UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

FACULTAD 2 "TELECOMUNICACIONES Y SEGURIDAD INFORMÁTICA"

TRABAJO DE DIPLOMA PARA OPTAR POR EL TÍTULO DE INGENIERO EN CIENCIAS INFORMÁTICAS

Título: *Notificación de Vulnerabilidades de Tecnologías y Sistemas Informáticos.*

Autor:

Enrique Alfonso Carmona García.

Tutor:

Ing. Rogfel Thompson Martínez.

Ciudad de La Habana, Junio de 2010.

"Año 52 de la Revolución"

DECLARACIÓN DE AUTORÍA

Se declara que es el único autor de este trabajo y se autoriza a la Universidad de las Ciencias Informáticas (UCI) y a la Facultad (2) para que hagan el uso que estimen pertinente con este trabajo.

Para que así conste se firma la presente a los ____ días del mes de julio del 2009.

_____

Firma del Autor

Enrique Alfonso Carmona García

_____

Firma del Tutor

Ing. Rogfel Thompson Martínez

FRASE

¿Queréis contar a vuestros amigos? Caed en el infortunio.

Napoleón Bonaparte (1769-1821).

AGRADECIMIENTOS

A mi mamá que siempre ha estado hay para apoyarme cuando más lo he necesitado.

A mi papa que es el mejor padre del mundo por todo lo que ha hecho por mí.

Al bicho morboso por quererme tanto.

A toda mi familia que me ha ayudado a seguir adelante aunque las cosas sean difíciles.

A los hermanos del barrio que han compartido tanto conmigo.

A todos los amigos de estos cinco años que hemos pasado tantas cosas y servers juntos.

A todos aquellos que han creído en mí.

A mi tutor por toda la ayuda y sugerencias que me brindó.

DEDICATORIA

A mi madre que tanto se preocupa.

RESUMEN

En la actualidad no se puede pensar en el desarrollo de un país sin que este cuente con un nivel elevado en el campo de la Informática y las Telecomunicaciones. Las grandes naciones desarrolladas cuentan con una infraestructura informática de gran envergadura, por la cual circula constantemente millones de datos. Todo esto está sustentado por sistemas informáticos los cuales son victimas frecuentes de ataques por parte de personas mal intencionadas. El robo de la información que circula por las redes es algo que siempre se debe evitar ya que esto se traduce en grandes pérdidas para la institución o país afectado.

El objetivo fundamental de la investigación es crear un sistema que pueda informar a todos los interesados, de todas las Vulnerabilidades que puedan tener las diferentes tecnologías y servicios usados por estos y la forma de eliminarlas, lo cual permitirá sin lugar a dudas aumentar el conocimiento sobre este tipo de peligro y a la par aumentar la cultura general.

ABSTRATC

Today it is unthinkable in the development of a country without that holds a fairly level in the field of Informatics and Telecommunications. Great nations have developed a large infrastructure, for which million of data is constantly circulated. All this is supported by computer systems which are frequent victims of attacks by malicious people. The theft of information flowing through networks is something that always should be avoided as this result in great losses to the institution or country concerned.

The objective of this investigation is create a system that can inform all concerned of all vulnerabilities that may have different technologies and services used by these and how to eliminate them, which will undoubtedly increase the knowledge about this type of danger and even increase the general culture.

ÍNDICE

ÍNDICE DE TABLAS

ÍNDICE DE ILUSTRACIONES

INTRODUCCIÓN

La Universidad de las Ciencias Informáticas cuenta con un gran número de equipamiento informático desplegado en diversas áreas, entre estas se puede mencionar los laboratorios productivos y docentes y los nodos, estos últimos son los encargados de proveer y sustentar los servicios telemáticos con los que cuenta dicha institución estudiantil. En la actualidad en este centro educativo se brindan diferentes servicios como es el caso de los servicios de correo y mensajería instantánea, entre otros, los cuales son de vital importancia para el centro. Si alguno de estos servicios dejara de brindarse por el lapso de algunos días podría convertirse en un importante escollo para el desarrollo de la Institución, ya que de estos depende el correcto funcionamiento de muchas de las actividades que se desarrollan en la Universidad.

Un ejemplo de los peligros que pueden significar esta serie de vulnerabilidades quedó evidenciado con la aparición del virus **Kido**, en todas sus variantes, el cual tuvo colapsado muchos de estos servicios en la Universidad, afectando de igual manera a todos los usuarios conectados a la red. Se puede señalar que el daño ocasionado por este código maligno no fue solo en la Universidad de las Ciencias Informáticas, el mundo en general también se vio afectado por este mal, llegando a afectar incluso la soberanía de un país, como se pudo apreciar en el caso de la Fuerza Aérea Francesa, la cual se vio imposibilitada de descargar sus planes de vuelo, porque no podían acceder al servidor donde se encontraban los mismos (1). La Universidad de las Ciencias Informáticas también fue afectada por este virus, llevando a la toma de una serie de medidas para su eliminación.

Para evitar situaciones futuras de esta índole, a la par de aumentar el conocimiento sobre estos problemas para todos los relacionados con un sistema informático, en especial para aquellos usuarios que se encuentran conectados a cualquier red pública o brinden cualquier servicio en la misma, es de vital importancia tener información de las diferentes vulnerabilidades que sus sistemas puedan tener así como su solución.

Por lo antes mencionado surge el siguiente **Problema Científico:** ¿Cómo garantizar la notificación de vulnerabilidades informáticas a los usuarios sobre los peligros que presentan las tecnologías y sistemas informáticos? Definiendo como **Objeto de Estudio** para esta investigación los procesos de obtención de

información de las diferentes vulnerabilidades informáticas. Como **Campo de Acción** los procesos que garantizan la notificación de vulnerabilidades informáticas en la Universidad de las Ciencias Informáticas.

El **Objetivo General** del presente trabajo es el desarrollo de una herramienta que permita la notificación de vulnerabilidades informáticas. En correspondencia a la problemática y el objetivo planteado se han enfocado las **Tareas de Investigación** a:

- Análisis del proceso actual de gestión de vulnerabilidades para lograr una comprensión correcta de la lógica de negocio que implementan estos sistemas de notificación.

- Análisis de las herramientas más usadas en la notificación de vulnerabilidades.

- Estudio de las tecnologías, metodologías y herramientas existentes para el desarrollo de aplicaciones web.

- Selección de la metodología de desarrollo y herramientas que se adapten para el desarrollo de la aplicación web.

- Identificación de los Requisitos de un producto que cumpla con las exigencias de los diferentes usuarios.

- Implementación de la aplicación que permita llevar un control de las vulnerabilidades en sistemas informáticos.

El documento se encuentra estructurado en resumen, introducción, 5 capítulos de contenidos, conclusiones, recomendaciones, bibliografía y glosario de términos.

Capítulo 1, *"Fundamentación Teórica",* incluye un estado del arte de la Notificación de Vulnerabilidades de Tecnologías y Sistemas Informáticos, de las tendencias, tecnologías y software usados en la actualidad.

Capítulo 2, *"Características del Sistema",* se define una propuesta del sistema, especificándose los requisitos de la herramienta, los cuales ayudan a desarrollar los casos de uso basándose en la metodología del RUP.

Capítulo 3, *"Análisis y Diseño del Sistema",* se define como se continúa el desarrollo del sistema mediante el análisis y diseño del mismo.

Capítulo 4, *"Implementación",* se realizan el desarrollo del Diagrama de despliegue, y los diagramas de componentes.

Capítulo 5, *"Factibilidad del Sistema",* se realiza un análisis del costo total del sistema a desarrollar y si es factible para la Institución que desee usarlo.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA.

1.1 Introducción

En el mundo informático, el problema de las vulnerabilidades en el software y su creciente explotación es real y creciente. El tiempo que transcurre entre el descubrimiento de una vulnerabilidad y su uso indebido actualmente ha disminuido grandemente, pasando de ser días a solamente horas. A cada momento salen más programas, con más funcionalidades y, por tanto, cada vez más completos. Pero este crecimiento va unido a un aumento de los posibles fallos en el software; a la existencia de más códigos, mayor la cantidad de errores que podrán contener, fallos que, por otra parte, presentan su origen en un desarrollo demasiado rápido de los productos a causa de la gran demanda del mercado, la necesidad de estar constantemente por delante de la competencia, o por la continua detección de fallos detectados una vez puesto este software en ejecución. De todos estos problemas una parte importante, más por su impacto que por su volumen, son los fallos de software que derivan en una vulnerabilidad de seguridad. Por eso ha surgido la necesidad de tener un control constante de las vulnerabilidades conocidas para poder mantener un grado de seguridad satisfactorio en nuestras aplicaciones.

La cantidad de información asociada a las vulnerabilidades es claramente muy elevada, lo que ha llevado a la creación de bases de datos de vulnerabilidades. Estas bases de datos son fuentes de información que proporcionan información detallada de las vulnerabilidades conocidas, habitualmente cuentan con información de gran interés como: información de sus riesgos, cómo mitigarla de forma temporal, o, en caso de existir, el método de solución.

Unos de los principales problemas de los responsables de seguridad de cualquier tipo de empresa, ya sea grande o pequeña, es gestionar un número de sistemas de información heterogéneos: sistemas operativos distintos con diversas aplicaciones, dispositivos de sistemas de red como conmutadores y enrutadores, software específico para sus tareas. El número de vulnerabilidades que puedan existir en un entorno de estas características es potencialmente elevado, y crece a medida que aumenta el equipamiento y software utilizado.

Aunque algunos fabricantes puedan desarrollar productos más robustos, la tendencia de crecimiento de vulnerabilidades estos años ha aumentado al igual que el número de fabricantes de software y

equipamiento que han surgido al mismo tiempo. Por un lado, a más fabricantes y productos, se crearán distintas vulnerabilidades que afectarán a distintos equipos a medida que se descubran. Por otro lado, un entorno homogéneo tendrá un número menor de vulnerabilidades e iguales para todos los equipos, pero cuando se descubra una nueva vulnerabilidad afectará a todos los sistemas en lugar de a una pequeña parte.

En esta situación, y facilitando la labor de los consumidores de productos de gestión de información, las bases de datos de vulnerabilidades se convierten en fuentes de información que van a permitir determinar las vulnerabilidades que puedan afectar a los productos que se están utilizando y, de estar actualizadas, podrán ser utilizadas para determinar si hay vulnerabilidades nuevas que puedan afectar a los equipos que se están gestionando, cuál es su criticidad y su solución.

La aparición de una nueva vulnerabilidad podrá estar acompañada de la aparición de un ataque directo sobre ese producto, relación que depende en gran medida de la persona que realiza el descubrimiento y sus intenciones al respecto. Estos ataques directos pueden variar desde la aparición de un nuevo virus, o en la incorporación del ataque a herramientas automáticas utilizadas para atacar sistemas conectados a Internet. En cualquier caso, la disponibilidad o no de este tipo de ataques relacionados será fundamental también para determinar las actuaciones a realizar una vez descubierto que los equipos administrados están afectados por una nueva vulnerabilidad.

Cualquier empresa o persona para poder detectar las nuevas vulnerabilidades que puedan presentar las herramientas o servicios que utilizan pueden hacerlo de varias maneras, la forma más frecuente de conocer esta información es:

1. Utilizar las fuentes de información públicas de vulnerabilidades para analizar y determinar cuáles pueden afectarles, entre las entidades que brindan estos servicios podemos citar a National Vulnerability Database, Common Vulnerabilities and Exposures y Open Source Vulnerability Database.

2. Contratar un servicio específico de auditorías a una empresa externa que provea de forma periódica información de aquellas vulnerabilidades que puedan afectar a nuestros sistemas o usar herramientas que se encuentren disponibles con ese propósito. Entre las empresas

que brindan este tipo de soluciones se encuentran Nessus, Nikto, Mageni.

3. Suscribirse a un servicio público genérico que envía avisos de las nuevas vulnerabilidades descubiertas, entre los cuales se pueden señalar Cisco Security Advisory List, Microsoft Security Alerts, HP Security Mailings.

En la Universidad de las Ciencias Informáticas no existe ningún sistema que se encargue de la notificación de vulnerabilidades a los diferentes usuarios, así como tampoco se cuenta con una fuente de información pública de vulnerabilidades, por lo que la única opción con que se cuenta es con solicitar el servicio de auditorías al proyecto LABSI ó utilizar propiamente una de las herramientas especializadas en este trabajo. Esto compromete la seguridad de la información personal y de la propia institución.

El Sistema de Notificación de Vulnerabilidades de Tecnologías y Sistemas Informáticos brindará la posibilidad de conocer la información relacionada con las vulnerabilidades y de recibir información actualizada mediante el envío de correos electrónicos, para que todas las personas interesadas en este servicio puedan obtenerlo de forma rápida y segura en la Universidad de las Ciencias Informáticas. Esto posibilitará aumentar la seguridad, ya que se dispondrá de una herramienta que permitirá conocer de todas las vulnerabilidades que pueden presentar los diferentes servicios y programas detrás de estos, así como las soluciones a las mismas.

1.2 Sistemas de Notificación de Vulnerabilidades

Para poder desarrollar un servicio que posibilite la conexión de múltiples usuarios de manera simultánea e independiente del lugar donde estos se encuentran, sin importar el Sistema Operativo instalado en el computador y sin tener que hacer uso de aplicaciones específicas que no estén incorporadas por defecto en el mismo, se ha decidido desarrollar un portal web que brinde varios servicios de consultas, ya que de esta forma permitirá conocer a los usuarios de los problemas de seguridad con que pueden contar las diferentes herramientas usadas para brindar los diferentes servicios. Esto no es una solución definitiva ya que muchas de las vulnerabilidades que podrían surgir son propias de las malas políticas de seguridad que se apliquen, así como de las configuraciones que se necesite hacer a los servicios, del conocimiento y dedicación de las personas encargadas de estos. Un sistema de Firewall que permita la mayoría de las conexiones a los diferentes puertos, es una gran vulnerabilidad surgida de una mala política de seguridad y de configuraciones inapropiadas de la herramienta, no de una vulnerabilidad propia de ella.

Existen en el mundo actualmente varios sitios de Vulnerabilidades que cuentan con una base de datos que informa de ellas:

1.2.1 CVE (Common Vulnerabilities and Exposures)

Es un diccionario para nombres comunes para publicar la información de las vulnerabilidades de seguridad conocidas. Provee identificadores para las configuraciones de las vulnerabilidades, el cual ya cuenta con 10 años de creada. Los identificadores comunes de **CVE** hacen más fácil el compartir datos a través de base de datos de seguridad y herramientas en redes separadas. Provee una base para la evaluación de la cobertura de las herramientas de seguridad de las organizaciones. Si un reporte de una de las herramientas de seguridad incorpora identificadores **CVE**, entonces se podrá acceder de forma más rápida y efectiva a uno o más bases de datos compatibles con **CVE** para encontrar la información necesaria y solucionar el problema. **CVE** es patrocinado por el National Cyber Security Division del U.S Department of Homeland Security, es decir, por la División de Seguridad Cibernética Nacional del Departamento de Seguridad de la Patria de los Estados Unidos de América (2).

CVE (Common Vulnerabilities and Exposures) es (2):

- Un único nombre para cada vulnerabilidad.
- Una descripción estandarizada para cada vulnerabilidad.
- Un diccionario independiente de las bases de datos.
- La forma en que las bases de datos y las herramientas puedan hablar el mismo lenguaje.
- La manera de operar una mejor cobertura de seguridad.
- Una guía para la evaluación entre las herramientas y las bases de datos.

1.2.2 NVD (National Vulnerability Database)

Es el repositorio del gobierno de los Estados Unidos de América, para la administración de los estándares basados en vulnerabilidades, el cual se encuentra en la versión 2.2, los datos se representan usando **SCAP (Security Content Automation Protocol).** Esto permite automatizar la administración de las

vulnerabilidades y las amenazas de seguridad. **NVD** incluye listas de control de seguridad de Bases de Datos, fallas de seguridad relacionadas con el software, malas configuraciones de aplicaciones, nombres de productos y métricas de impacto (3).

1.2.3 Base de Datos de Bugtrag

Esta base de datos está basada en gran parte en la información publicada en la lista de correo de seguridad del mismo nombre, fue adquirida por la empresa de seguridad Symantec. Sobre esta Base de Datos Symantec ha desarrollado un servicio comercial (4).

1.2.4 La Base de Datos de Xforce

Desarrollada por el fabricante de productos de seguridad Internet Security Sistems (ISS), perteneciente a IBM. Sirve de base tanto a los productos de seguridad de la compañía (herramientas de detección de intrusos, sistema de análisis de vulnerabilidades), como de servicios comerciales basados en ésta (5).

1.2.5 OSVDB (Open Source Vulnerability Database)

Es una base de datos independiente y de código abierto, creada por y para la comunidad. El proyecto promoverá una mayor colaboración entre las compañías y los desarrolladores individuales, eliminar el trabajo redundante, reducir los costos inherentes al desarrollo y mantenimiento de las bases de datos de vulnerabilidades. Fundada en Agosto de 2002, **OSVDB**, fue creada para proveer una base de datos independiente y de código abierto. El objetivo era proveer información precisa, detallada, actual, e imparcial información técnica acerca de todos los tipos de vulnerabilidades (6).

Una de las utilidades más importantes es que se permiten descargas de los script de la base de datos, los cuales son actualizados diariamente a la 1:00 AM. El script que se brinda en el sitio contiene una secuencia de comandos SQL para importar los datos a una base de datos MySQL o para una base de datos SQLite. OSVDB cuenta con el apoyo de algunos productos como es el caso de Nessus, Snort y Nikto. En la actualidad la base de datos contiene información de 60998 vulnerabilidades de 26587 diferentes software pertenecientes a 4735 desarrolladores (6).

1.3 Lenguajes de Programación

Desde los inicios del desarrollo de Internet, sus diferentes usuarios comenzaron a demandar un cierto número de servicios a los cuales se les dio solución mediante el uso de lenguajes estáticos. Con el posterior desarrollo de las comunicaciones y de la red surgieron nuevos problemas los cuales requerían una solución más avanzada. Esto propició el surgimiento y desarrollo de lenguajes de desarrollo web dinámicos, los cuales permiten interactuar con los diferentes usuarios y utilizar sistemas de base de datos.

1.3.1 HTML

Desde el propio surgimiento de Internet, ha sido el lenguaje que ha permitido la publicación de sitios web. Es un lenguaje estático, acrónimo en inglés de HyperText Markup Language, en español Lenguaje de Marcas Hipertextuales. Desarrollado por World Wide Web Consortium **W3C**. Los archivos pueden tener las extensiones html y htm.

Ventajas	Desventajas
Sencillo que permite describir hipertexto.	Lenguaje estático.
Texto presentado de forma estructurada y agradable.	La interpretación de cada navegar puede ser diferente.
No necesita de grandes conocimientos cuando se cuenta con un editor de páginas web.	Guarda muchas etiquetas que pueden convertirse en “basura” y dificultan la corrección.
Archivos pequeños.	El diseño es más lento.
Despliegue rápido.	Las etiquetas son muy limitadas.
Lenguaje de fácil aprendizaje.	
Lo admiten todos los exploradores.	

Tabla 1: Ventajas y Desventajas del lenguaje HTML

1.3.2 JavaScript

Este es un lenguaje interpretado, no requiere compilación. Fue creado por Brendan Eich en la empresa Netscape Communications. Es similar a Java, aunque no es un lenguaje orientado a objetos, el mismo no dispone de herencias. La mayoría de los navegadores en sus últimas versiones interpretan código JavaScript. Este código puede ser interpretado dentro de nuestras páginas web, para evitar incompatibilidades con **W3C**, el autor diseño un estándar denominado **DOM**, en inglés Document Object Model, en español Modelo de Objetos del Documento.

Ventajas	Desventajas
Lenguaje seguro y fiable.	Código visible por cualquier usuario.
Los scripts tienen capacidades limitadas por razones de seguridad.	El código debe descargarse completamente.
El código JavaScript se ejecuta en el cliente.	Puede poner en riesgo la seguridad del sitio, con el actual problema llamado **XSS** (Cross Site Scripting, renombrado a **XSS** por su similitud con las hojas de estilo **CSS**).

Tabla 2: Ventajas y Desventajas del lenguaje Javascripts

1.3.3 PHP

Es un acrónimo recursivo que significa “PHP Hypertext Pre-Processor”, inicialmente se llamó Personal Home Page. Surgió en 1995, desarrollado por PHP Group. **PHP** es un lenguaje de script interpretado en el lado del servidor utilizado para la generación de páginas web dinámicas, embebidas en páginas HTML y ejecutadas en el servidor. **PHP** no necesita ser compilado para ejecutarse, pero para su funcionamiento necesita tener instalado Apache o IIS con las librerías de PHP. La mayor parte de su sintaxis ha sido tomada de **C**, **Java** y **Perl** con algunas características específicas. Los archivos se identifican por tener la extensión php.

Ventajas	Desventajas
Fácil de aprender.	Se necesita instalar un servidor web.
Se caracteriza por ser un lenguaje rápido.	Todo el trabajo lo realiza el servidor y no delega al cliente, por lo que puede ser más ineficiente a medida que las solicitudes aumenten.
Es orientado a objetos.	La legibilidad del código puede verse afectada al mezclar sentencias HTML y PHP.
Es un lenguaje multiplataforma: Linux, Windows, entre otros.	
Capacidad de conexión con la mayoría de los gestores de base de datos: MySQL, PostgreSQL, Oracle, MSSQL, entre otras.	
Capacidad para expandir su potencial usando módulos.	
Posee documentación en su página oficial la cual incluye descripción y ejemplos de cada una de sus funciones.	
Es libre, por lo que constituye una alternativa para su uso.	
Incluye gran cantidad de funciones.	
No requiere definición de tipo de variables ni manejo detallado de bajo nivel.	

Tabla 3: Ventajas y Desventajas del lenguaje PHP.

PHP es un poderoso lenguaje e intérprete, ya sea incluido como parte de un servidor web en forma de modulo o ejecutado como un binario CGI separado, es capaz de acceder a archivos, ejecutar comandos y abrir conexiones de red en el servidor. Estas propiedades hacen que cualquier cosa que sea ejecutada en un servidor web sea insegura por naturaleza. **PHP** está diseñado específicamente para ser un lenguaje más seguro para escribir programas **CGI** que Perl o C, y con la selección correcta de opciones de configuración en tiempos de compilación y ejecución, y siguiendo algunas prácticas correctas de programación.

1.3.4 ASP

Es una tecnología del lado del servidor desarrollada por Microsoft para el desarrollo de sitios web dinámicos. **ASP** (Active Server Pages), fue liberado por Microsoft en 1996. Las páginas web desarrolladas bajo este lenguaje necesitan tener instalado Internet Information Server IIS. **ASP** no necesita ser compilado para ejecutarse. Existen varios lenguajes para desarrollar este tipo de páginas, el más usado es VBScript, nativo de Microsoft. Los archivos cuentan con la extensión asp.

Ventajas	Desventajas
Usa Visual Basic Script, siendo fácil para los usuarios.	Código desorganizado.
Comunicación óptima con SQL Server.	Se necesita escribir mucho código para realizar funciones sencillas.
Soporta el lenguaje JSCript (JavaScript de Microsoft).	Tecnología propietaria.
	Hospedaje de sitios web costosos.

Tabla 4: Tabla 4: Ventajas y Desventajas del lenguaje ASP.

1.3.4.1 ASP.NET

ASP.NET es el sucesor de la tecnología ASP, fue lanzado al mercado mediante una estrategia denominada .NET. Fue desarrollado para resolver las limitantes que presentaba ASP. Para el desarrollo de ASP.NET se puede utilizar C#, VB.NET o J#. Para su funcionamiento se necesita tener instalado IIS. Los archivos cuentan con la extensión aspx.

Ventajas	Desventajas
Completamente orientado a objetos.	Mayor consumo de recursos.
Controles de usuarios personalizados.	
División entre la capa de aplicación o diseño y el código.	
Facilita el mantenimiento de grandes aplicaciones.	
Incremento de velocidad de respuesta del servidor.	
Mayor velocidad.	
Mayor seguridad.	

Tabla 5: Ventajas y Desventajas del lenguaje ASP.NET.

1.3.5 JSP

Acrónimo de Java Server Page, está orientado a desarrollar páginas web en Java. Es un lenguaje multiplataforma creado para ejecutarse del lado del servidor. Fue desarrollado por Sun Microsystems, comparte ventajas similares a las de ASP.NET, desarrollado para desarrollo de aplicaciones web potentes. Posee un motor de páginas basado en los servlets de Java. Para su funcionamiento necesita tener instalado un servidor Tomcat.

Características	Ventajas	Desventajas
Código separado de la lógica del programa.	Ejecución rápida de servlets.	Complejidad de aprendizaje.
Las páginas son compiladas en la primera petición.	Crear páginas del lado del servidor.	

Permite separar la parte dinámica de la estática en las páginas web.	Multiplataforma.	
Los archivos se encuentran con la extensión jsp.	Código bien estructurado.	
El código JSP puede ser incrustado en código HTML.	Integridad con los módulos de Java.	
	La parte dinámica está escrita en Java.	
	Permite la utilización de servlets	

Tabla 6: Ventajas y Desventajas del lenguaje JSP.

