



Universidad de las Ciencias
Informáticas

Universidad de las Ciencias Informáticas

Facultad 7

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

Mecanismo de control de acceso en los servicios web y gestión de trazas en el
Sistema Nomenclador de Información

Autores: Arlety Sánchez Santos
Yunior Pacheco Correa

Tutores: MSc. Annia Arencibia Morales
Ing. José Mojena Alpizar

La Habana, junio de 2013

“Año 55 de la Revolución”



“Lo que sabemos es una gota, lo que ignoramos es un océano. La admirable disposición y armonía del universo, no ha podido sino salir del plan de un Ser omnisciente y omnipotente”.

Isaac Newton (1643-1727)

Datos de Contacto

MSc. Annia Arencibia Morales (aarencibia@uci.cu): graduada de Ingeniero en Ciencias Informáticas en la Universidad de las Ciencias Informáticas en el año 2007. Profesora de la Universidad de las Ciencias Informáticas, pertenece al Centro de Informática Médica (CESIM), actualmente posee la categoría docente de profesor Asistente. Máster en Informática Aplicada. Profesora de Metodología de la Investigación Científica e imparte Postgrado de Ingeniería de Software. Pertenece al Departamento de Sistemas de Apoyo a la Salud.

Ing. José Mojena Alpizar (jmojena@uci.cu): obtuvo el título de Ingeniero en Ciencias Informáticas en la graduación del curso 2010-2011, de la Universidad de las Ciencias Informáticas (UCI). Es miembro del Centro de Informática Médica (CESIM), se desempeña como programador del producto Synta del Departamento. Sistemas de Apoyo a la Salud.

Dedicatoria

A mis padres por tanto amor y confianza depositada en la realización de mis sueños y conquistas.

A la Revolución y a su guía el Comandante Fidel Castro por permitirme estudiar en esta universidad de excelencia, forjadora de ejércitos de luz que siempre abrazarán estos ideales revolucionarios.

A mis compañeros de universidad y de vida por compartir tantos momentos especiales que quedarán impregnados en el libro de mi pasado, presente y futuro.

A todas esas personas que siempre se involucraron en mis proyectos y supieron albergar en sí la convicción profunda de que era capaz de afrontar con creces las circunstancias que me depara la vida.

A los que aportaron su granito de arena en el desarrollo del trabajo que me certificará como ingeniera en ciencias informáticas.

A todos por su entrega infinita,

¡Muchas Gracias!

Arlety

A mis padres, por todos los años de sacrificio para que sus hijos tengan un mejor futuro.

A mi hermana, la mejor hermana que tengo.

Yunior

Agradecimientos

A mis padres Rosita y Mario por regalarme esta vida repleta de grandes emociones, por educarme sabiamente inculcando a mi personalidad valores como la honestidad, la modestia y la solidaridad.

A mis abuelos Luis y Modesta por obsequiarme una familia tan especial y ser esa fuente inspiradora de confianza y amor.

A mis tías por estar en todo momento participando en mi educación y compartiendo mis alegrías y tristezas. A Osvaldo por ser como mi segundo padre. A mi primo Diley por ser ese hermano que no tuve.

A mi adorable novio Jaime por su comprensión, apoyo y confianza, por estar siempre al doblar de un abrazo cuando más lo necesitaba, impregnándose de amor y optimismo. A sus padres gracias por acogerme con tanto cariño como a una hija más.

A mi hermana Rita Milena por involucrarse profundamente en cada paso que diera en esta universidad y en la vida. Gracias por ser mi eterna confidente y por brindarme esa fuerza que me hace recordar a cada instante quien soy a hacia donde voy.

A mis compañeros de grupo por compartir estos cinco años inolvidables y admitirme tal y como soy, con mis defectos y virtudes. Especialmente a mis compañeros de equipo por tanta ayuda y apoyo.

A mis tutores Annia y Mojena por tanta dedicación, apoyo y enseñanzas tras la materialización de esta investigación.

A mi excelente compañero de tesis por tanta dedicación y constancia para con el desarrollo de este trabajo.

A mis compañeras de apartamento, a amigos Claudia y Enrique por su grata compañía y apoyo, a Odelys, Zula, Betty por ser tan buenas y especiales conmigo, a Yeni y a su novio Arieskjen por brindarme su mano amiga cuando más lo necesitaba, gracias.

A mis segundas madres Edith, Leticia y Mayra por confiar en mis capacidades y aconsejarme a transitar por los senderos más importantes de la vida.

A mis profesores de todos los tiempos por enseñarme el valor de una carrera universitaria y la importancia de elevar nuestros conocimientos profesionales y culturales.

A mis amigos de la FEU por permitirme realizar grandes sueños y agradecer mi vida universitaria con pequeños detalles que se guardan como grandes obras y caudal de realizaciones concretas.

A la familia de Radio Ciudad Digital y del Centro Cultural por enseñarme a vincular los estudios con el arte aficionado, por ser esa fuente inspiradora de destreza que ha contribuido en mi vocación más anhelada: la locución.

A todas las personas que estuvieron un día y hoy no están físicamente, pero que su amor y ejemplo invaden mi interior.

Gracias UCI por esta fortuna de ser hoy una profesional a la altura de este momento histórico.

Gracias a mi madre por ser esa luz de bondad y optimismo que me enseñó a creer en mí en los momentos más difíciles; a ti dedico especialmente este triunfo que será el comienzo de otra etapa de mi vida que espero siga siendo fruto de orgullo para ti.

Arlety

A los profesores que han marcado mi carrera de estudiante, en la secundaria, en el pre y ahora que termino en la universidad.

A mis tutores Annia y José que me apoyaron y me ayudaron cuando lo necesitaba.

A mis compañeros de aula, a la gente del apartamento. A mis amigos, los que conocí aquí y los que han arrastrado conmigo por casi ya 10 años.

A mi compañera de tesis, por ser compañera de tesis y amiga.

A toda mi familia, los que veo poco, los más cercanos, los que viven en el extranjero, mis tías y tíos, mis primos, mi prima abogada. A mi hermana, que hace por mí lo que sea, y a mi cuñado, que sin compartir el apellido, es uno más de la casa.

A mis padres, las personas que más me quieren en el mundo. Mi mamá: cariñosa, trabajadora y dedicada y mi papá, que me ha dado el mejor ejemplo de hombre que un hijo puede tener.

Yunior

Resumen

El Sistema Nomenclador de Información es un sistema reutilizable y generalizable, que se puede integrar a cualquier sistema que cuente en su desarrollo con información nomenclada. Para dicha integración es necesario el consumo de los servicios web que este brinda. Estos servicios están expuestos a cualquier persona (sea usuario o no del sistema); por no contar con un mecanismo de control de acceso. El consumo de dichos servicios no garantiza la confidencialidad de la información. Además se imposibilita desde el sistema realizar procesos de auditoría (teniendo en cuenta las trazas de los usuarios, por no contar con funcionalidades desarrolladas).

Como respuesta al problema planteado, se incorporaron funcionalidades al Sistema Nomenclador de Información; estas permiten la gestión de trazas teniendo en cuenta el registro de las operaciones que se realizan en el sistema y quiénes las ejecutan. Además se incorporó un mecanismo de control de acceso a los servicios web que brinda el sistema. Para el desarrollo de la solución, se utilizó Symfony como *framework* de desarrollo de aplicaciones web, este utiliza el patrón de arquitectura de software Modelo Vista Controlador. Como lenguaje de programación se utilizó Hypertext Pre-processor, el cual incluye el Mapeador de Objetos Relacional Doctrine. Como herramienta de modelado se empleó Enterprise Architect, y para el análisis y diseño el Lenguaje Unificado de Modelado. Además se utilizó como sistema de gestión de base de datos PostgreSQL.

Palabras clave: *auditoría, nomenclador, servicios web, trazas*

Índice

Introducción	4
Capítulo 1: Fundamentación teórica de la seguridad en los servicios web y trazas en los sistemas informáticos	8
1.1 Definiciones relacionadas con el campo de acción	8
1.2 Antecedentes	9
1.2.1 Características de algunos sistemas internacionales.....	9
1.2.2 Características de algunos sistemas nacionales	10
1.3 Modelo de Autenticación.....	11
1.3.1 Sistema de Autenticación Básica HTTP	11
1.3.2 Capa de Conexiones Seguras.....	11
1.3.3 HTTPS	12
1.4 Tecnologías, técnicas, metodologías y software en las que se apoya la solución del problema...	12
Capítulo 2: Características de las funcionalidades gestionar trazas y seguridad de los servicios web.....	20
2.1 Propuesta de solución.....	20
2.2 Modelo de Dominio.....	21
2.2.1 Conceptos fundamentales	21
2.2.2 Diagrama del Modelo de Dominio	23
2.3 Requerimientos del Software.....	23
2.3.1 Requerimientos Funcionales (RF)	24
2.3.2 Requerimientos no Funcionales (RNF)	26
2.4 Actores del Sistema.....	29
2.5 Casos de Uso.....	29
2.6 Diagrama de Casos de Uso del Sistema.....	30
2.7 Descripción textual de casos de uso del sistema	32
Capítulo 3: Análisis, diseño e implementación de las funcionalidades gestionar trazas y seguridad de los servicios web	34
3.1 Descripción de la arquitectura	34

3.2	Modelo de análisis	35
3.2.1	Diagramas de Clases del Análisis	35
3.2.2	Diagramas de Comunicación.....	36
3.2.3	Definición de elementos del diseño.....	38
3.2.4	Diagramas de Clases del Diseño	39
3.2.5	Descripción de las clases.....	40
3.3	Modelo de datos	42
3.4	Descripción de la funcionalidad gestión de trazas y control de acceso en los servicios web XML	44
3.4.1	Gestión de trazas.....	44
3.4.2	Control de acceso en los servicios web XML.....	45
3.5	Diagrama de Despliegue.....	46
3.6	Modelo de implementación	46
3.7	Diagrama de componentes.....	47
	Conclusiones.....	54
	Recomendaciones	55
	Referencias Bibliográficas.....	56
	Bibliografía.....	61
	Anexos	69

Introducción

Durante las dos últimas décadas, el desarrollo tecnológico mundial ha mostrado una convergencia cada vez mayor entre la Informática, las Telecomunicaciones y la Automatización; procesos que han devenido en una nueva rama del saber, denominada Tecnologías de la Información y las Comunicaciones (TIC) (MIC, 2012). Con la llegada de las TIC, surgió la necesidad de crear una nueva sociedad conocida como Sociedad Digital o Sociedad de las Telecomunicaciones y la Información (Díaz, 2012). La utilización de las tecnologías ha permitido compartir los recursos de información, quebrantar las fronteras geográficas y acelerar los procesos de comunicación, fomentando un nuevo paradigma, sin importar clases sociales o diferencia de idioma. Disponer de una adecuada organización de la información ahorra tiempo y esfuerzo. (Delgado, 2010)

En la actualidad, la aplicación de las herramientas informáticas, requiere de un sistema organizado que permita tener la capacidad de transferir la información de forma rápida y eficaz, logrando acortar los tiempos de desarrollo. La mayoría de los sistemas desarrollados poseen la gestión de los nomencladores¹ de forma aislada, implicando que todos la definan de manera similar, ajustando cada uno al negocio al cual pertenece. En algunos casos la gestión de los nomencladores presenta implementaciones muy básicas (crear, leer, actualizar y eliminar), lo cual atenta considerablemente contra las aplicaciones informáticas que los utilizan; debido a que si cambia esta información, el sistema puede correr el riesgo de tener que volver a implementarse desde cero. (Morales, 2012 pág. 7) (sic)

A nivel internacional y nacional se crearon diversos sistemas (Fernández, 2009) (Particulares, 2008) (Díaz, 2009) que gestionaban y organizaban información nomenclada, pero no eran flexibles al cambio de estructura, atributos e inserción de nueva información. Para resolver estos problemas en el Centro de Informática Médica (CESIM), específicamente en el Departamento de Sistemas de Apoyo a la Salud (SAS) se creó un Sistema Nomenclador de Información (SNI); este presenta un alto nivel de reutilización y generalización, permitiendo su integración con cualquier sistema que necesite información nomenclada.

El sistema fue desarrollado para gestionar información común poco variable en el tiempo (más conocida como nomencladores) de forma jerárquica. Se utiliza como mecanismo de apoyo a otras aplicaciones,

¹ nomencladores: catálogo que tiene la nomenclatura de una ciencia. (Diccionario, 2013)

disminuyendo su tiempo de desarrollo y permitiendo que estas se centren en el consumo de servicios web XML² y no en la implementación de los nomencladores. Los sistemas que consumen la información que gestiona el SNI no tienen que reprogramarse ante cambios inesperados en la información nomenclada, pues realizan la integración haciendo uso de servicios web XML, independientemente de la plataforma en que estén desarrollados.

El SNI presenta un conjunto de funcionalidades que posibilitan gestionar la información de forma jerárquica, creando para cada grupo un conjunto de campos flexibles al cambio, en función de las necesidades propias de cada negocio. A través del sistema se pueden gestionar los usuarios que van a interactuar con él. Una vez creados, se le deben asignar los nomencladores a los cuales tendrán acceso. Posteriormente estos usuarios pueden incorporar, a cada nomenclador, la información necesaria para su uso. El SNI presenta una estructura interna compleja, donde toda la información (relacionada con la jerarquía de los nomencladores) es almacenada en una sola tabla de la base de datos. Cada tupla de esta tabla está relacionada con otra tupla; esta relación permite saber quién es el padre y quién es el hijo en un nomenclador determinado (el sistema usa esa relación para crear la jerarquía del nomenclador).

Toda la información que el usuario modifica o elimina en el sistema es imposible de recuperar, en caso de que se necesite o se quiera llevar el sistema al estado anterior. Para recuperar esta información, el usuario debe hacerlo de forma manual (debe volver a insertar la información en el sistema), repercutiendo esto en las aplicaciones que consumen los servicios del SNI, las cuales no podrán utilizar su información variable en el tiempo, lo que impediría o retrasaría su funcionamiento. Además no se puede conocer quién fue el responsable de dicho cambio, porque el sistema no registra las operaciones ejecutadas por los usuarios, impidiendo de esta forma la realización de auditorías. Actualmente en el SNI no se puede identificar el usuario que consume los servicios web, ni conocer si es válido y cuenta con los permisos necesarios para acceder a la información solicitada; poniendo en riesgo la confidencialidad de la información.

² XML: *eXtensible Markup Language*, lenguaje de marcas.

El SNI presenta otras deficiencias que se listan a continuación:

- No permite exportar información nombrada en formatos como: excel, .doc., .xml; lo que trae consigo que los usuarios no puedan adquirir la información que necesitan, para utilizar fuera del sistema.
- No permite importar la información ya nombrada en formatos como: excel, .doc., .xml; de esta manera los usuarios que deseen introducir alguna información al sistema lo tienen que realizar manualmente, resultando complejo y engorroso.
- No permite definir valores fijos o rango de valores en posiciones específicas del código de un grupo, ocasionando rigidez y poca flexibilidad al crear el código correspondiente al grupo. Provocando que el sistema no pueda ser utilizado por nomencladores con estructuras de código avanzadas y de gran complejidad.

Luego de un análisis de la situación problemática, se plantea como **problema a resolver**: el Sistema Nomenclador de Información presenta vulnerabilidades, debido a que no se controla la validez de los consumidores de los servicios web XML que brinda, ni se registran las acciones ejecutadas por los usuarios del sistema. Definiéndose como **objeto de estudio**: Proceso de Autenticación, Autorización y Auditoría. Centrándose en el **campo de acción**: gestión de trazas y mecanismo de control de acceso en los servicios web XML, del Sistema Nomenclador de Información. Planteándose como **objetivo general**: implementar funcionalidades que permitan la gestión de trazas de los usuarios que interactúan con el sistema y un mecanismo de control de acceso de los servicios web XML, en el Sistema Nomenclador de Información.

Para solucionar el objetivo planteado se proponen las siguientes **tareas de la investigación**:

- Evaluar los sistemas que gestionen trazas de usuarios y seguridad en los servicios web a nivel internacional y nacional, estableciendo similitudes y diferencias con la investigación en curso.
- Actualizar los artefactos correspondientes a los flujos de trabajo propuestos por la metodología de desarrollo Proceso Unificado de Desarrollo (RUP), sirviendo de base para el desarrollo del sistema.
- Asimilar las herramientas y tecnologías propuestas por el Departamento de Sistemas de Apoyo a la Salud (SAS), para el desarrollo de la solución.
- Implementar funcionalidades que brinden seguridad a los servicios web XML del Sistema Nomenclador de Información y gestionen las trazas de los usuarios que interactúan con el mismo.

Entre los **métodos científicos** utilizados en la investigación se destacan:

➤ **Métodos Teóricos:**

- **Analítico-Sintético:** se empleó para descomponer el problema de investigación en el estudio por separado de los procesos de gestión de trazas y la seguridad de los servicios web.
- **Histórico-Lógico:** se utilizó para profundizar en los antecedentes de sistemas que brindan seguridad en sus servicios web y permitan gestionar las trazas de los usuarios que interactúen con el sistema.
- **Inductivo-Deductivo:** para obtener un grupo de conocimientos, analizando los datos generales válidos sobre cómo se realiza la gestión de trazas y seguridad en los sistemas informáticos, llegando a conclusiones sobre ambos procesos.
- **Modelación:** para crear abstracciones de la realidad, con vistas a explicar la misma a partir de diagramas.

El desarrollo del trabajo se estructura en tres capítulos que se describen a continuación:

Capítulo 1: Fundamentación teórica de la seguridad en los servicios web y trazas en los sistemas informáticos: se incluye el estado del arte del tema a tratar tanto a nivel internacional como nacional. Se realiza un análisis sobre la seguridad en los servicios web en los sistemas informáticos. Además se describen las herramientas, tecnologías y metodologías a usar en la solución del problema planteado.

Capítulo 2: Características de las funcionalidades gestionar trazas y seguridad de los servicios web: se especifican los requerimientos funcionales y no funcionales, a partir de los cuales se representan los casos de uso del sistema y la descripción de los mismos. Además se implementan los mecanismos definidos para la seguridad de los servicios web y la gestión de trazas de los usuarios.

Capítulo 3: Análisis, diseño e implementación de las funcionalidades gestionar trazas y seguridad de los servicios web: se justifican los patrones a usar en el diseño de las funcionalidades, se define la estructura y elementos del análisis y el diseño, mostrando los diagramas de clases de análisis y diseño.

Capítulo 1: Fundamentación teórica de la seguridad en los servicios web y trazas en los sistemas informáticos

En el presente capítulo se realiza un estudio del estado del arte sobre la seguridad de los servicios web en sistemas informáticos, tanto a nivel internacional como nacional. Además se analizan herramientas, tecnologías y metodologías a usar en la solución del problema planteado.

1.1 Definiciones relacionadas con el campo de acción

Servicios web XML

El servicio web, es una tecnología que permite a las aplicaciones comunicarse sin presentar dependencia de la plataforma ni del lenguaje de programación. Un servicio web constituye una interfaz de software que describe un conjunto de operaciones, a las cuales se puede acceder por la red, a través de mensajería xml estandarizada. Usa protocolos basados en el lenguaje XML con el objetivo de describir una operación para ejecutar o para intercambiar datos con otro servicio web. (IBM, 2013)

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 una computadora u otro medio electrónico, una computadora por sí misma o un programa. En otras palabras, es un modo de asegurar que los usuarios son realmente quienes dicen ser y que tienen la autorización para realizar funciones en el sistema. (Velázquez, y otros, 2009)

Autorización

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 a aquellos recursos sobre los cuales tienen privilegios. (Velázquez, y otros, 2009)

Auditoría

Es la capacidad de un sistema de registrar eventos y rastrear la actividad del usuario mientras accede a los recursos, soportando sistemas de trazas. A través de la auditoría se detectan irregularidades en un sistema (indicando qué ha pasado, sobre qué información y quién ha accedido; formando parte imprescindible de cualquier sistema que pretenda proporcionar seguridad a la información). (RedUsers, 2013)

Trazas

Las trazas se gestionan en cualquier tipo de sistema donde se requiera hacer una revisión de los eventos, errores cometidos y las acciones realizadas por los usuarios dentro del mismo. Además permiten crear un registro de sucesos para un dispositivo o aplicación. A través de estos datos, se puede monitorear el funcionamiento de la aplicación y se ofrece una oportunidad para corregir problemas de seguridad.

