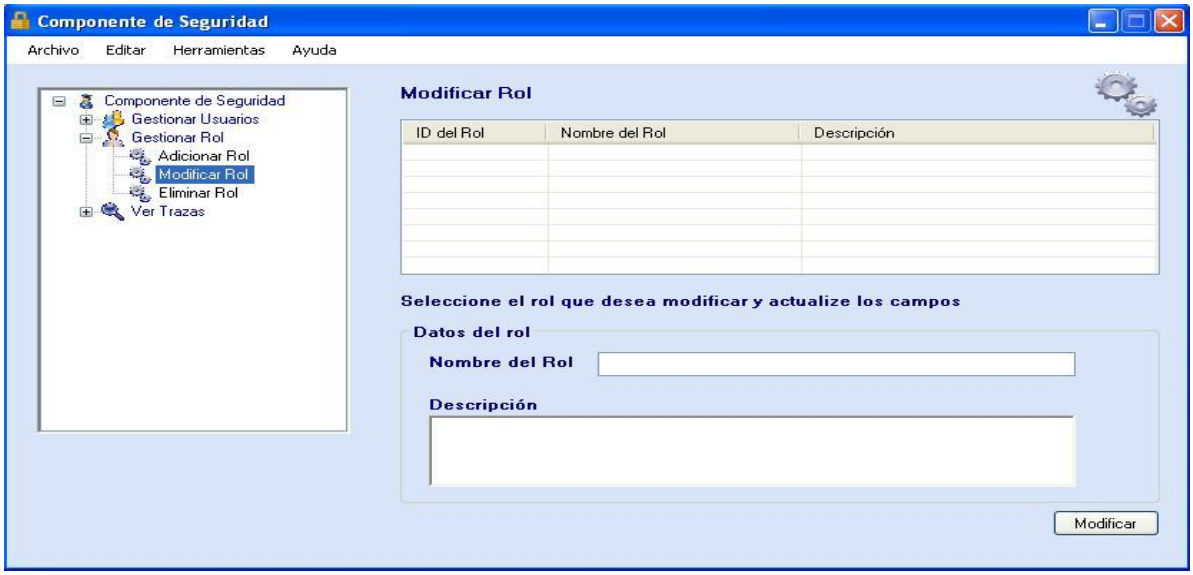
Interfaz: Eliminar usuario.



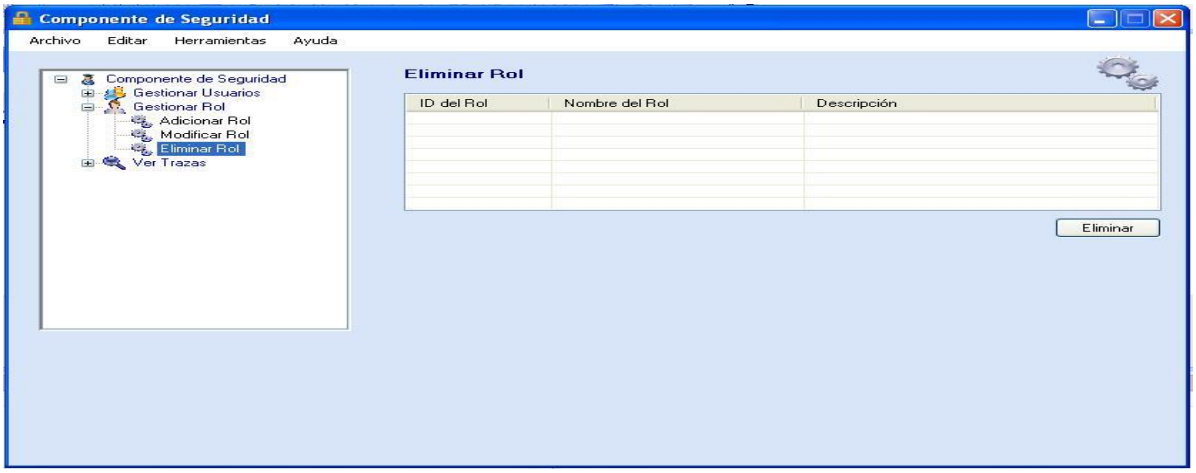
Interfaz: Gestionar Roles.