1.3.6 Ruby

Es un lenguaje interpretado de muy alto nivel y orientado a objetos. Desarrollado en 1993 por el programador japonés Yukihiro Matsumoto. Su sintaxis está inspirada en Perl y Python. Es distribuido bajo licencia Open Source. Ruby es un lenguaje dinámico para una programación orientada a objetos rápida y sencilla.

Características	Ventajas
Existe diferencia entre mayúsculas y minúsculas.	Permite desarrollar soluciones a bajo costo.
Múltiples expresiones por línea, separadas por punto y coma.	Software libre.
Dispone de manejo de excepciones.	Multiplataforma.
Puede cargar librerías de expresión dinámicas si el sistema operativo lo permite.	

Portátil.	

Tabla 7: Ventajas y Desventajas del lenguaje Ruby.

1.3.7 Python

Es un lenguaje de programación creado en el año 1990 por Guido Van Rossum, es el sucesor del lenguaje de programación ABC, esto viene haciendo de Python un lenguaje muy maduro. Python es comparado habitualmente con Perl. Presenta gran cantidad de librerías estándar, lo que permite ahorrar mucho tiempo y trabajo a la hora de desarrollar, además de que estas librerías presentan una gran posibilidad de soporte. Los usuarios lo consideran como un lenguaje más limpio para programar y permite la creación de todo tipo de programas, incluyendo que es muy fácil de aprender. Su código no necesita ser compilado, por lo que se le dice que el código es interpretado. En la actualidad hay un gran número de empresas que usan este lenguaje como el principal para las aplicaciones que estas desarrollan, entre estas podemos citar a Yahoo, Google, NASA, ILM, como algunas de las de mayor prestigio a nivel internacional. Python es un lenguaje de programación multiparadigma, lo que motiva a los programadores a escoger un estilo de programación particular, esto trae además la facilidad de la independencia de los tipos de paradigma, ya que los diferentes usuarios pueden escoger el que más relacionado con las características de la aplicación que se desee desarrollar:

- Programación orientada a objetos.
- Programación estructurada.
- Programación funcional.
- Programación orientada a aspectos.

Ventajas	Desventajas
Libre y de fuente abierta.	
Lenguaje de propósito general.	
Gran cantidad de funciones y librerías.	

Sencillo y rápido de programar.	
Multiplataforma.	
Licencia de código abierto.	
Orientado a objetos.	
Portable.	

Tabla 8: Ventajas y Desventajas del lenguaje Python.

1.4 Framework de Desarrollo

El concepto framework se emplea en muchos ámbitos del desarrollo de sistemas de software, no solo en el del desarrollo web. Se pueden encontrar infinidad de framework para el desarrollo de diferentes aplicaciones, como es el caso de los juegos, diseño gráfico e infinidad de ejemplos. De forma general el término framework se refiere a una estructura de software compuesta de componentes intercambiables y personalizables para el desarrollo de una aplicación (7).

Los objetivos principales para utilizar un framework son poder acelerar el proceso de desarrollo, reutilizar código ya existente y promover buenas prácticas mediante el uso de los patrones. Se puede definir un framework como un conjunto de componentes que forman un diseño reutilizable que facilita y agiliza el desarrollo de sistemas (7).

La mayoría de los framework web se encargan de ofrecer una capa de control de acuerdo con el patrón Modelo-Vista-Controlador o MVC, o alguno similar, ofreciendo mecanismos para facilitar la integración con otras herramientas para la implementación de las capas de negocio y presentación.

Prácticamente la mayoría de los framework web que se pueden encontrar presentan las siguientes características (7):

Abstracción de URL y sesiones.	No es necesario manipular directamente las URL ni las sesiones, el framework se encarga de hacerlo

	de forma transparente.
Acceso a Datos.	Incluyen las herramientas e interfaces necesarias para integrarse con herramientas de acceso a datos, generalmente en diversos gestores de bases de datos.
Controladores.	La mayoría de los framework implementan una serie de controladores para gestionar eventos, como una introducción de datos mediante un formulario o el acceso a una página. Estos controladores suelen ser fácilmente adaptados a las necesidades de un proyecto concreto.
Autenticación y control de acceso.	Incluyen mecanismos para la autenticación de usuarios mediante usuario y contraseña y permiten restringir el acceso a determinadas páginas a determinados usuarios.
Internacionalización	Permiten hacer cambios en el idioma de la información.
Separación entre diseño y contenido.	Esto los diferentes framework lo logran mediante el uso de patrones, generalmente el patrón Modelo-Vista-Controlador.

Tabla 9: Características de los framework.

Entre los framework para desarrollo de aplicaciones web que usan Python como lenguaje se pueden encontrar a CherryPy, Django, Grok, Karrigell, Nevow, Pyjamas, Pylons, Quixote, Spyce, TurboGears, Twisted, Web2py y Zope.

1.4.1 CherryPy

Es un framework para aplicaciones web orientado a objetos el cual usa como lenguaje de programación a Python. Tiene más de seis años de creado y en la actualidad es bastante rápido y estable. Está diseñado para el desarrollo rápido de aplicaciones web mediante uso de funcionalidades del protocolo HTTP, pero a un bajo nivel. CherryPy puede ser por sí mismo un servidor web, además de que las aplicaciones creadas con este pueden ser lanzadas usando cualquier ambiente WSGI compatible, como es el caso de Apache. No puede realizar tareas tales como renderizar las salidas, controlar los accesos traseros y los protocolos de autenticación. Este framework puede aumentar sus funcionalidades mediante el uso de diferentes paquetes y filtros.

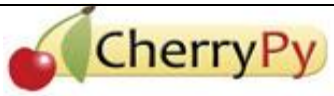
CherryPy	
Desarrolladores	Equipo de CherryPy
Última liberación estable	12 de Abril de 2009. Versión 3.1.2
Lenguaje usado	Python
Sistema Operativo	Multiplataforma
Tipo	Framework de aplicaciones web
Licencia	Licencia BSD
Sitio web	http://www.cherrypy.org

Tabla 10: Breve descripción del framework CherryPy.

1.4.2 Django

Es un framework de desarrollo web de código abierto desarrollado en Python. Fue desarrollado originalmente para gestionar páginas orientadas a noticias y fue liberado bajo licencia BSD en julio de 2005. La meta fundamental de Django es facilitar la creación de sitios web complejos. Pone énfasis en el

re-uso, la conectividad y extensibilidad de los componentes, del desarrollo rápido y del principio de DRY, del inglés *Don't Repeat Yourself*. Python es usado en todas las partes del framework, incluso en configuraciones, archivos y en los modelos de datos.

Django	
Desarrolladores	Django Software Fundation
Última liberación estable	24 de Mayo de 2010. Versión 1.2.1
Lenguaje usado	Python
Sistema operativo	Multiplataforma
Tipo	Framework de aplicaciones web
Licencia	Licencia BSD
Sitio web	http://www.djangoproject.com

Tabla 11: Breve descripción del framework Django.

1.4.3 Grok

Se basó para su desarrollo en la tecnología usada por Zope 3. El proyecto fue iniciado en 2006, y a partir de ese momento ha tenido liberaciones regulares. El principal motivo para la creación de Grok es hacer la tecnología de Zope 3 más accesible y al mismo tiempo acelerar el desarrollo de las aplicaciones, siguiendo el paradigma de programación ágil. Grok usa Python para los componentes de configuración.

Grok	

Desarrolladores	Desarrolladores de Grok, desarrolladores de Zope, Zope Fundation
Última liberación estable	18 de Mayo de 2010. Versión 1.1
Lenguaje Usado	Python
Tipo	Framework de aplicaciones web
Licencia	Licencia ZPL
Sitio web	http://grok.zope.org/

Tabla 12: Breve descripción del framework GROK.

1.4.4 Pylons

Es un framework para aplicaciones web de código abierto escrito en Python. Hace un uso extensivo de las Web Server Gateway Interface, con el fin de estandarizar y promover la reusabilidad así como separar las funcionalidades en diferentes módulos. Es fuertemente influenciado por Ruby on Rails.

Pylons	
Desarrolladores	Ben Bangert, James Gardner
Última liberación estable	23 de Febrero de 2009. Versión 0.9.7
Lenguaje usado	Python
Sistema Operativo	Multiplataforma
Tipo	Framework de aplicaciones web

Licencia	Licencia BSD
Sitio web	http://www.pylonshq.com

Tabla 13: Breve descripción del framework Pylons.

1.4.5 TurboGears

Es un megaframework para el desarrollo de aplicaciones web por el alto número de herramientas que contiene. Combina CherryPy, Kid, SQLAlchemy, Mochikit, entre otras. Presenta un buen número de herramientas de bases de datos, lo que permite tener una aplicación lista para su despliegue en muy corto tiempo. Presenta un uso de Ajax muy sencillo, tanto en el lado del cliente como en el del servidor, con un flexible y poderoso Object Relational Mapper (ORM).

TurboGears	TURBOGEARS
Desarrolladores	Kevin Dangoor, creador original y Mark Ramm, líder del grupo TG2
Última liberación estable	27 de Mayo de 2010. Versión 2.1b2
Lenguaje Usado	Python
Sistema Operativo	Multiplataforma
Tipo	Framework de aplicaciones web
Licencia	Licencia MIT, LGPL
Sitio Web	http://www.turbogears.org

Tabla 14: Breve descripción del framework TurboGears.

1.4.6 Web2py

Presenta todos los componentes en un mismo paquete, el cual no tiene otras dependencias. El desarrollo, despliegue, debug, testeo, administración de bases de datos y el mantenimiento de las aplicaciones puede hacerse utilizando la interfaz web que brinda el framework. Web2py usa Python para el modelo, la vista y los controladores y presenta un sistema de tickets para administrar los errores. Tiene internacionalización, es decir brinda soporte para múltiples lenguajes. Presenta soporte para MySQL, PostgreSQL, SQLite, Oracle, MSSQL, FireBird, IBM DB2, Informix, Ingres y Google App Engine a través de una capa de abstracción ORM u Object Relational Mapper. La seguridad protege de los tipos más comunes de vulnerabilidades. Incluye librerías para manejar HTML/XML, RSS, ATOM, CSV, RTF, JSON, AJAX, XML-RPC, WIKI, REST, Flash/AMF. Presenta compatibilidad con todas las versiones anteriores desde la versión 1.0 en el 2007 (8).

Web2py	
Desarrolladores	Massimo Di Pierro, creador, Desarrolladores web2py.
Última liberación estable	17 de Mayo de 2010. Versión 1.78.3
Lenguaje usado	Python
Sistema operativo	Multiplataforma
Tipo	Framework de aplicaciones web
Licencia	GNU GPL v2.0
Sitio web	http://www.web2py.com

Tabla 15: Breve descripción del framework web2py.

1.5 Metodología de desarrollo

Para organizar, guiar y controlar el desarrollo del sistema se ha decidido utilizar Proceso Unificado del Modelado o RUP como generalmente es conocido, como la metodología de software a utilizar. RUP junto al Lenguaje Unificado de Modelado o UML, constituye la metodología estándar más usada para el análisis, implementación y documentación de sistemas orientados a objetos, ya que RUP es un marco de trabajo genérico que puede especializarse para una gran variedad de sistemas de software, para diferentes aéreas de aplicación, diferentes tipos de organizaciones, diferentes niveles de aptitud y diferentes tamaños de proyectos.

1.6 Lenguaje de Modelado

Lenguaje Unificado de Modelado o UML por sus siglas en inglés, Unified Modeling Language, es el lenguaje de modelado de software más conocido y utilizado en la actualidad. Se encuentra respaldado por el Object Management Group u OMG. Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema. UML ofrece un estándar para describir un modelo del sistema, incluyendo aspectos conceptuales tales como procesos del negocio y funciones del sistema y aspectos concretos como expresiones del lenguaje de programación, esquemas de bases de datos y componentes reutilizables.

1.7 Herramienta CASE

Las herramientas CASE, Computer Aided Software Engineering o Ingeniería de Software Asistida por Ordenador, son diversas aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de software reduciendo el costo de las mismas en términos de tiempo y dinero. Estas herramientas proporcionan ayuda en todos los aspectos del ciclo de vida de desarrollo del software, en tareas tales como realización de diseños de proyecto, calculo de costos, implementación de parte del código automáticamente con el diseño dado, documentación o detección de errores.

1.7.1 Rational Rose

Rational Rose es una herramienta que soporta el ciclo de vida completo de un software, ofrece una agradable interfaz gráfica y facilita el trabajo en equipo lo que permite desarrollar un software con mayor

calidad y rapidez. Fue diseñado para una amplia gama de usuarios que incluye a los Ingenieros de Software, Analistas de Sistemas, Analistas de Negocios, Arquitectos.

1.8 Sistema Operativo

Linux es uno de los ejemplos más prominentes del software libre y del desarrollo del código abierto, cuyo código fuente está disponible públicamente, lo que permite estudiarlo, redistribuirlo y con los conocimientos adecuados, modificarlo.

Presenta además un gran número de ventajas con respecto a otros sistemas operativos, como es el caso de las diferentes versiones de Windows. Entre estas ventajas podemos mencionar que es más rápido, más seguro y más económico.

1.8.1 Debian GNU/Linux 5.0

Debian es un sistema operativo libre. El sistema operativo es el conjunto de programas básicos y utilidades que hacen que funcione su computadora. Debian utiliza el núcleo Linux, pero la mayor parte de las herramientas básicas vienen del Proyecto GNU, de ahí el nombre GNU/Linux.

La última versión estable de Debian es la 5.0, así como la última actualización de esa versión fue publicada el 30 de Enero de 2010.

1.9 Herramienta para el almacenamiento de datos

Los sistemas de bases de datos son programas especializados en servir como interfaz entre las bases de datos y las aplicaciones que las utilizan, tienen como objetivo general manejar de forma clara y sencilla los datos que se gestionan en el sistema. Algunos de los gestores más conocidos son: PostgreSQL, MySQL, SQLite, Oracle.

1.9.1 MySQL

MySQL es un sistema de gestión de bases de datos relacional. MySQL desde enero de 2008 es una subsidiaria de Sun Microsystems y ésta a su vez de Oracle Corporation desde abril de 2009. MySQL se desarrolla como software libre en un esquema de licenciamiento dual.

Por un lado se ofrece bajo la licencia GNU GPL para cualquier uso compatible con esta licencia, pero para aquellas empresas que quieran incorporarlo en productos privativos deben comprar a la empresa una licencia específica que les permita este uso. MySQL es propietario y está patrocinado por una empresa privada, que posee el copyright de la mayor parte del código. Se hace uso de MySQL debido a que los scripts que brinda OSVDB son para este gestor.

1.10 Servidor Web

Un servidor web es un programa que está diseñado para transferir hipertextos, páginas web o páginas HTML, textos complejos con enlaces, figuras, formularios, botones y objetos incrustados como animaciones o reproductores de música. El programa emplea el protocolo HTTP que pertenece a la capa de aplicación del modelo OSI.

1.10.1 Servidor Apache

El servidor HTTP Apache es un servidor web de código abierto para plataformas UNIX, Windows, Macintosh que implementa el protocolo HTTP/1.1. Apache presenta entre otras características altamente configurables, bases de datos de autenticación y de contenido. La mayoría de las vulnerabilidades descubiertas y resueltas tan solo pueden ser aprovechadas por usuarios locales y no remotamente. Sin embargo, algunas pueden ser accionadas remotamente en ciertas situaciones, o explotadas por los usuarios locales en las disposiciones de recibimiento compartidas que utilizan PHP como módulo de Apache.

1.11 Lenguaje Programación

Un lenguaje de programación es un lenguaje artificial diseñado para expresar ideas que pueden ser llevadas a cabo por máquinas como las computadoras. Pueden usarse para crear programas que controlen el comportamiento físico y lógico de una máquina y expresar algoritmos con precisión. Está formado por un conjunto de símbolos y reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos y expresiones.

1.11.1 Python

De todos los lenguajes mencionados, para el desarrollo del sistema web se seleccionó el lenguaje de programación Python.

1.12 Framework web2py

Se seleccionó el framework web2py por sus facilidades a la hora de desarrollar aplicaciones haciendo uso de esta herramienta, así como por las características y las utilidades que presenta.

1.13 Conclusiones

En el presente capítulo se ha presentado una investigación sobre el estado actual de las bases de datos que a nivel internacional brindan información sobre vulnerabilidades informáticas, la forma en que funcionan y las posibilidades que brindan. De forma paralela se hizo un estudio de los lenguajes de programación usados para el desarrollo de aplicaciones web, así como de los framework que hacen uso de los mismos.

CAPÍTULO 2: CARACTERÍSTICAS DEL SISTEMA

2.1 Introducción

En este capítulo se presentan las características del sistema, se describen los requisitos funcionales y no funcionales, así como un análisis del problema a resolver y a partir de este punto una propuesta del sistema. Se detallará el modelo de dominio, diagrama de actividades, modelo de objeto, el de casos de usos del sistema y la descripción de estos casos de usos.

2.2 Objeto de estudio

En el mundo para conocer de las vulnerabilidades que puede sufrir un determinado sistema se contratan los servicios de una empresa que realice pruebas de vulnerabilidades, o mediante el uso de las bases de datos de vulnerabilidades. En la actualidad hay varias instituciones que brindan estos servicios, entre estas se encuentra OSVDB, la cual los brinda de forma libre y confiable.

2.2.1 Problema

La Universidad de las Ciencias Informáticas en la actualidad cuenta con una infraestructura de red bastante compleja y de la cual depende el correcto funcionamiento de la institución, ya que por la misma circula información importante para el desarrollo de los diferentes proyectos productivos y para la superación académica de estudiantes y profesores. A esto hay que agregar el gran número de aplicaciones y servicios que las utilizan. En la actualidad no se dispone de un servicio que informe de las vulnerabilidades existentes y que a la vez puedan presentar las diferentes computadoras o aplicaciones que de una forma u otra interactúan en la red. Teniendo en cuenta que una red es tan segura como lo sea la estación o el servicio más débil que se encuentre en la misma, tratar de tener un nivel de seguridad adecuado es tarea de todos los usuarios de la misma, no solamente de los administradores.

2.2.2 Objeto de automatización

Se hace necesario desarrollar un sistema web que mantenga informado a los usuarios de las posibles vulnerabilidades que pueden tener los diferentes sistemas y tecnologías usadas por los mismos, además de que les brinde información relacionada con las vulnerabilidades y su solución.

El sistema permitirá que los usuarios realicen consultas acercas de las vulnerabilidades, y que se suscriban al mismo para que se les informe mediante envío de correos electrónicos de las vulnerabilidades nuevas, evitando de esta forma que el usuario tenga que acceder al mismo cada vez que desee comprobar si hay información actualizada.

2.2.3 Información que se maneja

La información que se manipula es la proveniente de la Base de Datos de Vulnerabilidades de Código Abierto u OSVDB. De cada vulnerabilidad se conocerá: identificador, nombre, día de descubrimiento, día de explotación, día que se agregó a la base de datos, día que se modificó la información referente a la misma por última vez, día que se publicó el exploit, día que apareció la solución, descripción de la vulnerabilidad, solución. Además, se manejará otras informaciones de interés como es el caso de los vendedores de los objetos afectados, nombre y versión de los productos afectados, datos de las personas que agregaron la vulnerabilidad a la base de datos, y la clasificación de los productos afectados.

2.3 Propuesta de sistema

Se propone un sistema que desde un ambiente web un usuario pueda solicitar información sobre las vulnerabilidades conocidas del sistema o tecnología que desee. El sistema se encargará de ofrecerle la información solicitada, así como brindarle la posibilidad de suscribirse al servicio de envió de información y de esta forma reciba de forma periódica datos de las nuevas vulnerabilidades que se agreguen al sistema.

El sistema será creado y administrado usando el framework web2py y para la implementación del mismo se utilizará el lenguaje de programación Python. Para garantizar el correcto funcionamiento del sistema se han creado roles que tributan al nivel de acceso y de servicios permitidos por el sistema, estos son: Administrador, Cliente, Cliente Agregado.

El Administrador del sistema es el encargado de supervisar el correcto funcionamiento del mismo, así como podrá eliminar Clientes Agregados que ya no cumplan con los requisitos. Los Clientes son todos aquellos usuarios que accedan al sistema con el fin solicitar alguna información específica. Los Clientes Agregados son aquellos que además de solicitar una determinada información se suscriben al sistema para que este de forma automática le informe de las nuevas vulnerabilidades.

2.4 Modelo de Dominio

El objetivo del modelado del dominio es contribuir a la comprensión del contexto del sistema, y por lo tanto contribuir a la comprensión de los requisitos del sistema que se desprende de este contexto. En otras palabras el modelado del dominio debería contribuir a la comprensión del problema que se supone que el sistema soluciona en relación a su contexto (9).

A continuación se identifican un glosario de términos correspondientes al Modelo de Dominio para favorecer el entendimiento del mismo y del negocio a desarrollar:

Cliente: Persona que accede al sistema para solicitar información.

Cliente Agregado: Persona que accede al sistema para solicitar información y además se suscribe al mismo para recibir información periódica.

Administrador: Persona encargada de administrar el sistema.

Reporte: Información generada por el sistema para responder la solicitud hecha por un Cliente Agregado.

SBDV: Sistema de Notificación de Vulnerabilidades del Proyecto LABSI.

Vulnerabilidades: Base de datos de Vulnerabilidades del Sistema de Notificación de Vulnerabilidades del Proyecto LABSI.

Usuarios: Base de datos de usuarios del Sistema de Notificación de Vulnerabilidades del Proyecto LABSI.

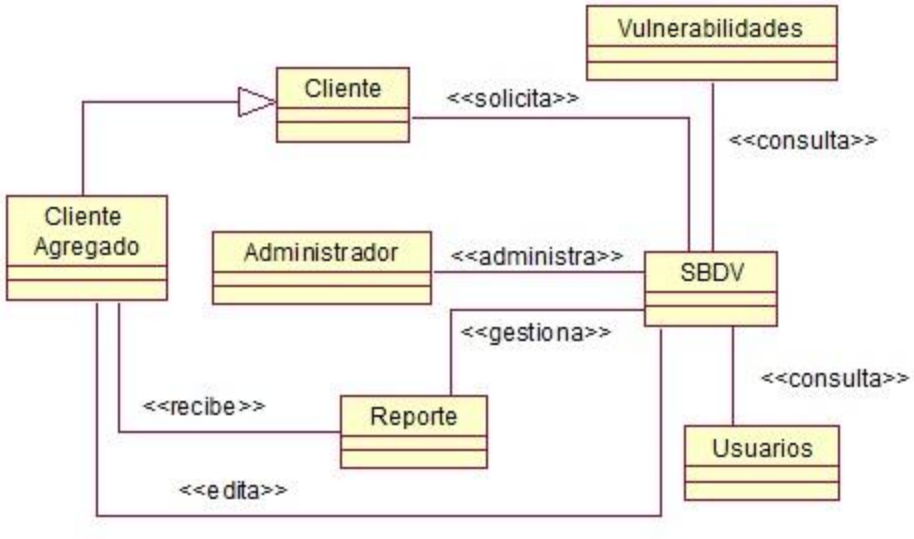


Figura 1: Modelo de Dominio

2.5 Relación de los requerimientos

Los requisitos de software son condiciones o capacidades que tienen que ser alcanzadas o poseídas por un sistema o componente de un sistema para satisfacer un contrato, estándar u otro documento impuesto formalmente. Define que es lo que el sistema debe hacer, para lo cual se identifican las funcionalidades requeridas y las restricciones que se imponen. Es una característica que el sistema debe poseer para cubrir alguna de las necesidades de los usuarios que lo motivan para resolver un problema o lograr un objetivo.

2.5.1 Listado de los requerimientos funcionales

A continuación se listan las funciones que el sistema debe ser capaz de realizar:

RF1: Inscribir Usuario.
RF2: Editar Perfil.
RF3: Gestionar Usuario
    RF3.1: Listar Usuario
    RF3.2: Eliminar Usuario
RF4: Autenticar Usuario.

RF5: Mostrar Vulnerabilidades.
RF6: Realizar Búsquedas.
RF7: Crear Reporte.
RF8: Enviar Reporte.
RF9: Descargar Base de Datos.

2.5.2 Definición de los requerimientos no funcionales

Los requerimientos no funcionales, atributos de calidad, aseguran que se disponga de un sistema manejable y gestionable que ofrezca la funcionalidad requerida de manera fiable, ininterrumpida o con el tiempo mínimo de interrupción, incluso antes situaciones inusuales. Estos requerimientos están dados en las siguientes categorías:

Rendimiento.

El sistema deberá responder en el mínimo de tiempo posible ante las solicitudes de información por parte de otros sistemas y en el procesamiento de la información. La eficiencia de la aplicación estará en gran medida determinada por el aprovechamiento de los recursos que se dispone en el modelo de capas, y en la velocidad de las consultas realizadas a la base de datos.

Soporte.

Se documentará la aplicación con un manual de ayuda con el fin de explicar el funcionamiento del sistema para garantizar un soporte óptimo a la herramienta. El proyecto se realizará extensible de forma que permita darle mantenimientos al sistema a fin de aumentar funcionalidades y/o corregir errores del mismo a través de versiones posteriores. El servicio de envío de reportes y actualización de la información de la base de datos del sistema se realizará de forma automatizada, a pesar de esto los servicios de instalación, mantenimiento y supervisación del sistema será tarea del administrador del mismo en la entidad donde sea utilizado.

Seguridad.