(Velázquez, y otros, 2009)

1.2 Antecedentes

A partir de un estudio de los sistemas internacionales y nacionales existentes que gestionan trazas e implementan la seguridad en servicios web, se detectaron algunos sistemas que manejan en su solución algunas de las funcionalidades propuestas.

1.2.1 Características de algunos sistemas internacionales

➤ Bitácora

Sistema producido por la empresa S21sec, encargado de la gestión de trazas a partir de las visitas que registra una web. Esta aplicación permite archivar los datos facilitando su consulta por parte de las personas autorizadas con el fin de realizar auditorías, así como generar alertas en caso de ataques. “El número de *logs* generados por una organización está sufriendo un incremento exponencial”, explican desde la compañía guipuzcoana, que ha prestado especial atención a todo lo relacionado con la seguridad. (Moral, 2004)

➤ Conjunto de Productos Log Logic

Log Logic es una solución de seguridad, que consiste en la instalación de un equipo, permitiendo centralizar y agregar datos de registros crudos de cualquier fuente de datos conectada, analizar esos datos en tiempo real, fijar las alarmas para advertir de algún comportamiento sospechoso y almacenar con seguridad todos los datos para su posterior utilización. Esta solución agrega confiablemente altos volúmenes de *logs* y ofrece búsquedas rápidas y detalladas, esenciales para la mitigación de amenazas, a la vez que automatiza los datos de registros históricos y proporciona el almacenamiento seguro de los *logs*. (Tecnocomputación, 2012)

➤ **Sawmill**

Potente herramienta de análisis de *logs* (registros de sucesos/eventos). Está especialmente diseñado para analizar accesos a servidores web, pero puede procesar casi cualquier registro. Publica una intuitiva interfaz gráfica de usuario, que puede ser usada desde cualquier navegador, para configurar y ejecutar Sawmill, o para ver estadísticas de páginas. Las estadísticas son jerárquicas, atractivas y llenas de enlaces que facilitan la navegación. El programa incluye una completa documentación. Sawmill ofrece una gran cantidad de opciones, incluyendo una base de datos persistente, control sobre la apariencia de las páginas de estadísticas, y opciones de filtrado sobre los *logs*. (Softonic, 2013)

1.2.2 Características de algunos sistemas nacionales

➤ **Herramienta para la gestión y el análisis de trazas en el Sistema de Gestión Integral CEDRUX**

Con el desarrollo de este componente se obtienen beneficios al proceso de desarrollo en Centro de Informatización para la Gestión de Entidades (CEIGE), como la optimización del tiempo de trabajo en la identificación de errores. La correcta gestión de los registros almacenados por este sistema es de gran importancia para apoyar la toma de decisiones en la entidad donde se implante la solución. Permitirá a los equipos de desarrollo saber hacia dónde deben enfocar esfuerzos para optimizar las soluciones que puedan atender contra el rendimiento del sistema. (Camejo, et al., 2012 p. 1)

El componente fue desarrollado en la UCI y persigue el registro de los eventos que ocurren en el sistema y específicamente controla: la ocurrencia de acciones, excepciones, accesos al sistema, integración entre sistemas, integración entre componentes de un sistema y accesos a base de datos. La correcta gestión de las trazas, brindan mayor información y conocimiento de los procesos que se desarrollan a través del sistema, apoyando de esta forma la toma de decisiones. (Camejo, et al., 2012 p. 1)

➤ **Componente de Seguridad para Aplicaciones del Área Temática Sistema de Apoyo a la Salud**

El Componente de Seguridad, permite realizar una gestión eficiente de todos los requerimientos de seguridad, de aquellos sistemas externos que consuman los servicios proporcionado por este, haciendo uso de eficaces procesos de Autenticación, Autorización y Auditoría (creación de usuarios, asignaciones de roles, privilegios de acceso). Por otra parte brinda un eficiente y óptimo proceso de trazabilidad y auditoría, de manera que se lleva un control estricto de las operaciones en que se involucran los usuarios

del sistema. La implementación de servicios web es importante en el sistema, debido a que brinda la posibilidad a componentes externos a consumir algunas de sus principales funcionalidades (autenticar, autorizar, adicionar traza, buscar traza, etc.). Para evitar ataques al sistema usa protocolos seguros de comunicación como HTTPS. (Díaz, y otros, 2008 pág. 85)

Los sistemas expuestos anteriormente no son viables como solución al problema planteado, por no ser de código abierto, imposibilitando modificar el código fuente en función de las necesidades propias. No es posible su integración con el SNI, debido a la compleja estructura del sistema. La recopilación de la información y la gestión de traza deben ser implementadas internamente. Una vez analizados los sistemas, se concluye que estos establecen una estrecha relación entre trazas y seguridad; brindando la posibilidad de realizar auditorías sobre los registros almacenados en busca de amenazas.

1.3 Modelo de Autenticación

1.3.1 Sistema de Autenticación Básica HTTP

Entre los diferentes sistemas de autenticación a utilizar en una aplicación web, la autenticación básica es la más utilizada habitualmente. Esto se debe principalmente, a su sencillez y al amplio soporte que dispone desde hace mucho tiempo en todo tipo de navegadores, aunque adolece de importantes problemas de seguridad que no la hacen recomendable en muchas situaciones. (Patxi, 2006)

La autenticación básica está codificada en la petición HTTP, utilizando el Protocolo Simple de Acceso a Datos (SOAP por sus siglas en inglés). Cuando el servidor de aplicaciones recibe la petición HTTP, se recuperan el nombre de usuario y la contraseña, y se verifican utilizando el mecanismo de autenticación específico del servidor. Al usar esta autenticación, al cliente se le piden nombre y clave de un usuario válido para acceder al recurso, que se mandan sin cifrar por la red desde el cliente al servidor. (IBM, 2009) (López, y otros, 2010)

1.3.2 Capa de Conexiones Seguras

Secure Sockets Layer (SSL) traducido al español como Capa de Conexiones Seguras, es un protocolo que hace uso de certificados digitales para establecer comunicaciones seguras a través de Internet. Utiliza la criptografía de llave pública y provee: autenticación del servidor, cifrado de datos e integridad de los datos en las comunicaciones cliente/servidor. Permite confiar información personal a sitios web, ya que los

datos se ocultan a través de métodos criptográficos mientras se navega en sitios seguros. (Ramírez, y otros, 2011)

1.3.3 HTTPS

Es una combinación del protocolo HTTP (consumido en cada transacción web) con el protocolo SSL/TLS, usada para establecer comunicaciones cifradas en sitios web (Ramírez, y otros, 2011). Básicamente, la página web codifica la sesión con certificado digital. De este modo, el usuario tiene ciertas garantías de que la información que envíe desde dicha página, no podrá ser interceptada y utilizada por terceros. Su funcionamiento consiste en cinco pasos (IBM, 2010):

1. El cliente envía una petición de sesión segura
2. El servidor envía un certificado X.509 que contiene su llave pública
3. El cliente autentica el certificado con una lista de CA (Autoridad Certificadora) conocidas
4. El cliente genera una clave simétrica aleatoria y la cifra utilizando la llave pública del servidor
5. Tanto el cliente como el servidor conocen la clave simétrica y cifran los datos del usuario final utilizando la clave simétrica mientras dure la sesión.(ver Anexo 1)

1.4 Tecnologías, técnicas, metodologías y software en las que se apoya la solución del problema

➤ **Servidor Web Apache**

Apache es un servidor web de código abierto para plataformas Unix (BSD, GNU/Linux), Windows, Macintosh y otras, que implementa el protocolo HTTP y la noción de sitio virtual. Apache es altamente configurable, admite bases de datos de autenticación y negociado de contenido, aunque carece de una interfaz gráfica que ayude en su configuración. Es una aplicación que permite montar un servidor web en cualquier equipo y sistema operativo. Apache soporta PHP como lenguaje de programación. (Pérez, 2012)

➤ **Arquitectura Cliente-Servidor**

La arquitectura cliente-servidor consiste en un cliente que realiza peticiones a otro programa (el servidor) que le da respuesta. Es un patrón arquitectónico para el desarrollo de sistemas distribuidos en una aplicación entre dos o más componentes especializados, cuya ejecución se distribuyen entre uno o más equipos. Define dos tipos de entidades diferenciadas (asimétricas) que se responsabilizan de acciones diferentes: clientes y servidores. Constituye un modelo de interacción basado en el concepto de servicio

implementado sobre un diálogo petición-respuesta. El cliente inicia el diálogo mediante el envío de peticiones y el servidor presta el servicio y responde las peticiones recibidas. (SCS, 2009)

➤ **Modelo Vista Controlador**

El patrón conocido como Modelo-Vista-Controlador (MVC) separa el modelado del dominio, la presentación y las acciones, basadas en datos ingresados por el usuario en tres clases diferentes, proporcionando múltiples vistas sobre un mismo modelo de datos. El patrón MVC se usa frecuentemente en aplicaciones web, donde se utilizan diferentes interfaces de usuario y el código que provee los datos de la página es dinámico. Dado que la vista se encuentra separada del modelo y no existe dependencia directa del modelo respecto a la vista, la interfaz de usuario puede mostrar variadas vistas de los mismos datos simultáneamente. Los tres elementos esenciales de este patrón son (Reynoso, y otros, 2004):

- Modelo: representa la información con la que trabaja la aplicación, es decir, su lógica de negocio.
- Vista: transforma el modelo en una página web que permite al usuario interactuar con ella, manejando la visualización de la información.
- Controlador: se encarga de procesar las interacciones del usuario y realiza los cambios apropiados en el modelo o en la vista.

➤ **Lenguaje de Marcado de Hipertexto**

Hypertext Markup Language (HTML, en español Lenguaje de Marcado de Hipertexto), es un lenguaje de composición de documentos y especificación de ligas de hipertexto, que define la sintaxis y coloca instrucciones especiales que no muestra el navegador, aunque sí le indica cómo desplegar el contenido del documento, incluyendo textos, imágenes y otros medios soportados. HTML también indica cómo hacer un documento interactivo a través de vínculos especiales de hipertexto, las cuales conectan diferentes documentos, ya sea en su computadora o en otras, así como otros recursos de Internet. (Musciano, 2010)

➤ **Lenguaje de Marcado Extensible**

Lenguaje de Marcado Extensible (XML), es una tecnología muy sencilla que tiene a su alrededor otras tecnologías que la complementan y lo hacen mucho más grande. XML permite compartir los datos con los que se trabaja a todos los niveles, por todas las aplicaciones y soportes. El uso del XML permite compartir información de forma segura y realizar el intercambio de documentos entre las aplicaciones. (XML, 2010)

➤ **Estilo de Hoja en Cascada**

CSS es un lenguaje de hojas de estilos, creado para controlar el aspecto o presentación de los documentos electrónicos definidos con HTML y XHTML. Constituye la mejor forma de separar los contenidos y su presentación y es imprescindible para crear páginas web complejas. Al crear una página web, se utiliza en primer lugar el lenguaje HTML/ XHTML para marcar los contenidos, designando una función a cada elemento dentro de la página: párrafo, titular, texto destacado, tabla, lista de elementos. Una vez creados los contenidos, se utiliza el lenguaje CSS para definir el aspecto de cada elemento: color, tamaño y tipo de letra del texto, separación horizontal y vertical entre elementos, posición de cada elemento dentro de la página. (O'Brien, 2010)

➤ **JavaScript 1.1**

Es un lenguaje de programación del lado del cliente, que se utiliza principalmente para crear páginas web dinámicas. Una página web dinámica es aquella que incorpora efectos como: texto que aparece y desaparece, animaciones, acciones que se activan al pulsar botones y ventanas con mensajes de aviso al usuario. Técnicamente, JavaScript (JS) es un lenguaje de programación interpretado, por lo que no es necesario compilar los programas para ejecutarlos. Los programas escritos con JS se pueden probar directamente en cualquier navegador sin necesidad de procesos intermedios. JS puede funcionar como un procedimiento y un lenguaje orientado a objetos. Los objetos se crean mediante la programación en este lenguaje, fijando los métodos y propiedades. Una vez que un objeto se ha construido puede ser utilizado como un modelo (o prototipo) para la creación de objetos similares. (Pérez, 2010)

➤ **JavaScript Asíncrona + XML**

El término AJAX es un acrónimo de *Asynchronous JavaScript + XML*, que se puede traducir como "JavaScript asíncrono + XML". Es una técnica de desarrollo web para crear sistemas interactivos mediante la combinación de tres tecnologías ya existentes. AJAX no es una tecnología en sí mismo. (ajax, 2010)

Las tecnologías (ajax, 2010) que forman AJAX son:

- XHTML y CSS, para crear una presentación basada en estándares
- DOM, para la interacción y manipulación dinámica de la presentación
- XML, XSLT y JSON, para el intercambio y la manipulación de información
- XML Http Request, para el intercambio asíncrono de información

- JavaScript, para unir todas las demás tecnologías

AJAX permite mejorar completamente la interacción del usuario con la aplicación, evitando las recargas constantes de la página, ya que el intercambio de información con el servidor se produce en un segundo plano. Las aplicaciones construidas con AJAX eliminan la recarga constante de páginas mediante la creación de un elemento intermedio entre el usuario y el servidor. La nueva capa intermedia de AJAX mejora la respuesta de la aplicación, ya que el usuario nunca se encuentra con una ventana del navegador vacía esperando la respuesta del servidor. (ajax, 2010)

➤ **ExtJS 3.1**

ExtJS es un *framework* JavaScript que permite construir aplicaciones complejas en Internet. Este conjunto de librerías incluye (Rosas, 2008):

Componentes Interfaz de Usuario (UI) del alto performance y personalizables.

- Modelo de componentes extensibles
- Licencias Open Source y comerciales

ExtJS se ajusta dentro de este esquema como un motor, que permite crear Ricas Interfaces de Aplicaciones (RIA) mediante JavaScript. Posibilita crear aplicaciones complejas utilizando componentes predefinidos así como un manejador de *layout* similar al que provee Java Swing; gracias a esto provee una experiencia consistente sobre cualquier navegador, evitando el tedioso problema de validar que el código escrito funcione bien en cada uno (Firefox, IE, Safari, entre otros). (Rosas, 2008)

➤ **PHP 5.3**

PHP es un acrónimo recursivo que significa *Hypertext Pre-processor* (PHP). El lenguaje fue diseñado para la creación de páginas web dinámicas. Es usado principalmente en interpretación del lado del servidor (server-side scripting), pero actualmente puede ser utilizado desde una interfaz de línea de comandos o en la creación de otros tipos de programas, incluyendo aplicaciones con interfaz gráfica. (King, 2010)

Entre sus principales características están (King, 2010):

- Es un lenguaje multiplataforma.
- Completamente orientado al desarrollo de aplicaciones web dinámicas con acceso a información almacenada en una Base de Datos.

- El código fuente escrito en PHP, es invisible al navegador y al cliente, por ser es el servidor el que se encarga de ejecutar el código y enviar su resultado HTML al navegador. Esto hace que la programación en PHP sea segura y confiable.
- Capacidad de conexión con la mayoría de los motores de base de datos que se utilizan en la actualidad, destaca su conectividad con MySQL y PostgreSQL.
- Posee una amplia documentación en su página oficial, entre la cual se destaca que todas las funciones del sistema están explicadas y ejemplificadas en un único archivo de ayuda.
- Es libre, por lo que se presenta como una alternativa de fácil acceso para todos.
- Permite aplicar técnicas de Programación Orientada a Objetos (POO).

➤ **Symfony 1.4**

Symfony es un marco de trabajo completo (o *framework* de desarrollo), diseñado para optimizar el desarrollo de las aplicaciones web gracias a sus características. Inicialmente, separa la lógica de negocio, la lógica de servidor y la presentación de la aplicación web. Proporciona varias herramientas y clases encaminadas a reducir el tiempo de desarrollo de una aplicación web compleja. Además, automatiza las tareas más comunes, permitiendo al desarrollador dedicarse por completo a los aspectos específicos de cada aplicación. Symfony está desarrollado completamente con PHP 5. Ha sido probado en numerosos proyectos reales y se utiliza en sitios web de comercio electrónico de primer nivel. Es compatible con la mayoría de gestores de bases de datos, como MySQL, PostgreSQL, Oracle y SQL Server de Microsoft. Se puede ejecutar tanto en plataformas *nix (Unix, Linux, etc.) como en plataformas Windows. (Zaninotto, et al., 2008 p. 7)

Características y ventajas (Zaninotto, et al., 2008 p. 7):

- Fácil de instalar y configurar en la mayoría de plataformas (y con la garantía de que funciona correctamente en los sistemas Windows y *nix estándares).
- Independiente del sistema gestor de bases de datos.
- Sencillo de usar en la mayoría de casos, pero lo suficientemente flexible como para adaptarse a los casos más complejos.
- Basado en la premisa de "convenir en vez de configurar", en la que el desarrollador solo debe configurar aquello que no es convencional.

- Sigue la mayoría de mejores prácticas y patrones de diseño para la web
- Preparado para aplicaciones empresariales y adaptables a las políticas y arquitecturas propias de cada empresa, además de ser lo suficientemente estable como para desarrollar aplicaciones a largo plazo. Código fácil de leer que incluye comentarios y que permite un mantenimiento muy sencillo.
- Fácil de extender, lo que permite su integración con librerías desarrolladas por terceros.

➤ **Netbeans IDE 6.9**

El IDE Netbeans es un reconocido entorno de desarrollo integrado disponible para Windows, Mac, Linux y Solaris. El proyecto Netbeans está formado por un IDE de código abierto y una plataforma de aplicación, que permite a los desarrolladores crear con rapidez: aplicaciones web, empresariales, de escritorio y móviles utilizando la plataforma Java, así como JavaFX, PHP, JavaScript y Ajax, Ruby y Ruby on Rails, Groovy and Grails y C/C++. (Corporation, 2013)

➤ **PostgreSQL 8.3**

PostgreSQL es un sistema de gestión de bases de datos objeto-relacional, distribuido bajo licencia BSD y con su código fuente disponible libremente. Es el sistema de gestión de bases de datos de código abierto, más fuerte del mercado y en sus últimas versiones no admite comparación con otras bases de datos comerciales. PostgreSQL utiliza un modelo cliente-servidor y usa multiprocesos en vez de multihilos para garantizar la estabilidad del sistema. Un fallo en uno de los procesos no afectará el resto y el sistema continuará funcionando. (postgresql, 2010)

Características más generales de PostgreSQL (postgresql, 2010):

- Integridad referencial
- Replicación asíncrona / *Streaming replication - Hot Standby*
- Unicode
- Multi-Version Concurrency Control (MVCC)
- Múltiples métodos de autenticación
- Acceso cifrado vía SSL
- Completa documentación

- Disponible para Linux y UNIX en todas sus variantes (AIX, BSD, HP-UX, SGI IRIX, Mac OS X, Solaris, Tru64) y Windows 32/64bit

➤ **Proceso Unificado de Desarrollo**

El Proceso Unificado de Desarrollo (RUP, por sus siglas en inglés), es una infraestructura flexible de desarrollo de software, que proporciona prácticas recomendadas probadas y una arquitectura configurable (GSInnova, 2011). RUP es una metodología de desarrollo de software que está basado en componentes e interfaces bien definidas, y junto con el Lenguaje Unificado de Modelado (UML), constituye la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos. (RUP, 2011)

➤ **Lenguaje de Modelado Unificado 2.1**

Lenguaje Unificado de Modelado (UML) es un lenguaje para visualizar, especificar, construir y documentar los artefactos de un sistema. Este ofrece un estándar para describir un plano del sistema, incluyendo aspectos conceptuales tales como: procesos de negocio, funcionalidades del sistema, esquemas de base de datos y componentes de software reutilizables. El lenguaje UML tiene una notación gráfica muy expresiva que permite representar en mayor o menor medida todas las fases de un proyecto informático: desde el análisis con los casos de uso, el diseño con los diagramas de clases, objetos, etc., hasta la implementación y configuración con los diagramas de despliegue. (Orallo, 2007 pág. 1)

➤ **Enterprise Architect 7.1**

Enterprise Architect (EA) es una herramienta comprensible de diseño y análisis UML, cubriendo el desarrollo de software desde el paso de los requerimientos a través de las etapas del análisis, modelos de diseño, pruebas y mantenimiento. Es una herramienta multi-usuario, basada en Windows, diseñada para ayudar a construir software robusto y fácil de mantener. Ofrece salida de documentación flexible y de alta calidad. El UML provee beneficios significativos para ayudar a construir modelos de sistemas de software rigurosos y donde es posible mantener la trazabilidad de manera consistente; EA soporta este proceso en un ambiente fácil de usar, rápido y flexible. Entre sus características está (EA, 2010):

- 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 ha creado
- Generar código para el software que está construyendo
- Realizar ingeniería reversa del código existente en varios lenguajes
- Importación/ Exportación XMI 2.1

Capítulo 2: Características de las funcionalidades gestionar trazas y seguridad de los servicios web

En el presente capítulo se hace una descripción de las características del sistema, definiéndose la propuesta para la gestión de trazas y el mecanismo de seguridad de los servicios web XML del Sistema Nomenclador de Información.