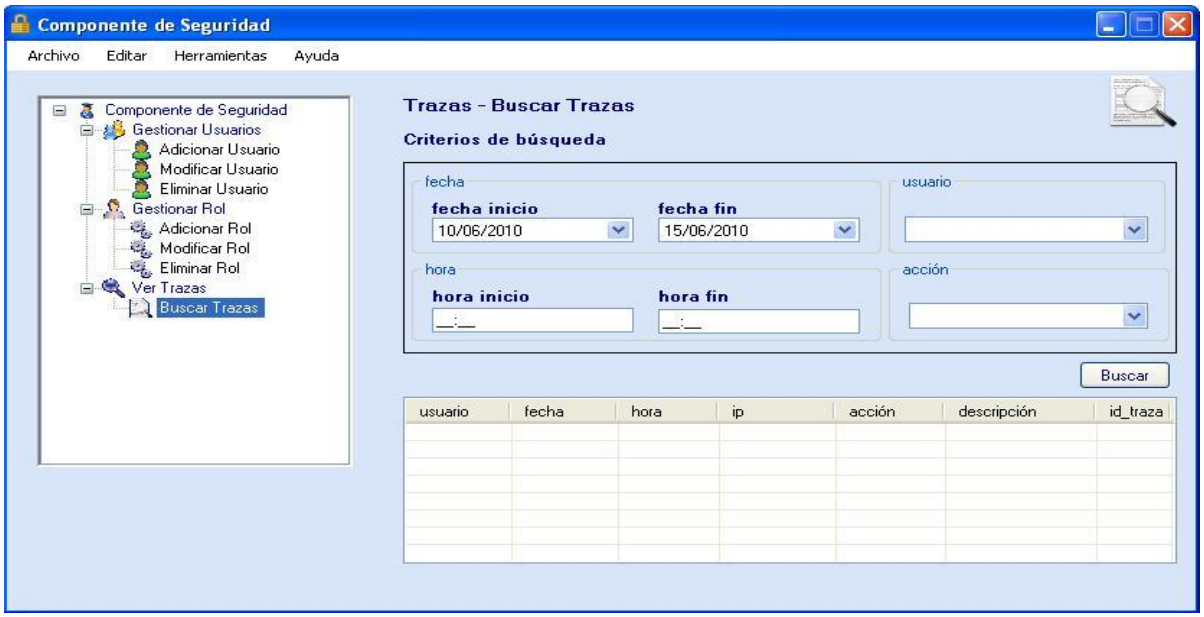
Interfaz: Adicionar rol.



Interfaz: Modificar rol.



Interfaz: Eliminar rol.



Interfaz: Buscar Traza.

Interfaz: Editar Perfil.

Anexo 2: Diagramas de Clases del Análisis

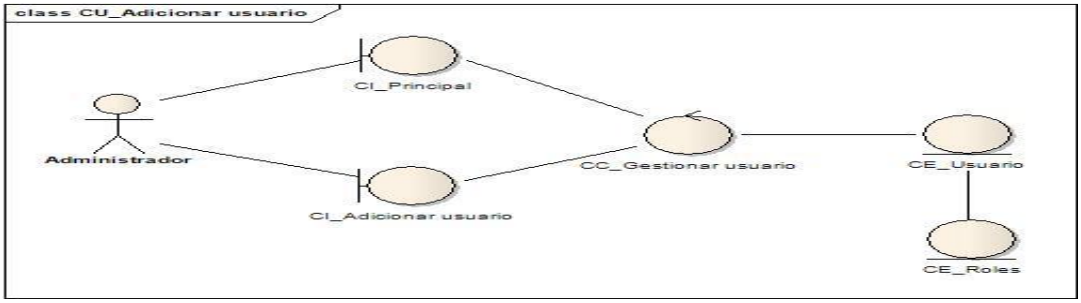


Figura 20: Diagrama de Clase del Análisis: CU_Adicionar usuario.

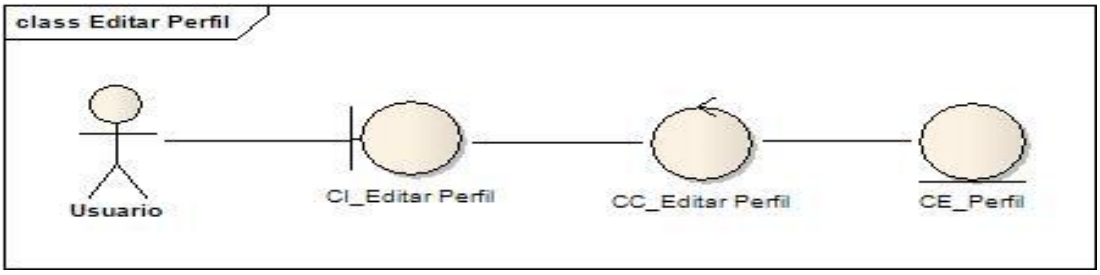


Figura 23: Diagrama de Clase del Análisis: CU_Editar Perfil.