La información estará protegida contra accesos no autorizados utilizando mecanismos de validación que puedan garantizar el cumplimiento de esto: cuenta, contraseña y nivel de acceso, de manera que cada

uno de los usuarios del sistema pueda tener disponible solamente las opciones relacionadas con su actividad y tenga datos de acceso propios, garantizando así la confidencialidad.

Se usarán mecanismos de encriptación de los datos que por cuestiones de seguridad no deben viajar al servidor en texto plano, como es el caso de las contraseñas. Se guardará encriptado esta información en la base de datos usando para ello los mecanismos de encriptación adecuados.

Software.

El servidor trabajará sobre una distribución Linux, con servidor web Apache y usará MySQL como gestor de base de datos.

Hardware.

Se requiere de un servidor de base de datos con 80 GB de disco duro a lo sumo, ya que la información que se necesita almacenar, usuarios registrados en el sistema e información sobre vulnerabilidades, se encuentra en constante crecimiento. Es importante señalar que también se guarda un registro de todas las acciones que realizan los diferentes usuarios en el sistema. Se requiere de un servidor dedicado en la capa de negocio para la aplicación web con 1GB o 2GB de RAM para cubrir la demanda de al menos de 150 usuarios de manera simultánea, ya que una de las funcionalidades principales es la realización de búsquedas por los usuarios, las cuales pueden llegar a ser complejas en algunos casos.

2.6 Modelo de Casos de Uso del Sistema

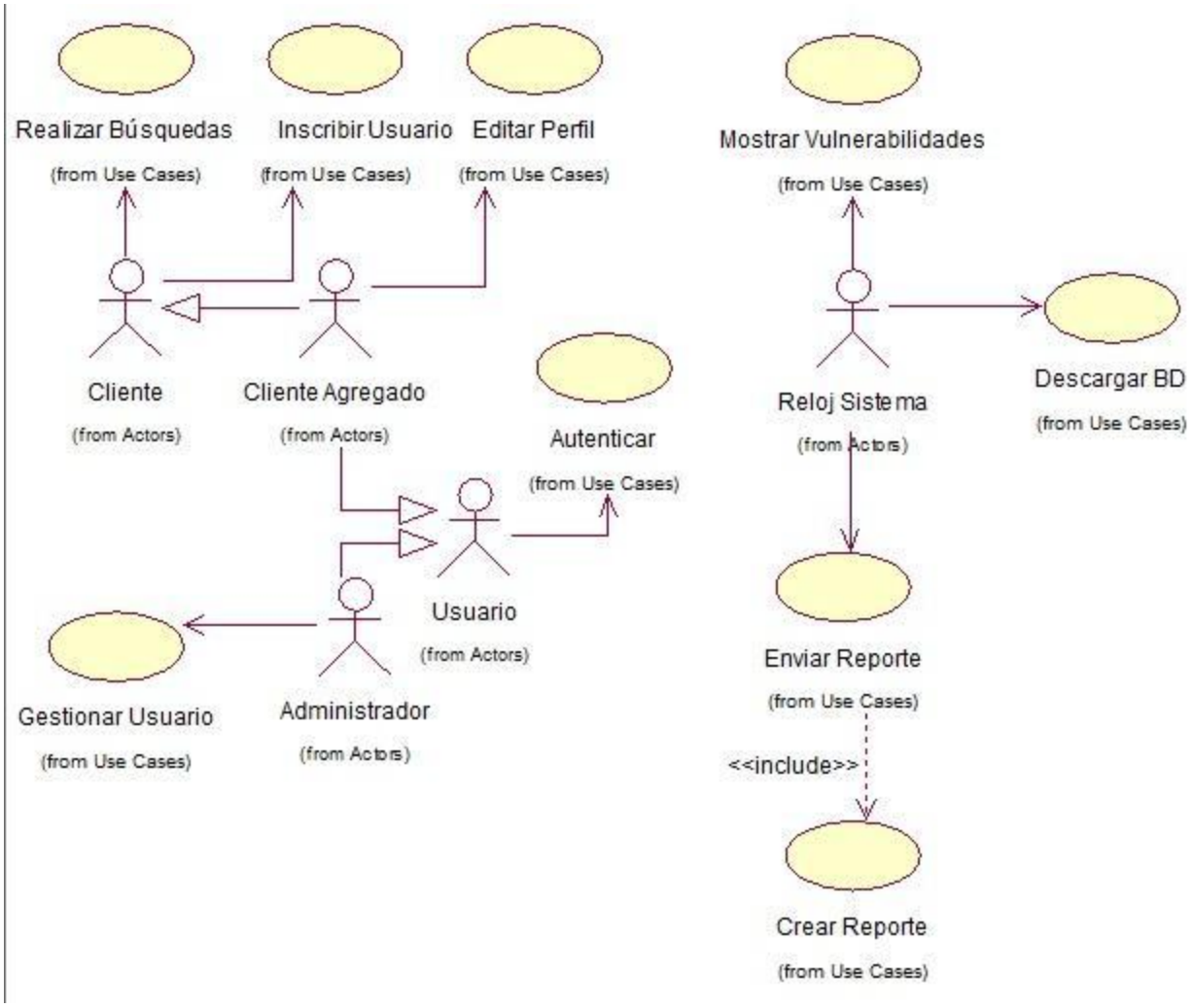


Figura 2: Modelo de Casos de Uso del Sistema.

2.6.1 Definición de los actores del sistema a automatizar

Actores del sistema	Justificación
Cliente	Persona que puede realizar búsquedas en el sistema, pero que no posee información personal en el mismo.

Cliente Agregado	Persona que puede realizar búsquedas en el sistema y que posee información registrada en el mismo, por lo que puede estar suscrito al servicio de envío de reportes.
Administrador	Persona encargada de la supervisión del sistema. Puede eliminar Clientes Agregados que no cumplan los requerimientos del sistema.
Reloj Sistema	Este es un actor ficticio que se encarga de iniciar procesos en el sistema cada cierto tiempo.

Tabla 16: Listado de actores del sistema.

2.6.2 Descripción de los Casos de Uso del sistema.

Caso de Uso	
Caso de Uso 1	Inscribir Usuario.
Propósito	Permite al Cliente ingresar información personal en el sistema, y solicitar servicios especializados.
Actores	Cliente
Resumen	El caso de uso se inicia cuando el cliente solicita registrarse en el sistema y culmina con la inserción satisfactoria de los datos del mismo.
Referencias	RF 1
Pre condiciones	
Post condiciones	El usuario queda registrado en el sistema.

Flujo normal de eventos de sección Inscribir Usuario.	
Acción del Actor	**Respuesta del Sistema**
1. El Cliente solicita registrarse en el sistema.	2. El sistema muestra la interfaz para registrar usuarios.
3. El Cliente entra los datos solicitados por el sistema	4. El sistema verifica que los datos ingresados son correctos. 5. El sistema informa que los datos son correctos. 6. El sistema inserta los datos en la base de datos. 7. El sistema muestra la interfaz principal.
Flujo alterno de eventos de sección Inscribir Usuario.	
Acción del actor	**Respuesta del Sistema**
	5.1 El sistema informa que los datos son incorrectos. 6.1 El sistema vuelve al paso 2 del flujo normal de eventos de sección Inscribir Usuario.

Tabla 17: Descripción textual del CUS-Inscribir Usuario.

Caso de Uso	
Caso de Uso 2	Editar Perfil
Propósito	Permite al Cliente Agregado modificar su información personal y cambiar la configuración de

	los servicios especializados.
Actores	Cliente Agregado
Resumen	El caso de uso se inicia cuando el cliente solicita cambiar su información personal o la solicitud de servicios especializados y culmina con la inserción de los nuevos datos en el sistema.
Referencias	RF 2
Pre condiciones	El Cliente Agregado debe estar autenticado en el sistema
Post condiciones	La nueva información del Cliente Agregado es registrada con éxito.
Flujo normal de eventos de sección Editar Perfil.	
Acción del Actor	**Respuesta del Sistema**
1. El Cliente Agregado solicita modificar su información personal ó la configuración de los servicios especializados	2. El sistema muestra la información personal del Cliente Agregado.
3. El Cliente Agregado realiza los cambios deseados.	4. El sistema verifica que los datos ingresados son correctos. 5. El sistema informa que los datos son correctos. 6. El sistema inserta los datos en la base de datos. 7. El sistema muestra la interfaz principal.

Flujo alterno de eventos de sección Editar Perfil.	
Acción del Actor	**Respuesta del Sistema**
	5.1 El sistema informa que los datos son incorrectos. 6.1 El sistema vuelve al paso 2 del flujo normal de eventos de sección editar Perfil.

Tabla 18: Descripción textual del CUS-Editar Perfil.

Caso de Uso	
Caso de Uso 3	Gestionar Usuario
Propósito	Permite al Administrador listar los Clientes Agregados y darles de baja si dejan de cumplir los requisitos del sistema.
Actores	Administrador
Resumen	El caso de uso se inicia cuando el Administrador solicita listar los usuarios o darle baja a un determinado usuario.
Referencias	RF 3
Pre condiciones	El Administrador debe estar autenticado en el sistema
Post condiciones	El Cliente Agregado es eliminado del sistema
Flujo normal de eventos de sección Listar Usuarios.	

Acción del Actor	Respuesta del Sistema
1. El Administrador solicita un listado de los Clientes Agregados del sistema.	2. El sistema muestra un listado con todos los Clientes Agregados solicitados.
Flujo normal de eventos de sección Eliminar Usuario.	
Acción del Actor	**Respuesta del Sistema**
1. El Administrador solicita dar baja del sistema a un Cliente Agregado.	2. El sistema muestra la interfaz para eliminar usuarios.
3. El Administrador selecciona al Cliente Agregado al cual se le dará baja del sistema	4. El sistema elimina la información personal del Cliente Agregado de la base de datos.

Tabla 19: Descripción textual del CUS-Gestionar Usuario.

Caso de Uso	
Caso de Uso 4	Autenticar Usuario
Propósito	Permite comprobar las credenciales ofrecidas por el Usuario.
Actores	Usuario.
Resumen	El caso de uso se inicia cuando un Usuario provee un juego de credenciales para autenticarse en el sistema.
Referencias	RF 4
Pre condiciones	El Usuario debe estar registrado en la Base de Datos del sistema.

Post condiciones	El Usuario queda autenticado en el sistema.
Flujo normal de eventos de sección Autenticar Usuario.	
Acción del actor	**Respuesta del sistema**
1. El usuario solicita autenticarse en el sistema.	2. El sistema muestra la interfaz para autenticarse en el sistema.
3. El usuario entra los datos correspondientes.	4. El sistema verifica que los datos ingresados sean correctos. 5. El sistema informa que los datos son correctos. 6. El sistema muestra la interfaz correspondiente dependiendo del rol del usuario.
Flujo alterno sección Autenticar Usuario.	
Acción del actor	**Respuesta del sistema**
	5.1 El sistema informa que los datos son incorrectos. 6.1 El sistema vuelve al paso 2 del flujo normal de eventos de sección Autenticar Usuario.

Tabla 20: Descripción textual del CUS-Autenticar Usuario.

Caso de Uso	
Caso de Uso 5	Mostrar Vulnerabilidades
Propósito	Informar de forma general de nuevas

	vulnerabilidades.
Actores	Cliente
Resumen	El caso de uso se inicia cuando un Cliente ingresa a la página principal del sistema y le es mostrada información sobre las últimas vulnerabilidades ingresadas al sistema.
Referencias	RF 5
Pre condiciones	
Post condiciones	
Flujo normal de eventos de sección Mostrar Vulnerabilidades.	
Acción del actor	**Respuesta del sistema**
1. El usuario solicita la interfaz principal del sistema.	2. El sistema busca información sobre nuevas vulnerabilidades. 3. El sistema muestra información en la interfaz principal.

Tabla 21: Descripción textual del CUS-Mostrar Vulnerabilidades.

Caso de Uso	
Caso de Uso 6	Realizar Búsquedas
Propósito	Buscar información específica en el sistema.
Actores	Cliente

Resumen	El caso de uso se inicia cuando un Cliente desea buscar información específica acerca de las vulnerabilidades y culmina cuando se muestra el resultado de la búsqueda.
Referencias	RF 6
Pre condiciones	
Post condiciones	
Flujo normal de eventos de sección Realizar Búsquedas.	
Acción del actor	**Respuesta del sistema**
1. El usuario solicita buscar información específica de las vulnerabilidades.	2. El sistema muestra la interfaz para realizar búsquedas.
3. El usuario introduce los datos de la búsqueda.	4. El sistema verifica que los datos ingresados sean correctos. 5. El sistema informa que los datos son correctos. 6. El sistema muestra la información solicitada.
Flujo alterno sección Realizar Búsquedas.	
Acción del actor	**Respuesta del sistema**
	5.1 El sistema informa que los datos son incorrectos o no existe información sobre el criterio de búsqueda seleccionado. 6.1 El sistema vuelve al paso 2 del flujo normal de

	eventos de sección Realizar Búsquedas.

Tabla 22: Descripción textual del CUS-Realizar Búsquedas.

Caso de Uso	
Caso de Uso 7	Crear Reporte
Propósito	Creación del reporte que será enviado al cliente agregado.
Actores	Reloj Sistema
Resumen	El caso de uso se inicia cuando el Reloj del Sistema solicita crear un reporte y culmina con la creación del mismo.
Referencias	RF 7
Pre condiciones	
Post condiciones	El reporte es creado
Flujo normal de eventos de sección Crear Reporte.	
Acción del actor	**Respuesta del sistema**
1. Se solicita crear un reporte.	2. El sistema busca información solicitada. 3. El sistema crea el reporte.

Tabla 23: Descripción textual del CUS-Crear Reporte.

Caso de Uso

Caso de Uso 8	Enviar Reporte
Propósito	Enviar reporte al Cliente Agregado con información solicitada por el mismo en los servicios especializados.
Actores	Reloj Sistema
Resumen	El caso de uso se inicia cuando el Reloj del Sistema solicita enviar un reporte a un Cliente Agregado y culmina cuando el reporte es enviado.
Referencias	RF 8
Pre condiciones	El reporte debe estar creado
Pos condiciones	El reporte es enviado
Flujo normal de eventos de sección Enviar Reporte.	
Acción del actor	**Respuesta del sistema**
1. Se solicita enviar un reporte.	2. El sistema busca el reporte solicitado. 3. El sistema envía el reporte al Cliente Agregado.

Tabla 24: Descripción textual del CUS-Enviar Reporte.

Caso de Uso	
Caso de Uso 9	Descargar Base de Datos
Propósito	Descargar el script de la base de datos del sistema.

Actores	Reloj Sistema
Resumen	El caso de uso se inicia cuando el Reloj del Sistema solicita descargar el script de la base de datos del sistema y culmina con la descarga del mismo.
Referencias	RF 9
Pre condiciones	
Post condiciones	Queda descargado script de la base de datos del sistema
Flujo normal de eventos de sección Descargar Base de Datos.	
Acción del actor	**Respuesta del sistema**
1. Se solicita descargar script de base de datos del sistema.	2. El sistema accede a URL específica para descargar script de la base de datos. 3. El sistema descarga el script de base de datos.
Flujo alterno de sección Descargar Base de Datos.	
Acción del actor	**Respuesta del sistema**
	2.1 El sistema informa que no se puede acceder a URL especificada o que no se puede descargar la información solicitada.

Tabla 25: Descripción textual del CUS-Descargar Base de Datos.

2.7 Conclusiones.

En este capítulo se ha planteado la necesidad de que exista un sistema de notificación de vulnerabilidades y los objetivos que este cumple, así como la necesidad de que el mismo sea desarrollado en un entorno web. Se hace propuesta de un sistema capaz de resolver estas dificultades y de un análisis de los requerimientos funcionales y no funcionales del sistema, facilitando de esta manera la identificación de los casos de uso del sistema.

CAPÍTULO 3: ANÁLISIS Y DISEÑO.

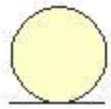
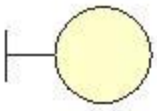
3.1 Introducción.

En este capítulo se introducen los conceptos de Análisis y Diseño, que son una parte fundamental para el desarrollo de la aplicación. Al realizar el análisis se garantiza un buen diseño y posteriormente una implementación correcta. Se mostrarán los diagramas de clases del análisis, también los diagramas de clases del diseño con estereotipos Web. Además, se realizan los diagramas de secuencia del diseño, que ayudarán a la implementación del sistema.

3.2 Análisis.

Durante el análisis, se analizan los requisitos que se describen en la captura de requerimientos, refinándolos y estructurándolos. El objetivo de hacerlo es conseguir una comprensión más precisa de los requisitos y una descripción de los mismos que sea fácil de mantener y que ayude a estructurar el sistema entero, incluyendo su arquitectura (9).

El artefacto más importante que genera el análisis es el modelo del análisis, este contiene las clases del análisis estas se centran en los requisitos funcionales, además presentan conceptos y relaciones del dominio. Estas clases del análisis se pueden clasificar en:

Nombre	Características	Representación
Entidad	Modelan información que posee larga vida y que es a menudo persistente.	
Interfaz	Modelan la interacción del sistema y sus actores.	

Control	Coordinan la relación entre uno u otros casos de uso, coordinado de las actividades de los objetos que implementan la funcionalidad del caso de uso.	

Tabla 26: Clases del análisis.

3.2.1 Diagramas de Clase del Análisis.

Este es un artefacto del análisis, que representa las clases del análisis y la relación entre ellas. Este diagrama da una vista estática del sistema. Representa el mundo real, no la implementación.

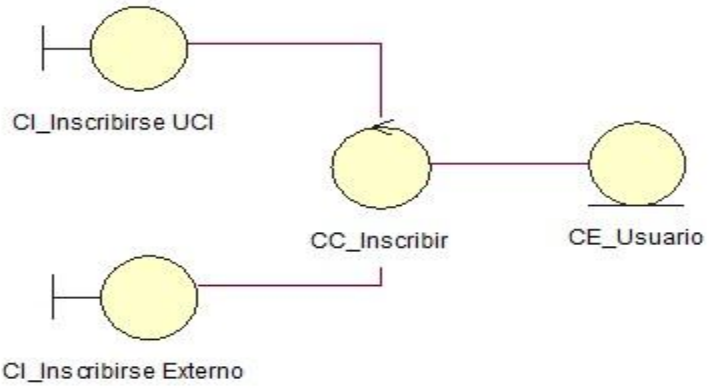


Figura 3: Diagrama de clase del análisis CU-Inscribir Usuario.

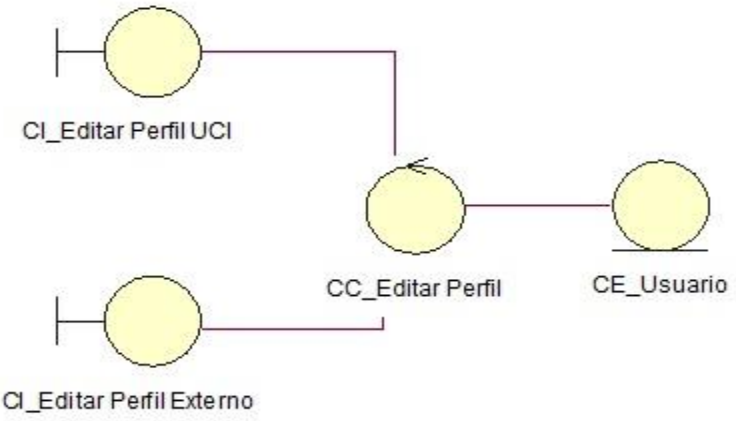


Figura 4: Diagrama de clase del análisis CU-Editar Perfil.

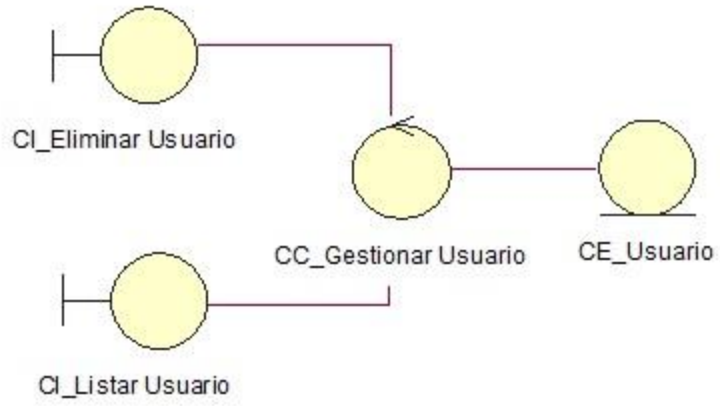


Figura 5: Diagrama de clase del análisis CU-Gestionar Usuario.

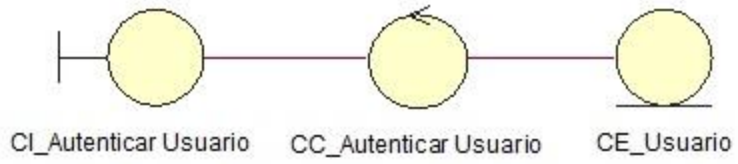


Figura 6: Diagrama de clase del análisis CU-Autenticar Usuario.

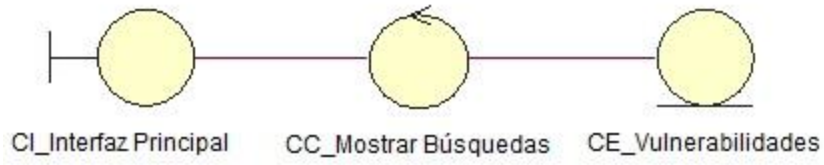


Figura 7: Diagrama de clase del análisis CU-Mostrar Vulnerabilidad.

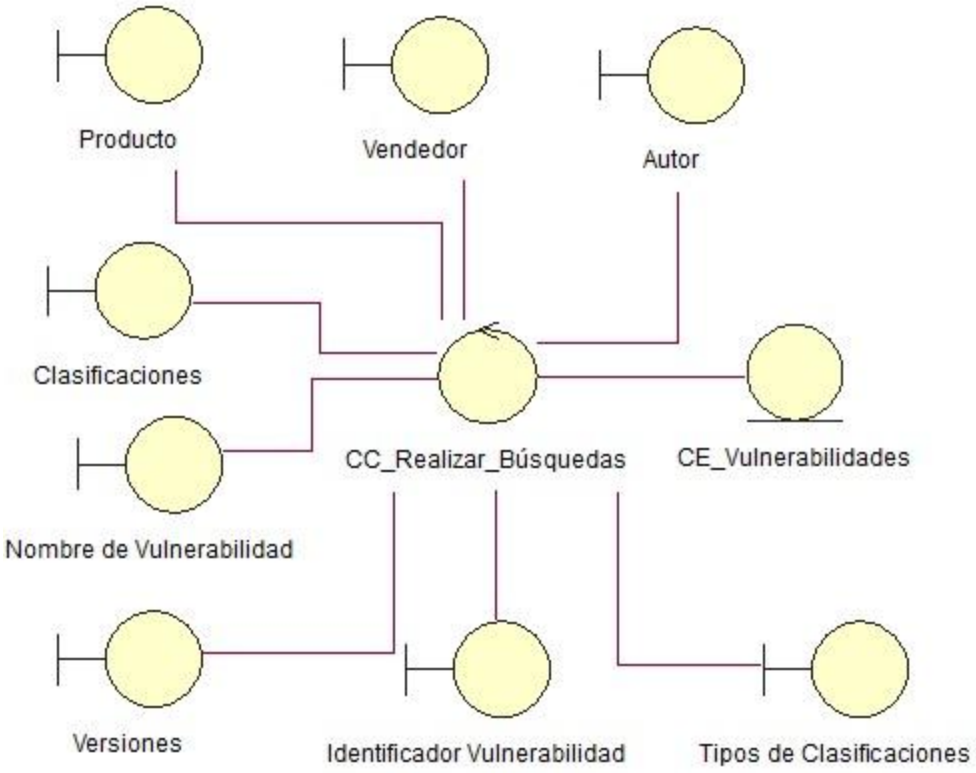


Figura 8: Diagrama de clase del análisis CU-Realizar Búsquedas.

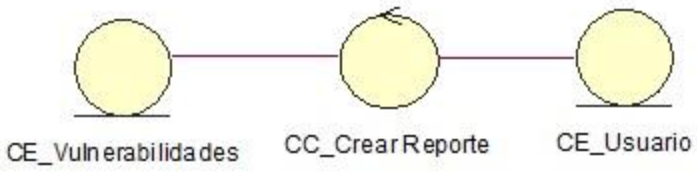


Figura 9: Diagrama de clase del análisis CU-Crear Reporte.

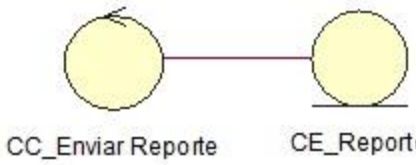


Figura 10: Diagrama de clase del análisis CU-Enviar Reporte.

Figura 11: Diagrama de clase del análisis CU-Descargar Base de Datos.

3.3 Diseño

El propósito del diseño es adquirir una comprensión a fondo de los aspectos relacionados con los requisitos no funcionales y restricciones relacionadas con los lenguajes de programación, componentes reutilizables, sistemas operativos, tecnologías de distribución y concurrencia, tecnología de interfaz de usuario, tecnología de gestión de transacciones (9).