2.1 Propuesta de solución

Una vez analizado el SNI, se propone la implementación de funcionalidades, que permitan lograr la gestión de trazas de los usuarios que interactúan con el sistema y crear un mecanismo de control de acceso en los servicios web XML que brinda el mismo. La gestión de trazas permitirá deshacer cambios no deseados en el sistema (recuperar información) y realizar controles de operaciones, por personal autorizado.

La información almacenada respecto a las trazas, se debe dividir en dos grupos: la información que se puede recuperar (referente a las acciones: eliminar nomenclador y usuario, y modificar nomenclador) y la que solo puede ser consultada (acciones: crear, modificar, eliminar, crear reportes, restaurar información y consumir servicios web). Al realizarse una acción (por el usuario) en el sistema, este debe generar una traza con la información de la operación realizada. Esta traza debe contener información respecto a: usuario que realiza la acción, la fecha, el nombre de la operación, el objeto sobre el que se realiza la acción y la dirección IP (de la PC) donde se encuentra el usuario; con el objetivo de ser solamente consultada.

El control de acceso en los servicios web XML, permitirá realizar los procesos de autenticación y autorización a los usuarios que consumen dichos servicios. Para lograr un mecanismo de control de acceso, en el consumo de los servicios web que brinda el SNI, es necesario incluir en las peticiones que se realizan al sistema, información que contenga las credenciales del usuario que realiza la petición (usuario y contraseña). Con esta información, se pueden realizar en el servidor los procesos de autenticación y autorización, esenciales en el control de acceso a la información de los nomencladores.

2.2 Modelo de Dominio

Es una representación visual de clases conceptuales o de objetos reales en un dominio de interés. Es la representación de los conceptos más importantes y significativos en el desarrollo de un sistema. Su principal objetivo es ayudar a comprender los conceptos que utilizan los usuarios, las definiciones con las que trabajan y con las que deberá trabajar la aplicación. (Martínez, 2009)

2.2.1 Conceptos fundamentales

A continuación se proporciona un marco conceptual con las definiciones identificadas en la elaboración del desarrollo del software. Conceptos fundamentales tomados de la versión anterior del sistema, incorporándose los nuevos relacionados con la investigación en curso (los nuevos incorporados se enuncian en letra cursiva). (Vázquez, y otros, 2011)

Nomenclador: representa el contenedor de la información que se desea gestionar, está estructurado jerárquicamente por grupos y hojas.

Grupos: representa la raíz de la estructura jerárquica de un nomenclador, conformado a su vez por grupos y hojas.

Información: representa el nivel final de la estructura jerárquica de un nomenclador. (Hoja)

Usuario: representa una persona o individuo que interactúa con el sistema, encargado de llenar la información de los nomencladores. Usuario del sistema encomendado a la gestión de la información del sistema.

Administrador: representa el usuario del sistema encargado de la configuración y la gestión de la información que posee privilegios administrativos para el uso y configuración del sistema, así como de la construcción de su estructura jerárquica, es decir grupos y hojas.

Campo: representa los distintos atributos o características con los que puede contar un nomenclador.

Código: constituye un indicador que representa únicamente a un nomenclador específico. Este se construye a partir de la unión de los respectivos códigos de los grupos y hojas del nomenclador.

Asociar: operación que permite establecer si un campo será agregado o asignado a un nomenclador.

Agregado: clasificación que se le da a un campo cuando este solo va a pertenecer a un elemento padre.

Asignado: clasificación que se le da a un campo cuando este no va a pertenecer a un elemento padre y será heredado por sus hijos, formando así parte de estos últimos.

Estado: concepto que posibilita conocer el estado actual de la información en el sistema, este puede ser activo o pasivo.

Activo: representa el estado activo de la información cuando está siendo usada en el sistema.

Pasivo: representa el estado pasivo de la información; aunque ha sido eliminada en el sistema, permanece físicamente en la base de datos.

Traza: *representa el registro de las operaciones ejecutadas por el usuario al interactuar con el sistema.*

Servicios web XML: *representan funcionalidades que brinda el sistema a otras aplicaciones, permitiendo que estas se comuniquen en una forma que no dependan de la plataforma ni del lenguaje de programación.*

Seguridad: *representa la implementación de mecanismos de seguridad en el consumo de los servicios web XML, por los usuarios autenticados en el sistema.*

2.2.2 Diagrama del Modelo de Dominio

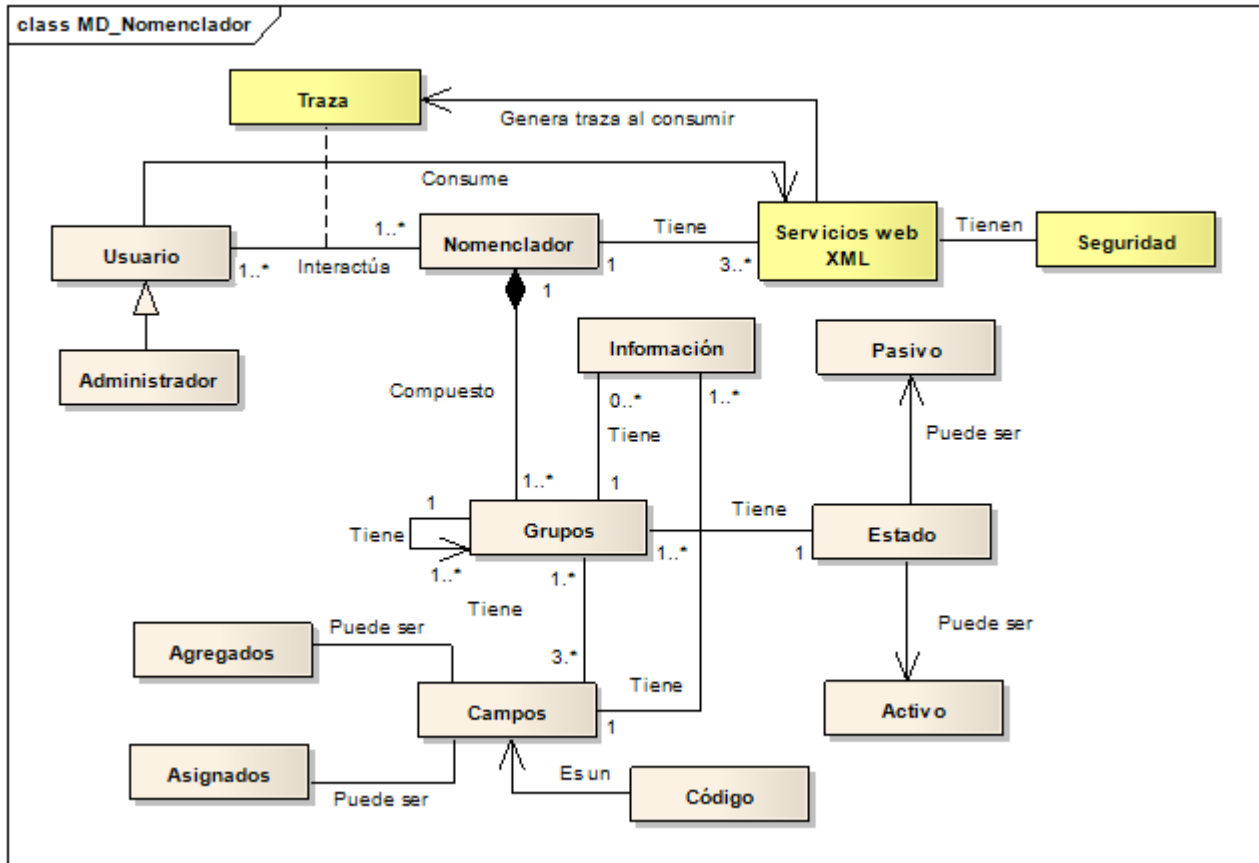


Figura 1. Diagrama de Dominio³

2.3 Requerimientos del Software

En la especificación de los requisitos de software, se definen las condiciones o capacidades necesarias para uno o varios usuarios; con el fin de solucionar un problema o conseguir un objetivo. Los requerimientos del software no son más que la propiedad o restricción determinada con precisión, que un producto software debe satisfacer (Sommerville, 2005). Estos se clasifican en dos grupos: los requisitos funcionales y los requisitos no funcionales.

³ **Leyenda:** Los elementos resaltados en color amarillo, representan las nuevas clases incorporadas al modelo de dominio existente.

2.3.1 Requerimientos Funcionales (RF)

Los requisitos funcionales son capacidades o condiciones que el sistema debe cumplir para dar solución al problema identificado (ISW, 2012). A continuación se muestra un listado (Mojena, y otros, 2011) con los RF tomados de la versión anterior del sistema, incorporando al mismo los nuevos requerimientos.

Tabla1. Requerimientos Funcionales⁴

Requisitos funcionales	
RF1 Insertar Grupo	RF20 Listar Usuario
RF2 Modificar Información Grupo	RF21 Crear Reporte
RF3 Eliminar Grupo	RF22 Insertar Información Grupo
RF4 Buscar Grupo	<u>RF 23 Crear Traza</u>
RF5 Listar Grupo	<u>RF24 Actualizar Árbol para Nomencladores Eliminados</u>
RF6 Insertar Campo	<u>RF25 Actualizar Árbol para Nomencladores Modificados</u>
RF7 Modificar Campo	<u>RF26 Eliminar Trazas de Nomencladores Modificados</u>
RF8 Eliminar Campo	<u>RF27 Eliminar Trazas de Usuarios Eliminados</u>
RF9 Listar Campo	<u>RF28 Eliminar Trazas de Nomencladores Eliminados</u>
RF10 Buscar Campo	<u>RF29 Eliminar Trazas de Operaciones Realizadas</u>
RF11 Asociar Campo	<u>RF30 Recuperar Información sobre Nomencladores Modificados</u>
RF12 Insertar Estructura de Código	<u>RF31 Recuperar Información sobre Nomencladores Eliminados</u>
RF13 Modificar Estructura de Código	<u>RF32 Recuperar Información sobre Usuarios Eliminados</u>
RF14 Buscar Estructura de Código	<u>RF33 Importar Ficheros</u>
RF15 Autenticación	<u>RF34 Exportar Ficheros</u>
RF16 Crear Usuario	<u>RF35 Buscar Trazas de Nomencladores Eliminados</u>

⁴ Los RF resaltados en **negrita** son los modificados (en el nuevo RF crear traza) y los que se encuentran subrayados, son los nuevos incorporados.

RF17 Modificar Usuario	<u>RF36 Buscar Trazas de Operaciones Realizadas</u>
RF18 Eliminar Usuario	<u>RF37 Buscar Trazas de Nomencladores Modificados</u>
RF19 Buscar Usuario	<u>RF38 Buscar Trazas de Usuarios Eliminados</u>

A continuación se describen los nuevos RF:

RF23 Crear Traza: permite crear una traza en el sistema cuando se ejecuten las operaciones (insertar, modificar, eliminar, crear, consumir servicios web XML y recuperar información). Para poder crear las trazas fue necesario modificar todos los RF relacionados con las acciones antes enunciadas.

RF24 Actualizar Árbol para Nomencladores Eliminados: permite actualizar la información visual en forma de árbol que muestran los nomencladores sin necesidad de recargar la página.

RF25 Actualizar Árbol para Nomencladores Modificados: permite actualizar la información visual en forma de árbol

RF26 Eliminar Trazas de Nomencladores Modificados: permite recuperar la información modificada en el Sistema Nomenclador de Información.

RF27 Eliminar Trazas de Usuarios Eliminados: permite eliminar las trazas de los usuarios eliminados en el sistema.

RF28 Eliminar Trazas de Nomencladores Eliminados: permite eliminar las trazas de los nomencladores eliminados.

RF29 Eliminar Trazas de Operaciones Realizadas: permite eliminar las trazas de las operaciones realizadas por el usuario en el sistema.

RF30 Recuperar Información sobre Nomencladores Modificados: permite recuperar la información modificada en el Sistema Nomenclador de Información.

RF31 Recuperar Información sobre Nomencladores Eliminados: permite recuperar la información eliminada en el sistema.

RF32 Recuperar Información sobre Usuarios Eliminados: permite recuperar la información de los usuarios eliminados en el sistema.

RF33 Importar Ficheros: permite importar ficheros en formatos como: excel, .doc., .pdf, xml e incorpora dicha información a la estructura de los nomencladores.

RF34 Exportar Ficheros: permite exportar ficheros en los formatos: excel, .doc, .xml.

RF35 Recuperar Información sobre Nomencladores Eliminados: permite recuperar la información eliminada en el sistema.

RF36 Buscar Trazas de Operaciones Realizadas: permite buscar las trazas de todas las operaciones realizadas en el sistema por un usuario registrado.

RF37 Buscar Trazas de Nomencladores Modificados: permite recuperar la información modificada en el Sistema Nomenclador de Información.

RF38 Buscar Trazas de Usuarios Eliminados: permite recuperar la información de los usuarios eliminados en el sistema.

2.3.2 Requerimientos no Funcionales (RNF)

Los requerimientos no funcionales son restricciones de los servicios o funciones ofrecidos por el sistema. Incluyen restricciones de tiempo, sobre el proceso de desarrollo y estándares. Los requerimientos no funcionales a menudo se aplican al sistema en su totalidad (ISW, 2012). A continuación se muestra un listado con los RNF (Mojena, y otros, 2011) tomados de la versión anterior del sistema.

➤ Usabilidad

RNF1 El sistema solo podrá ser utilizado por los usuarios definidos en el sistema, ya sea ejerciendo el rol administrador o usuario.

RNF2 El usuario definido como administrador debe tener conocimientos básicos de informática.

RNF3 El sistema debe presentar un acceso fácil y rápido, para facilitar el uso del mismo por usuarios (definidos como usuarios) con pocos conocimientos en el campo de la informática.

RNF4 El usuario final deberá recibir una capacitación mínima de siete días.

➤ Seguridad

RNF5⁵ Los servicios web XML, deben poseer mecanismos de seguridad que garanticen la confidencialidad de la información que se consume.

RNF6 Se harán validaciones de la información tanto en el cliente como en el servidor contra ataques de inyección HTML o SQL.

➤ Confidencialidad

RNF7 La información estará protegida contra accesos no autorizados utilizando mecanismos de

⁵ Este es el único requisito no funcional adicionado.

autenticación propios.

➤ **Integridad**

RNF8 La información manejada por el sistema será objeto de protección contra estados inconsistentes.

RNF9 Se implementarán políticas de resguardo de información, así como la realización de copias periódicas de seguridad, que permitan restaurar el sistema en caso de fallo crítico o pérdida total de la información.

➤ **Disponibilidad**

RNF10 El sistema debe poseer la capacidad de seguir brindando servicio ante errores presentados en el ciclo de vida de la aplicación.

➤ **Eficiencia**

RNF11 El sistema deberá ser rápido ante las solicitudes de los usuarios, el tiempo de respuesta deberá ser el menor posible.

➤ **Soporte**

RNF12 Una vez terminado el sistema se realizarán procesos de despliegue, capacitación y mantenimiento de software. El personal que trabaja con el sistema debe contar con el nivel técnico requerido mediante adiestramiento de servicio.

➤ **Restricciones de diseño**

RNF13 Se utilizará PostgreSQL versión 8.4.

RNF14 Se utilizará tecnología Apache versión 2.2 o superior para el servidor web.

RNF15 Para el desarrollo con PHP 5.3 y el *Framework* Symfony 1.4 se utilizará el NetBeans 6.9.

RNF16 Se utilizará un servidor con el sistema operativo instalado Windows XP o superior, o con un sistema operativo GNU/Linux. Ubuntu v12.04 preferentemente.

RNF17 En las computadoras de los clientes se requiere de un navegador web (Internet Explorer versión 8.0 o superior, Mozilla Firefox versión 3.0 o superior).

RNF18 La comunicación de las computadoras clientes con el servidor será a través de conexiones de una red WAN o LAN.

RNF19 Todos los componentes del sistema deben desarrollarse siguiendo el patrón de diseño bajo acoplamiento.

RNF20 Para diseño de las páginas se utilizará CCS, HTML, JavaScript 1.1 y ExtJS 3.0.

➤ **Requisitos para la documentación de usuarios en línea y ayuda del sistema**

RNF21 Se dispondrá de un Manual de Usuario que indicará como interactuar con las funcionalidades del sistema.

RNF22 Se dispondrá de la documentación del sistema realizada con la metodología de software RUP.

➤ **Interfaz**

RNF23 El sistema debe tener una interfaz fácil de usar y amigable para que pueda ser utilizada sin mucho entrenamiento por el usuario.

RNF24 Las interfaces de usuario deben estar definidas y creadas siguiendo las pautas establecidas por el departamento SAS. (Ejemplo: íconos, mensajes del sistema, etiquetado).

➤ **Hardware para el cliente**

RNF25 Ordenador Pentium IV

RNF26 Se requiere tarjeta de red

➤ **Hardware para el servidor**

RNF27 Se requiere que tenga al menos 512 MB de memoria RAM, 1GB de disco duro, 512MHz como mínimo en el cliente.

➤ **Hardware para el servidor**

RNF28 Se requiere que tenga al menos 1GB de memoria RAM y 100GB de disco duro, 2GHz como mínimo.

➤ **Interfaces de Comunicación**

RNF29 La comunicación con el sistema se hará a través de servicios web xml si la aplicación que la consume está desplegada en otro servidor. Si la aplicación se encuentra en el mismo servidor, la comunicación se realizará a través de agentes de implementación.

➤ **Estándares Aplicables**

RNF30 Para la implementación del sistema se deberán seguir los estándares de codificación y diseño definidos por el departamento SAS.

2.4 Actores del Sistema

Representan papeles que las personas (o dispositivos) juegan como impulsores del sistema. Definido más formalmente, un actor es algo que comunica con el sistema o producto y que es externo al sistema en sí mismo. (Pressman, 2001)

A continuación se describen (Vázquez, y otros, 2011) los actores del sistema:

Tabla 2. Descripción de los actores del sistema

Actor	Descripción
Administrador	Usuario que posee privilegios administrativos para el uso y configuración del sistema.
Usuario	Usuario del sistema encargado de llenar la información de los nomencladores.

2.5 Casos de Uso

Los casos de uso modelan el sistema desde el punto de vista del usuario y son creados durante la obtención de requisitos. El caso de uso describe la manera en que los actores interactúan con el sistema. (Pressman, 2001)

Tabla 3. Lista de casos de uso del sistema⁶

Casos de uso del sistema	
CUS1 Insertar Grupo	CUS13 Modificar Estructura de Código
CUS2 Modificar Información Grupo	CUS14 Autenticar
CUS3 Eliminar Grupo	CUS15 Mostrar Detalles
CUS4 Insertar Campo	CUS16 Reportes
CUS5 Modificar Campo	CUS17 Crear Usuario
CUS6 Buscar Campo	CUS18 Modificar Usuario
CUS7 Eliminar Campo	CUS19 Buscar Usuario

⁶ Los casos de uso subrayados fueron adicionados al sistema

CUS8 Asociar Campo	CUS20 Eliminar Usuario
CUS9 Crear Estructura de Código	CUS21 Insertar Información Grupo
<u>CUS10 Crear Trazas</u>	<u>CUS22 Exportar Fichero</u>
<u>CUS 11 Buscar Trazas</u>	<u>CUS23 Importar Fichero</u>
<u>CUS12 Eliminar Trazas</u>	<u>CUS24 Recuperar Información</u>

2.6 Diagrama de Casos de Uso del Sistema

Un Diagrama de Casos de Uso muestra la relación entre los actores y los casos de uso del sistema. Representa la funcionalidad que ofrece el sistema en lo que se refiere a su interacción externa (Pressman, 2001). A continuación el diagrama de Casos de Uso del Sistema, resaltando en color amarillo los requisitos incorporados.