Anexo 3: Diagramas de Colaboración del Análisis

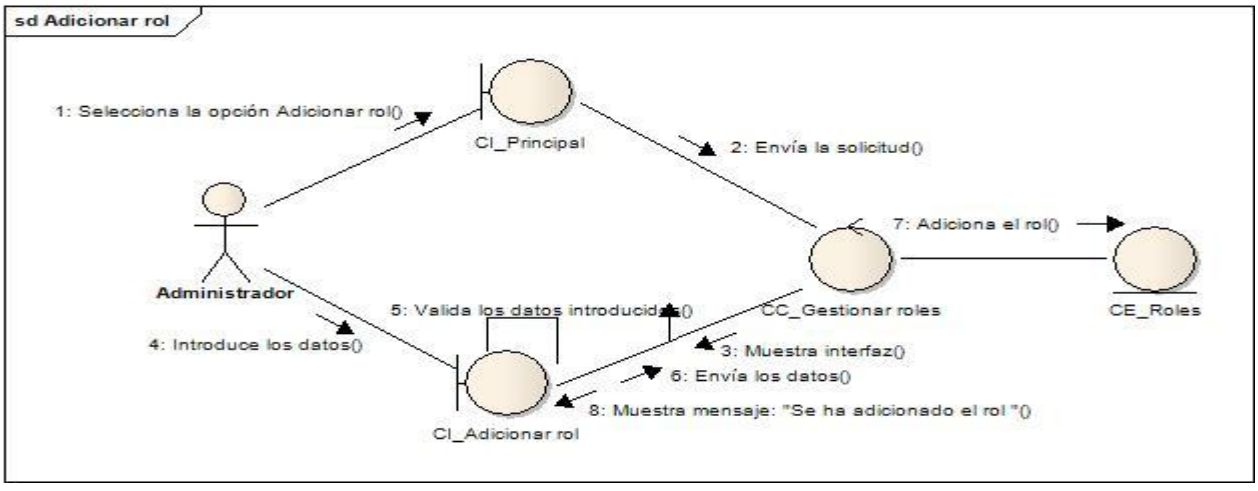


Figura 24: Diagrama de Colaboración del Análisis: CU_Adicionar rol.

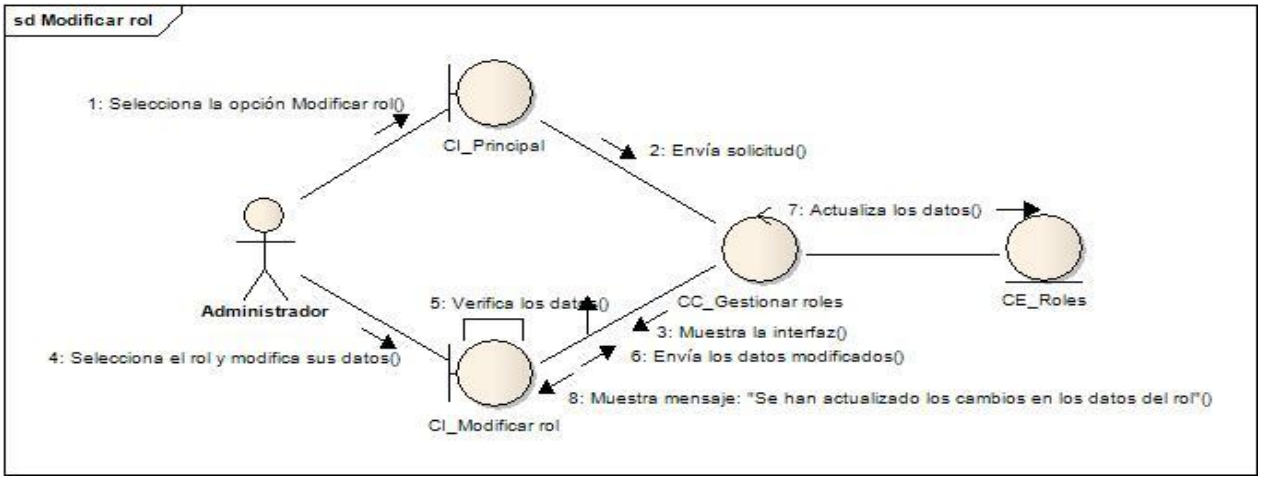


Figura 25: Diagrama de Colaboración del Análisis: CU_Modificar rol.