El artefacto más importante es el modelo de diseño, el cuál.es un modelo de objetos que describe la realización física de los casos de uso centrándose en cómo los requisitos funcionales y no funcionales, junto con otras restricciones relacionadas con el entorno de implementación, tienen un impacto en el sistema a considerar (9).

3.3.1 Patrones de Diseño

Una arquitectura orientada a objetos bien estructurada contiene el uso de múltiples patrones. La calidad de un sistema orientado a objetos se mide por la atención que los diseñadores han prestado a la colaboración entre sus objetos. Los patrones conducen a arquitecturas más pequeñas, más simples y más comprensibles (10).

Los patrones de diseño son soluciones simples y elegantes a problemas específicos y comunes del diseño orientado a objetos. Son soluciones basadas en la experiencia y que se ha demostrado que funcionan. Es evidente que a lo largo de multitud de diseños de aplicaciones hay problemas que se repiten o que son análogos, es decir, que responden a un cierto patrón. Sería deseable tener una colección de dichos patrones con las soluciones óptimas para cada caso. Los patrones de diseño no son fáciles de entender, pero una vez entendido su comportamiento, los diseños serán mucho más flexibles, modulares y reutilizables. Han revolucionado el diseño orientado a objetos y todo buen arquitecto de software debería conocerlos (11).

Un patrón es un esquema o micro arquitectura que supone una solución a problemas (dominios de aplicación) semejantes. Los patrones son patrones del dominio de la solución. En resumen, un patrón es el denominador, una estructura común que tiene aplicaciones semejantes.

Para el diseño de aplicaciones con sofisticados interfaces se utiliza el patrón Modelo-Vista-Controlador. La lógica de una interfaz de usuario cambia con más frecuencia que los almacenes de datos y la lógica de negocio. Si se realiza un diseño confuso, que mezcle los componentes de interfaz y negocio, entonces el resultado será que, cuando se necesite cambiar la interfaz, se tendrá que modificar trabajosamente los elementos del negocio. Mayor trabajo y más riesgo de error. Se trata de realizar un diseño que desacople la vista del modelo, con la finalidad de mejorar la reusabilidad. De esta forma, las modificaciones en la vista impactan en menor medida en la lógica de negocio o datos (12).

Este patrón cuenta con tres elementos fundamentales. El modelo contiene datos y reglas de negocio. La vista muestra la información del modelo al usuario y el controlador que se encarga de la gestión de las entradas del usuario. Un modelo puede tener diversas vistas, cada una con su correspondiente controlador.

De forma general se puede definir (12):

1. El modelo es el responsable de:
    a. Acceder a la capa de almacenamiento de datos. Lo ideal es que el modelo sea independiente de la capa de almacenamiento.
    b. Define las reglas de negocio (la funcionalidad del sistema).
    c. Lleva un registro de las vistas y los controladores del sistema.
    d. Si estamos ante un modelo activo, notificará a las vistas de los cambios que en los datos pueda producir un agente externo.
2. El controlador es responsable de:
    a. Recibir los eventos de entradas.

b. Contener las reglas de gestión de eventos.

3. Las vistas son responsables de:

   a. Recibir los datos del modelo y mostrarlos a los usuarios.

   b. Tener un registro de su controlador asociado.

Web2py obliga al desarrollador a la representación por separado los datos, el modelo, la presentación de datos, la vista, y el flujo de trabajo de la aplicación, el controlador.

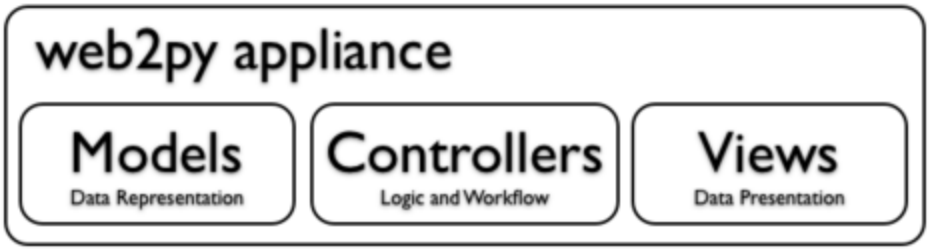


Figura 12: Modelo Vista Controlador.

El flujo de trabajo típico de una petición de web2py es como se muestra en el siguiente diagrama.

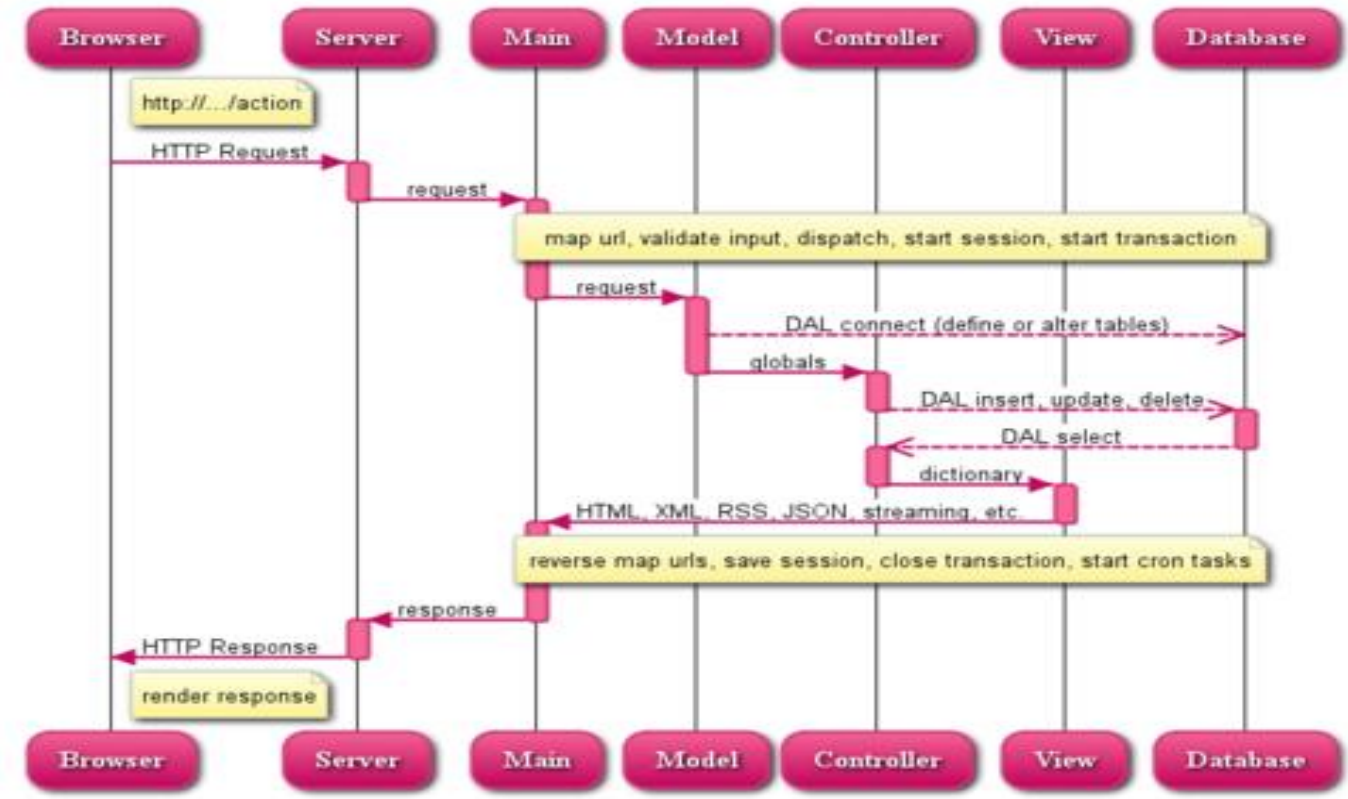


Figura 13: Flujo de trabajo de peticiones web2py.

En el diagrama anterior, el servidor puede ser el que trae incorporado el web2py o un servidor de terceros, como sería en el caso del servidor Apache. Cabe señalar que el servidor por defecto de web2py maneja multi-threading. Main es la aplicación principal, la cual realiza todas las tareas comunes y envuelve las aplicaciones de usuario. Los Model, Controller y View, componen la aplicación del usuario. Las líneas de puntos representan la comunicación con los motores de base de datos (13).

Todas las llamadas se envuelven en una transacción y cualquier excepción no detectada hace la transacción retroceder. El framework se ocupa de las sesiones y las cokies de sesión de forma automática. Permite registrar las tareas periódicas, cron, para ejecutarse a horas fijas y/ó después de la realización de determinadas acciones. De esta manera, es posible la realización de acciones en segundo plano sin ralentizar la navegación (13).

3.3.2 Diagrama de clases del diseño

Una clase del diseño es una abstracción de una clase; y hay que tener algunas consideraciones con dichas clases, por ejemplo que deben ser especificadas en el mismo lenguaje en el que se va a implementar (9).

En los diagramas de clases del diseño se muestran las clases del diseño, los subsistemas y sus relaciones. Este permite coordinar todos los requisitos para la realización de los diferentes casos de uso (9).

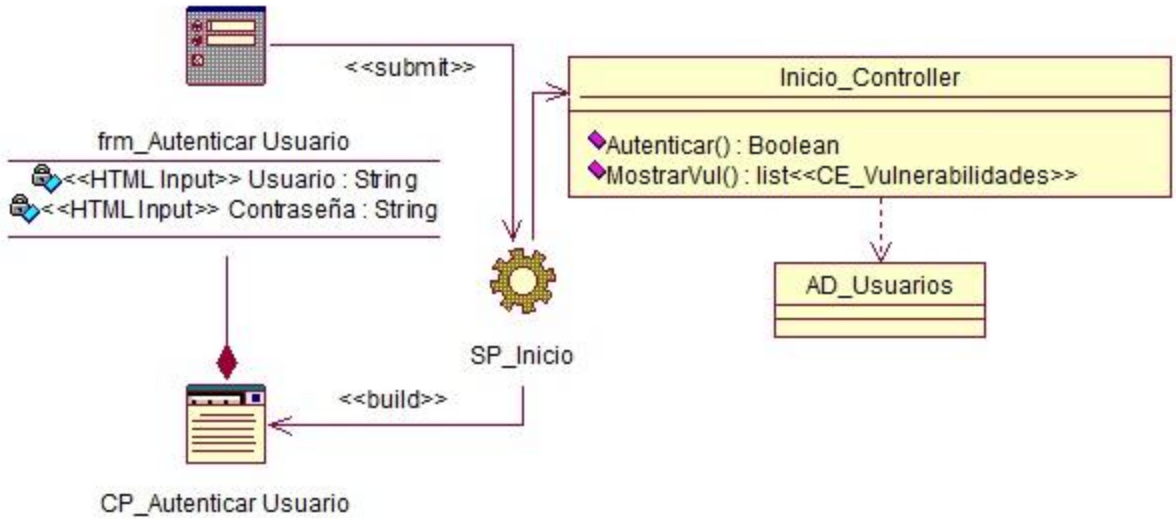


Figura 14: Diagrama de clases del diseño CU- Autenticar Usuario.

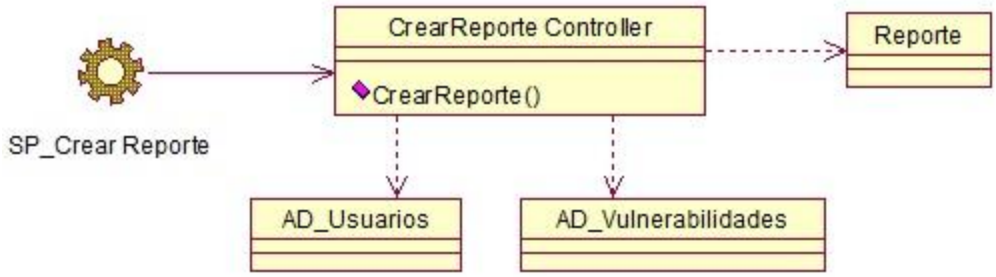


Figura 15: Diagrama de clases del diseño CU- Crear Reporte.

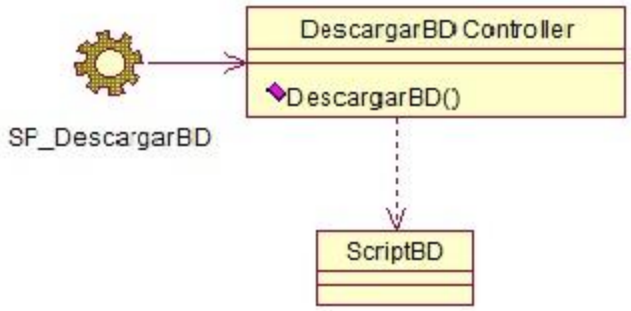


Figura 16: Diagrama de clases del diseño CU- Descargar Base de Datos.

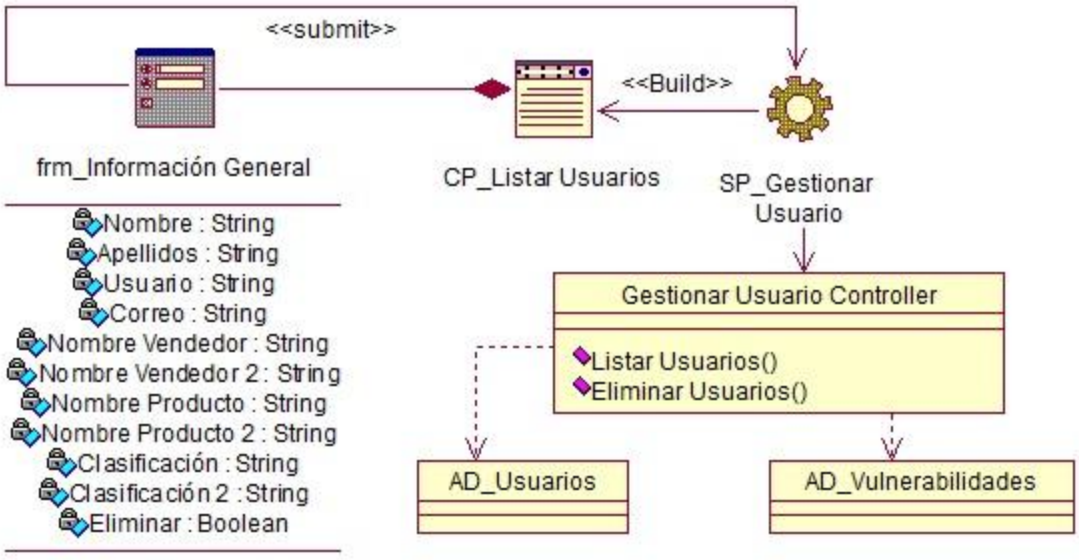


Figura 17: Diagrama de clases del diseño CU- Gestionar Usuario.

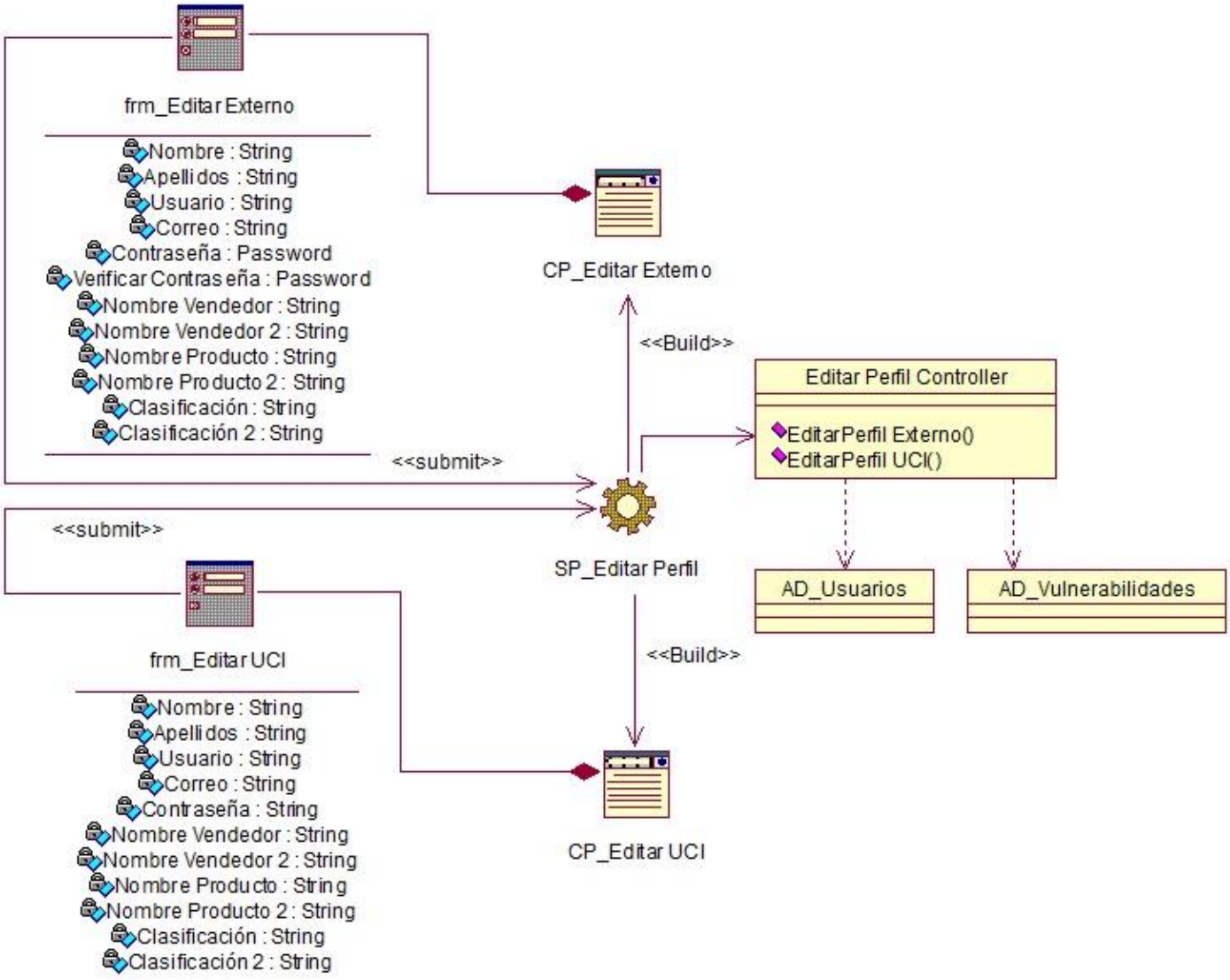


Figura 18: Diagrama de clases del diseño CU- Editar Perfil.

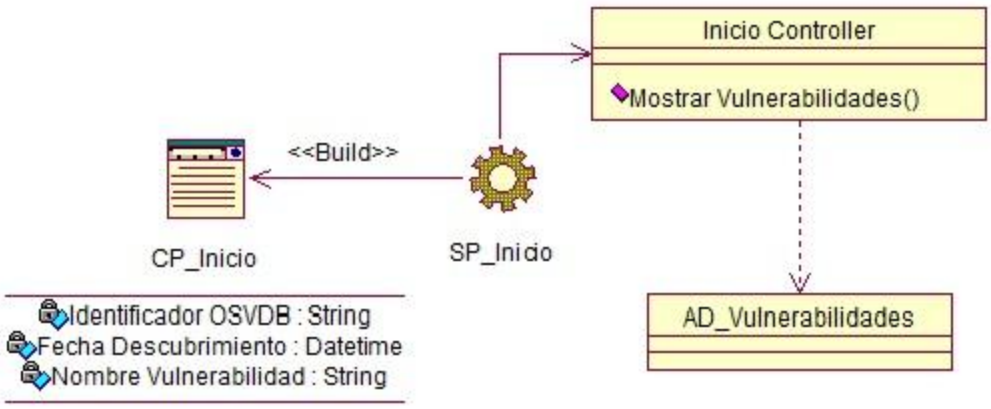


Figura 19: Diagrama de clases del diseño CU- Mostrar Vulnerabilidades.

Figura 20: Diagrama de clases del diseño CU- Enviar Reporte.

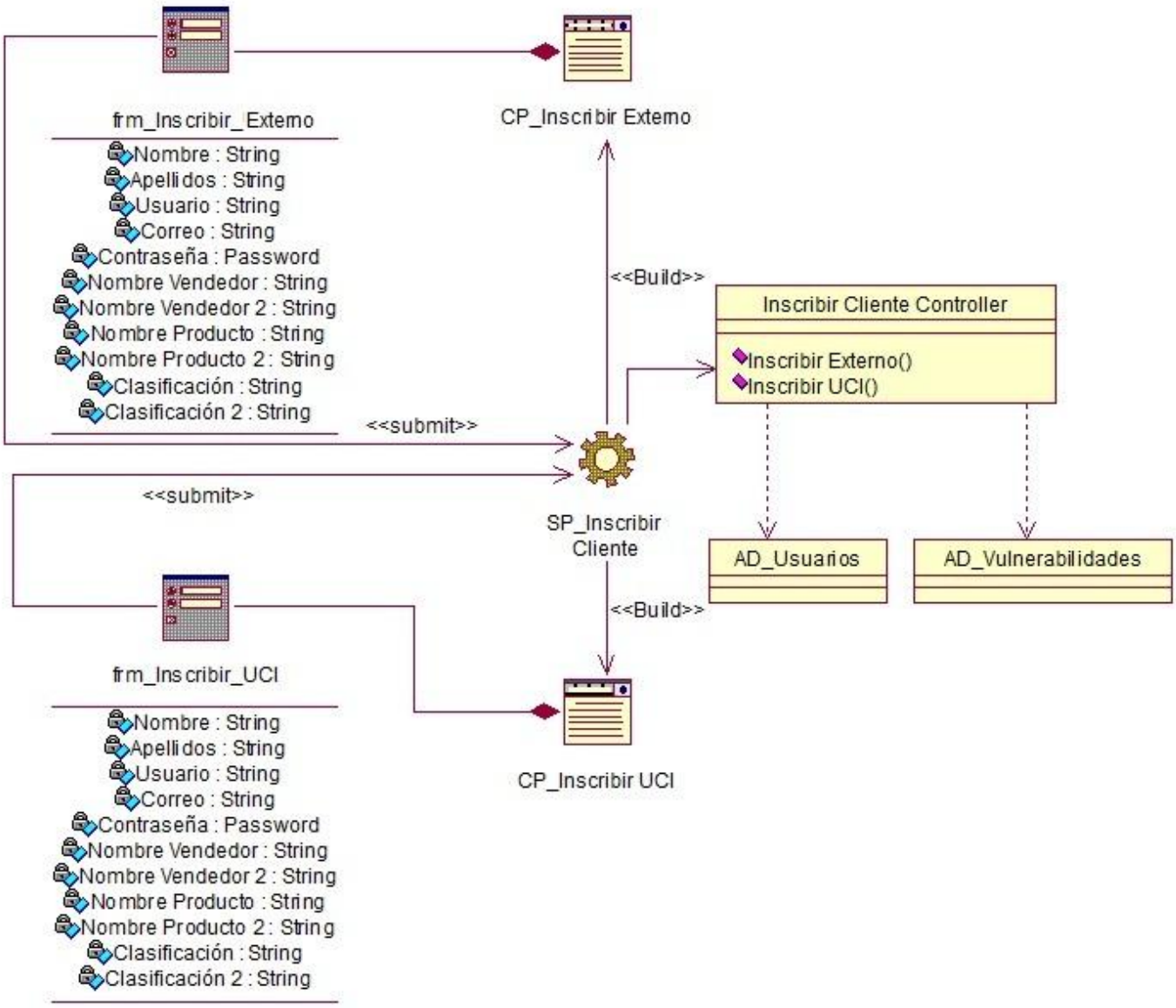


Figura 21: Diagrama de clases del diseño CU- Inscribir Usuario.