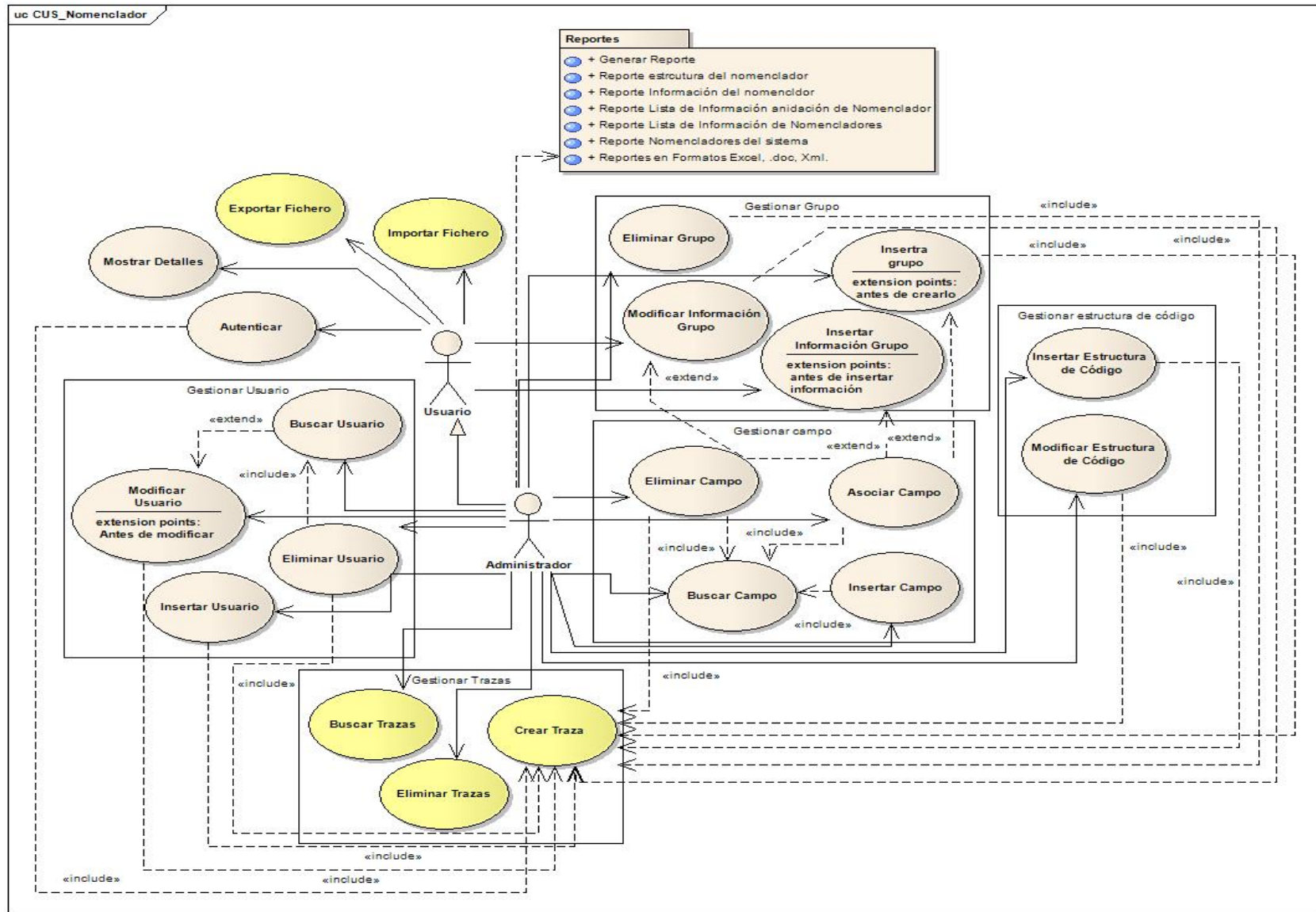


Figura 2. Diagrama de Casos de Uso del Sistema

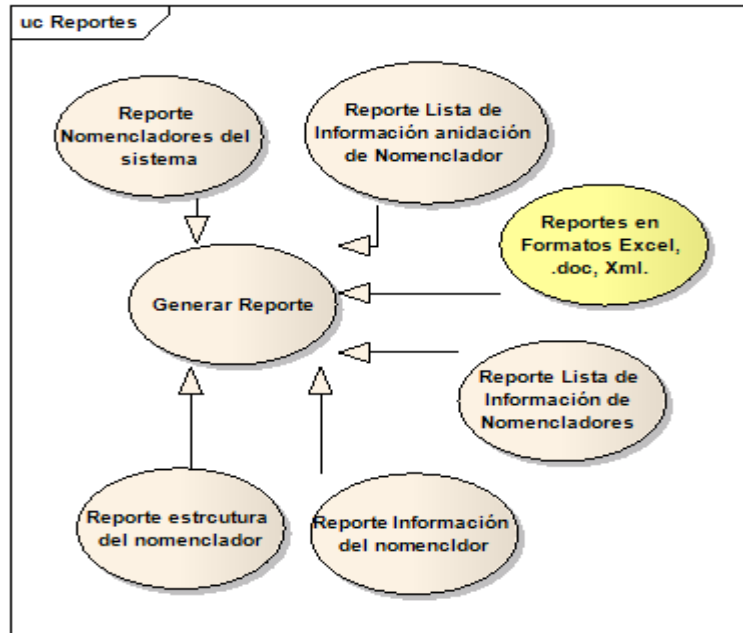


Figura 3. Diagrama del paquete de reportes⁷

2.7 Descripción textual de casos de uso del sistema

A continuación se realiza la descripción de dos casos de usos. Las descripciones restantes serán mostradas en el expediente de proyecto.

Tabla 4. Descripción textual CUS Buscar Traza

Caso de Uso:	Buscar Traza
Actores:	Administrador
Resumen:	El caso de uso inicia cuando el actor accede a la opción Gestión de Trazas, el sistema brinda la posibilidad de seleccionar los tres tipos de trazas existentes, el actor escoge el tipo de traza y el sistema brinda la posibilidad de introducir criterios de búsqueda, el actor introduce los datos que considera como criterios para realizar una búsqueda, buscando y mostrando los campos que cumplen con los criterios de búsqueda, el caso de uso termina.

⁷ El CUS en color amarillo es el adicionado en este paquete

Precondiciones:	El usuario autenticado debe ser Administrador
Referencias	RF36, RF37, RF38
Prioridad	Alta

Tabla 5. Descripción textual CUS Recuperar Información

Caso de Uso:	Recuperar Información
Actores:	Administrador
Resumen:	El caso de uso inicia cuando el actor selecciona una traza y accede a la opción Recuperar, se recupera la información en el sistema, el caso de uso termina.
Precondiciones:	El usuario debe ser Administrador
Referencias	RF25, RF30, RF31
Prioridad	Alta

Capítulo 3: Análisis, diseño e implementación de las funcionalidades gestionar trazas y seguridad de los servicios web

El objetivo del presente capítulo es convertir, en especificaciones del sistema, los requisitos funcionales. Para ello se describe la arquitectura sobre la que está implementado el sistema y se realizan artefactos como: Diagramas de Clases del Análisis, Diseño y Despliegue.

3.1 Descripción de la arquitectura

Para el desarrollo de las funcionalidades que permitan gestionar trazas se utilizó el Framework Symfony, que está basado en un patrón clásico del diseño web, conocido como arquitectura MVC. (Potencier, y otros, 2011)

Atendiendo a las características propias del *Framework* Symfony, la aplicación cuenta en el Controlador, la Vista y el Modelo, con sus características propias:

El Controlador en Symfony normalmente se divide en un controlador frontal, que es único para cada aplicación, y las acciones, que incluyen el código específico del controlador de cada página. Una de las principales ventajas de utilizar un controlador frontal, es que ofrece un punto de entrada único para toda la aplicación. Así, en caso de que sea necesario impedir el acceso a la aplicación, solamente es necesario editar el script correspondiente al controlador frontal. Si la aplicación no dispone de un controlador frontal, se debería modificar cada uno de los controladores. (Potencier, y otros, 2011)

En la Vista se puede aprovechar la separación del código. Symfony la separa: en *layouts* (capa externa), *templates* (plantilla) y lógica de la vista; agrupando en las primeras la parte de esta, que permanece invariable para todas o parte de las páginas de la aplicación, mientras que las segundas, solo se encargan de visualizar las variables definidas en el controlador. En esta capa se utiliza además, la librería ExtJS que incluye un conjunto de componentes que aportan mejoras visuales y facilitan un mejor intercambio con el sistema. (Potencier, y otros, 2011)

El Modelo se puede dividir en la capa de acceso a los datos y en la capa de abstracción de la base de datos. De esta forma, las funciones que acceden a los datos no utilizan sentencias ni consultas que dependen de una base de datos, sino que utilizan funciones escritas en el lenguaje propio del Mapeador

de Objetos Relacional (ORM) Doctrine. Si se cambia de sistema gestor de base de datos, que en el caso particular del SNI es PostgreSQL 8.3, solamente es necesario actualizar la capa de abstracción de la base de datos, ya que el acceso a datos se realiza mediante el uso del motor de persistencia Doctrine. Este motor abstrae al sistema del uso de cualquier sistema gestor de base de datos, mediante el uso de la programación orientada a objeto, haciendo posible tratar las tablas como objetos del sistema. (ver Anexo 3) (Potencier, y otros, 2011)

3.2 Modelo de análisis

El modelo de análisis ofrece una especificación más precisa de los requisitos, de modo que facilita su comprensión y en general su mantenimiento. Un modelo de análisis puede considerarse como una primera aproximación al modelo de diseño y es por tanto una entrada fundamental cuando se da forma al sistema en el diseño y la implementación. (Jacobson, 2000)

3.2.1 Diagramas de Clases del Análisis

Las clases de análisis siempre se ajustan a uno de tres estereotipos básicos: de interfaz, de control o de entidad. Cada estereotipo implica una semántica específica que constituye un método potente y consistente de identificar y describir las clases de análisis. (Jacobson, 2000)

A continuación se muestran algunos de los Diagramas de Clases del Análisis de los casos de uso incorporados al sistema; los restantes (Vázquez, y otros, 2011) pueden encontrarse en el expediente de proyecto.

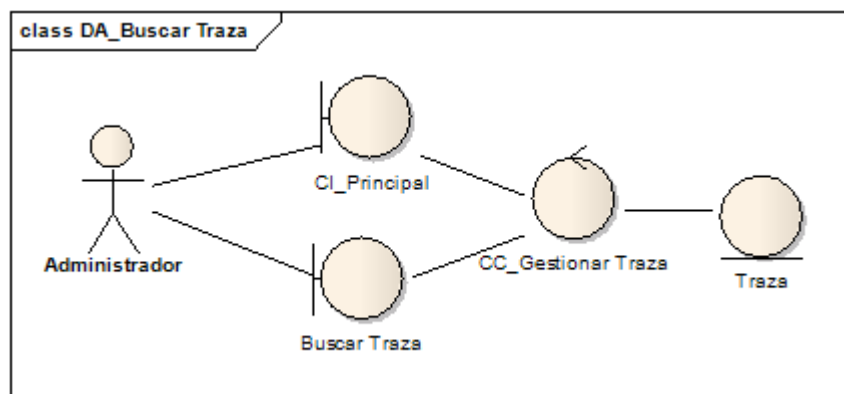


Figura 4. Diagrama de Clases de Análisis CUS_ Buscar Traza

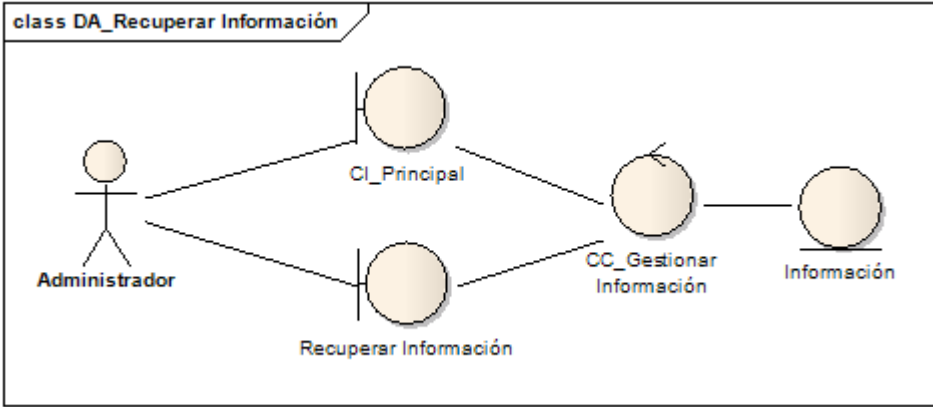


Figura 5. Diagrama de Clases de Análisis CUS_ Recuperar Información

3.2.2 Diagramas de Comunicación

En los diagramas de comunicación, se muestran las interacciones entre objetos, creando enlaces entre ellos y añadiendo mensajes a esos enlaces. El nombre de un mensaje debería denotar el propósito del objeto invocante en la interacción con el objeto invocado (Jacobson, 2000). A continuación se muestran algunos de los diagramas de comunicación de los casos añadidos al sistema; los restantes pueden encontrarse en el expediente de proyecto (Vázquez, y otros, 2011).

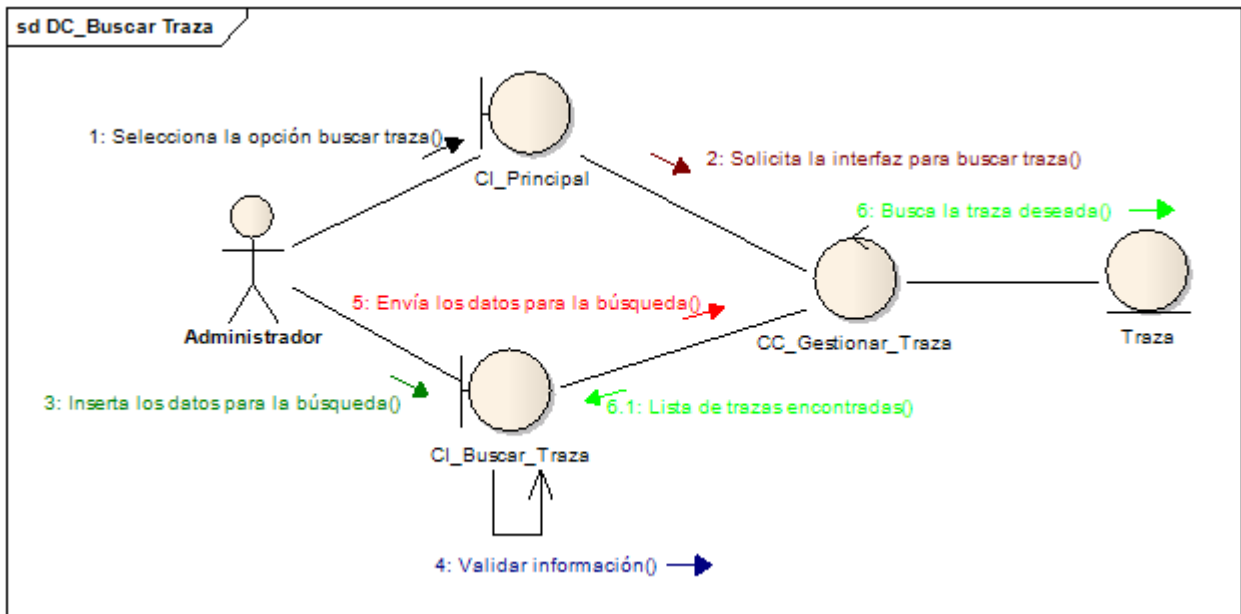


Figura 6. Diagrama de Colaboración CUS_Buscar Traza

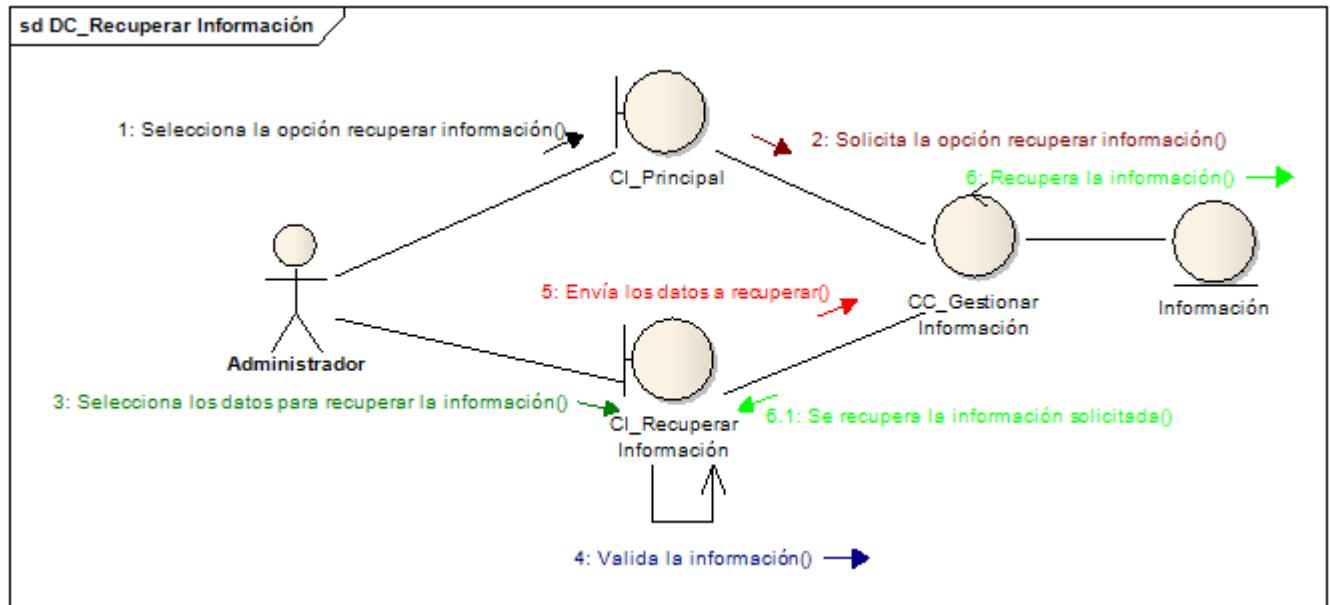


Figura 7. Diagrama de Colaboración CUS_Recuperar Información

3.2.3 Definición de elementos del diseño

El sistema es una aplicación web que se modeló utilizando las clases UML estereotipadas “Server Page”, “Client Page” y “Form”, empleadas para el código servidor, código cliente y formularios respectivamente, permitiendo además, representar ficheros contenedores de sentencias script como por ejemplo PHP y JavaScript.

Tabla 8. Clases del diseño estereotipadas



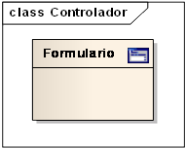
Estereotipos web para las clases del diseño	
ESTEREOTIPO	DESCRIPCIÓN
	Server Page: Representa una página web que tiene scripts que son ejecutados por el servidor. Estos scripts interactúan con recursos del servidor como bases de datos, lógica de negocio, sistemas externos y se encarga de construir (build) o generar el resultado HTML.
	Client Page: Página web con formato HTML y una mezcla de datos, presentación e incluso lógica. Las páginas clientes son representadas por los navegadores clientes, y pueden contener scripts que son interpretados por el navegador.
	Form: Colección de campos de entrada que forman parte de una página cliente. Los formularios envían sus datos al código servidor para ser procesados los pedidos (submit).