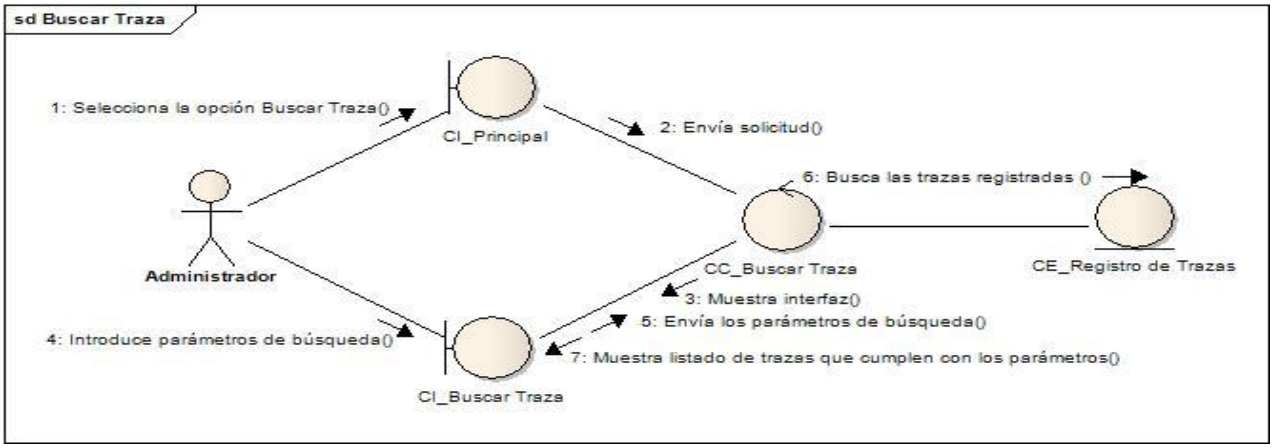


Figura 27: Diagrama de Colaboración del Análisis: CU_Buscar Traza.

Anexo 4: Diagramas de Secuencia

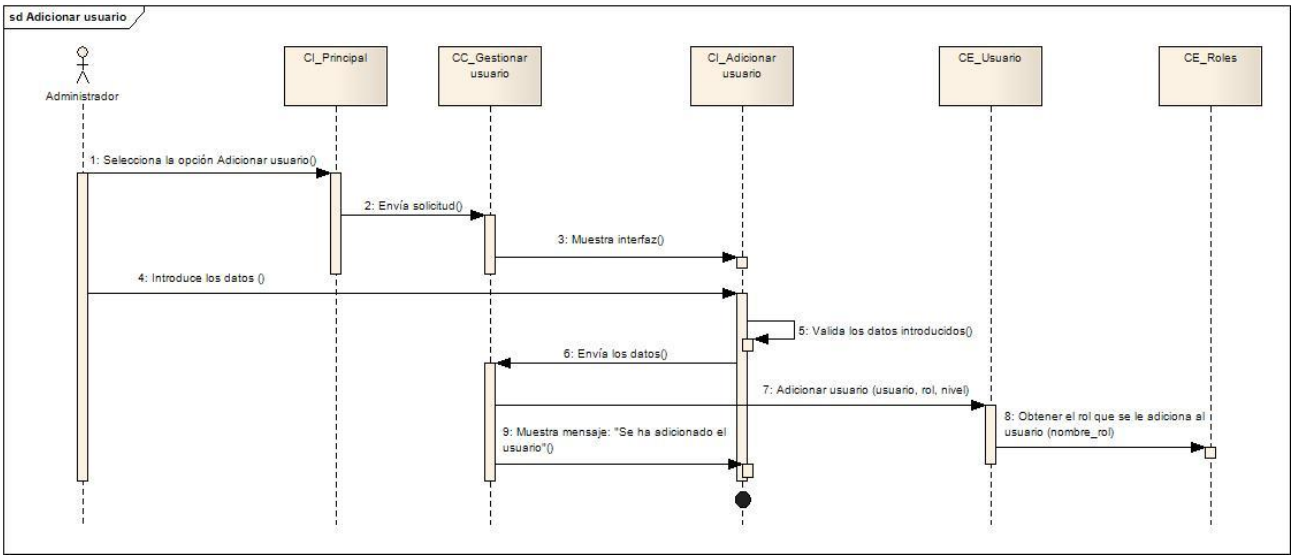


Figura 28: Diagrama de Secuencia: CU_Adicionar usuario.