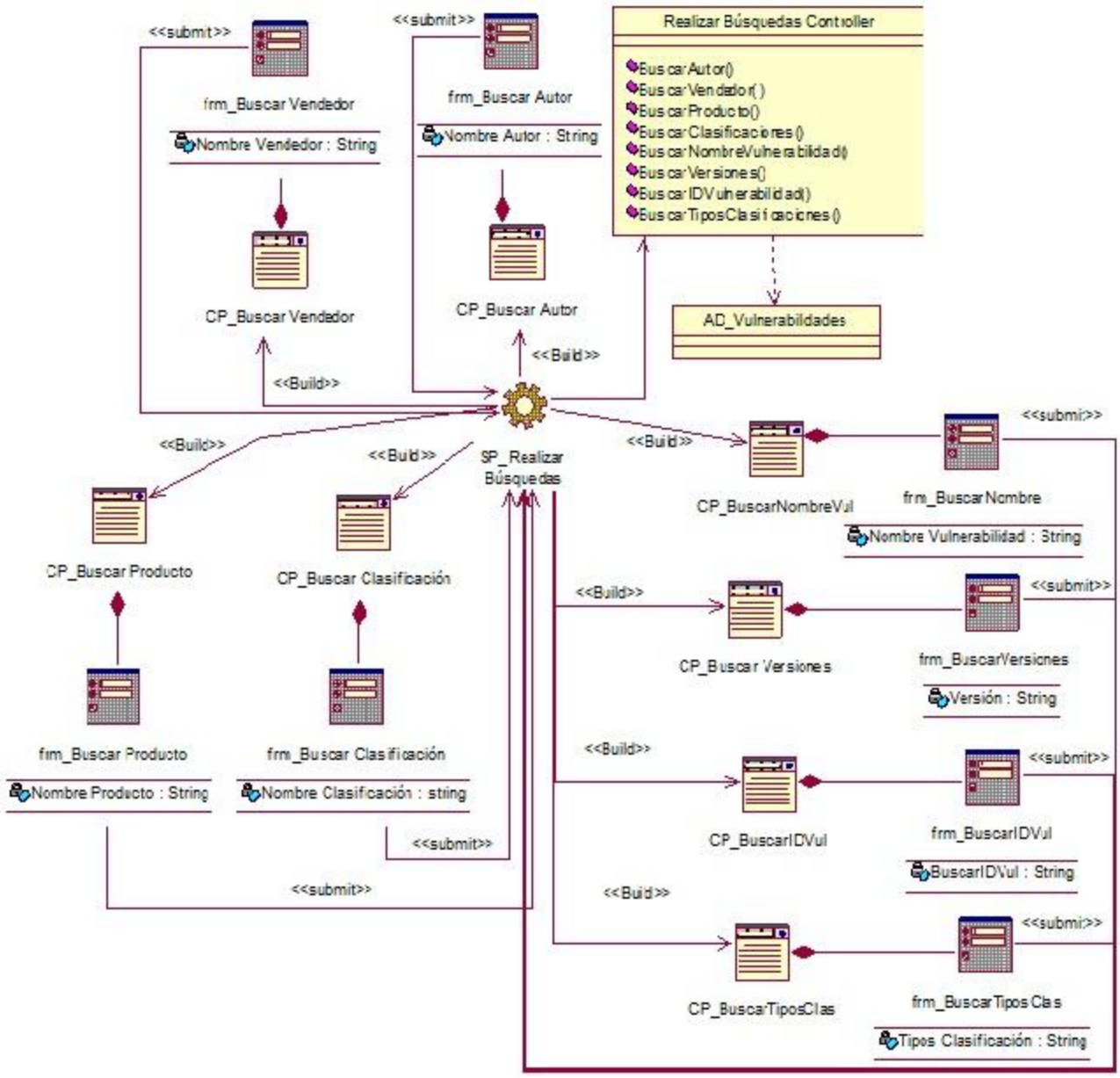


Figura 22: Diagrama de clases del diseño CU- Realizar Búsqueda.

3.3.3 Diagramas de Secuencia del Diseño.

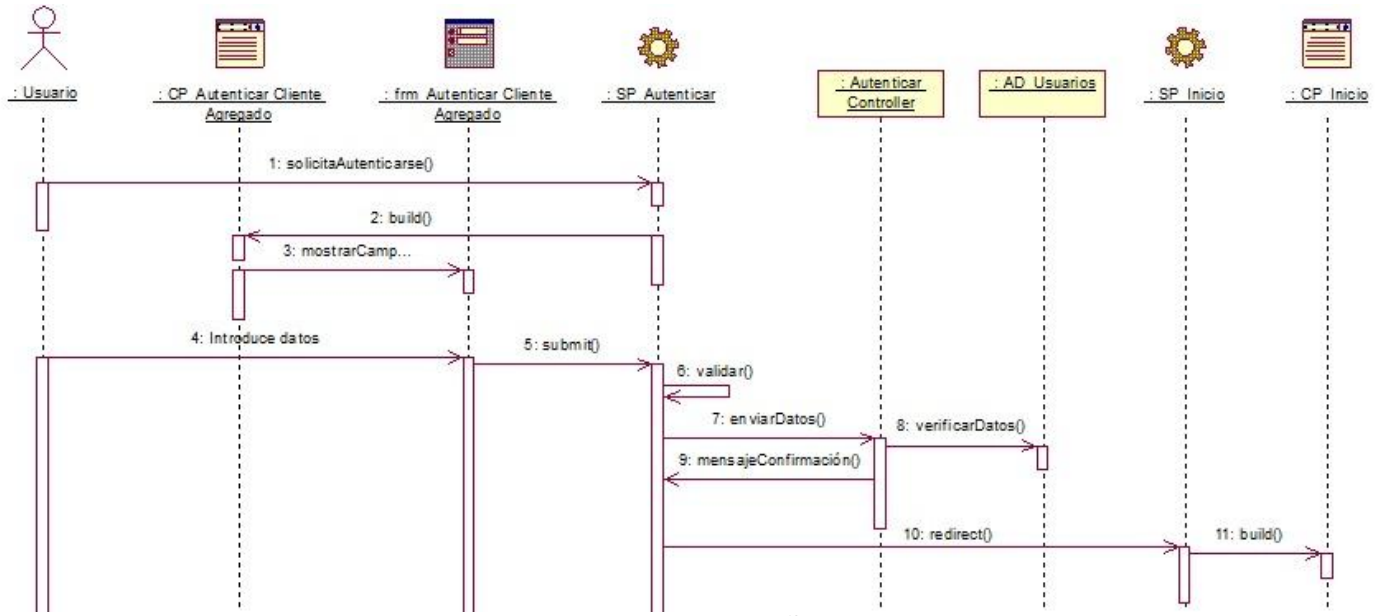


Figura 23: Diagrama de secuencia del diseño CU-Autenticar Usuario.

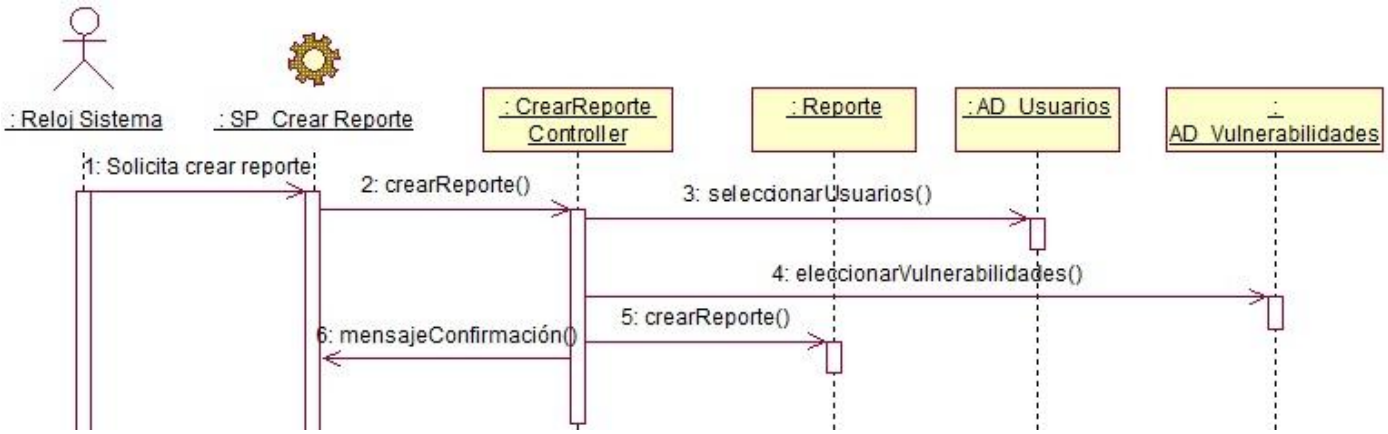


Figura 24: Diagrama de secuencia del diseño CU-Crear Reporte.

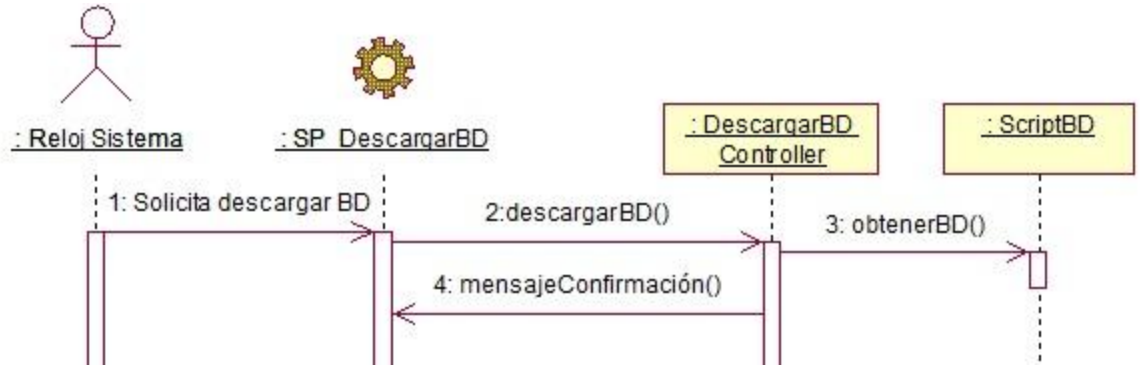


Figura 25: Diagrama de secuencia del diseño CU-Descargar Base de Datos.

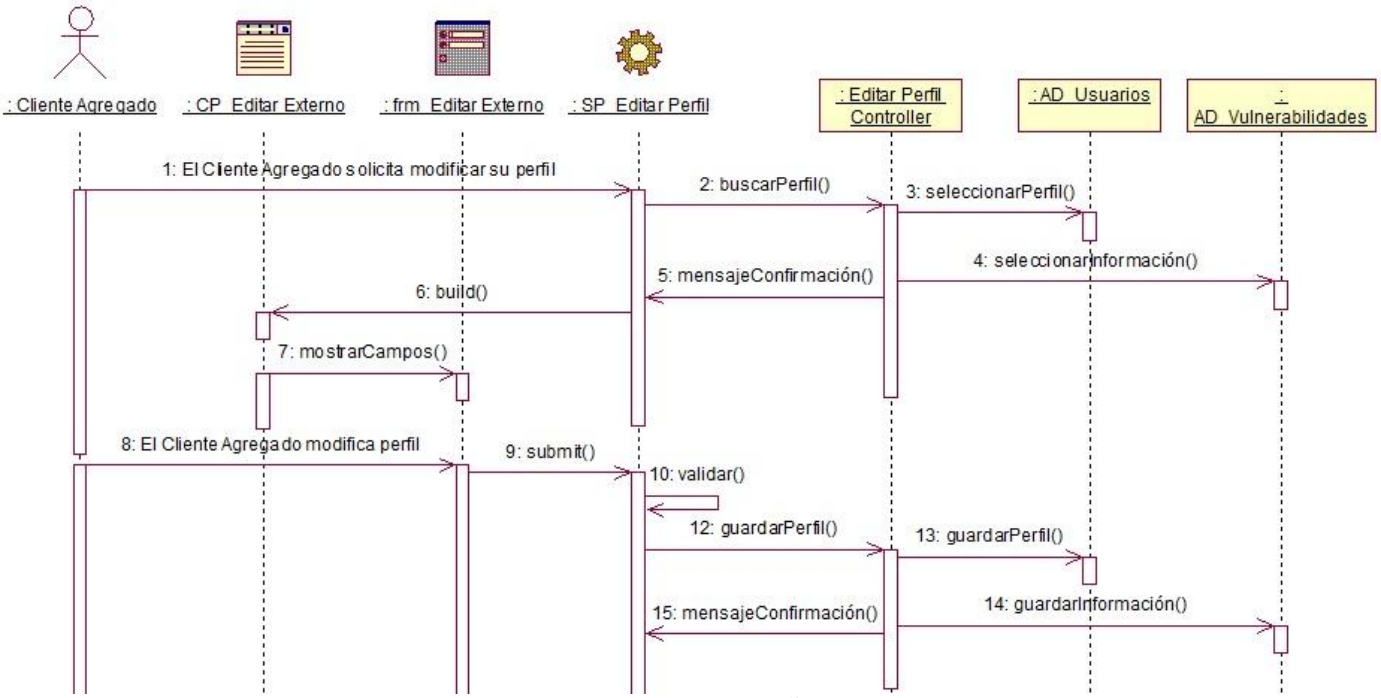


Figura 26: Diagrama de secuencia del diseño CU-Editar Perfil Externo.

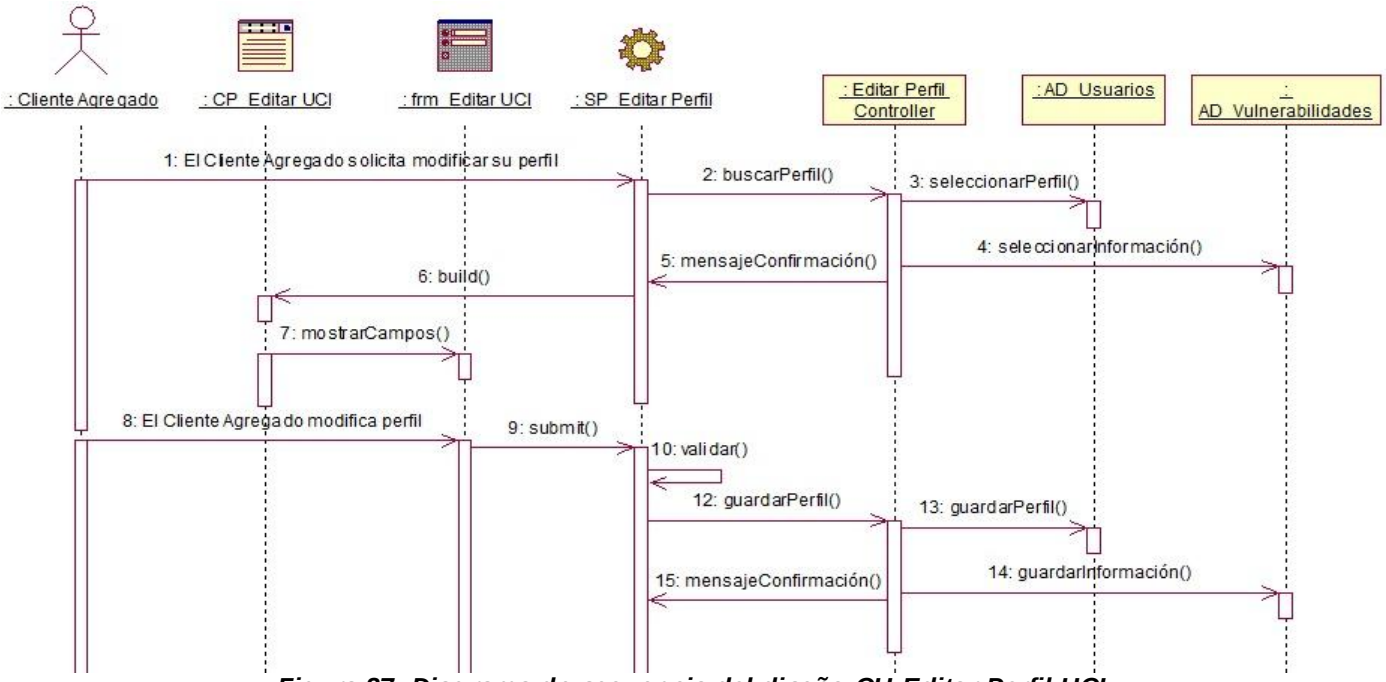


Figura 27: Diagrama de secuencia del diseño CU-Editar Perfil UCI.

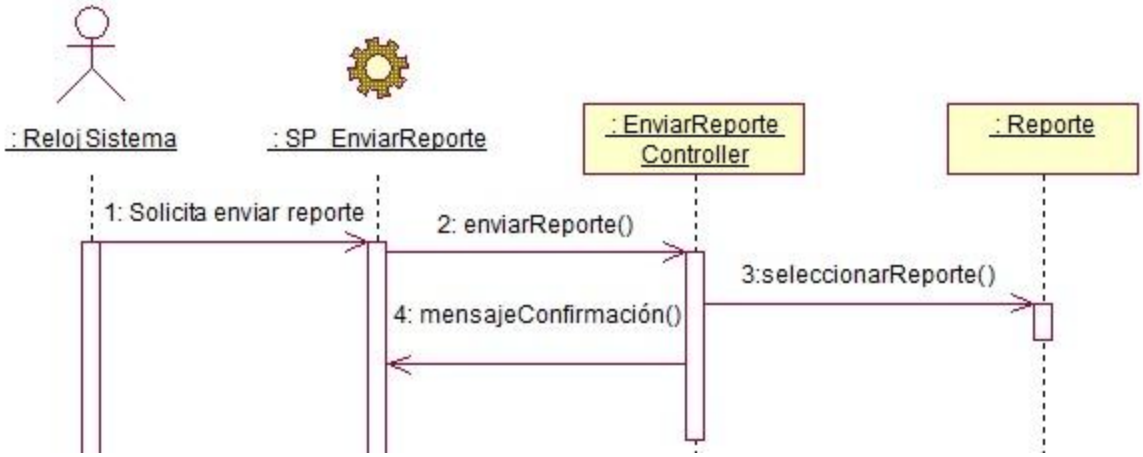


Figura 28: Diagrama de secuencia del diseño CU-Enviar Reporte.

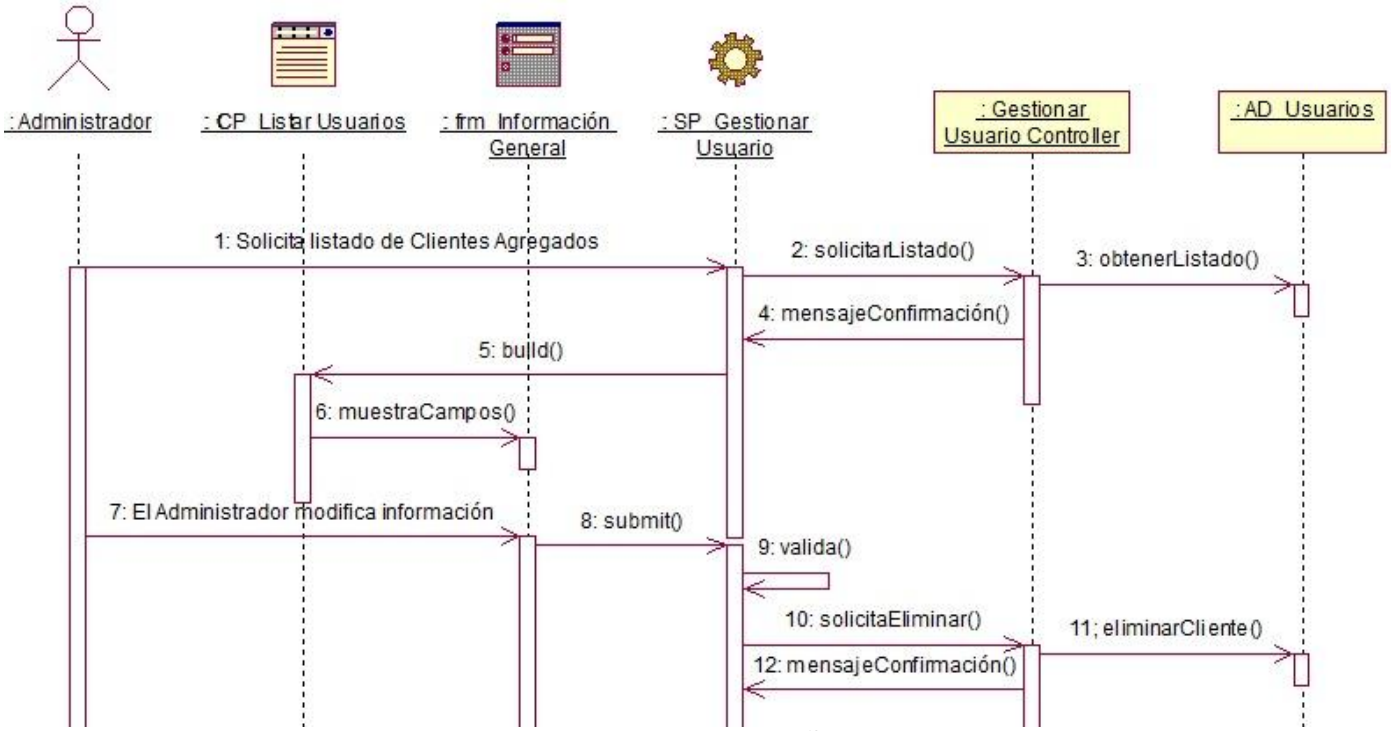


Figura 29: Diagrama de secuencia del diseño CU-Gestionar Usuario.

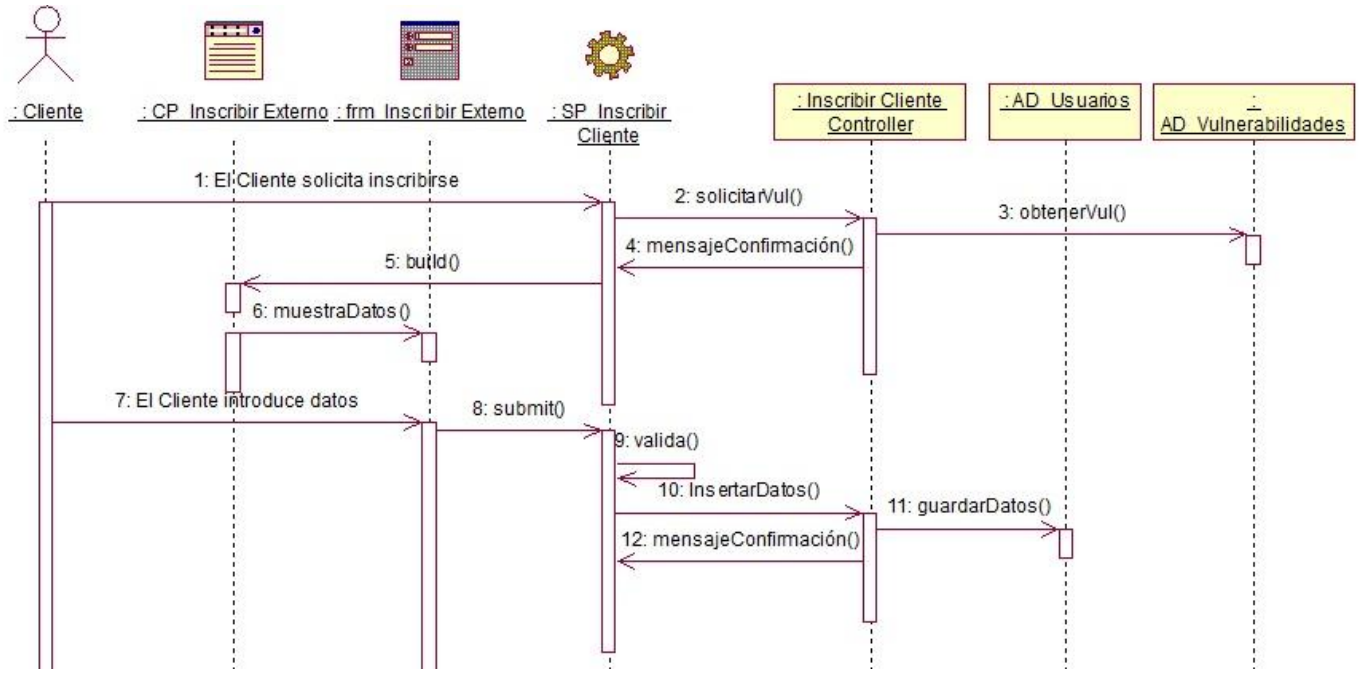


Figura 30: Diagrama de secuencia del diseño CU-Inscribir Externo.

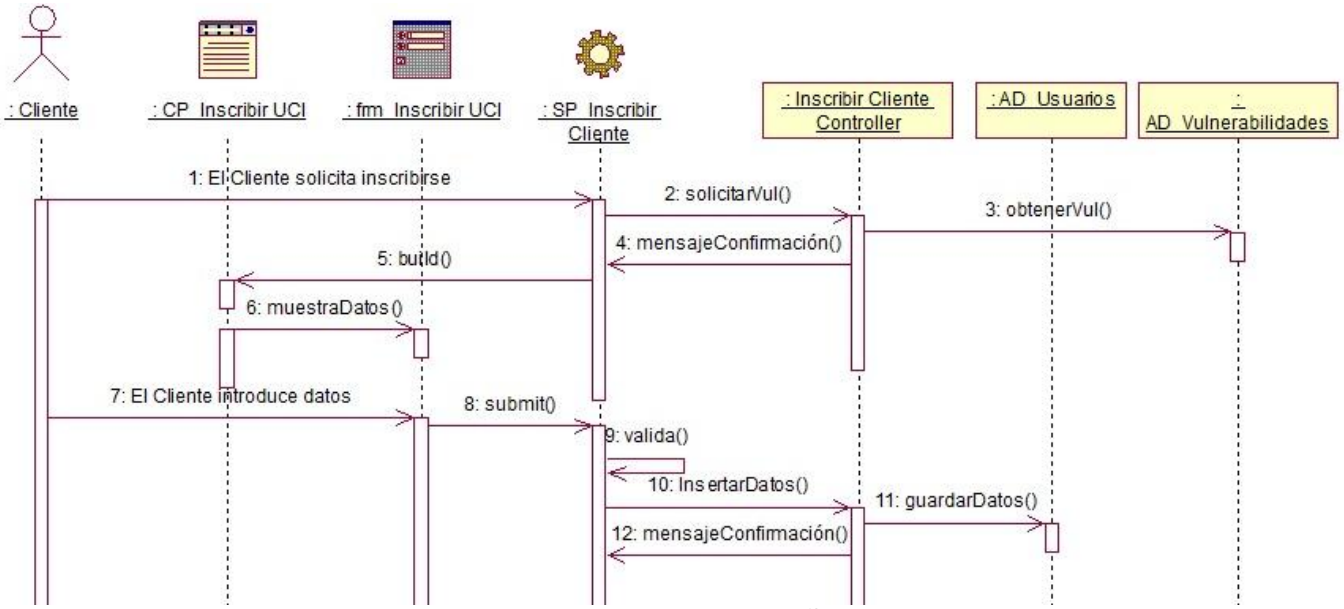


Figura 31: Diagrama de secuencia del diseño CU-Inscribir UCI.