Tabla 9. Estereotipos para las relaciones entre las clases

Estereotipo para las relaciones entre las clases	
link	Representa un apuntador desde una “client page” hacia una “client page” o “server page”.
submit	Esta relación siempre ocurre entre una “form” y una “server page”, la “server page” procesa los datos que la “form” le envía.
build	Se establece cuando la “server page” crea una “client page”. Una “server page” puede crear varias “client page”, pero una “client page” sólo puede ser creada por una sola “server page”.
redirect	Esta es también una relación unidireccional que indica que una página web redirecciona el

procesamiento a otra página.

3.2.4 Diagramas de Clases del Diseño

Son clases conectadas a la realización de un caso de uso, mostrando sus clases principales, subsistemas y sus relaciones, de esta forma se puede guardar la pista de los elementos participantes en la realización de un caso de uso (ISW, 2012). A continuación se muestran algunos de los diagramas de Clases del Diseño de los casos de uso incorporados al sistema; los restantes pueden encontrarse en el expediente de proyecto. (Vázquez, y otros, 2011)

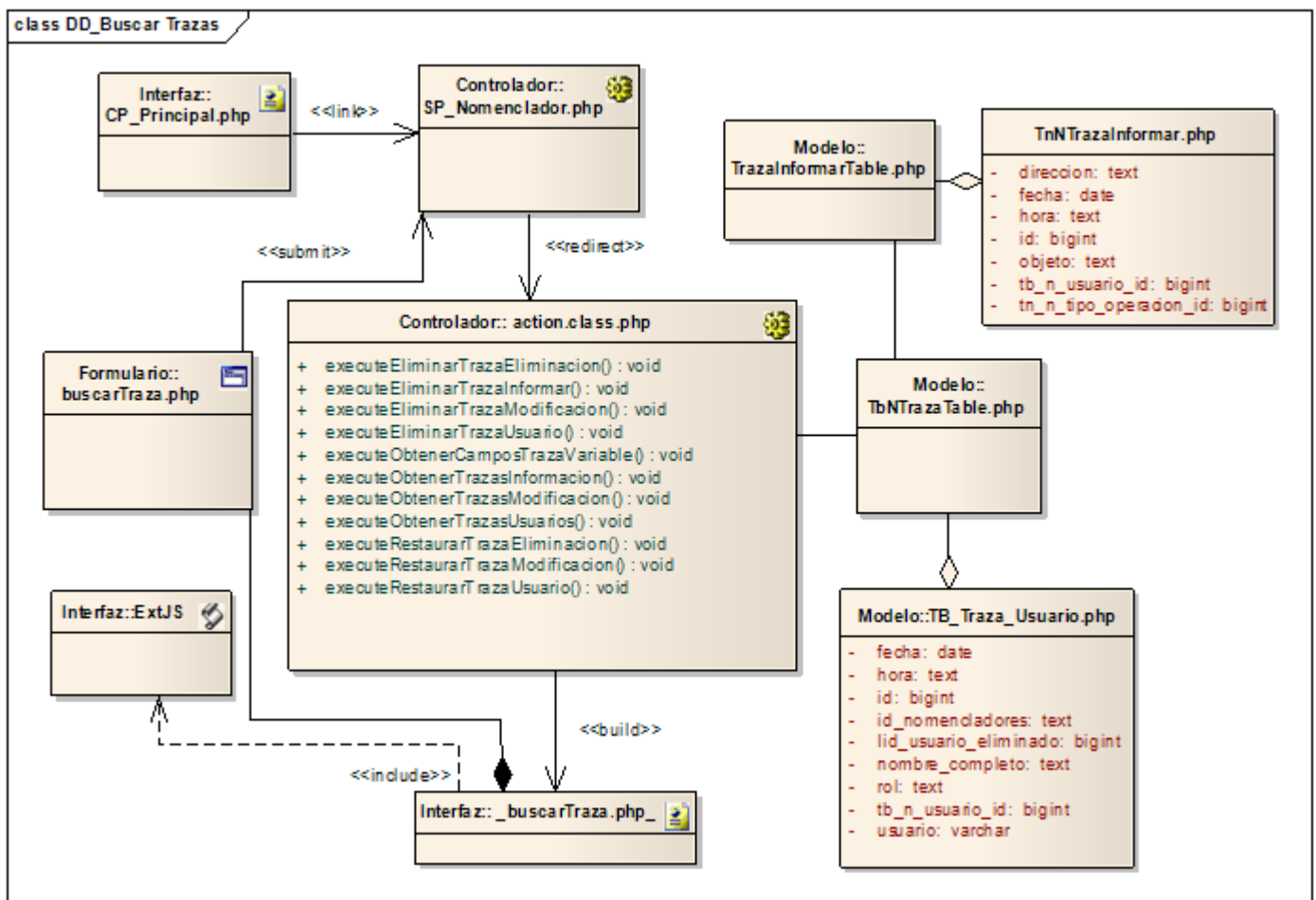


Figura 10. Diagrama de Clases del Diseño: DD_Buscar Trazas

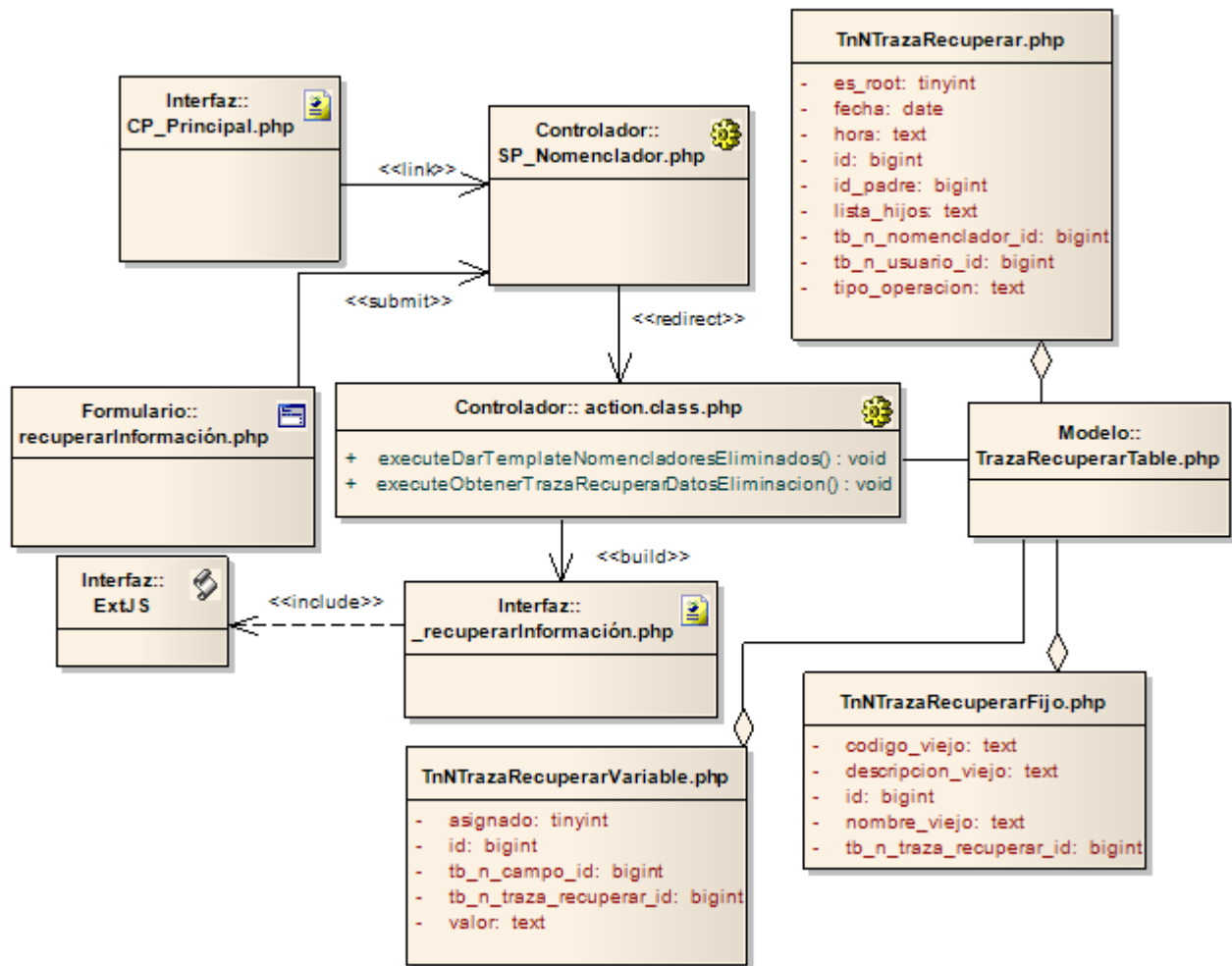


Figura 11. Diagrama de Clases del Diseño: DD_Recuperar Información

3.2.5 Descripción de las clases

A continuación se describen algunas clases que han sido identificadas para la posterior implementación del sistema. De esta manera se poseerá un mejor entendimiento sobre el funcionamiento del mismo.

Tabla 10. Descripción de la clase SP_Nomencladores_dev.php

Nombre	SP_Nomenclador_dev.php
Tipo	Página servidora
Descripción	El controlador frontal es el único punto de entrada a la aplicación. Carga la configuración y

	determina la acción a ejecutarse. Se encarga de procesar las interacciones del usuario y realiza los cambios apropiados en el modelo o en la vista.
--	---

Tabla 11. Descripción de la clase ExtJS.php

Nombre	Extjs.php
Tipo	Librería
Descripción	Es utilizada para la creación de las interfaces de usuario. Permite crear interfaces de usuario dinámicas y organizadas, capaces de brindar todas las funcionalidades necesarias para la gestión de la información.

Tabla 12. Descripción de la clase TbTabla.php

Nombre	TbNombreTabla.php
Tipo	Acceso a datos
Descripción	Esta clase hereda todos los métodos de las clases bases y no les afectan las modificaciones en el esquema, permite empezar a programar desde el primer momento. La estructura de archivos creada permite personalizar y evolucionar el modelo. Es una clase objeto que representan un registro de la base de datos. Permiten acceder a las columnas de un registro y a los registros relacionados.

Tabla 13. Descripción de la clase TbNombreTablaTable.php

Nombre	TbNombreTablaTable.php
Tipo	Acceso a datos
Descripción	Esta clase hereda todos los métodos de las clases bases y no les afectan las modificaciones en el esquema, permite empezar a programar desde el primer momento. La estructura de archivos creada permite personalizar y evolucionar el modelo. Tienen métodos estáticos para trabajar con las tablas de la base de datos. Proporcionan los medios necesarios para obtener los registros de las tablas. Sus métodos devuelven normalmente un objeto o una colección de objetos de la clase objeto relacionado.

Tabla 14. Descripción de la clase action.class.php

Nombre	action.class.php
Tipo	Página servidora
Descripción	Estas clases contienen toda la lógica del módulo al que pertenece. Las acciones son métodos de una clase que utilizan el modelo y definen variables para la vista. Cuando se realiza una petición web en una aplicación Symfony, la URL define una acción y los parámetros de la petición.

3.3 Modelo de datos

3.3.1 Descripción de algunas de las tablas de la base de datos

Tabla 15. Descripción de tb_n_traza_recuperar

Nombre: tb_n_traza_recuperar			
Descripción: Almacena información que permite recopilar los registros de las trazas sobre las que se realizan recuperación de información de nomencladores. Registra información descriptiva como la fecha y hora, además de apuntadores a las tablas que contienen la información mediante las llaves foráneas.			
Atributo	Tipo	Nulo	Descripción
id_traza_recuperar (PK)	Integer	No	Identificador único para la traza_recuperar.
fecha	Date	Si	Fecha en que se realizó la acción.
hora	Character (variable)	Si	Hora en que se realizó la acción.
tipo_operación	Character (variable)	No	Indica el tipo de acción realizada: 'modificación' o 'eliminación'.
lista_hijos	character(variable)	Si	Subgrupos de nomencladores pertenecientes al nomenclador sobre el que se realiza la acción.
es_root	boolean	Si	Define si el nomenclador eliminado aparece como raíz del árbol de nomencladores eliminados
id_padre	Integer	Si	Id del nomenclador padre, usado al momento de restaurar un nomenclador eliminado.
id_nomenclador (FK)	Integer	No	Identificador asociado al nomenclador sobre el que se realiza la acción, está nombrada en el

			esquema Nomencladores. Tb_n_nomenclador.
id_usuario(FK)	Integer	No	Identificador asociado al usuario que realiza la acción, está nombrada en el esquema Nomencladores.tb_n_usuario.

Tabla 16. Descripción de tb_n_traza_usuario

Nombre: tb_n_traza_usuario			
Descripción: Almacena información referente a los usuarios eliminados para ser recuperados en caso de ser necesario.			
Atributo	Tipo	Nulo	Descripción
id_traza_usuario (PK)	integer	No	Identificador único para la traza_usuario .
id_usuario_eliminado	integer	No	Identificador del usuario eliminado en la tabla tb_n_usuario
nombre_completo	character(variable)	No	Nombre con apellidos del usuario eliminado.
usuario	character(variable)	No	Usuario (user) del usuario eliminado.
fecha	date	Si	Fecha en que se eliminó el usuario.
hora	character(variable)	Si	Hora en que se eliminó el usuario.
id_nomencladores_asignados	character(variable)	Si	Grupos asignados al usuario
id_usuario(FK)	integer	No	Identificador asociado al usuario que realiza la acción, está nombrada en el esquema Nomencladores.tb_n_usuario.

Tabla 17. Descripción de tb_n_traza_informar

Nombre: tb_n_traza_informar			
Descripción: Almacena información referente a los acciones que se realizan en el sistema y los usuarios involucrados, manteniendo un registro para posibles auditorías.			
Atributo	Tipo	Nulo	Descripción
id_traza_informar (PK)	integer	No	Identificador único para la traza_informar.
fecha	date	Si	Fecha en que se realizó la acción.
hora	character(variable)	Si	Hora en que se realizó la acción.
objeto	character(variable)	No	Especifica sobre qué elemento se realiza la acción.
dirección	character(variable)	No	Dirección de IP del usuario que realiza la acción.
id_tipo_operación(FK)	integer	No	Identificador para representar el tipo de operación, está nombrada en el esquema Nomencladores.tb_n_tipo_operación.
id_usuario(FK)	integer	No	Identificador asociado al usuario que realiza la acción, está nombrada en el esquema Nomencladores.tb_n_usuario

3.4 Descripción de la funcionalidad gestión de trazas y control de acceso en los servicios web XML

3.4.1 Gestión de trazas

La gestión de las trazas permite la recuperación de los nomencladores eliminados. Para poder implementar esta funcionalidad en el sistema, los desarrolladores se basan en el comportamiento SoftDelete del ORM Doctrine. El funcionamiento del comportamiento SoftDelete es muy simple, consiste en sobrescribir la funcionalidad “delete ()” y adicionar la columna “deleted” a las tablas que lo implementen. Cuando el método “delete ()” es llamado, en lugar de eliminar físicamente la tupla de la base de datos, la columna “deleted” es activada como verdadero. (SENSIOLABS, 2009 pág. 273)

En la implementación de la solución, la columna que se agrega tiene el nombre “deleted_at” y almacena la fecha en que se realiza la eliminación. Este comportamiento es aplicado en la tabla que contiene la información de los nomencladores (tb_n_nomenclador), de esta forma al eliminar un nomenclador en el sistema, este no se elimina físicamente en la base de datos, sino que el campo “deleted_at” que inicialmente se encontraba en nulo (null), se actualiza con la fecha en que se realizó la acción. En la traza de nomencladores eliminados se registra el identificador del nomenclador eliminado, así el proceso de recuperación de la información que registra la traza, consiste, de forma general, en restaurar el valor del campo “deleted_at” a “null”, en la tupla que se corresponde con el identificador del nomenclador.

3.4.2 Control de acceso en los servicios web XML

En la solución propuesta la manera de proporcionar la información necesaria para realizar el control de acceso es a través de la autenticación básica HTTP. Este sistema de autenticación está diseñado para permitir a un navegador o programa, aportar unas credenciales basadas en nombre de usuario y contraseña, que le permitan autenticarse ante un determinado servicio. El sistema es muy sencillo de implementar, sin embargo no está pensado para ser utilizado sobre líneas públicas, debido a que las credenciales que se envían desde el cliente al servidor, se envían en texto plano. (Patxi, 2006)

La información enviada en la petición, se codifica en base64. Esta codificación no se hace por seguridad, sino para aceptar en la información transmitida caracteres que no entran dentro del estándar ASCII, sin romper el protocolo HTTP. La autenticación básica tiene dos ventajas: es parte de HTTP 1.0 (por lo que está soportada por todos los browsers actuales), es sencilla y puede pasar a través de firewalls y proxy. (Oviedo, 2010)

De esta forma es posible obtener las credenciales necesarias en el servidor usando la variable php “\$_SERVER”, mediante esta variable superglobal, se puede conocer el usuario y la contraseña enviada y realizar la autenticación y autorización. La principal desventaja de utilizar la autenticación básica, es que se transmiten las contraseñas sin cifrar. Si un usuario supervisa las comunicaciones de la red, fácilmente puede interceptar y conocer estas contraseñas mediante herramientas disponibles al público. Si se usa este tipo de autenticación se tienen que emplear métodos adicionales para hacerla segura. Esto consiste en cifrar el tráfico por otros medios como, SSL. Esto permite que nadie sin autorización pueda acceder al contenido de las transmisiones (Oviedo, 2010). El protocolo HTTPS protege la información enviada

cifrándola mediante SSL, se habilita el protocolo HTTPS en el servidor y de esta forma el principal problema de seguridad de la autenticación básica queda resuelto.

3.5 Diagrama de Despliegue

Un Diagrama de Despliegue modela la arquitectura en tiempo de ejecución de un sistema. Se utiliza para modelar el hardware utilizado en las implementaciones de sistemas y las relaciones entre sus componentes. Muestra la configuración de los elementos de hardware (nodos) y cómo los elementos y artefactos del software se trazan en esos nodos.

La especificación detallada de cada nodo se encuentra en el expediente de proyecto⁸.