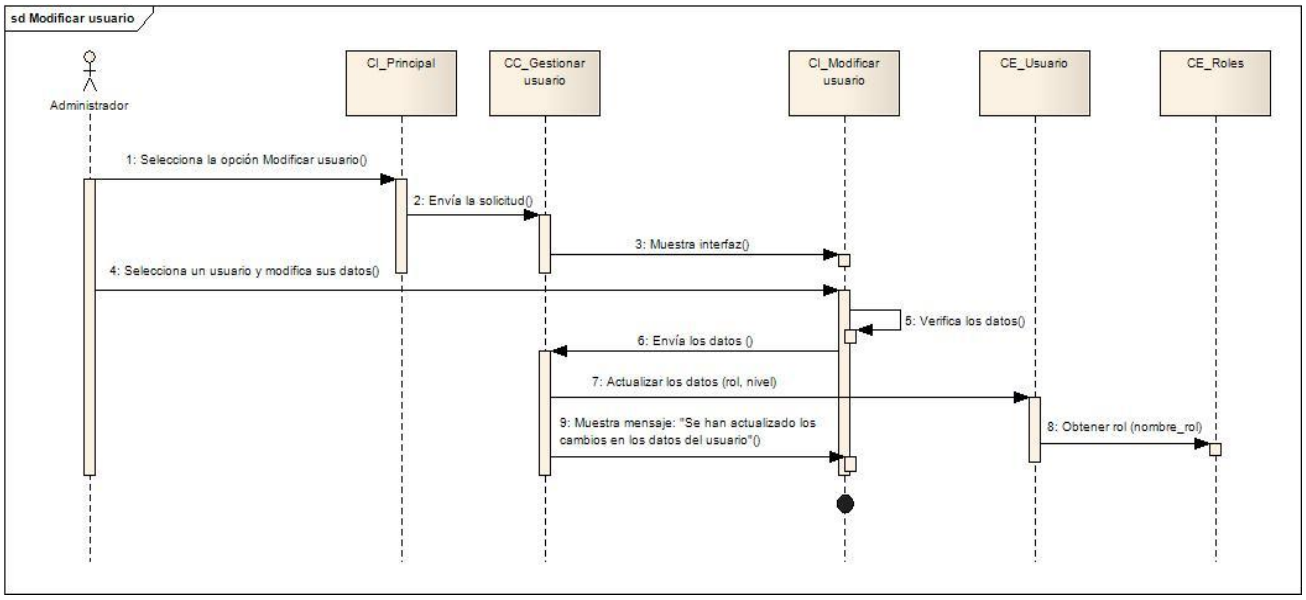


Figura 29: Diagrama de Secuencia: CU_Modificar usuario.

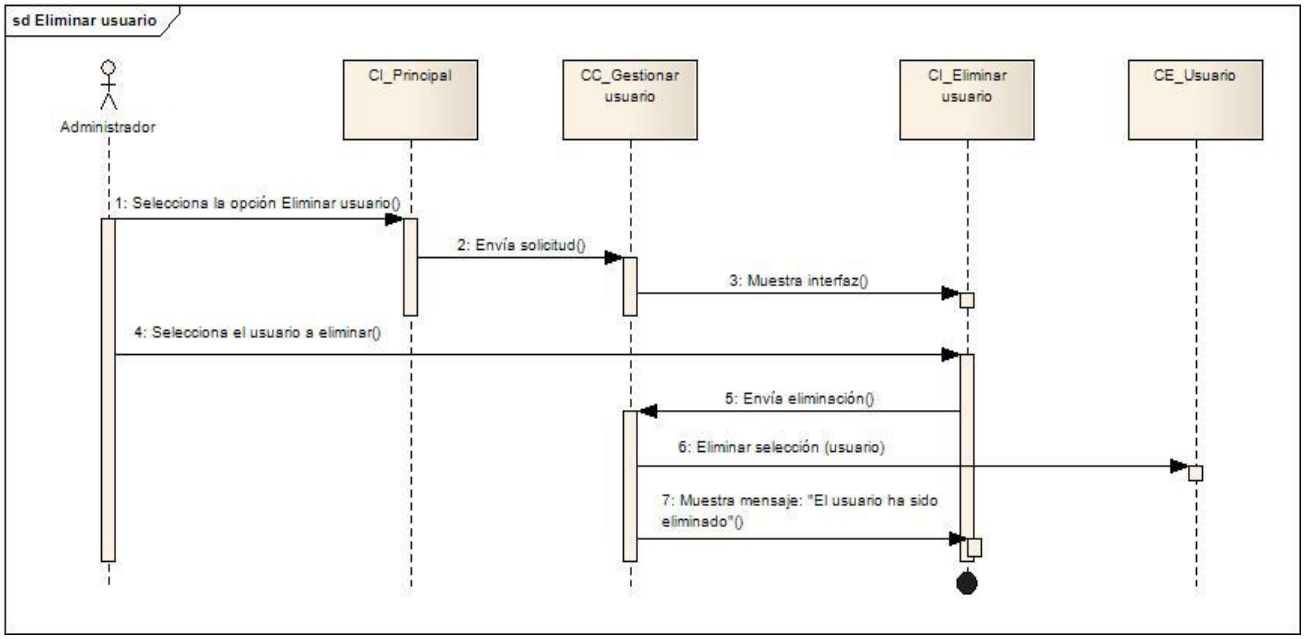


Figura 30: Diagrama de Secuencia: CU_Eliminar usuario.