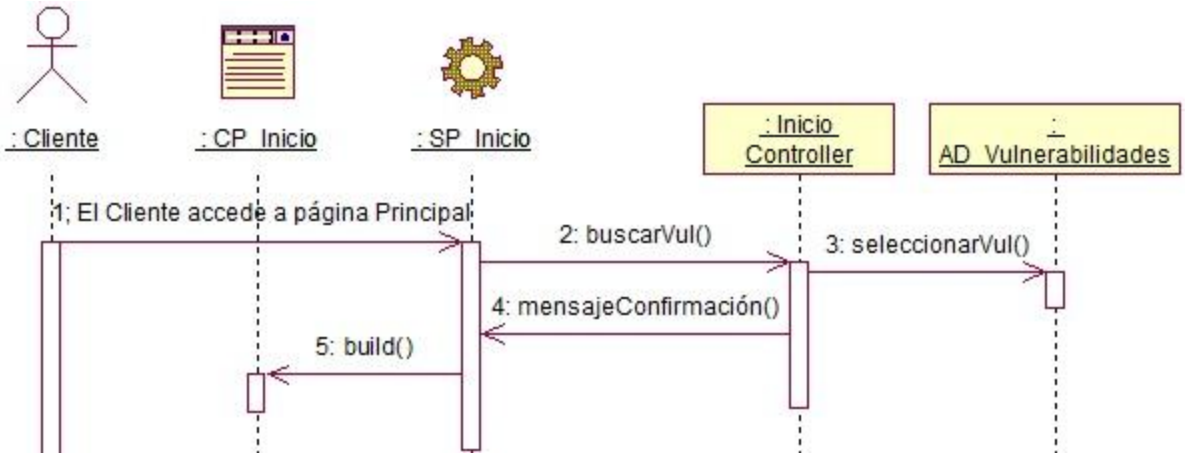


Figura 32: Diagrama de secuencia del diseño CU-Mostrar Vulnerabilidades.

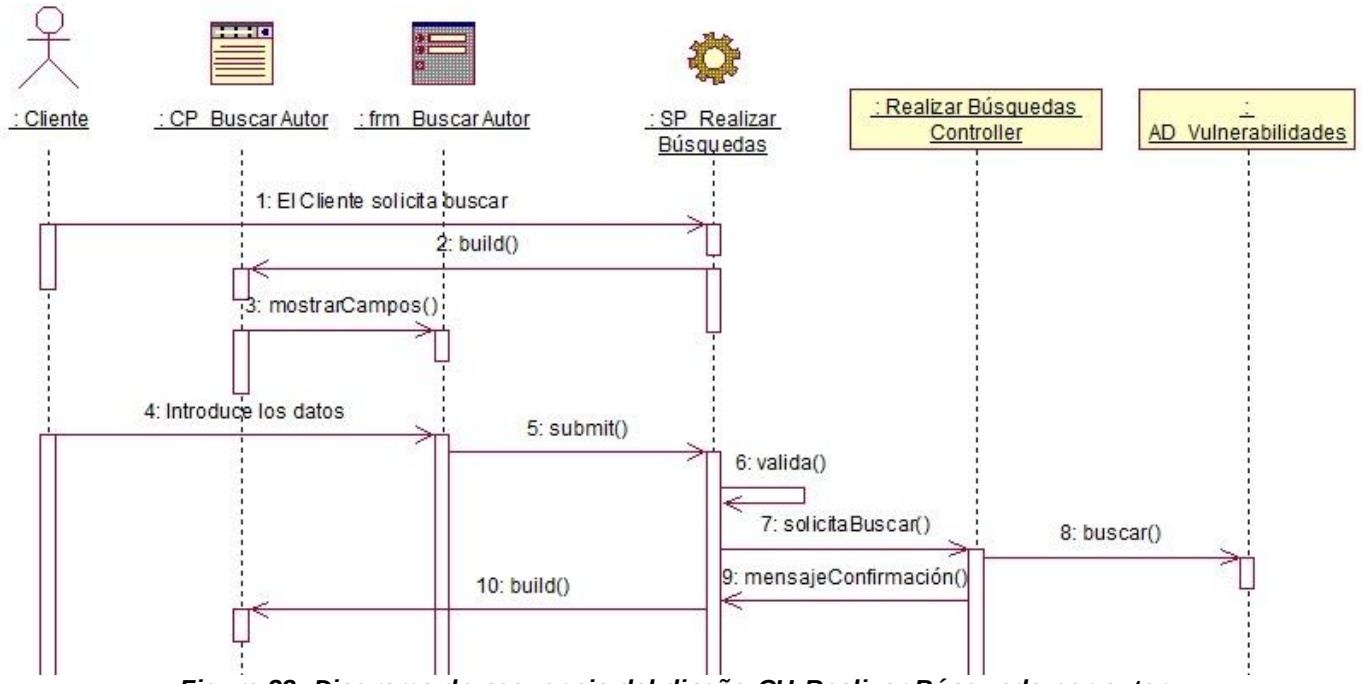


Figura 33: Diagrama de secuencia del diseño CU-Realizar Búsqueda por autor.

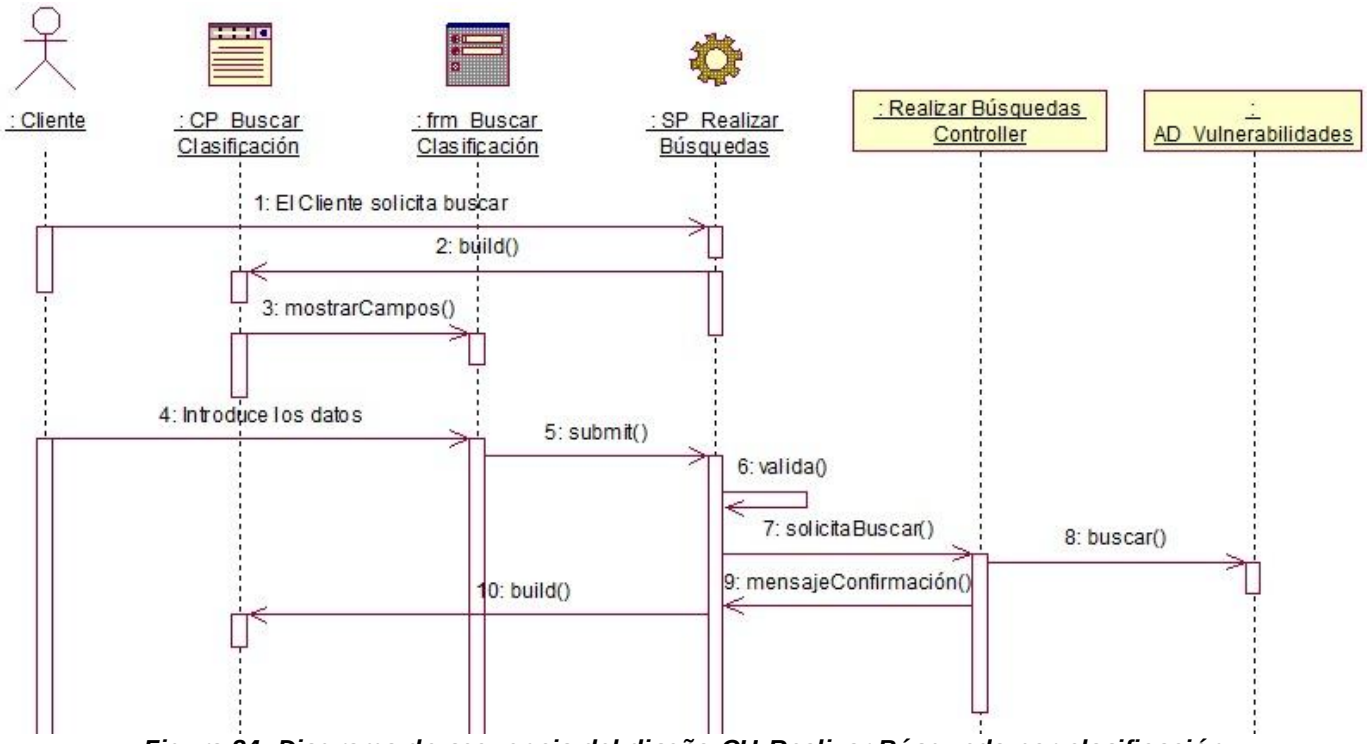


Figura 34: Diagrama de secuencia del diseño CU-Realizar Búsqueda por clasificación.

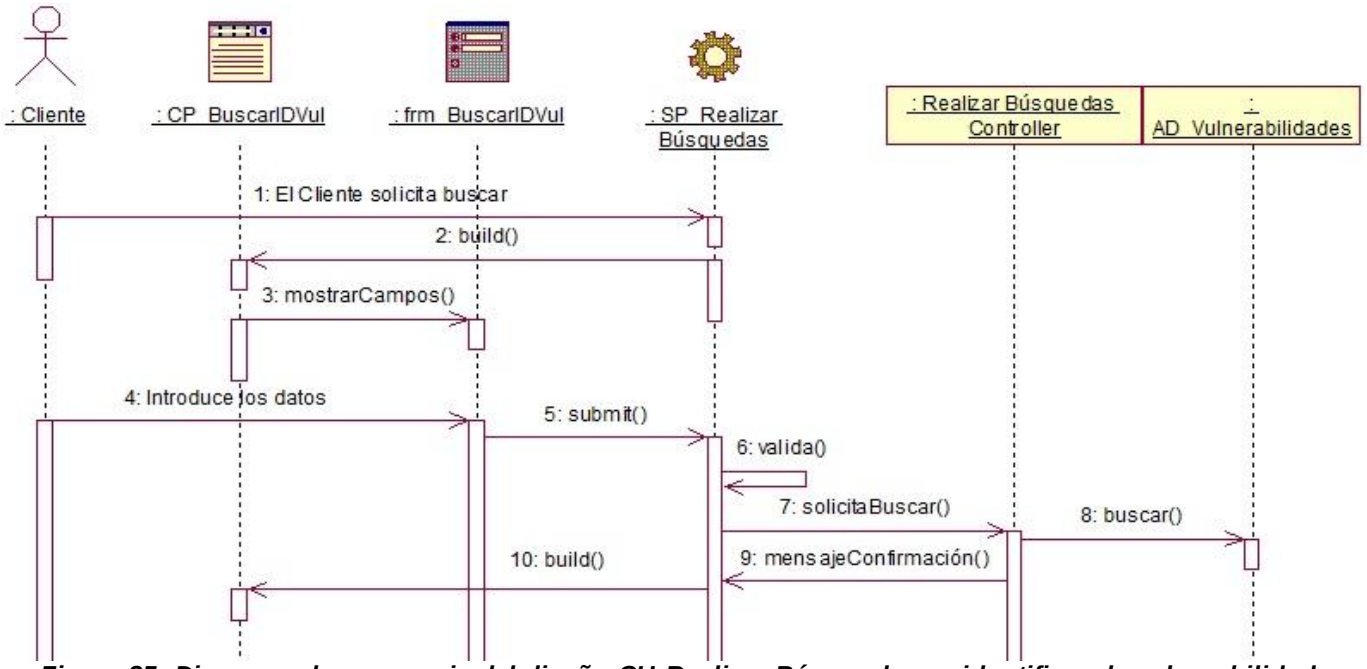


Figura 35: Diagrama de secuencia del diseño CU-Realizar Búsqueda por identificar de vulnerabilidad.

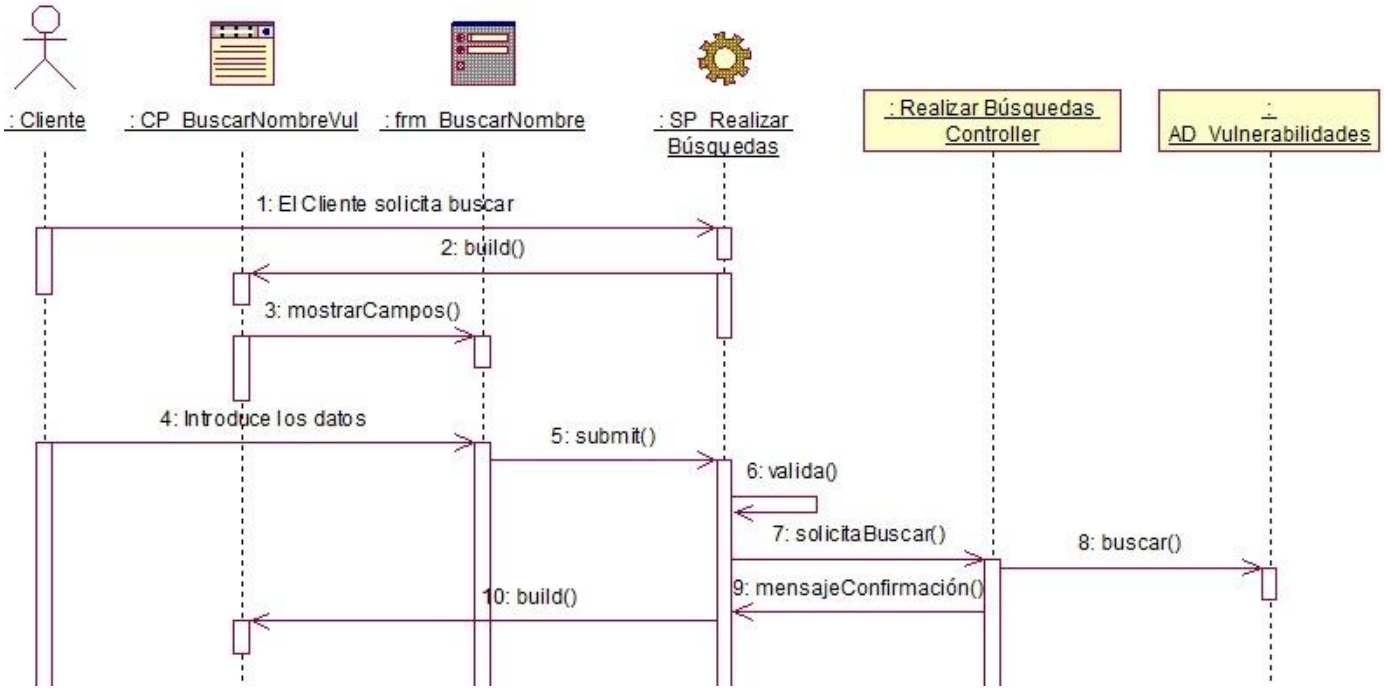


Figura 36: Diagrama de secuencia del diseño CU- Realizar Búsquedas por Nombre de Vulnerabilidad.

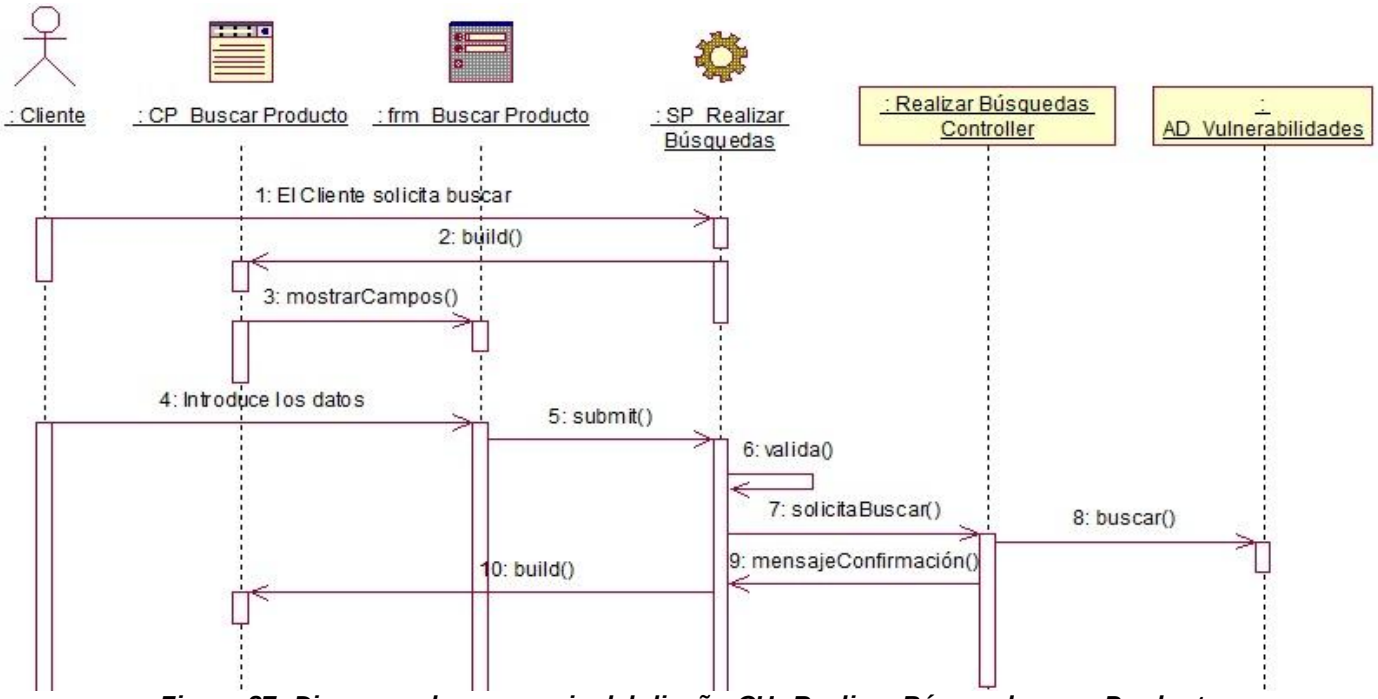


Figura 37: Diagrama de secuencia del diseño CU- Realizar Búsquedas por Producto.

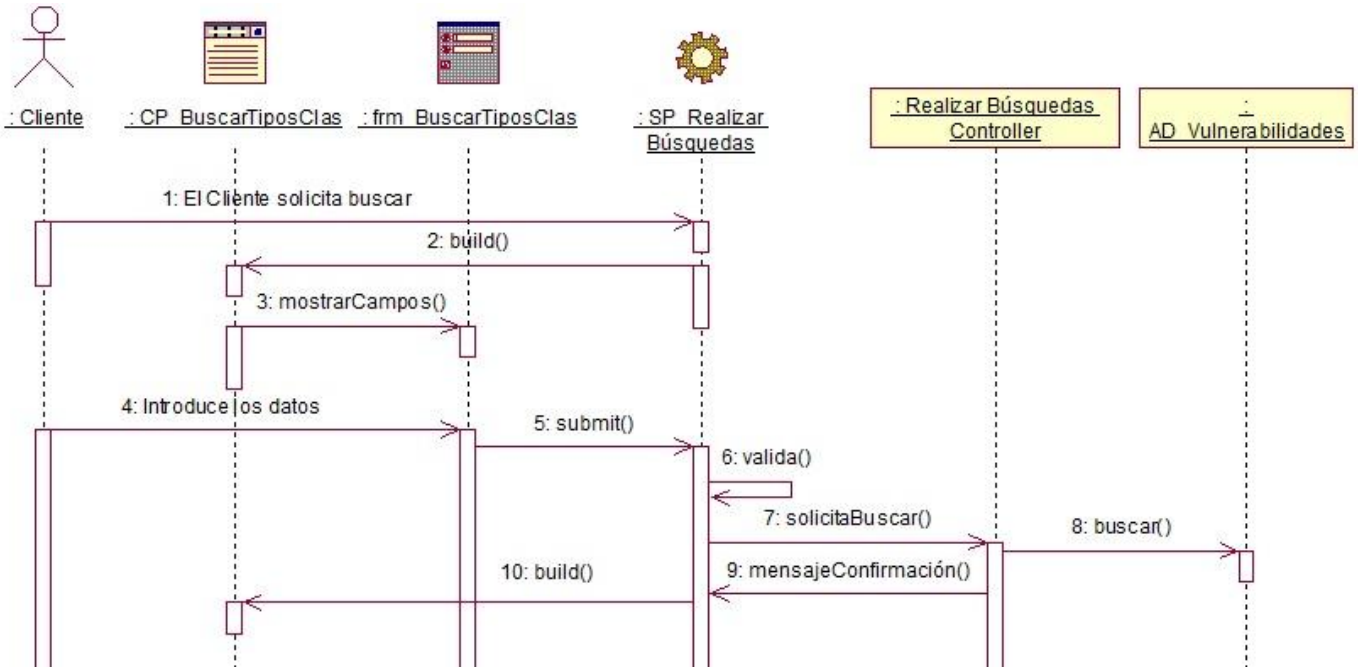


Figura 38: Diagrama de secuencia del diseño CU- Realizar Búsquedas por tipo de clasificación.

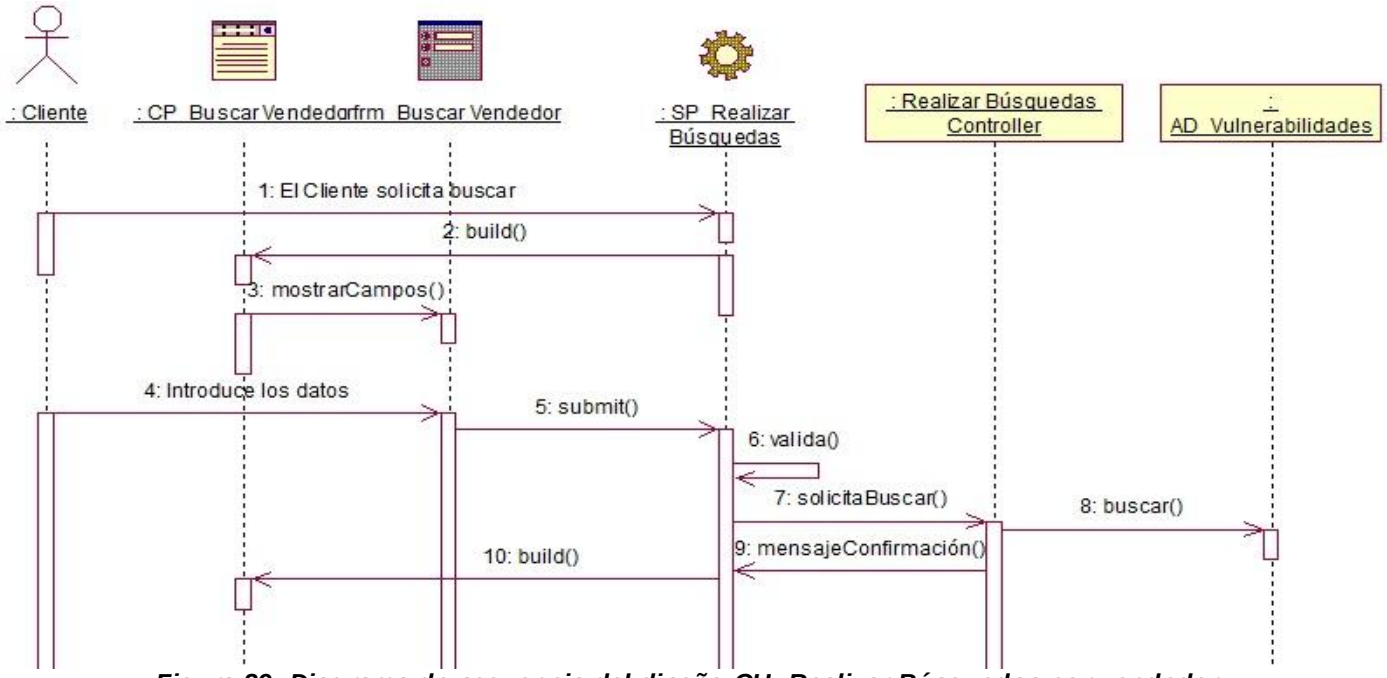


Figura 39: Diagrama de secuencia del diseño CU- Realizar Búsquedas por vendedor.