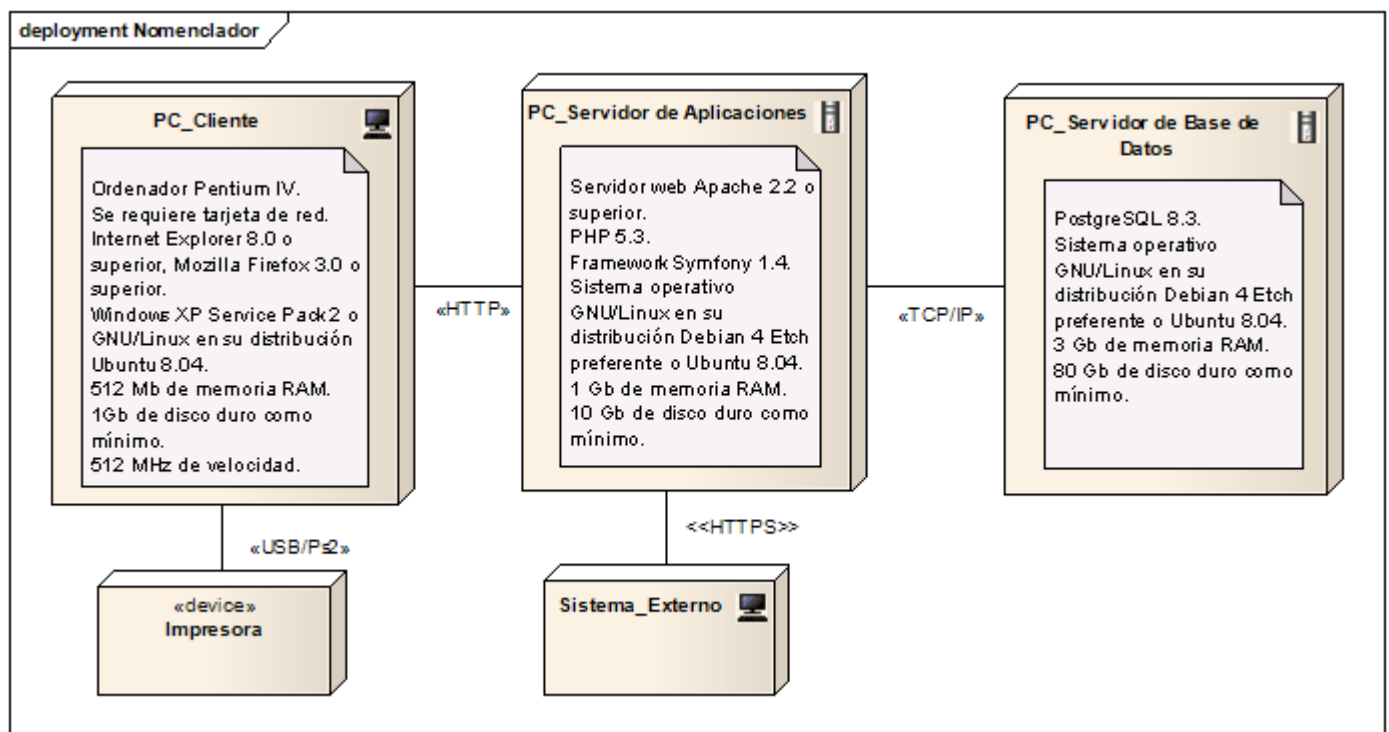


Figura 12. Diagrama de Despliegue

3.6 Modelo de implementación

El modelo de implementación describe cómo los elementos del modelo de diseño, se implementan en términos de componentes, como ficheros de código de fuentes y ejecutables. El modelo de

⁸ No se explican los nodos, pues esta investigación no ejecuta ningún cambio en la primera versión del sistema

implementación describe también cómo se organizan los componentes de acuerdo con los mecanismos de estructuración disponibles en el entorno de implementación y en el lenguaje o lenguajes de programación utilizados, y cómo dependen los componentes unos de otros. (Jacobson, 2004)

3.7 Diagrama de componentes

Los Diagramas de Componentes ilustran las piezas del software y controladores que conformarán un sistema. Tiene un nivel más alto de abstracción que un diagrama de clase. Usualmente un componente se implementa por una o más clases (u objetos) en tiempo de ejecución. Representa la separación de un sistema de software en componentes físicos (por ejemplo: archivos, cabeceras, módulos, paquetes, etc.) y muestran las dependencias entre estos componentes. Estos diagramas contienen: relaciones de dependencia, interfaces, generalización, asociación y paquetes o subsistemas de realización. (Cruz, 2012)

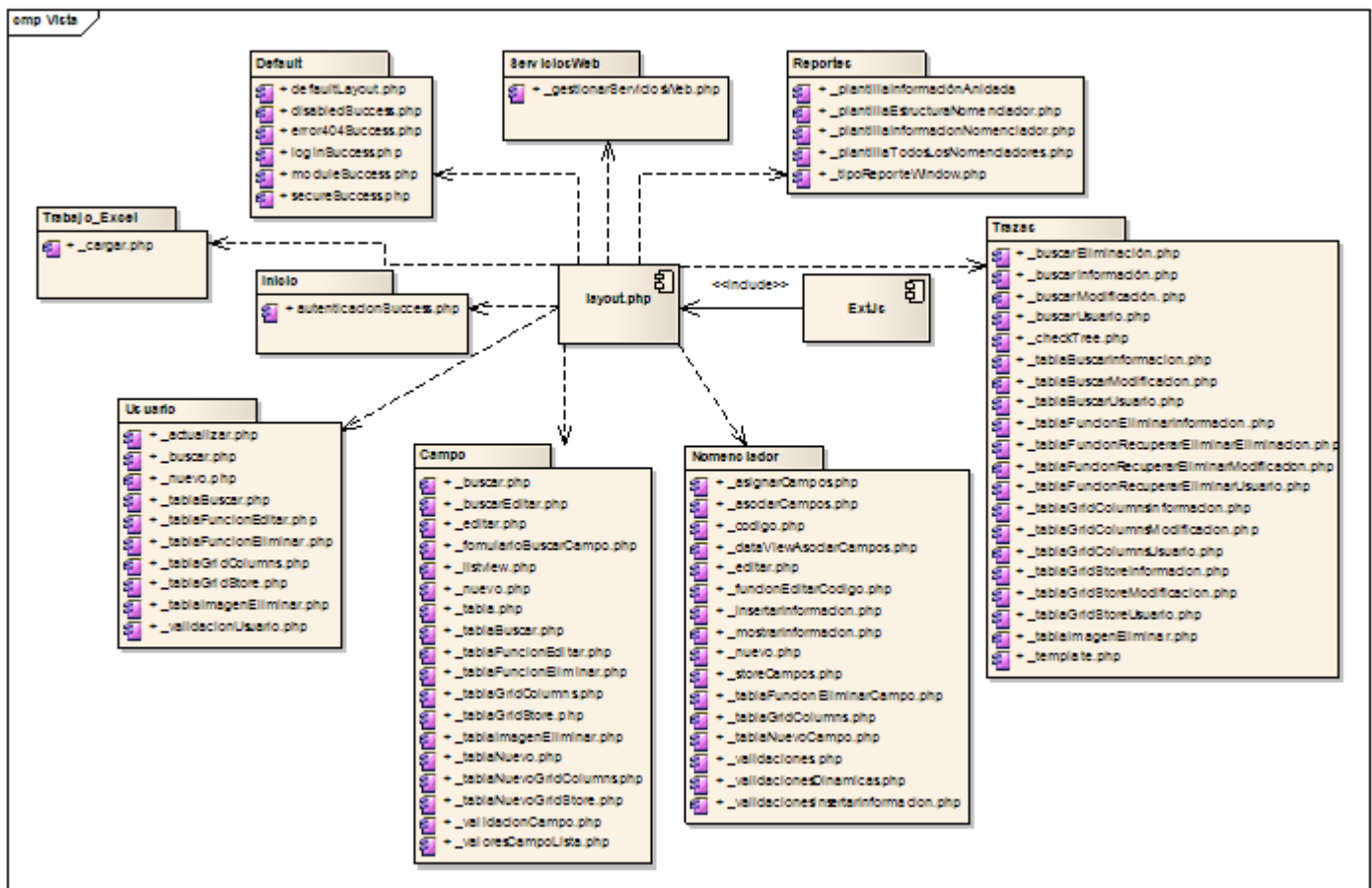


Figura 14. Diagrama de Componentes (Vista)

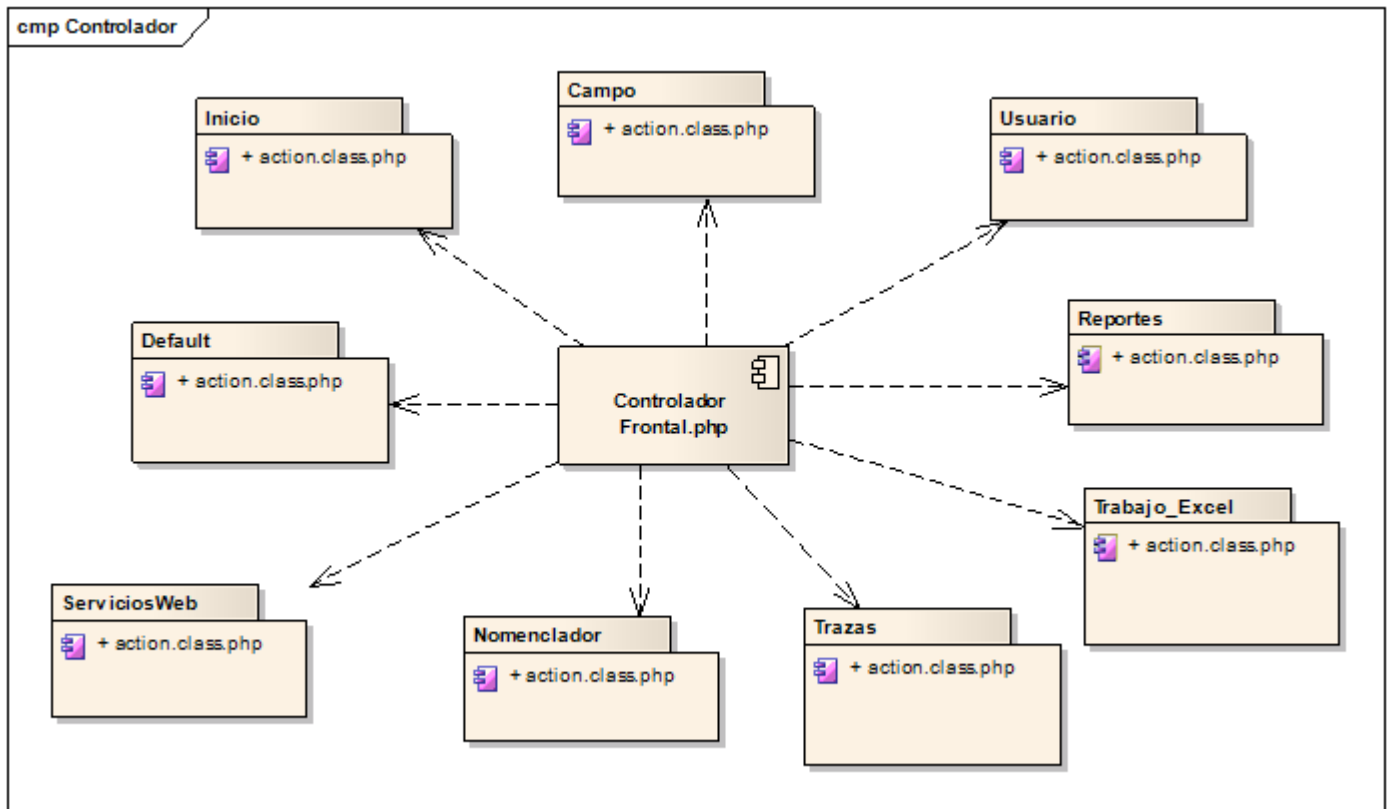


Figura 15. Diagrama de Componentes (Controlador)

A continuación serán expuestas algunas de las clases más importantes adicionadas al sistema, para lograr un mejor entendimiento sobre el funcionamiento del mismo.

Clase controladora perteneciente a la clase Traza

Tabla 18. Descripción de la clase `sp_actions.class.php` de la clase traza

Nombre: <code>sp_actions.class.php</code>
Descripción: Esta clase es la encargada de gestionar toda la información relacionada con las trazas. Entre sus tareas se encuentra mostrar los diferentes tipos de trazas y realizar la recuperación o eliminación de la información referente a cada traza. Esta clase tiene gran importancia ya que a través de la gestión de trazas, se logra mantener un registro de las acciones realizadas en el sistema y en caso necesario la recuperación de esta información.

Al modelo se adicionan dos clases auxiliares que contienen las funcionalidades para la gestión de trazas y a través de las cuales las clases controladoras de cada módulo pueden, en dependencia de la acción realizada por el usuario, guardar un registro en la base de datos. Estos registros (trazas) son los que luego la clase Traza gestiona, valiéndose de las funcionalidades que brindan estas clases. Estas son: TrazaAdicionar.class.php y TrazaRecuperar.class.php, a continuación se describen estas clases y algunas de sus funcionalidades más importantes.

Clase TrazaAdicionar.class.php

Tabla 19. Descripción de la clase TrazaAdicionar.class.php del modelo

Nombre: Clase TrazaAdicionar.class.php
Descripción: Esta clase cuenta con las funcionalidades necesarias para adicionar, según la operación realizada por el usuario, la traza con la información necesaria. Entre sus funciones se encuentran (Adicionar Traza Nomenclador Modificado, Adicionar Traza Nomenclador Eliminado, Adicionar Traza Usuario Eliminado, Adicionar Traza Informar). A continuación se muestran dos de las funcionalidades anteriores así como sus parámetros de entrada y salida.
Nombre de la función: AdicionarTrazaNomencladorModificado ()

Parámetros de entrada	
Descripción	Tipo de dato
\$obj	TbNNomencladorTable
\$ form_nomenclador	TbNNomencladorForm
\$ form_dinámico	FormaDinámica
Parámetros de salida	
Descripción	Tipo de dato
respuesta	bool
Descripción de la funcionalidad	
Luego de realizarse una modificación a un nomenclador la función adiciona una traza con la información que contenía el nomenclador antes de la modificación para el caso de que se desee restaurar la	

información. Estos datos incluyen, además de los valores de los campos fijos para cada nomenclador: código, nombre, descripción; los valores de los campos asociados al nomenclador.

Nombre de la función: AdicionarTrazaInformar ()	
Parámetros de entrada	
Descripción	Tipo de dato
\$tipo_operación	TnNTipoOperaciónTable
\$objeto	TbNNomencladorTable, TbNCampoTable, TbNUsuarioTable
Parámetros de salida	
Descripción	Tipo de dato
respuesta	bool
Descripción de la funcionalidad	
Luego de realizar cualquier acción de las que sea necesario mantener un registro, la función adiciona una traza con la información necesaria. Estos datos incluyen: usuario, dirección ip, objeto sobre el que se realiza la acción, entre otros.	

Clase TrazaRecuperar.class.php

Tabla 20. Descripción de la clase TrazaRecuperar.class.php del modelo

Nombre: Clase TrazaRecuperar.class.php	
Descripción: Esta clase cuenta con las funcionalidades necesarias para recuperar o eliminar la información de las trazas según desee el usuario. Entre sus funciones se encuentran (Restaurar Nomenclador Eliminado, Restaurar Nomenclador Modificado, Restaurar Usuario Eliminado, Eliminar Trazas, entre otras). A continuación se muestran dos de las funcionalidades anteriores así como sus parámetros de entrada y salida.	
Nombre de la función: RestaurarNomencladorModificado ()	
Parámetros de entrada	
Descripción	Tipo de dato
\$id_traza	int
Parámetros de salida	

Descripción	Tipo de dato
respuesta	bool
Descripción de la funcionalidad	
La función restaura en el sistema la información guardada en la traza referente al nomenclador antes de que se realizara la modificación sobre él. En caso de que el nomenclador esté eliminado, la función retorna 'false' indicando que debe restaurar primero el nomenclador y luego proceder a restaurar los valores modificados.	

Nombre: Clase TrazaRecuperar.class.php	
Descripción: Esta clase cuenta con las funcionalidades necesarias para recuperar o eliminar la información de las trazas según desee el usuario. Entre sus funciones se encuentran (Restaurar Nomenclador Eliminado, Restaurar Nomenclador Modificado, Restaurar Usuario Eliminado, Eliminar Trazas, entre otras). A continuación se muestran dos de las funcionalidades anteriores así como sus parámetros de entrada y salida.	
Nombre de la función: RestaurarNomencladorModificado ()	
Parámetros de entrada	
Descripción	Tipo de dato
\$id_traza	int
Parámetros de salida	
Descripción	Tipo de dato
respuesta	bool
Descripción de la funcionalidad	
La función restaura en el sistema la información guardada en la traza referente al nomenclador antes de que se realizara la modificación sobre él. En caso de que el nomenclador esté eliminado, la función retorna 'false', indicando que debe restaurar primero el nomenclador y luego proceder a restaurar los valores modificados.	

Nombre de la función: RestaurarUsuarioEliminado ()	
Parámetros de entrada	
Descripción	Tipo de dato
\$id_traza	int
Parámetros de salida	
Descripción	Tipo de dato
respuesta	bool
Descripción de la funcionalidad	
La función restaura en el sistema la información de la traza referente al usuario eliminado. En caso de que el usuario no sea administrador se le asignan los nomencladores que tenían asignados antes de ser eliminado.	

Para garantizar la seguridad en los servicios web que brinda el sistema, en la clase `sp_actions.class.php`, la clase controladora del módulo Servicios Web, se agregan funcionalidades para garantizar la autenticación y autorización de los usuarios que consuman estos servicios. El usuario que desee consumir cualquiera de los métodos que brinda el sistema, debe enviar los datos de autenticación: usuario, contraseña. Estos datos viajan protegidos y se transmiten sobre el protocolo HTTPS, el servidor entonces se encarga de validar que sean correctos, y en ese caso comprueba que el usuario tenga permiso sobre el nomenclador solicitado. A continuación se describen dos de las funcionalidades más importantes adicionadas a esta clase para garantizar la autenticación y autorización de los usuarios: Autenticación () y Autorización ().

Tabla 21. Descripción de las funcionalidades Autenticación () y Autorización ()

Nombre de la función: Autenticación ()	
Parámetros de entrada	
Descripción	Tipo de dato
Parámetros de salida	

Descripción	Tipo de dato
respuesta	bool
Descripción de la funcionalidad	
La función a partir de los datos de usuario: usuario y contraseña, valida que el usuario exista y la contraseña para ese usuario sea correcta.	
Nombre de la función: Autorización ()	
Parámetros de entrada	
Descripción	Tipo de dato
\$user	TbNUsuarioTable
\$criterio	int, string
\$tipo	bool
Parámetros de salida	
Descripción	Tipo de dato
respuesta	bool
Descripción de la funcionalidad	
La función verifica que el usuario autenticado tenga acceso el nomenclador solicitado. O sea, que el usuario pueda obtener información solamente de los nomencladores que tiene asignados.	

Conclusiones

Con la realización del presente trabajo de diploma se ha cumplido con el objetivo general propuesto, así como con las tareas de la investigación definidas, obteniéndose las siguientes conclusiones:

- El estudio de los principales sistemas que gestionan trazas y seguridad encontrados a nivel internacional y nacional evidenció, que estos integran en su solución, la información proveniente del registro de las acciones realizadas por los usuarios en forma de *logs*; centrando su fuerza en la vinculación traza-seguridad.
- Se asimilaron y analizaron las herramientas y tecnologías propuestas por el departamento de Sistemas de Apoyo a la Salud para el desarrollo de la solución, concluyendo que este conjunto de técnicas responden al entorno de desarrollo del proyecto viabilizando las soluciones y el objeto de trabajo. Por lo que se debe continuar el manejo de estas herramientas y tecnologías para futuras implementaciones; generando todos los artefactos relacionados con los flujos definidos por la metodología de desarrollo de software RUP.
- Se implementaron funcionalidades que gestionan las trazas e incorporan un mecanismo de control de acceso a los servicios web XML que brinda el sistema, posibilitando la realización de auditorías y controles por personal autorizado, así como brindando autenticación y autorización a los sistemas externos que consuman su información.
- Se obtuvo un sistema más seguro que corrige las deficiencias anteriores, y brinda nuevas funcionalidades que aumentan la calidad del producto final.

Recomendaciones

Una vez concluido el trabajo de diploma, el equipo de desarrollo recomienda:

- Incorporar nuevas funcionalidades al sistema:
 1. Permitir la comunicación entre nomencladores creados. Con esta funcionalidad se garantizará la reutilización de la información común para varios nomencladores, al permitir que un nomenclador pueda agregar a su estructura información de otro nomenclador, evitando que esta sea repetida en la base de datos del sistema.
 2. Permitir la búsqueda de nomencladores específicos. Con esta funcionalidad y la presencia de un buscador en el sistema se agilizará la captura de la información haciendo el proceso más eficaz y rápido.
 3. Permitir la comprobación de los datos que pertenezcan a campos de multiselección, evitando de esta forma la entrada de valores incorrectos cuando se cargue información de un excel.
 4. Permitir a los usuarios que hacen uso de los servicios web, introducir desde su sistema información en los nomencladores a los que tengan acceso.
 5. Definir el uso del protocolo HTTPS en todas las conexiones con el sistema, no solo en los servicios web.