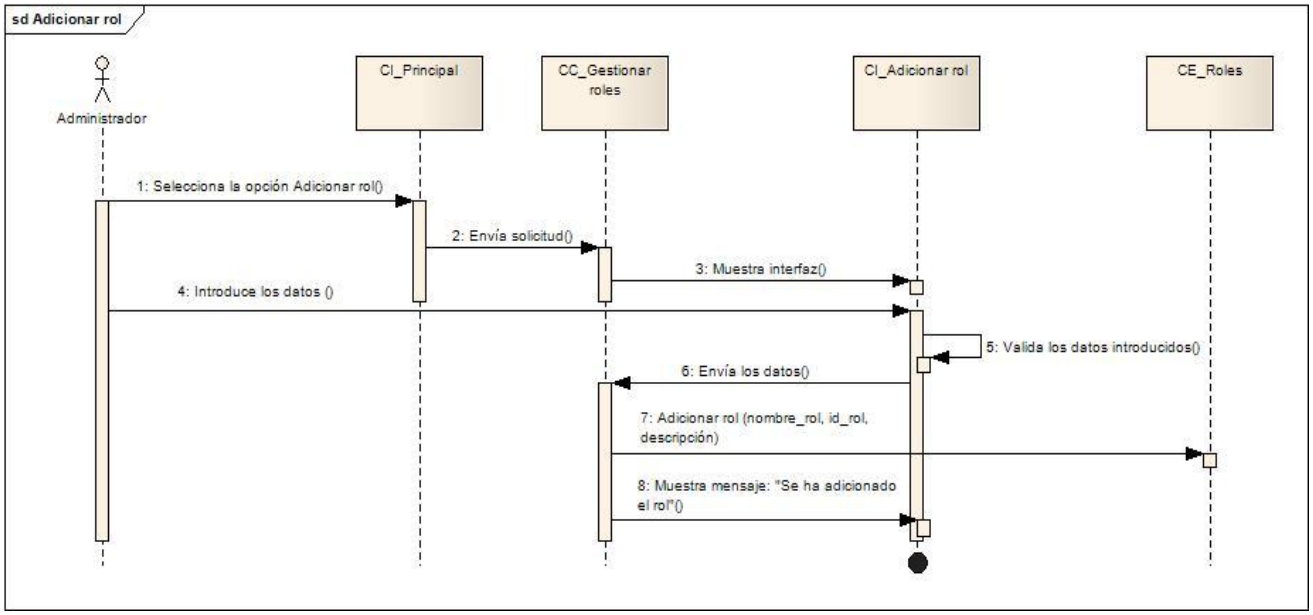


Figura 31: Diagrama de Secuencia: CU_Adicionar rol.

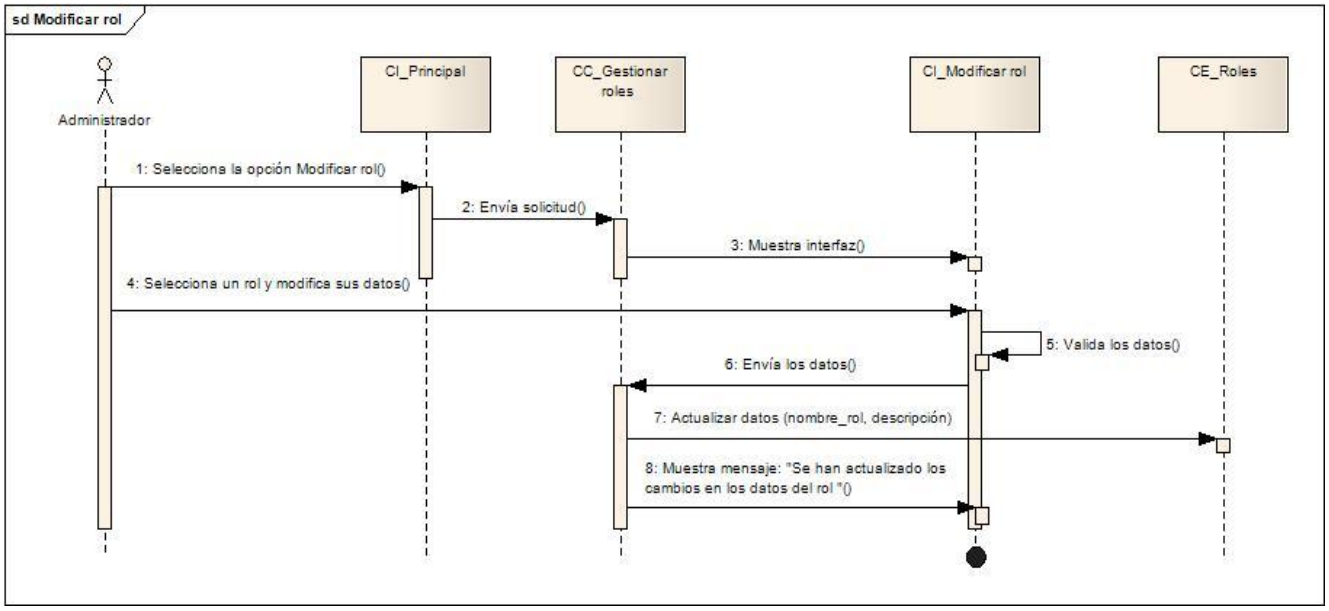


Figura 32: Diagrama de Secuencia: CU_Modificar rol.