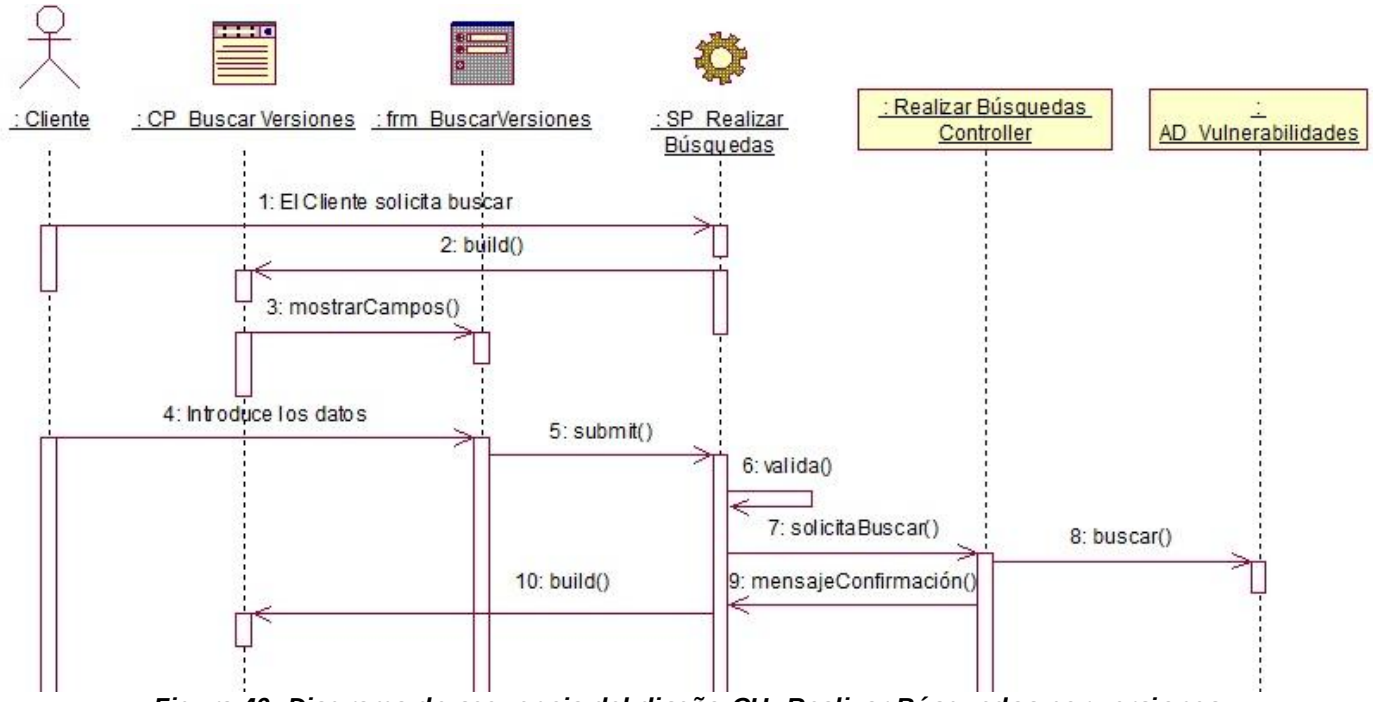


Figura 40: Diagrama de secuencia del diseño CU- Realizar Búsquedas por versiones.

3.5 Conclusiones

En este capítulo se hizo una introducción al análisis y el diseño de la aplicación. Se brindaron los diagramas de clases del análisis, los de clases del diseño y los de secuencia del diseño. Se describe la arquitectura del sistema así como los patrones de diseño utilizados.

CAPÍTULO 4: IMPLEMENTACIÓN

4.1 Introducción.

Teniendo ya los resultados del análisis y el diseño, la implementación se simplifica a implementar el sistema en término de componentes, es decir, ejecutables, código fuente y scripts. Aquí se desarrolla la arquitectura y el sistema como un todo. Además, se implementan las clases del diseño como componentes que contienen código fuente.

4.2 Diagrama de despliegue

Se puede decir que un diagrama de despliegue es un grafo de nodos unidos por conexiones de comunicación, donde un nodo puede contener instancias de componentes. En general, un nodo se entiende como una unidad de computación de algún tipo como es el caso de impresoras o la misma computadora (9).

En un diagrama de despliegue se muestran las relaciones físicas entre los componentes de hardware y de software, es decir, la configuración de los elementos de procesamiento en tiempo de ejecución y los componentes de software, que no son más, que los procesos que se ejecutan en ellos.

A continuación se muestra el diagrama de despliegue:

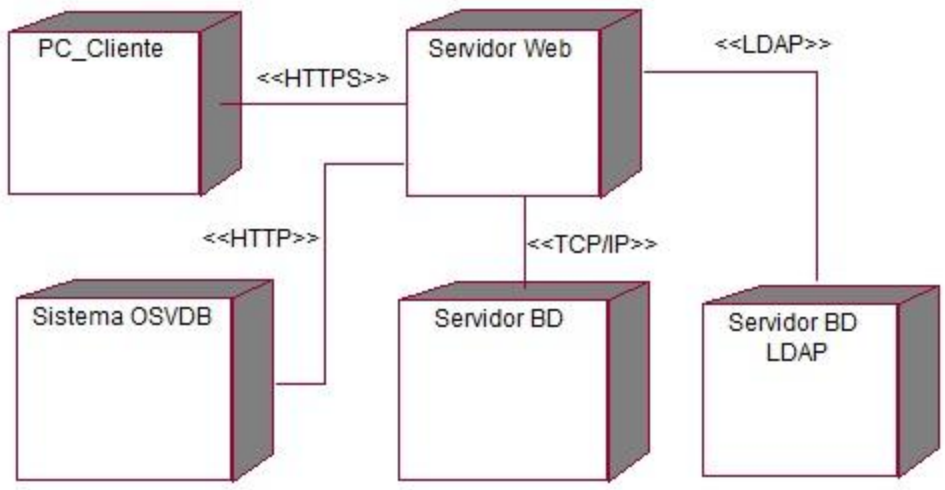


Figura 41: Diagrama de Despliegue.

4.3 Diagrama de componentes

Se puede decir que un componente es la parte modular de un sistema, desplegable y reemplazable que encapsula implementación y un conjunto de interfaces y proporciona la realización de los mismos. Un componente típicamente contiene clases y puede ser implementado por uno o más artefactos.

Los diagramas de componentes estructuran el modelo de implementación en términos de subsistemas de implementación y muestra las relaciones entre los elementos de implementación. Estos diagramas muestran la estructura de alto nivel del modelo de implementación.

Un diagrama de componentes representa las dependencias entre componentes de software, incluyendo componentes de código fuente, de código binario y ejecutable. Algunos componentes existen en tiempo de compilación, algunos en tiempo de enlace y algunos en tiempo de ejecución, otros en varias de éstas (9).

A continuación se muestran los diagramas de componentes:

Figura 42: Diagrama de Componentes Base de Datos.

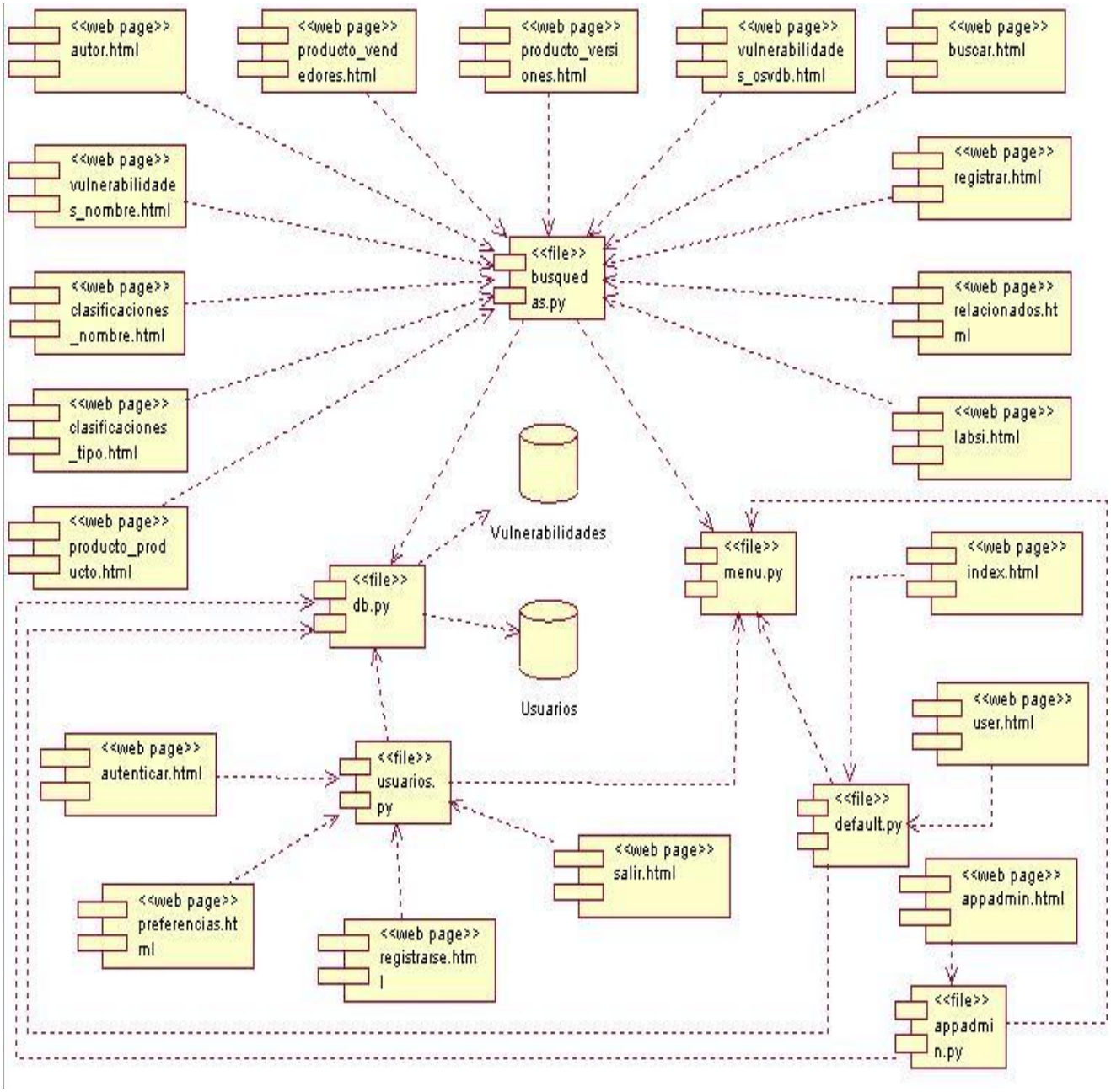


Figura 43: Diagrama Componentes Modelo, Vista, Controlador.

4.4 Conclusiones

En el presente capítulo se abordó el tema de la implementación del sistema. Se muestra el diagrama de despliegue. También los diagramas de componentes. Todo esto representa la forma en que está construido el sistema.

CAPÍTULO 5: FACTIBILIDAD DEL SISTEMA

5.1 Introducción

En el presente capítulo se hará un estudio sobre la factibilidad del sistema. El estudio de factibilidad es el análisis de una empresa para determinar si el negocio que se propone será factible, y en cuales condiciones se debe desarrollar para que sea exitoso. Factibilidad es el grado en que lograr algo es posible o las posibilidades que se tiene de lograrse.

Iniciar un proyecto de producción o fortalecerlo significa invertir recursos como tiempo, dinero y equipos. Como los recursos siempre son limitados, es necesario tomar una decisión; las buenas decisiones solo pueden ser tomadas sobre la base de evidencias y cálculos correctos, de manera que se tenga mucha seguridad de que el negocio se desempeñará correctamente y que producirá ganancias.

Comprender el concepto de proyecto es muy importante para el desarrollo de criterios y comportamientos, principalmente si se trata de propiciar cambios culturales y de mentalidad. Esto incluye los conceptos de ahorro, generación de excedentes e inversiones, imprescindibles para desarrollar proyectos sostenibles.

Para evaluar la complejidad de un sistema de software existen varias técnicas como son: el método de puntos de función, método COCOMO, puntos de caso de uso y model driven architecture. Los cuales han surgido por la misma necesidad, minimizar el impacto de la denominada “crisis del software”, provocada por la complejidad de este proceso, tanto en el desarrollo propio de los sistemas como en la gestión de los mismos. En las últimas décadas se han desarrollado y perfeccionado estas técnicas para la estimación de proyectos de software.

Para la realización de este análisis se utiliza el método de Puntos de Casos de Uso, fue introducido por Gustav Karner en 1993, y supervisado por Ivar Jacobson.

5.2 Método de estimación por Casos de Uso

El método utiliza los actores y casos de uso identificados para calcular el esfuerzo que costará desarrollarlos. A los casos de uso se les asigna una complejidad basada en transacciones, que no es más que pares de pasos acción-usuario-respuesta del sistema, de los escenarios de los casos de uso. A los actores se les asigna una complejidad basada en el tipo de actor, es decir, si son interfaces con usuarios o

si son interfaces con otros sistemas. También se utilizan factores de entorno y de complejidad técnica para mejorar el resultado.

Una vez asignada complejidad a actores y casos de uso, y establecidos los factores técnicos y de entorno, se calculan los puntos de caso de uso no ajustados o UUCP, el factor de complejidad técnica o TCF y el factor de entorno o EF. Con ellos se calculan los puntos de caso de uso o UCP, que finalmente se traducen a esfuerzo en horas-hombre con un sencillo cálculo.

5.2.1 Cálculo de Puntos de Caso de Uso sin ajustar.

Al inicio de un proyecto de software, cuando apenas se conocen los casos de uso y sus actores asociados, se puede proyectar una breve descripción de cada caso de uso, en el cual se describe de forma la funcionalidad que éste debe brindar.

Los puntos de caso de uso sin ajustar son de utilidad para tener una idea un poco más precisa de la dificultad de los casos de uso e interfaces, tomando en cuenta los pesos de los actores y los pesos de los casos de uso. Por lo que la fórmula para calcular esto sería:

UUCP = UAW + UUCW

El significado de estas siglas es:

UUCP: puntos de caso de uso sin ajustar.

UAW: factor de peso de los actores sin ajustar.

UUCW: factor de peso de los casos de uso sin ajustar.

Una vez calculado tanto el **UAW** como el **UUCW**, el resultado sería:

UUCP = 3 + 55

UUCP = 58

5.2.2 Factor de Peso de los Actores sin ajustar.

Consiste en la evaluación de la complejidad de los actores con los que tendrá que interactuar el sistema. Este puntaje se calcula determinando si cada actor u otro sistema, además evalúa la forma en la que interactúa con el caso de uso, y la cantidad de actores de cada tipo. La fórmula para esto sería:

UAW = Sum (cantidadDeUnTipoDeActor * Factor)

Para realizar esta operación sería necesario contar cuantos actores de cada tipo existen en el sistema, este representaría el valor cantidadDeUnTipoDeActor en la formula y se tiene que multiplicar por el valor que tenga su factor correspondiente, para obtener el resultado por cada tipo de actor. Una vez terminado esto se procede a sumar cada producto para obtener el **UAW**.

Tipo	**Descripción**	**Peso**	**Cantidad**	**Cant * Peso**
Simple	Otro sistema que interactúa con el sistema a desarrollar mediante una interfaz de programación. (Aplication Programming Interface).	1	0	0 * 1
Medio	Otro sistema que interactúa con el sistema a desarrollar mediante un protocolo o interfaz basada en texto.	2	0	0 * 2
Complejo	Una persona que interactúa con el sistema mediante una interfaz gráfica.	3	1	1 * 3
Total:				3

Tabla 27: Cálculo del factor de peso de los actores sin ajustar.

5.2.3 Factor de Peso de los Casos de Uso sin ajustar.

Para determinar el nivel de complejidad se puede realizar mediante dos métodos: basado en transacciones o basado en clases de análisis. Una transacción es un conjunto de actividades atómicas, lo que quiere decir que se ejecutan todas o no se ejecuta ninguna.

Basado en transacciones: Toma en cuenta el número de transacciones que se pueden realizar en un caso de uso y lo evalúa.

Basado en clases de análisis: Toma en cuenta el número de clases que tiene un caso de uso y lo evalúa.

Independiente del camino utilizado para determinar el tipo de caso de uso, la formula es la misma:

UUCW = Sum (cantidadDeUnTipoDeCasoUso * Factor)

Para realizar esta operación se debe contar cuantos casos de uso de cada tipo hay en el sistema y esta cantidad se sustituiría en el campo nombrado cantidadDeUnTipoDeCasoUso y se multiplica por el valor que tenga su factor correspondiente, para obtener el resultado por cada tipo de caso de uso a la vez. Una vez hecho esto se suma cada producto para obtener el factor de peso de los casos de uso sin ajustar.

Tipo de Caso de Uso	Descripción	Factor
Simple	3 transacciones o menos.	5
Medio	4 a 7 transacciones.	10
Complejo	Más de 7 transacciones	15

Tabla 28: Peso de las Transacciones.

No.	Nombre de Caso de Uso	Cantidad de Transacciones	Tipo	Peso	Cant * Peso
1	Gestionar Usuarios	4	Medio	10	7 * 5
2	Autenticar	2	Simple	5	
3	Mostrar Vulnerabilidades	2	Simple	5	
4	Realizar Búsquedas	3	Simple	5	2 * 10
5	Crear Reporte	2	Simple	5	

6	Enviar Reporte	4	Medio	10	
7	Inscribir Usuario	3	Simple	5	0 * 15
8	Actualizar Base de Datos	3	Simple	5	
9	Editar Perfil	2	Simple	5	
Total:					55

Tabla 29: Factor de Peso de los Casos de Uso sin ajustar.

5.2.4 Puntos de Casos de Uso ajustados

Para esto se utilizan las siglas **UCP** y se obtiene al multiplicar el **UUCP**, el **TCF** y el **EF** quedando la operación de la siguiente forma:

UCP = UUCP * TCF * EF

Donde el significado de estas siglas es:

UCP: Puntos de Casos de Uso ajustados.

UUCP: Puntos de Casos de Uso sin ajustar.

TFC: Factores Técnicos.

EF: Factores de Ambiente.

Una vez conocidos los valores de **UUCP, TCF** y **EF**, el valor de los puntos de Casos de Uso ajustados seria:

UCP = 58 * 0.99 * 0.935

UCP = 53.6877

5.2.5 Factores de Complejidad Técnica.

Los Factores de Complejidad técnica se componen de 13 puntos que evalúan la complejidad de los módulos del sistema que se desarrolla, cada uno de estos factores tiene un peso definido con los cuales se obtendrán puntos ponderados para cada uno de ellos, según la valoración que se le asigne.

Para calcular el factor de complejidad técnica se utilizan las siguientes fórmulas:

TFactor = Sum (Valor * Peso)

TCF = 0.6 + (0.01 * TFactor)

Para realizar este cálculo, se debe evaluar cada factor, asignándole un valor como se menciona anteriormente, después se multiplican y se suman cada producto para obtener el TFactor. Luego, se debe seguir la segunda fórmula multiplicando el TFactor por 0.01 y sumar el resultado a 0.6 obteniendo como resultado el TCF.

Factor	Descripción.	Peso
T1	Sistema distribuido.	2
T2	Objetivos de performance o tiempo de respuesta.	1
T3	Eficiencia del usuario final.	1
T4	Procesamiento interno complejo.	1
T5	Código reutilizable.	1
T6	Facilidad de instalación.	0.5
T7	Facilidad de uso.	0.5
T8	Portabilidad.	2

T9	Facilidad de cambio.	1
T10	Concurrencia	1
T11	Incluye objetivos especiales de seguridad.	1
T12	Provee acceso directo a terceras partes.	1
T13	Requiere facilidades especiales de entrenamiento a usuarios.	1

Tabla 30: Peso de los Factores de Complejidad Técnica.

Cada uno de estos factores de complejidad técnica se evalúa según la siguiente escala:

Irrelevante: De 0 a 2

Medio: De 3 a 4

Esencial: 5

Factor	Peso	Valor	Comentario	Valor * Peso
T1	2	4	El sistema es distribuido.	8
T2	1	5	Se requiere que sistema tenga buen rendimiento.	5
T3	1	1	No hay restricciones de eficiencia del usuario final.	1
T4	1	1	No hay procesamiento interno complejo.	1
T5	1	4	El código debe ser reutilizable.	4
T6	0.5	3	El sistema debe ser fácil de instalar	1.5
T7	0.5	5	Debe ser fácil de usar.	2.5

T8	2	0	El sistema debe ser portable.	0
T9	1	3	Debe ser flexible ante los cambios.	3
T10	1	4	Debe presentar concurrencia.	4
T11	1	5	El sistema gestiona información de carácter limitado	5
T12	1	3	Provee acceso directo a terceras partes	3
T13	1	1	No se requieren facilidades especiales de entrenamiento de usuarios.	1
Total:				39

Tabla 31: Cálculo del Valor de TFactor.

Calculado el TFactor, el resultado del Factor de Complejidad Técnica seria:

TCF = 0.6 + (0.01 * 39)

TCF = 0.99

5.2.6 Factores de Ambiente.

Los factores sobre los cuales se realiza la evaluación son 8 puntos, que están relacionados con las habilidades y experiencia del grupo de personas involucradas con el desarrollo del proyecto. Estos factores se muestran a continuación:

Factor	**Descripción**	**Peso**
E1	Familiaridad con el modelo de proyecto utilizado.	1.5
E2	Experiencia en la aplicación.	0.5
E3	Experiencia en orientación a objetos.	1

E4	Capacidad del analista líder.	0.5
E5	Motivación.	1
E6	Estabilidad de los requerimientos	2
E7	Personal part-time	-1
E8	Dificultad del lenguaje de programación	-1

Tabla 32: Peso de los Factores Ambientales.

Cada uno de estos factores se debe calificar con un valor de 0 a 5. Las fórmulas para este punto son:

EFactor = Sum (Valor * Peso)

EF = 1.4 + (-0.03 * EFactor)

Para obtener el **EFactor** se debe sumar todos los productos obtenidos al multiplicar e peso de cada punto por el valor asignado, después se multiplica por -0.03 y se le suma el 1.4. De esta forma, es que se obtiene el peso de los factores de ambiente.

Factor	Peso	Valor	Comentario	Valor * Peso
E1	1.5	2	El grupo no está familiarizado con el modelo de proyecto.	3
E2	0.5	2	No hay mucha experiencia en la aplicación.	1
E3	1	5	El grupo ha programado orientado a objetos.	5
E4	0.5	3	El analista principal comenzó recientemente en el proyecto.	1.5
E5	1	3	El grupo no está altamente motivado.	3
E6	2	3	Se esperan cambios.	6

E7	-1	0	El equipo es full-time.	0
E8	-1	4	Se usará el lenguaje de programación Python orientado a objetos, no estudiado, pero sin gran curva de aprendizaje.	-4
Total:				15.5

Tabla 33: Cálculo del Valor de EFactor.

Calculado el EFactor, el resultado del Factor de Ambiente sería:

EF = 1.4 + (-0.03 * 15.5)

EF = 0.935

5.2.7 Esfuerzo Horas-Hombres.

Este cálculo se realiza con el fin de tener una aproximación del esfuerzo, pensando solo en el desarrollo según las funcionalidades de los casos de uso. Está basado en los factores ambientales y se calcula de la siguiente manera, primero se debe contar la cantidad de factores de ambientes del E1 al E6 que tienen una puntuación menor a 3 y la cantidad de factores de ambiente del E7 al E8 que son mayores que 3. Este resultado se evalúa según la siguiente tabla:

Horas-Hombres	**Descripción**
20	Si el valor es <= 2
28	Si el valor es <= 4
36	Si el valor es >=5

Tabla 34: Cantidad de Horas-Hombres según el valor.

El esfuerzo en Horas-Hombres viene dado por:

E = UCP * CF

Donde el significado de estas siglas es el siguiente:

E: Esfuerzo estimado en Horas-Hombres.

UCP: Puntos de Casos de Uso ajustados.

CF: Horas-Hombres.

Al realizar la multiplicación del **UCP** por las Horas-Hombres, se consigue un esfuerzo estimado, que representa una parte del total del esfuerzo del proyecto.

Por tanto, el esfuerzo en Horas-Hombres seria:

E = 53.6877 * 28

E = 1503.2556

5.3 Costos

Salario medio de la fuerza de trabajo: $100.00

Cantidad de Trabajadores: 1

Al tener un estimado de 1503.2556 Horas-Hombre, este valor es dividido entre 6 horas, que es la cantidad real de horas que dedica cada trabajador del módulo, para calcular la cantidad de días de trabajo por hombres que se necesitan para realizar el módulo. Al realizar esta operación se llega a la conclusión que el módulo para su realización requiere de 251 días de trabajo por cada integrante del mismo. Por lo que un trabajador necesitaría aproximadamente 8 meses. El costo final del módulo es de:

CTM = SM * CT * DP

Donde el significado de las siglas sería:

CTM: Costo Total del módulo.

SM: Salario Medio.

CT: Cantidad de Trabajadores.

DP: Duración del Proyecto (Meses).

Por lo tanto, el costo total del módulo sería:

CTM = 100 * 1 * 9

CTM = 800

5.4 Beneficios tangibles e intangibles

El desarrollo del Sistema de Notificación de Vulnerabilidades de Tecnologías y Sistemas Informáticos del Laboratorio de Seguridad Informática de la Facultad 2 de la Universidad de las Ciencias Informáticas aporta para toda la comunidad universitaria un servicio que ayuda a la seguridad, así como un gran ahorro económico. El mayor aporte de este sistema es el importante servicio de notificación de vulnerabilidades, lo cual contribuirá a elevar la seguridad de las redes

Otro gran aporte que brinda este sistema, es que los existentes, se encuentran disponibles en Internet y la gran mayoría de los mismos son propiedad de empresas norteamericanas, conociendo lo difícil que es el acceso a Internet por las limitaciones impuestas a nuestro país, este servicio es una alternativa viable para todos aquellos usuarios que no tengan conexión a Internet, pero si a la Intranet.

5.5 Análisis de costos y beneficios

Teniendo en cuenta el elevado costo de la creación de software a nivel mundial debido a los precios que son puestos por las grandes compañías creadoras del mismo y que sistemas similares al presentado son codiciados por diferentes usuarios y a precios más elevados como se muestra en la fundamentación teórica de este trabajo. El soporte a la aplicación será realizado por personas pertenecientes a la misma organización que lo utilice, concluyendo que la implantación del Sistema de Notificación de Vulnerabilidades de Tecnologías y Sistemas Informáticos, es factible.