Referencias Bibliográficas

1. (ajax, 2010) **ajax. 2010.** librosweb.es. [En línea] 01 de 11 de 2010. [Citado el: 26 de 01 de 2013.] librosweb.es. librosweb.es. [En línea] [Citado el: 1 de noviembre de 2010.] <http://www.librosweb.es/ajax/capitulo1.html>.
2. (Alvarez, 2010) **Alvarez, Miguel Angel. 2010.** desarrolloweb. [En línea] 25 de 11 de 2010. [Citado el: 25 de 01 de 2013.] <http://www.desarrolloweb.com/articulos/449.php>.
3. (Anacleto, 2010) **Anacleto, Valerio Adrián. 2010.** epidataconsulting. [En línea] 15 de 11 de 2010. [Citado el: 22 de 01 de 2013.] http://www.epidataconsulting.com/tikiwiki/tiki-read_article.php?articleId=15.
4. (BussinessDictionary, 2013) **BussinessDictionary. 2013.** BussinessDictionary. [En línea] 2013. [Citado el: 30 de 05 de 2013.] <http://www.businessdictionary.com/>.
5. (Camejo, et al., 2012) **Delgado Ramos, Ariel, y otros. 2012.** rcim. [En línea] 2012. [Citado el: 20 de 01 de 2013.] http://www.rcim.sld.cu/revista_25/articulo_pdf/nomencladores.pdf.
6. (cibernetia, 2012) **cibernetia. 2012.** cibernetia. [En línea] 2012. [Citado el: 01 de 04 de 2013.] http://www.cibernetia.com/manuales/instalacion_servidor_web/1_conceptos_basicos.php.
7. (Cruz, 2012) **Cruz, Daniel Huerta. 2012.** slideshare. [En línea] 19 de 12 de 2012. [Citado el: 15 de 05 de 2013.] <http://www.slideshare.net/dhuertacruz/diagrama-de-componentes-15705604>
8. (Definición, 2008) **Definición.De. 2008.** Definición.DE. [En línea] 2008. [Citado el: 26 de 02 de 2013.] <http://definicion.de/seguridad-informatica/>.
9. (Definición, 2008) **Definición. 2008. Definición.DE.** [En línea] 2008. [Citado el: 14 de 03 de 20013.] <http://definicion.de/autorizacion/>.
10. (Delgado, 2010) **Moreira Delgado, Mercedes. 2010.** gestiopolis. [En línea] 2010. [Citado el: 19 de 01 de 2013.] <http://www.gestiopolis1.com/recursos7/Docs/ger/organizacion-de-la-informacion-para-la-gestion-del-conocimiento.htm>.

11. (Díaz, 2009) **Díaz, Dr. Miguel Eusebio Marín. 2009.** Informática en salud 2009. [En línea] febrero de 2009. [Citado el: 20 de octubre de 2011.] Disponible en: <http://informatica2009.sld.cu/Members/marin/nomencladores-medicos-nacionales-para-la-informatizacion-de-la-atencion-medica-en-el-sistema-nacional-de-salud/>.
12. (Díaz, 2012) **Díaz, Lester Reinier Hernández. 2012.** Un Modelo para la Implementación de la Seguridad de una Aplicación Web con el Uso de la Programación Orientada a Aspectos. Instituto Superior Politécnico José Antonio Echeverría. La Habana : s.n., 2012. Tesis.
13. (Técnicos, 2010) **Diccionario de Términos técnicos de Internet. 2010.** tecnologia.glosario. [En línea] 04 de 11 de 2010. [Citado el: 22 de 01 de 2013.] <http://tecnologia.glosario.net/terminos-tecnicos-internet/uml-1655.html>.
14. (Diccionario, 2011) **Diccionario. 2011.** elmundo.es. [En línea] 2011. [Citado el: 19 de 01 de 2013.] http://diccionarios.elmundo.es/diccionarios/cgi/diccionario/lee_diccionario.html?busca=nomenclador&diccionario=1&submit=Buscar+.
15. (EA, 2010) **Enterprise Architect. 2010.** sparx systems. [En línea] 21 de 11 de 2010. [Citado el: 22 de 01 de 2013.] <http://www.sparxsystems.com.ar/products/ea.html>.
16. (Española, 2011) **Española, Oficina. 2011.** w3c.es. [En línea] 2011. [Citado el: 22 de 01 de 2013.] <http://www.w3c.es/Divulgacion/GuiasBreves/ServiciosWeb>.
17. (Fernández, 2009) **Fernández. 2011.** Comprometidos con la educación mexicana. [En línea] 2011. [Citado el: 20 de 01 de 2013.] <http://www.tareasya.com.mx/index.php/tareas-ya/primaria/quinto-grado/matematicas/1309-Organizaci%C3%B3n-de-la-informaci%C3%B3n.html>.
18. (GSInnova, 2011) **GSInnova. 2011.** rational. [En línea] 04 de 02 de 2011. [Citado el: 22 de 01 de 2013.] <http://www.rational.com.ar/herramientas/rup.html>.
19. (IBM, 2013) **IBM. 2013.** IBM. [En línea] 2013. [Citado el: 21 de 01 de 2013.] <http://www.ibm.com/developerworks/ssa/webservices/newto/websvc.html>.
20. (ISW, 2012) **ISW. 2012.** Ingeniería de Software. [En línea] 22 de 09 de 2012. [Citado el: 03 de 04 de 2013.] <http://ingenieriadesoftware.bligoo.com.mx/requerimientos-funcionales-y-no-funcionales-rf-rnf>.
21. (King, 2010) **Hannibal King. 2010.** programacionweb. [En línea] 21 de 09 de 2010. [Citado el: 21 de 01 de 2013.]

- <http://www.programacionweb.net/articulos/articulo/?num=686>.
22. (Legrá, 2011) **Legrá Pérez, Niurvis, Román Castro, Rodrigo y López Muñoz, Francisco Javier. 2011.** Serie Científica Universidad de las Ciencias Informáticas. [En línea] 31 de 03 de 2011. [Citado el: 21 de 01 de 2013.] <http://publicaciones.uci.cu/index.php/SC/article/view/611>.
23. (Martínez, 2008) **Martínez, Ivette Carolina. 2008.** itescam. [En línea] 2008. [Citado el: 02 de 04 de 2013.] <http://www.itescam.edu.mx/principal/sylabus/fpdb/recursos/r88846.PDF>.
24. (MIC, 2012) **MIC. 2012.** Ministerio de Informática y las Comunicaciones. [En línea] 2012. [Citado el: 22 de noviembre de 2012.] <http://www.mic.gov.cu/sitiomic/servlet/hthememp?1>.
25. (Moral, 2004) **del Moral, José A. 2004.** gananzia. [En línea] 2004. [Citado el: 21 de 01 de 2013.] <http://gananzia.com/s21sec-ofrece-comercialmente-una-aplicacion-de-gestion-de-logs-que-desarrollo-para-bankinter>.
26. (Morales, 2012) **Morales, Annia Arencibia. 2012.** Propuesta de disminución del tiempo de desarrollo en aplicaciones informáticas que gestionen información poco variable en el tiempo. Universidad de las Ciencias Informáticas. La Habana : s.n., 2012. págs. 4-5, Tesis.
27. (Musciano, 2010) **Chuck Musciano, Kennedy Bill. 2010.** bibliodoc. [En línea] 01 de 11 de 2010. [Citado el: 25 de 01 de 2013.] Chuck Musciano, Kennedy Bill. [En línea] [Citado el: 1 de noviembre de 2010.] <http://bibliodoc.uci.cu/pdf/reg01313.pdf>.
28. (Corporation, 2013) **Oracle Corporation. 2013.** NetBeans. [En línea] 2013. [Citado el: 25 de 05 de 2013.] https://netbeans.org/community/releases/69/index_es.html.
29. (Orallo, 2007 pág. 1) **Orallo, Enrique Hernández. 2007.** *El Lenguaje Unificado de Modelado(UML)*. España : Valencia s.n, 2007.
30. (Particulares, 2008) **Particulares, Asociación de Clínicas. 2008.** Superintendencia nacional de aseguramiento en salud. [En línea] marzo de 2008. [Citado el: 20 de 10 de 2011.] Disponible en: http://www.seps.gob.pe/servicios/nomenclador/nomenclador_presentacion.aspx?opcion=12&seccion=178.
31. (Patxi, 2006) **Patxi. 2006.** eslomás.com. [En línea] 17 de 10 de 2006. [Citado el: 5 de 05 de 2013.] <http://www.eslomas.com/2006/10/funcionamiento-autenticacion-http-basic/>.

32. (O'Brien, 2010) **Paul O'Brien. 2010.** librosweb. [En línea] 06 de 11 de 2010. [Citado el: 25 de 01 de 2013.] <http://www.librosweb.es/css/capitulo1.html>.
33. (Pérez, 2010) **JavaScript. 2010.** librosweb. [En línea] 03 de 11 de 2010. [Citado el: 22 de 01 de 2013.] http://www.librosweb.es/javascript/pdf/introduccion_javascript.pdf.
34. (Pérez, 2012) **Pérez, Alonso Javier. 2012.** ajpdsoft. [En línea] 2012. [Citado el: 24 de 01 de 2013.] <http://www.ajpdsoft.com/modules.php?name=Encyclopedia&op=content&tid=820> Proyecto AjpdSoft.
35. (PgAdmin, 2010) **PgAdmin ArPug. 2010.** arpug. [En línea] 07 de 12 de 2010. [Citado el: 21 de 01 de 2013.] <http://www.arpug.com.ar/trac/wiki/PgAdmin>.
36. (Pressman, 2001) **Pressman, Roger. 2001.** Ingeniería de Software: Un Enfoque Práctico. 2001.
37. (RUP, 2011) **Proceso Unificado de Rational. 2011.** buenastareas.com. [En línea] 04 de 02 de 2011. [Citado el: 22 de 01 de 2013.] <http://www.buenastareas.com/ensayos/Proceso-Unificado-De-Rational/1333981.html>.
38. (RedUsers, 2013) **RedUsers. 2013.** RedUsers Comunidad de Tecnología. [En línea] 2013. [Citado el: 10 de junio de 2013.] <http://www.redusers.com/noticias/seguridad-en-redes-autenticacion-con-servidores-aaa/>.
39. (Reynoso, y otros, 2004) **Reynoso, Carlos y Kicillof, Nicolás. 2004.** Estilos y Patrones en la estrategia de arquitectura de Microsoft. [En línea] 10 de 03 de 2004. [Citado el: 02 de 04 de 2013.] <http://carlosreynoso.com.ar/archivos/arquitectura/Estilos.PDF>.
40. (Rosas, 2008) **Desarrollo en Web. 2010.** Desarrollo en Web. [En línea] 12 de 11 de 2010. [Citado el: 26 de 01 de 2013.] <http://blogs.antartec.com/desarrolloweb/2008/10/extjs-lo-bueno-lo-malo-y-lo-feo>.
41. (RUP, 2011) **RUP. 2011.** ecured.cu. [En línea] 04 de 02 de 2011. [Citado el: 22 de 01 de 2013.] <http://www.ecured.cu/index.php/RUP>.
42. (S21Sec) **S21Sec. Comprometidos con la seguridad.** [En línea] [Citado el: 21 de 01 de 2013.] <http://www.s21sec.com/es/productos/lookwise/gestion-de-logs-sim>.
43. (SCS, 2009) **SCS. 2009.** uvigo. [En línea] 27 de 09 de 2009. [Citado el: 02 de 04 de 2013.]

- <http://ccia.ei.uvigo.es/docencia/SCS/1011/transparencias/Tema1.pdf>.
44. (SENSIOLABS, 2009 pág. 273) **SENSIOLABS. 2009.** *Doctrine ORM for PHP*. 2009. pág. 407, Manual.
45. (Softonic, 2013) **Softonic. 2013.** softonic. [En línea] 2013. [Citado el: 30 de 05 de 2013.] <http://sawmill.softonic.com/linux>.
46. (Sommerville, 2004) **Sommerville. 2004.** Requerimientos del Software. [En línea] 2012 de 09 de 2004. [Citado el: 03 de 04 de 2013.] <http://lsi.ugr.es/~ig1/docis/requeintro.pdf>.
47. (Tecnocomputación, 2012) **Tecnocomputación. 2012.** Tecnocomputación. [En línea] 2012. [Citado el: 21 de 01 de 2013.] http://www.tecnocomputacion.com/sol_si_geslog.asp.
48. (Techweek, 2008) **Techweek. 2008.** Techweek. [En línea] 13 de 11 de 2008. [Citado el: 22 de 01 de 2013.] <http://www.techweek.es/soa/informes/1003947005601/seguridad-fundamental-al-implementar.1.html>.
49. (Velázquez, et al., 2012) **Gómez Velázquez, Karel, y otros. 2012.** La Ingeniería y la Arquitectura por un Futuro Sustentable. [En línea] 2012. [Citado el: 21 de 01 de 2013.] <http://ccia.cujae.edu.cu/index.php/siia/siia2008/paper/view/1212>.
50. (Velázquez, y otros, 2009) **Velázquez, Karel Gómez, y otros. 2009.** Sistema de Autenticación, Autorización y Registro para aplicaciones basadas en Servicios WEB XML. [En línea] 2009. [Citado el: 05 de 06 de 2013.] <http://publicaciones.uci.cu/index.php/SC/article/view/113>. ISSN: 1813-5056.
51. (XML, 2010) **XML. 2010.** O'REILLY. *xml.com*. [En línea] 2010. [Citado el: 10 de 02 de 2013.] <http://www.xml.com/pub/a/98/10/guide0.html?page=2#AEN58>.
52. (Zaninotto, et al., 2008) **Potencier Fabien, Francois Zaninotto. 2011.** librosweb. [En línea] 14 de 03 de 2011. [Citado el: 26 de 01 de 2013.] http://www.librosweb.es/symfony_1_2/.

Bibliografía

1. **ajax. 2010.** librosweb.es. [En línea] 01 de 11 de 2010. [Citado el: 22 de 01 de 2013.] librosweb.es. librosweb.es. [En línea] [Citado el: 1 de noviembre de 2010.] <http://www.librosweb.es/ajax/capitulo1.html>.
2. **Alvarez, Miguel Angel. 2010.** desarrolloweb. [En línea] 25 de 11 de 2010. [Citado el: 25 de 01 de 2013.] <http://www.desarrolloweb.com/articulos/449.php>.
3. **Anacleto, Valerio Adrián. 2010.** epidataconsulting. [En línea] 15 de 11 de 2010. [Citado el: 22 de 01 de 2013.] http://www.epidataconsulting.com/tikiwiki/tiki-read_article.php?articleId=15.
4. **buenastareas. 2011.** buenastareas. [En línea] 2011. [Citado el: 22 de 01 de 2013.] <http://www.buenastareas.com/ensayos/Definicion-De-Logs/1934762.html> Auditabilidad. Los sistemas basados en servicios Web deben].
5. **buenastareas. 2010.** [En línea] 05 de 2010. [Citado el: 22 de 01 de 2013.] <http://www.buenastareas.com/ensayos/Definici%C3%B3n-y-Tipos-De-Aplicaciones-Web/317130.html> May 2010.
6. **BussinessDictionary. 2013.** BussinessDictionary. [En línea] 2013. [Citado el: 30 de 05 de 2013.] <http://www.businessdictionary.com/>.
7. **Camejo, René Bauta y Gálvez, Ariel Torres. 2012.** GESTEC. XIV Taller Internacional de Gestión Tecnológica e Innovación en las Organizaciones. [En línea] 2012. [Citado el: 21 de 01 de 2013.] http://www.gestec.disaic.cu/ponencias_Cuba_2010.htm.
8. **Carretero, Valentín y Marín, Omar. 2008.** [En línea] 2008. [Citado el: 07 de 06 de 2013.] http://iessanvicente.com/colaboraciones/berkeley_interbase.pdf.
9. **CESIM. 2012.** repositorio. [En línea] 2012. [Citado el: 06 de 04 de 2013.] <https://repositorio.cesim.prod.uci.cu/svn/sas/sas>.
10. **cibernetia. 2012.** cibernetia. [En línea] 2012. [Citado el: 01 de 04 de 2013.] http://www.cibernetia.com/manuales/instalacion_servidor_web/1_conceptos_basicos.php.
11. **Cruz, Daniel Huerta. 2012.** slideshare. [En línea] 19 de 12 de 2012. [Citado el: 15 de 05 de 2013.] <http://www.slideshare.net/dhuertacruz/diagrama-de-componentes-15705604>.
12. **DASUTeN. 2010.** Sistema para Prestadores de DASUTeN. Concordia : s.n., 2010, págs. 1,2.

13. **Definición. 2008.** Definición.DE.[En línea]2008.[Citado el: 14 de 03 de 2013.] <http://definicion.de/autorizacion/>.
14. **Definición. 2008.** Definición.DE.[En línea]2008.[Citado el: 26 de 02 de 2013.] <http://definicion.de/seguridad-informatica/>.
15. **Definición. 2008.** Definición.DE. [En línea] 2008. [Citado el: 16 de 03 de 2013.] <http://definicion.de/auditoria/>.
16. **Delgado Ramos, Ariel, y otros. 2012.** rcim. [En línea] 2012. [Citado el: 20 de 01 de 2013.] http://www.rcim.sld.cu/revista_25/articulo_pdf/nomencladores.pdf.
17. **Delgado, Lic. Mercedes Moreira. 2010.** GestioPolis.com. La organización de la información para la gestión del conocimiento en las empresas. [En línea] octubre de 2010. [Citado el: 20 de octubre de 2011.] Disponible en: <http://www.gestiopolis1.com/recursos7/Docs/ger/organizacion-de-la-informacion-para-la-gestion-del-conocimiento.htm>.
18. **Delgado, Mercedes Moreira. 2010.** gestiopolis. [En línea] 2010. [Citado el: 19 de 01 de 2013.] <http://www.gestiopolis1.com/recursos7/Docs/ger/organizacion-de-la-informacion-para-la-gestion-del-conocimiento.htm>.
19. **Díaz, Dr. Miguel Eusebio Marín. 2009.** Informática en salud 2009. [En línea] febrero de 2009. [Citado el: 20 de octubre de 2011.] Disponible en: <http://informatica2009.sld.cu/Members/marin/nomencladores-medicos-nacionales-para-la-informatizacion-de-la-atencion-medica-en-el-sistema-nacional-de-salud/>.
20. **Díaz, Karina Centeno, Ríos, Danisbel Rojas y Mulet, Héctor M. Solís. 2008.** Componente de seguridad para aplicaciones del Área temática Sistemas de Apoyo a la Salud. Área temática Sistemas de Apoyo a la Salud, Universidad de las Ciencias Informáticas. Ciudad de la Habana : s.n., 2008. pág. 124, Tesis de pregrado.
21. **Díaz, Lester Reinier Hernández. 2012.** Un Modelo para la Implementación de la Seguridad de una Aplicación Web con el Uso de la Programación Orientada a Aspectos. Instituto Superior Politécnico José Antonio Echeverría. La Habana : s.n., 2012. Tesis. <http://revistas.mes.edu.cu/greenstone/collect/repo/import/repo/201209/71208079810.pdf>.
22. **Diccionario de Términos Técnicos. 2010.** glosario.net. [En línea] 04 de 11 de 2010. [Citado el: 22 de 01 de 2013.] <http://tecnologia.glosario.net/terminos-tecnicos-internet/uml-1655.html>.
23. **Diccionario. 2011.** elmundo.es. [En línea] 2011. [Citado el: 19 de 01 de 2013.] http://diccionarios.elmundo.es/diccionarios/cgi/diccionario/lee_diccionario.html?busca=nomenclador&diccionario=1&submit=Buscar+.

24. **Diccionarios. 2013.** Real Academia Española. [En línea] 2013. [Citado el: 21 de octubre de 2012.] Disponible en: <http://diccionarios.elmundo.es/diccionarios>.
25. **DosIdeas. 2011.** dosideas. [En línea] 16 de 02 de 2011. [Citado el: 21 de 01 de 2013.] <http://www.dosideas.com/wiki/NetBeans>.
26. **EcuRed. 2013.** Conocimiento con todos y para todos. [En línea] 26 de 03 de 2013. [Citado el: 26 de 03 de 2013.] http://www.ecured.cu/index.php/Seguridad_Inform%C3%A1tica#Objetivos.
27. **ecured. 2011.** ecured. [En línea] 03 de 01 de 2011. [Citado el: 23 de 01 de 2013.] http://www.ecured.cu/index.php/Arquitectura_de_software.
28. **ecured. 2010.** ecured. [En línea] 3 de 11 de 2010. [Citado el: 03 de 01 de 2013.] EcuRed. [En línea] 14 de Septiembre de 2010. [Citado el: 3 de Noviembre de 2010.] http://www.ecured.cu/index.php/Arquitectura_Cliente_Servidor.
29. **EcuRed. 2011.** EcuRed. [En línea] 04 de 03 de 2011. [Citado el: 22 de 01 de 2013.] http://www.ecured.cu/index.php/Servidores_Web.
30. **EcuRed. 2010.** EcuRed. [En línea] 20 de 10 de 2010. [Citado el: 24 de 01 de 2013.] http://www.ecured.cu/index.php/Patr%C3%B3n_Modelo_Vista_Controlador.
31. **Enterprise Architect. 2010.** sparx systems. [En línea] 21 de 11 de 2010. [Citado el: 22 de 01 de 2013.] <http://www.sparxsystems.com.ar/products/ea.html>.
32. **Española, Oficina. 2011.** w3c.es. [En línea] 2011. [Citado el: 22 de 01 de 2013.] <http://www.w3c.es/Divulgacion/GuiasBreves/ServiciosWeb>.
33. **Fernández. 2011.** Comprometidos con la educación mexicana. [En línea] 2011. [Citado el: 20 de 01 de 2013.] <http://www.tareasya.com.mx/index.php/tareas-ya/primaria/quinto-grado/matematicas/1309-Organizaci%C3%B3n-de-la-informaci%C3%B3n.html>.
34. **Fernández, Andrés. 2009.** infouniversidades. *Divulgación y noticias universitarias*. [En línea] Universidad Nacional de Córdoba, 8 de 1 de 2009. [Citado el: 20 de 10 de 2011.] Disponible en: http://infouniversidades.siu.edu.ar/noticia.php?titulo=nomenclador_cartografico_para_ciegos&id=273.
35. **2011.** FlashTicSalut. [En línea] 2011. [Citado el: 20 de 01 de 2013.] <http://www.gencat.cat/salut/ticsalut/flashticsalut/html/es/articulos/doc34875.html>.
36. **GSInnova. 2011.** rational. [En línea] 04 de 02 de 2011. [Citado el: 22 de 01 de 2013.] <http://www.rational.com.ar/herramientas/rup.html>.