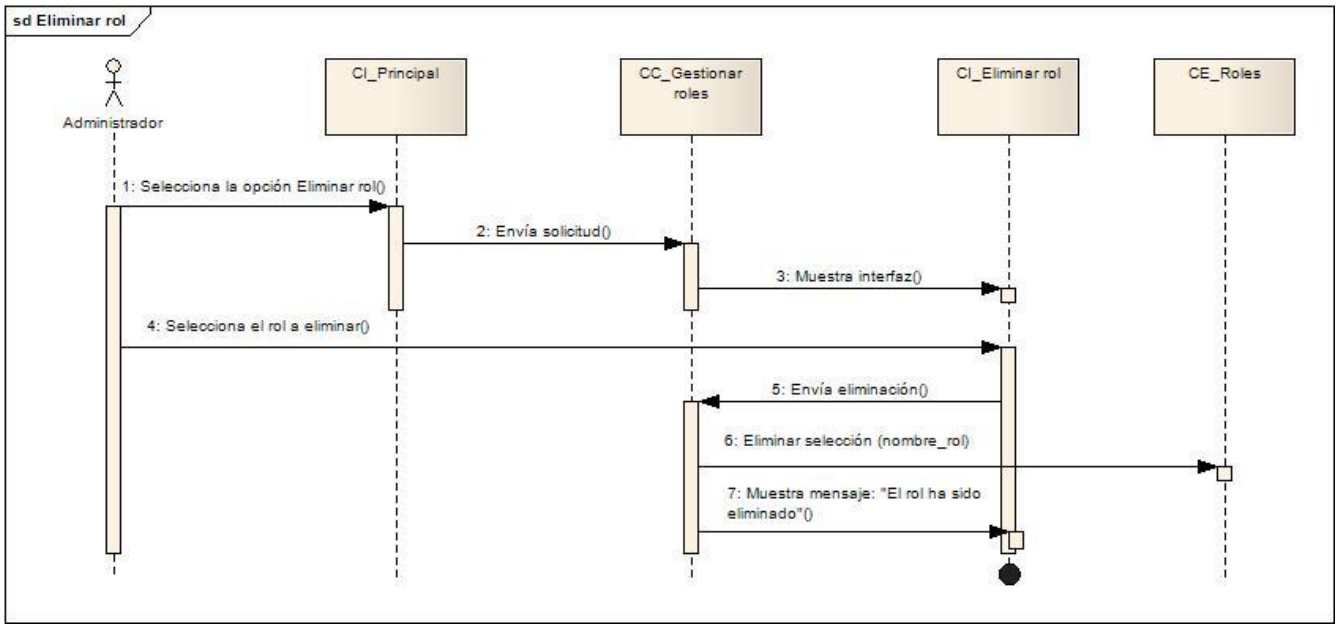


Figura 33: Diagrama de Secuencia: CU_Eliminar rol.

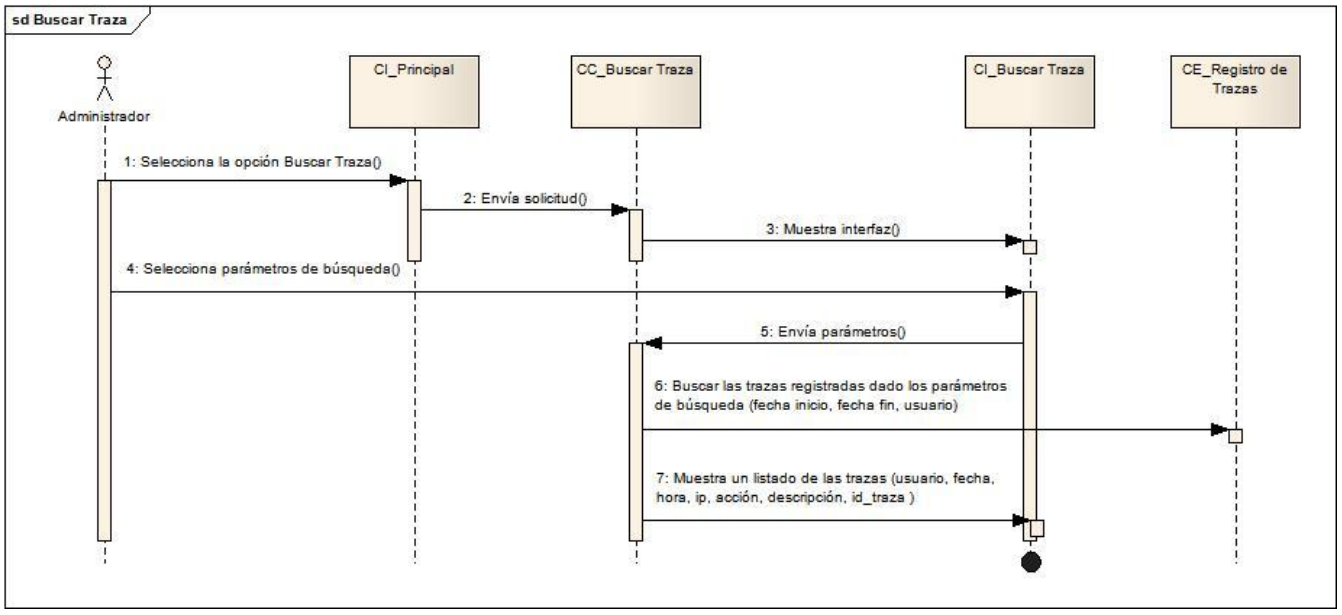


Figura 34: Diagrama de Secuencia: CU_Buscar Traza.