5.6 Conclusiones

En este capítulo se realizó un estudio de factibilidad para el sistema que se presenta, lo cual permitió llegar a la conclusión que el sistema a implementar resulta factible. Se analizaron los beneficios tangibles

e intangibles que proporciona la implementación del Sistema de Notificación de Tecnologías y Sistemas Informáticos en la Universidad de las Ciencias Informáticas, así como los costos que presuponen herramientas similares con respecto a la que se presenta y la factibilidad de esta para la implantación en dicha institución.

CONCLUSIONES

Una vez concluida la investigación propuesta, se puede asegurar que se cumplieron los objetivos trazados al inicio del desarrollo del mismo. Además, se pudo llegar a las siguientes conclusiones:

- Se realizó un estudio de las metodologías de desarrollo y las herramientas que se adaptan al desarrollo de aplicaciones web.
- Se lograron identificar los requisitos de los diferentes usuarios, lo que permitió la realización de un sistema que cumple con las exigencias de los mismos.
- Durante todo el proceso investigativo en la Universidad de las Ciencias Informáticas no se pudo encontrar un sistema que hiciera algo similar a lo que fue propuesto, por lo que el desarrollo del presente trabajo viene siendo una opción novedosa y de gran importancia para mantener y elevar el nivel de seguridad en las redes.
- Se logró realizar un sistema que permite el acceso a la información referente a las vulnerabilidades, así como el servicio de recepción de información mediante correo electrónico.

RECOMENDACIONES

Independientemente de que se lograron los objetivos trazados al inicio, se proponen las siguientes recomendaciones:

- Mejorar el diseño de las diferentes interfaces.
- Aumentar y optimizar el sistema de búsquedas implementado.
- Registrar el sistema en la CVE, Common Vulnerabilities and Exposure, para que de esta forma se tenga un producto con validez internacional.
- Realizar la migración de la información de la base de datos de vulnerabilidades para PostgreSQL.

BIBLIOGRAFÍA

Trabajos citados

1. **Kaspersky Lab.** Viruslist.com. *Viruslist.com.* [En línea] [Citado el: 5 de Junio de 2009.] http://www.viruslist.com/sp/news?id=208274291.

2. **Common Vulnerabilities and Exposures (CVE).** [En línea] [Citado el: 2009 de Noviembre de 26.] http://cve.mitre.org .

3. **National Vulnerability Database Home.** [En línea] [Citado el: 2009 de Noviembre de 26.] http://nvd.nist.gov/home.cfm.

4. **Security Focus.** [En línea] [Citado el: 2009 de Noviembre de 26.] http://www.securityfocus.com/about.

5. **Internet Security System-Research.** [En línea] IBM. [Citado el: 2009 de Noviembre de 26.] http://xforce.iss.net.

6. **OSVDB: The Open Source Vulnerability Database.** [En línea] [Citado el: 2009 de Noviembre de 26.] http://osvdb.org.

7. **J. Gutiérrez, Javier. Departamento de Lenguajes y Sistemas Informáticos.** [En línea] [Citado el: 2010 de Enero de 16.] http://www.lsi.us.es/~javierj/investigacion_ficheros/Framework.pdf.

8. **web2py:Enterprise Web Framework.** [En línea] [Citado el: 2010 de Enero de 25.] http://www.web2py.com.

9. **Jacobson, Ivar, Booch, Grady y Rumbaugh, James. 2000.** *El Proceso Unificado de Desarrollo de Software.* Madrid : Pearson Educación, S A., 2000.

10. **Proactiva Calidad.** [En línea] [Citado el: 2010 de Enero de 10.] http://www.proactiva-calidad.com/java/patrones/index.html.

11. **Prácticas y métodos para mejorar el desarrollo de proyectos. Ingeniería del Software.** [En línea] [Citado el: 2010 de Enero de 10.] http://www.ingenierosoftware.com/analisisydiseno/patrones-diseno.php.

12. **Proactiva Calidad.** [En línea] [Citado el: 2010 de Enero de 10.] http://www.proactiva-calidad.com/java/patrones/mvc.html.

13. **The Official web2py Book.** [En línea] [Citado el: 2010 de Enero de 10.] http://web2py.com/book.

Bibliografía Consultada

Comisión del Mercado de las Telecomunicaciones (CMT). CMT-Comisión del Mercado de las Telecomunicaciones. *CMT-Comisión del Mercado de las Telecomunicaciones.* [En línea] [Citado el: 5 de Junio de 2009.] http://www.cmt.es/es/publicaciones/anexos/2001ANEXO_3.pdf.

Urgente 24. Urgente 24|Información Confiable. *Urgente 24|Información Confiable.* [En línea] [Citado el: 5 de Junio de 2009.] http://www.urgente24.com/index.php?id=ver&tx_ttnews%5Btt_news%5D=116728&cHash=baf5acee68.

Mediapubli Sociedad de Publicaciones y Ediciones S.L. Público.es. *Público.es.* [En línea] [Citado el: 5 de Junio de 2009.] http://www.publico.es/ciencias/tecnologia/208704/virus/kido/amenaza/nuevo.

2008. *Kioskea.* [En línea] 18 de Octubre de 2008. [Citado el: 10 de Enero de 2010.]http://es.kioskea.net/contents/secu/secuintro.php3.

Estr@tegia Magazine. 2007. GestioPolis. [En línea] 26 de Octubre de 2007. [Citado el: 11 de Enero de 2010.]http://www.gestiopolis.com/administracion-estrategia/estrategia/seguridad-de-sistemas-informaticos.htm.

Eurologic. Eurologic Data Protection Systems. [En línea] [Citado el: 10 de Enero de 2010.] http://www.eurologic.es/conceptos/conbasics.htm.

Eurologic. Eurologic Data Protection Systems. [En línea] [Citado el: 10 de Enero de 2010.] http://www.eurologic.es/soluciones.htm.

Eurologic. Eurologic Data Protection Systems. [En línea] [Citado el: 10 de Enero de 2010.] http://www.eurologic.es/cifrado.htm.

Eurologic. Eurologic Data Protection Systems. [En línea] [Citado el: 10 de Enero de 2010.] http://www.eurologic.es/conceptos.htm.

H., Pedro Morales. 2010. GestioPolis. [En línea] 5 de Febrero de 2010. [Citado el: 12 de Febrero de 2010.]http://www.gestiopolis.com/administracion-estrategia/seguridad-transacciones-internet-ecommerce.htm.

Hispasec. 2009. Hispasec-seguridad informática. [En línea] 21 de Septiembre de 2009. [Citado el: 10 de Enero de 2010.] http://www.hispasec.com/laboratorio/Hispasec_Estudio_Vulnerabilidades.pdf.

2002. Instituto Nacional de Estadísticas y Geografía de Mexico. [En línea] INERGI, Noviembre de 2002. [Citado el: 10 de Enero de 2010.] http://www.inegi.gob.mx/inegi/contenidos/espanol/ciberhabitat/museo/cerquita/redes/seguridad/intro.htm.

Martín, Javier. 2009. *El País.* [En línea] El País.com, 20 de Octubre de 2009. [Citado el: 10 de Enero de 2010.]http://www.elpais.com/articulo/tecnologia/fabricantes/software/tardan/meses/arreglar/fallos/elpeputec/20091020elpeputec_06/Tes.

Romero, Pablo. 2009. Elmundo.es. [En línea] 23 de Septiembre de 2009. [Citado el: 10 de Enero de 2010.] http://www.elmundo.es/elmundo/2009/03/23/navegante/1237828040.html.

Aneiro Rodríguez, Lázaro Orlando. 2001. *elementos de arquitectura y seguridad informática.* Ciudad de La Habana : Editorial Pueblo y Educación, 2001.

García Feal, Miguel y Chouciño Ferreiro, Jose Luis. *Seguridad en Internet SSL.*

Gilfillan, Ian. *La Biblia MySQL.* s.l. : Anaya Multimedia.

Kabir, Mohammed J. *La Biblia Servidor Apache 2.* s.l. : Anaya Multimedia.

Leffingwell, Dean y Widrig, Don. 2003. *Managing Software Requirements. A Use Case Approach.* s.l. : Pearson Education, Inc, 2003.

Marzal, Andrés y Gracia, Isabel. 2003. *Introducción a la Programación en Python.* 2003.

Míguez Pérez, Carlos, Pérez Agudín, Justo y Matas García, Abel Mariano. *La Biblia Hacker.* s.l. : Anaya Multimedia.

Pollice, Gary, y otros. 2004. *Software Development for Small Teams. A RUP-Centric Approach.* Boston : PearsonEducation, Inc, 2004.

Pressman, Roger S. *Ingeniería del Software. Un enfoque práctico.*

Sasha Pachev, Alexander. 2003. *MySQL Enterprise Solutions.* Indianapolis, Indiana: Wiley Publishing, Inc, 2003.

Whittaker, James A. y Thompson, Herbert H. 2004. *How to Brake Software Security.* s.l. : Pearson Education, Inc., 2004.

Fernández Rivera, Javier. Desarrollador Front-End, Estándares Web y Accesibilidad . *Desarrollador Front-End, Estándares Web y Accesibilidad.* [En línea] [Citado el: 2010 de Mayo de 5.] http://aurea.es/wp-content/uploads/modelodedatos.pdf.

Gestión de Riesgos, Seguridad de Sistemas y Pruebas de Penetración. [En línea] Mageni. [Citado el: 2010 de Febrero de 12.] http://www.mageni.net/.

Junta de Andalucía. El Lenguaje HTML. [En línea] [Citado el: 2010 de Enero de 16.] http://www.juntadeandalucia.es/averroes/iesgaviota/informatica/html.html.

LLC-Departamento de Lenguajes y Ciencias de la Computación. Manual de JavaScript. [En línea] Universidad de Málaga. [Citado el: 2010 de Enero de 16.] http://www.lcc.uma.es/~eat/services/html-js/manual14.html.

Monografías.com-Tesis,Documentos, Publicaciones y Recursos Educativos. Test de Intrución. Principales metodologías. [En línea] [Citado el: 2010 de Febrero de 12.] http://www.monografias.com/trabajos71/ethical-hacking-test-intrusion-metodologias/ethical-hacking-test-intrusion-metodologias2.shtml.

Monografías.com-Tesis,Documentos,Publicaciones y Recursos Educativos. Lenguaje de programación para páginas web. [En línea] [Citado el: 2010 de Enero de 16.] http://www.monografias.com/trabajos7/html/html.shtml.

Nikto 2. CIRT.net. [En línea] [Citado el: 2010 de Febrero de 12.] http://cirt.net/nikto2.

OSVDB: The Open Source Vulnerability Database. [En línea] [Citado el: 2010 de Noviembre de 26.] http://osvdb.org/about.

OSVDB: The Open Source Vulnerability Database. [En línea] [Citado el: 2009 de Noviembre de 26.] http://osvdb.org/database_info.

Recursos Enseñanza Ciencia. Introducción al Lenguaje HTML. [En línea] [Citado el: 2010 de Enero de 16.] http://www.deciencias.net/disenoweb/elaborardw/paginas/intro_html.htm.

Security and Inteligence Advising. [En línea] [Citado el: 2010 de Febrero de 12.] http://www.siacorp.com/2k8/informatica.htm.

Security Focus. [En línea] [Citado el: 2009 de Noviembre de 26.] http://www.securityfocus.com/vulnerabilities.

Seguridad Informática/Seguridad de la Información. Amenazas Lógicas-Tipos de Ataques. [En línea] [Citado el: 2010 de Febrero de 12.] http://www.segu-info.com.ar/ataques/ataques.htm.

Snort. [En línea] [Citado el: 2010 de Febrero de 12.] http://www.snort.org/.

Tenable Network Security. [En línea] [Citado el: 2010 de Febrero de 12.] http://www.nessus.org/nessus/.

XPPS.net-El lugar de encuentro de usuarios y profesionales del ERP XPPS. [En línea] [Citado el: 2010 de Enero de 16.] http://www.xpps.net/contenido_xppsnet/areatec/HMTL.pdf.

XPPS.net-El lugar de encuentro de usuarios y profesionales del ERP XPPS. [En línea] [Citado el: 2010 de Enero de 16.] http://www.xpps.net/contenido_xppsnet/areatec/JavaScript.pdf.

CherryPy. [En línea] [Citado el: 2010 de Enero de 25.] http://www.cherrypy.org/.

CherryPy. [En línea] [Citado el: 2010 de Enero de 25.] http://www.cherrypy.org/wiki/CherryPyTutorial.

CherryPy. [En línea] [Citado el: 2010 de Enero de 25.] http://www.cherrypy.org/wiki/TableOfContents.

Developer Resources for Java Technology. [En línea] [Citado el: 2010 de Febrero de 5.] http://java.sun.com/products/jsp/.

Developer Resources for Java Technology. [En línea] [Citado el: 2010 de Febrero de 5.] http://java.sun.com/products/jsp/docs.html.

Django. [En línea] [Citado el: 2010 de Enero de 25.] http://www.djangoproject.com/.

Django. [En línea] [Citado el: 2010 de Enero de 25.] http://docs.djangoproject.com/en/1.2/.

Grok-A Smashing Web Framework. [En línea] [Citado el: 2010 de Enero de 25.] http://grok.zope.org/.

Grok-A Smashing Web Framework. [En línea] [Citado el: 2010 de Enero de 25.] http://grok.zope.org/documentation/.

Grok-A Smashing Web Framework. [En línea] [Citado el: 2010 de Enero de 25.] http://grok.zope.org/about.

Lenguaje de Programación Ruby. [En línea] [Citado el: 2010 de Febrero de 5.] http://www.ruby-lang.org/es/.

Lenguaje de Programación Ruby. [En línea] [Citado el: 2010 de Febrero de 5.] http://www.ruby-lang.org/es/documentation/.

Maestros del Web. [En línea] [Citado el: 2010 de Febrero de 5.] http://www.maestrosdelweb.com/editorial/phpintro/.

Manual de ASP. [En línea] [Citado el: 2010 de Febrero de 5.] http://www.manualdeasp.com/.

PHP HyperText Preprocessor. [En línea] [Citado el: 2010 de Febrero de 5.] http://php.net/index.php.

Programación en Castellano. [En línea] [Citado el: 2010 de Febrero de 5.] http://www.programacion.com/articulo/que_es_asp_net_227.

Pylons HQ. [En línea] [Citado el: 2010 de Enero de 25.] http://pylonshq.com/.

Pylons HQ. [En línea] [Citado el: 2010 de Enero de 25.] http://pylonshq.com/docs/en/0.9.7/.

Python Programing Languaje. [En línea] [Citado el: 2010 de Febrero de 5.] http://www.python.org/.

Python Programing Languaje. [En línea] [Citado el: 2010 de Febrero de 5.] http://www.python.org/doc/.

TurboGears. [En línea] [Citado el: 2010 de Enero de 25.] http://turbogears.org/.

TurboGears. [En línea] [Citado el: 2010 de Enero de 25.] http://turbogears.org/2.0/docs/.

TurboGears. [En línea] [Citado el: 2010 de Enero de 25.] http://turbogears.org/about/.

web2py: Enterprise Web Framework. [En línea] [Citado el: 2010 de Enero de 25.] http://web2py.com/book.

ANEXOS

Anexo 1: Esquema de la base de datos de OSVDB.

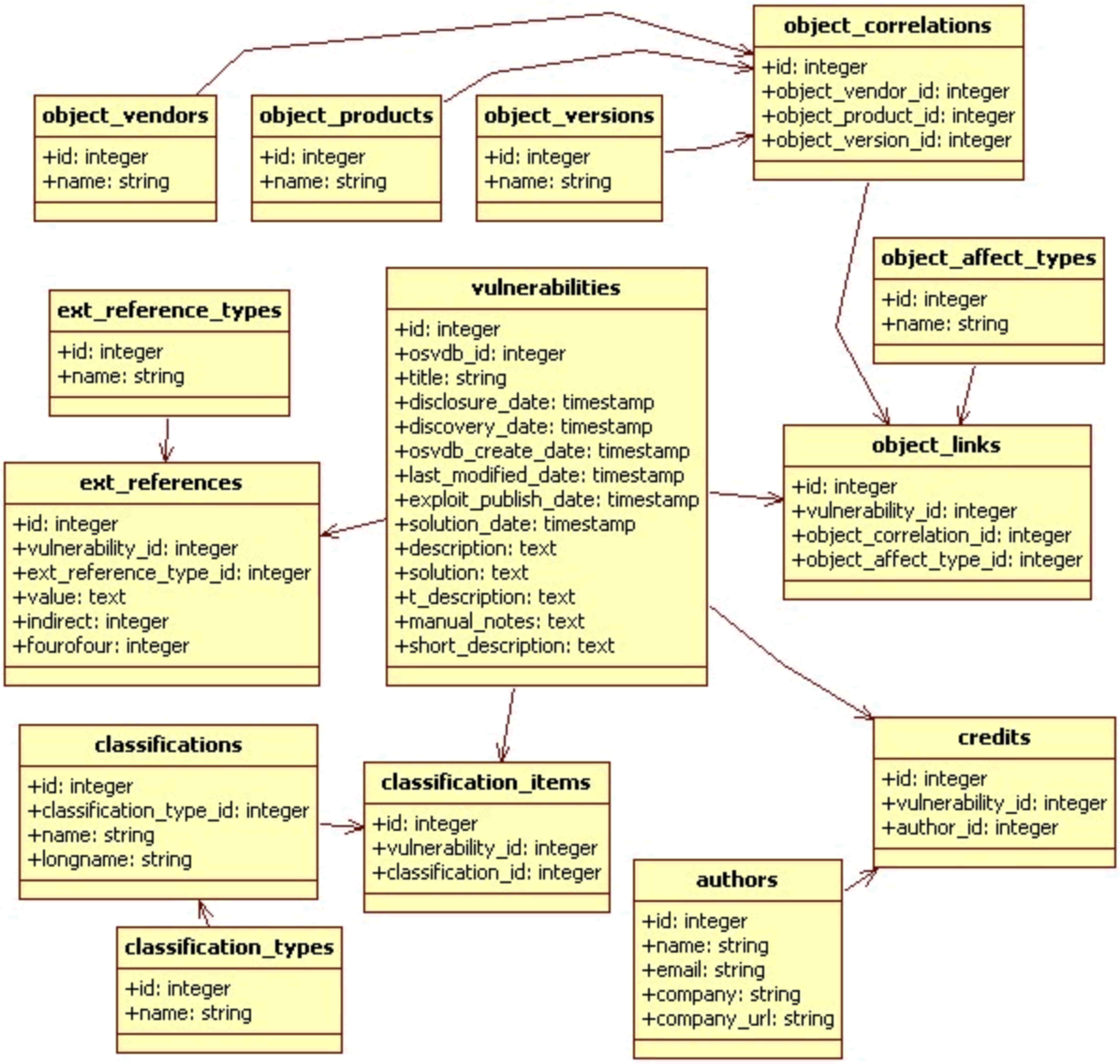


Ilustración 1: Esquema de base de datos de OSVDB.

GLOSARIO DE TÉRMINOS

C

CGI: Common Gateway Interface, en español, Interfaz de Entrada Común, es una importante tecnología web que permite a un navegador web solicitar datos de un programa ejecutado en un servidor web. Especifica un estándar para transferir datos entre el cliente y el servidor.

COCOMO: Constructive Cost Model, en español, Modelo constructivo de costes. Se considera un modelo matemático que se utiliza para la estimación de costes de proyectos de software.

D

DOM: Document Object Model, en español, Modelo de Objetos del Documento, es una interfaz de programación de aplicaciones que proporciona un conjunto estándar de objetos para representar documentos HTML. Es un modelo estándar sobre cómo pueden combinarse dichos objetos y una interfaz estándar para acceder a ellos y manipularlos.

H

HTML: Hyper Text Markup Language, en español, Lenguaje de marcación de Hipertexto, se puede definir como el lenguaje que se utiliza para la elaboración de páginas web.

HTTP: Hypertext Transfer Protocol, en español, protocolo de transferencia de hipertexto. Es un protocolo orientado a transacciones y sigue el esquema petición-respuesta entre un cliente y un servidor.

HTTPS: Hypertext Transfer Protocol Secure, en español Protocolo Seguro de Transferencia de Hipertexto. Es la versión segura del protocolo HTTP. Es un protocolo de red basado en el protocolo HTTP, destinado a la transferencia segura de datos.

HP: Hewlett-Packard, es una de las mayores empresas de tecnologías de la información del mundo. Fabrica y comercializa hardware y software además de brindar servicios de asistencias relacionados con la informática.

HP Security Mailings: Servicio de reporte de incidentes de seguridad de HP.

Herramienta Case: CASE (Computer Aided Software Engineering). Conjunto de programas y ayudas que dan asistencia a los analistas, ingenieros de software y desarrolladores, durante todos los pasos del Ciclo de Vida de desarrollo de un Software.

I

ISS: Internet Security Sistems, en español, Sistemas de Seguridad de Internet, es un servicio comercial proporcionado por IBM centrado en la seguridad.

IBM: International Business Machines, es una empresa multinacional que fabrica y comercializa herramientas, programas y servicios relacionados con la informática.

IIS: Internet Information Services, en español, Servicios de Información de Internet, es una serie de servicios para los ordenadores que funcionan con Windows. Este servicio convierte a un ordenador en un servidor de Internet o Intranet, lo que quiere decir que en las computadoras que esté instalado este servicio se pueden publicar páginas web.

M

Mageni: Es una empresa que ofrece servicios de seguridad de sistemas y aplicaciones, pruebas de penetración, gestión y administración de riesgos.

Microsoft: Es una empresa multinacional estadounidense. Desarrolla, fabrica, licencia y produce software y equipos electrónicos. Siendo sus productos más usados el sistema operativo Microsoft Windows y Microsoft Office.

Microsoft Security Alert: Servicio de reportes de incidentes de seguridad de Microsoft.

MySQL: Es un sistema de gestión de base de datos relacional, multihilo y multiusuario.

MSSQL: Es un sistema para la gestión de base de datos producido por Microsoft basado en el modelo relacional.

MVC: Modelo Vista Controlador es un estilo de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos

N

Nessus: Es un programa de escaneo de vulnerabilidades en diversos sistemas operativos. Está conformado por un cliente y un servidor.

Nikto: Es un escaneador de servidores web de código abierto. Es una herramienta muy usada para el testeo de vulnerabilidades.

O

ORM: Object Relational Mapping, en español, Mapeo Objeto Relacional, es una técnica de programación para convertir datos entre el sistema de tipos utilizado en un lenguaje de programación orientado a objetos y el utilizado en una base de datos relacional.

Oracle: Es un sistema de gestión de base de datos relacional desarrollado por Oracle Corporation.

P

PostgreSQL: Es un sistema de gestión de base de datos relacional orientada a objetos y libre, publicado bajo la licencia BSD.

R

RUP: Rational Unified Process, en español, Proceso Unificado de Racional, es el proceso de desarrollo del un software, que utiliza UML como lenguaje de modelado.

Ruby on Rails: es un framework o un entorno de desarrollo web de código abierto.

S

SCAP: Es el Protocolo de Seguridad de Contenidos de Automatización. Fue desarrollado para la gestión de la información de la Base de Datos de Vulnerabilidades de los Estados Unidos.

Snort: Es un sniffer de paquetes y un detector de intrusos basado en red. Es una de las herramientas de seguridad más usadas en la actualidad.

Security Advisory List: Servicio de reportes de seguridad de IBM.

SQLite: Es un sistema de gestión de base de datos relacional. Es bastante sencillo, pero a la vez robusto y simple.

SQL: Structured Query Language, en español, Lenguaje Estructurado de Consultas, es un lenguaje formal declarativo, para manipular información en una base de datos.

Servlets: La palabra servlets deriva de otra anterior, applet, que se refería a pequeños programas que se ejecutan en el contexto de un navegador web. Por contraposición servlets es un programa que se ejecuta en un servidor. El uso más común es generar páginas web de forma dinámica a partir de los parámetros que envíe el navegador web.

T

Tomcat: Funciona como un contenedor de servlets. Implementa las especificaciones de los servlets y de JavaServer Pages de Sun Microsystems.

U

UML: Unified Modeling Language. El Lenguaje Unificado de Modelado prescribe un conjunto de notaciones y diagramas estándar para modelar sistemas orientados a objetos, y describe la semántica esencial de lo que estos diagramas y símbolos significan. UML se puede usar para modelar distintos tipos de sistemas: sistemas de software, sistemas de hardware, y organizaciones del mundo real.

W

W3C: Es un consorcio internacional que produce recomendaciones para la web. Es dirigida por Tim Berners-Lee, el creador original de URL (Uniform Resource Locator, Localizador Uniforme de Recursos), HTTP (HyperText Transfer Protocol, Protocolo de Transferencia de HiperTexto) y HTML, Lenguaje de Marcado de HiperTexto, que son las principales tecnologías sobre las que se basa la web.

WSGI: Web Server Gateway Interface, en español, Interfaz de Entrada de Servidores Web, define una simple y universal interface entre los servidores web y las aplicaciones web ó framework para el lenguaje de programación Python.

X

XSS: Del inglés Cross-Site Scripting, es un tipo de vulnerabilidad informática o agujero de seguridad basado en la explotación de errores del sistema de validación del HTML.