37. **IBM. 2013.** IBM. [En línea] 2013. [Citado el: 21 de 01 de 2013.] <http://www.ibm.com/developerworks/ssa/webservices/newto/websvc.html>.
38. **IBM. 2009.** IBM. [En línea] 18 de 02 de 2009. [Citado el: 09 de 05 de 2013.] http://pic.dhe.ibm.com/infocenter/wasinfo/v6r1/index.jsp?topic=%2Fcom.ibm.websphere.base.doc%2Finfo%2Faes%2Fae%2Ftwbs_auwschta.html.
39. **IBM. 2010.** Tivoli.Software. [En línea] 2010. http://publib.boulder.ibm.com/tividd/td/TRM/SC23-4822-00/es_ES/HTML/user277.htm.
40. **ISW. 2012.** Ingeniería de Software. [En línea] 22 de 09 de 2012. [Citado el: 03 de 04 de 2013.] <http://ingenieriadesoftware.bligoo.com.mx/requerimientos-funcionales-y-no-funcionales-rf-rnf>.
41. **Jacobson, James. 2004.** El proceso unificado de desarrollo de software. España : Addison Wesley, 2004.
42. **Jacobson, James. 2000.** El proceso unificado de desarrollo de software. Madrid : Pearson Educación, S.A, 2000. págs. 22-25. Vol. 2.
43. **King, Hannibal. 2010.** programacionweb. [En línea] 21 de 09 de 2010. [Citado el: 21 de 01 de 2013.] <http://www.programacionweb.net/articulos/articulo/?num=686..>
44. kioskea. [En línea] [Citado el: 02 de 04 de 2013.] <http://es.kioskea.net/contents/cs/cs3tier.php3>.
45. **Legrá, Niurvis Pérez. 2011.** Serie Científica Universidad de las Ciencias Informáticas. [En línea] 31 de 03 de 2011. [Citado el: 21 de 01 de 2013.] <http://publicaciones.uci.cu/index.php/SC/article/view/611>.
46. **López, Juan Manuel Redondo y Soler, Francisco Ortín. 2010.** OpenCourseWare. [En línea] 01 de 08 de 2010. [Citado el: 09 de 05 de 2013.] <http://ocw.uniovi.es/mod/resource/view.php?id=817>.
47. **López, María del Carmen Paderni, y otros. 2009.** Informática en salud. [En línea] 2009. [Citado el: 20 de noviembre de 2012.] <http://informatica2009.sld.cu/Members/denis/nomencladores-nacionales-de-recursos-y-servicios-para-la-informatizacion-de-la-atencion-medica-en-el-sistema-nacional-de-salud/>.
48. **Marta Delgado Dapena. 2010.** revistaelectronica. [En línea] 2010. [Citado el: 02 de 04 de 2013.] <http://www.inf.udec.cl/~revista/ediciones/edicion8/Rbc.pdf>.
49. **Martínez, Ivette Carolina. 2008.** itescam. [En línea] 2008. [Citado el: 02 de 04 de 2013.] <http://www.itescam.edu.mx/principal/sylabus/fpdb/recursos/r88846.PDF>.

50. **Martínez, Prof. Ivette Carolina. 2009.** Clase 6: Modelo conceptual/Modelo de dominio. [En línea] 2009. [Citado el: 10 de 03 de 2013.] <http://www.itescam.edu.mx/principal/sylabus/fpdb/recursos/r88846.PDF>.
51. **masadelante. 2012.** masadelante.com. [En línea] 2012. [Citado el: 04 de 1 de 2013.] <http://www.masadelante.com/faqs/servidor>.
52. **MIC. 2012.** Ministerio de la Informática y las Comunicaciones. [En línea] 2012. [Citado el: 19 de noviembre de 2012.] <http://www.mic.gov.cu/sitiomic/servlet/hthememp?1>.
53. **Microsoft Soporte. 2007.** Microsoft Soporte. [En línea] 25 de 10 de 2007. [Citado el: 11 de 05 de 2013.] <http://support.microsoft.com/kb/282805/es>.
54. **Mojena, José y Vázquez, Renán. 2011.** *Especificación de Requisitos de Software*. La Habana : s.n., 2011.
55. **MongoDB. 2010.** mongoDB. [En línea] 2010. [Citado el: 07 de 06 de 2013.] <http://www.mongodb.org>.
56. **Monodevelop. 2010.** MonoDevelop. [En línea] 6 de 12 de 2010. [Citado el: 20 de 2 de 2013.] <http://monodevelop.com/>.
57. **Moral, José A. del. 2004.** gananzia. [En línea] 2004. [Citado el: 21 de 01 de 2013.] <http://gananzia.com/s21sec-ofrece-comercialmente-una-aplicacion-de-gestion-de-logs-que-desarrollo-para-bankinter>.
58. **Morales, Annia Arencibia. 2012.** *Propuesta de disminución del tiempo de desarrollo en aplicaciones informáticas que gestionen información poco variable en el tiempo*. Sistemas de Apoyo a la Salud, Universidad de las Ciencias Informáticas. La Habana : s.n., 2012. págs. 4-5, Tesis de maestría.
59. **Morales, Annia Arencibia. 2011.** Sistema Integral para el Control Farmacológico. [En línea] 2011.
60. **Mozilla Developer Network. 2010.** developer.mozilla. [En línea] 03 de 11 de 2010. [Citado el: 21 de 01 de 2013.] https://developer.mozilla.org/en/About_JavaScript.
61. **Musciano, Kennedy Bill. 2010.** bibliodoc. [En línea] 01 de 11 de 2010. [Citado el: 25 de 01 de 2013.] Chuck Musciano, Kennedy Bill. [En línea] [Citado el: 1 de noviembre de 2010.] <http://bibliodoc.uci.cu/pdf/reg01313.pdf>.
62. **Oracle Corporation. 2013.** NetBeans. [En línea] 2013. [Citado el: 25 de 05 de 2013.] https://netbeans.org/community/releases/69/index_es.html.

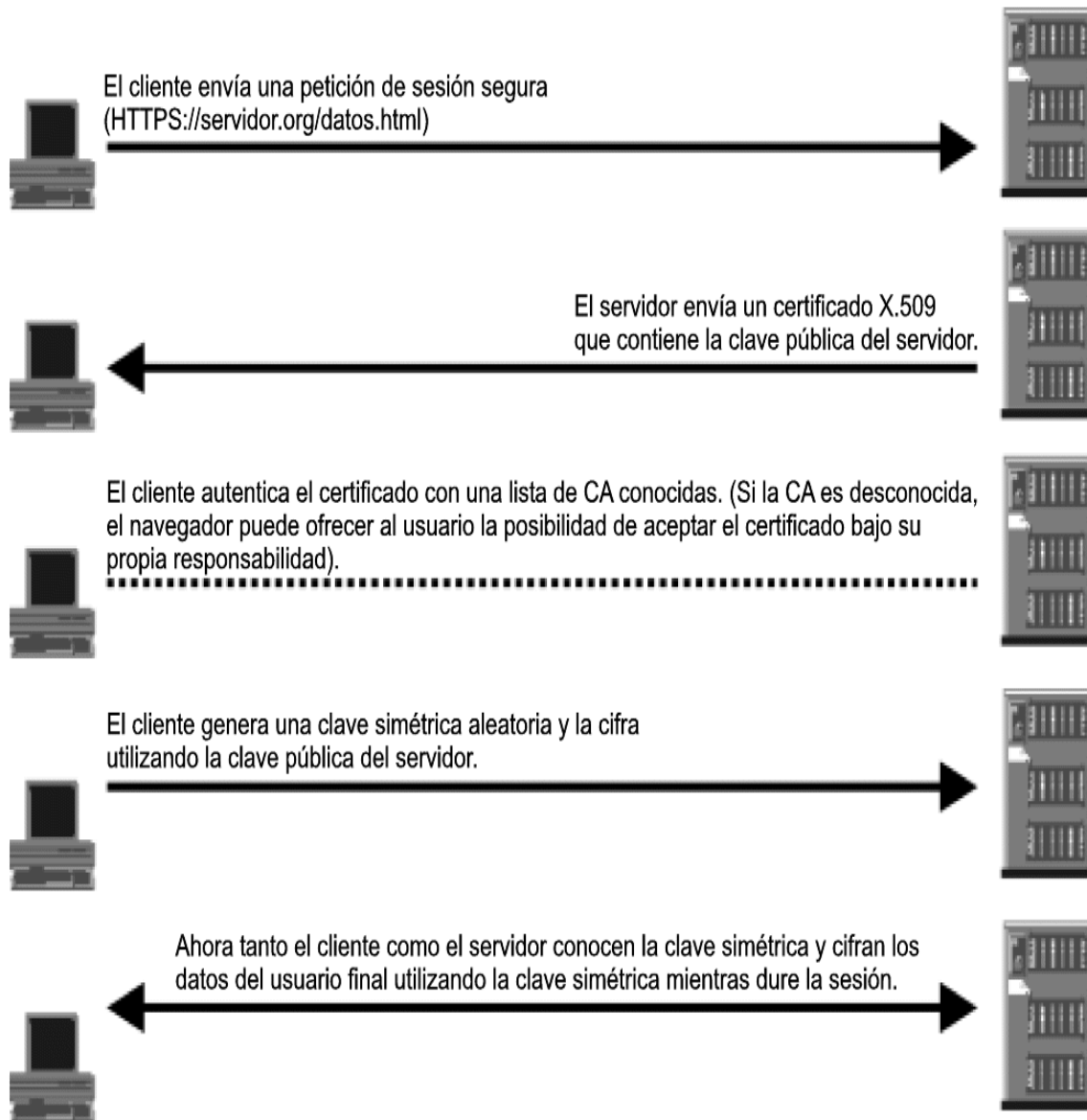
-
63. **Orallo, Enrique Hernández. 2007.** *El Lenguaje Unificado de Modelado(UML)*. España : Valencia s.n, 2007.
64. **Oviedo, Universidad de. 2010.** OpenCourseWare. [En línea] 01 de 08 de 2010. [Citado el: 09 de 05 de 2013.] <http://ocw.uniovi.es/mod/resource/view.php?id=817>.
65. **Particulares, Asociación de Clínicas. 2008.** Superintendencia nacional de aseguramiento en salud. [En línea] marzo de 2008. [Citado el: 20 de 10 de 2011.] Disponible en: http://www.seps.gob.pe/servicios/nomenclador/nomenclador_presentacion.aspx?opcion=12&seccion=178.
66. **Yanette Díaz González, Yenisleidy Fernández Romero. 2012.** Patrón Modelo Vista Controlador. 1, La Habana : s.n., 2012, Revista Digital de las Tecnologías de la Información y las Comunicaciones, Vol. 11, pág. 2. revistatelematica.cujae.edu.cu/index.php/tele/article/download/15/10. ISSN: 1729-3804.
67. **Patxi. 2006.** eslomás.com. [En línea] 17 de 10 de 2006. [Citado el: 5 de 05 de 2013.] <http://www.eslomas.com/2006/10/funcionamiento-autenticacion-http-basic/>.
68. **Paul O`Brien. 2010.** librosweb. [aut. libro] Paul O`Brien. 2010, págs. 5-6.
69. **Pérez, Alonso Javier. 2012.** ajpdsoft. [En línea] 2012. [Citado el: 24 de 01 de 2013.] <http://www.ajpdsoft.com/modules.php?name=Encyclopedia&op=content&tid=820> Proyecto AjpdSoft.
70. **Pérez, Javier Eguíluz. 2010.** librosweb. [En línea] 03 de 11 de 2010. [Citado el: 22 de 01 de 2013.] http://www.librosweb.es/javascript/pdf/introduccion_javascript.pdf.
71. **pergamino virtual. 2012.** Pergaminovirtual. [En línea] 2012. [Citado el: 10 de 05 de 2013.] <http://www.pergaminovirtual.com.ar/definicion/SSL.html>.
72. **PgAdmin. 2010.** arpug. [En línea] 07 de 12 de 2010. [Citado el: 21 de 01 de 2013.] <http://www.arpug.com.ar/trac/wiki/PgAdmin>.
73. **postgresql. 2010.** postgresql. [En línea] 10 de 10 de 2010. [Citado el: 21 de 01 de 2013.] <http://www.postgresql.org/about/>.
74. **Potencier, Fabien y Zaninotto, Francois. 2011.** librosweb. [En línea] 14 de 03 de 2011. [Citado el: 12 de 04 de 2013.] http://www.librosweb.es/symfony_1_2/pdf/.
75. **Pressman, Roger. 2001.** *Ingeniería de Software: Un Enfoque Práctico*. 2001.

76. **Proceso Unificado de Desarrollo. 2011.** buenastareas.com. [En línea] 04 de 02 de 2011. [Citado el: 22 de 01 de 2013.] <http://www.buenastareas.com/ensayos/Proceso-Unificado-De-Rational/1333981.html>.
77. **Ramírez, Dante Odín y Espinosa, Carmina Cecilia. 2011.** Seguridad. [En línea] 03 de 05 de 2011. [Citado el: 12 de 05 de 2013.] <http://revista.seguridad.unam.mx/numero-10/el-cifrado-web-ssltls>.
78. **RedUsers. 2013.** RedUsers Comunidad de Tecnología. [En línea] 2013. [Citado el: 10 de junio de 2013.] <http://www.redusers.com/noticias/seguridad-en-redes-autenticacion-con-servidores-aaa/>.
79. **Reynoso, Carlos y Kicillof, Nicolás. 2004.** Estilos y Patrones en la estrategia de arquitectura de Microsoft. [En línea] 10 de 03 de 2004. [Citado el: 02 de 04 de 2013.] <http://carlosreynoso.com.ar/archivos/arquitectura/Estilos.PDF>.
80. **Rosas, Juan Eladio Sánchez. 2008.** Desarrollo en Web. [En línea] 12 de 11 de 2008. [Citado el: 21 de 01 de 2013.] <http://blogs.antartec.com/desarrolloweb/2008/10/extjs-lo-bueno-lo-malo-y-lo-feo>.
81. **RUP. 2011.** ecured.cu. [En línea] 04 de 02 de 2011. [Citado el: 22 de 01 de 2013.] <http://www.ecured.cu/index.php/RUP>.
82. **S21Sec.** Comprometidos con la seguridad. [En línea] [Citado el: 21 de 01 de 2013.] <http://www.s21sec.com/es/productos/lookwise/gestion-de-logs-sim>.
83. **SCS. 2009.** uvigo. [En línea] 27 de 09 de 2009. [Citado el: 02 de 04 de 2013.] <http://ccia.ei.uvigo.es/docencia/SCS/1011/transparencias/Tema1.pdf>.
84. **SENSIOLABS. 2009.** *Doctrine ORM for PHP*. 2009. pág. 407, Manual.
85. **Softonic. 2013.** softonic. [En línea] 2013. [Citado el: 30 de 05 de 2013.] <http://sawmill.softonic.com/linux>.
86. **Sommerville. 2004.** Requerimientos del Software. [En línea] 2012 de 09 de 2004. [Citado el: 03 de 04 de 2013.] <http://lsi.ugr.es/~ig1/docis/requeintro.pdf>.
87. **Sommerville, Ian. 2005.** *Ingeniería de software*. Madrid : s.n., 2005.
88. **Sommerville.2004.** lsi.ugr.es. [En línea] 2004. [Citado el: 01 de 04 de 2013.] <http://lsi.ugr.es/~ig1/docis/requeintro.pdf> .
89. **Sparxsystems. 2008.** sparxsystems. [En línea] 21 de 11 de 2008. [Citado el: 8 de 02 de 2013.] <http://www.sparxsystems.es/New/products/ea.html>.

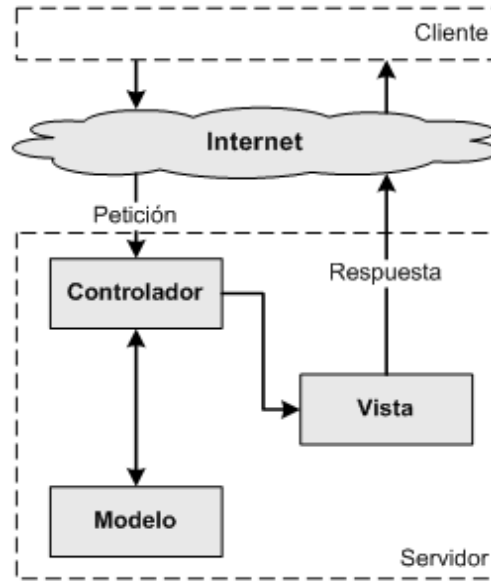
90. **Techweek. 2008.** Techweek. [En línea] 13 de 11 de 2008. [Citado el: 22 de 01 de 2013.] <http://www.techweek.es/soa/informes/1003947005601/seguridad-fundamental-al-implementar.1.html>.
91. **Tecnocomputación. 2012.** Tecnocomputación. [En línea] 2012. [Citado el: 21 de 01 de 2013.] http://www.tecnocomputacion.com/sol_si_geslog.asp.
92. **Vázquez, Renán y Mojena, José. 2011.** *Especificación de requisitos de software*. Sistemas de Apoyo a la Salud, Universidad de las Ciencias Informáticas. La Habana : s.n., 2011. desarrollo.
93. **Vázquez, Renán y Mojena, José. 2011.** *Modelo del Sistema*. La Habana : s.n., 2011.
94. **Velázquez, Karel Gómez, y otros. 2012.** *La Ingeniería y la Arquitectura por un Futuro Sustentable*. [En línea] 2012. [Citado el: 21 de 01 de 2013.] <http://ccia.cujae.edu.cu/index.php/siia/siia2008/paper/view/1212>.
95. **Velázquez, Karel Gómez, y otros. 2009.** Sistema de Autenticación, Autorización y Registro para aplicaciones basadas en Servicios WEB XML. [En línea] 2009. [Citado el: 05 de 06 de 2013.] <http://publicaciones.uci.cu/index.php/SC/article/view/113>. ISSN: 1813-5056.
96. **Velázquez, Karel Gómez, y otros. 2009.** Sistema de Autenticación, Autorización y Registro para aplicaciones basadas en Servicios WEB XML. [En línea] 2009. [Citado el: 06 de 05 de 2013.] <http://publicaciones.uci.cu/index.php/SC/article/view/113>. ISSN: 1813-5056. ISSN: 1813-5056.
97. **XML. 2010.** O'REILLY. *xml.com*. [En línea] 2010. [Citado el: 10 de 02 de 2013.] <http://www.xml.com/pub/a/98/10/guide0.html?page=2#AEN58>.
98. **Zaninotto, Francois y Fabien, Potencier. 2008.** Sympony la guía definitiva. 2008. pág. 425, Guía.

Anexos

Anexo1: Funcionamiento del protocolo HTTPS



Anexo 2: Diagrama de representación del patrón MVC



Anexo 3: El flujo de trabajo de Symfony