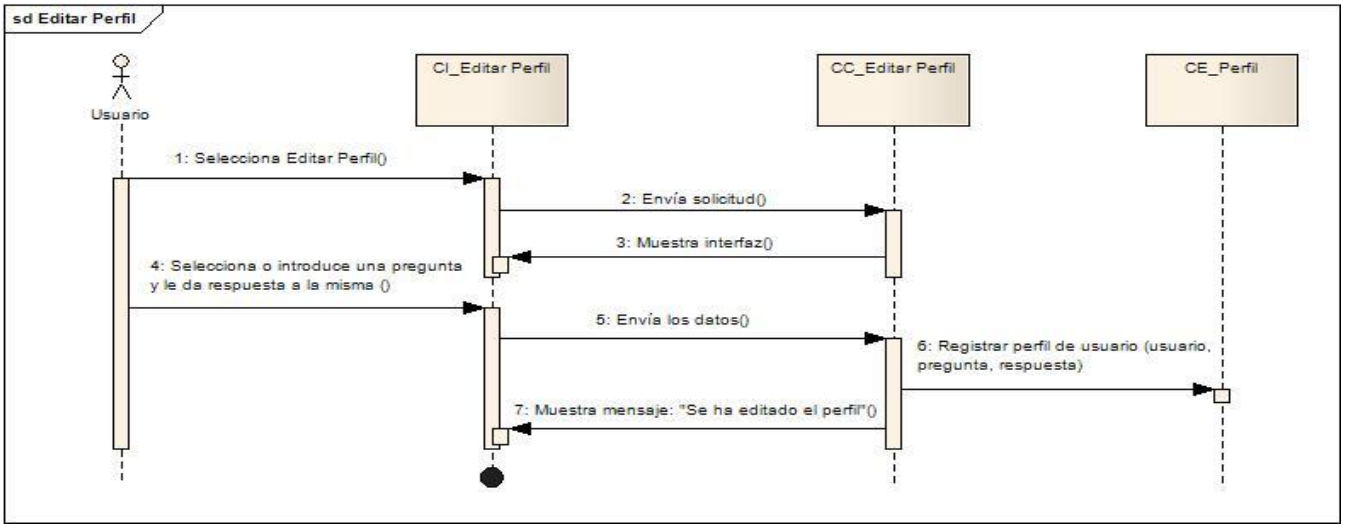


Figura 35: Diagrama de Secuencia: CU_Editar Perfil.

Glosario de Términos

Autenticación: Mecanismo del sistema de información para poder identificar a los usuarios que acceden a sus recursos, asegurando la integridad y autenticidad de los datos.

Autorización: Proceso por el cual se autoriza al usuario identificado a acceder a determinados recursos del sistema.

LDAP: Lightweight Directory Acces Protocol. Es un protocolo de tipo cliente – servidor para acceder a un servicio de directorio ordenado y distribuido para buscar información en un entorno de red.

SAAA: Sistema de Autenticación, Autorización y Auditoría. Su propósito es gestionar la seguridad de las aplicaciones del SNS, mediante la administración de usuarios y la asignación de privilegios a estos, posibilita que un usuario se autentique y acceda a las aplicaciones según lo permisos asignados.

SAML: Security Assertion Markup Language. Es un estándar XML que permite el intercambio de autenticaciones y autorizaciones entre entidades no relacionadas.

SGML: Standard Generalized Markup Language. Consiste en un sistema para la organización y etiquetado de documentos. Sirve para especificar las reglas de etiquetado de documentos.

SSL: Secure Socket Layer. Proporciona cifrado de datos, autenticación de servidores, integridad de mensajes y, opcionalmente, autenticación de cliente para conexiones TCP/IP.

SSO: Single Sing-On. Mecanismo que permite la autenticación única por parte del usuario para acceder a sus recursos, es decir, introducir una sola vez el usuario y contraseña, sin necesidad de volver a poner el login a la hora de acceder a otros recursos en los que aún no se había autenticado.

XACML: Extensible Access Control Markup Language. Es un estándar que describe un lenguaje y una política de control de solicitud de acceso/respuesta, escritos en XML.

XML: Extensible Markup Language. XML es un conjunto de reglas para definir etiquetas semánticas que organizan un documento en diferentes partes. Es un metalenguaje que define la sintaxis utilizada para definir otros lenguajes de etiquetas estructurados.