Universidad de las Ciencias Informáticas
Facultad 2

Título: Herramienta de Apoyo a la Metodología de Mitigación de Inyecciones SQL.

Trabajo de Diploma para optar por el título de

Ingeniero Informático

Autor(es): Mónica Ballaga Croublet

Susana Reyes Valle

Tutor(es): Msc. Rogfel Thompson Martínez

La Habana, Cuba

Junio 2013

"Lo fundamental es que seamos capaces de hacer cada día algo que perfeccione lo que hicimos el día anterior." Ernesto Guevara de la Serna (Che)

DECLARACIÓN DE AUTORÍA

Declaramos que somos los únicos autores de este trabajo y autorizamos a la Facultad 2 de la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.
Para que así conste firmo la presente a los ____ días del mes de _____ del año 2013.

_____ _____

Firma del Autor
Mónica Ballaga Croublet

Firma del Autor
Susana Reyes Valle

_____

Firma del Tutor
Msc. Rogfel Thompson Martínez

DATOS DE CONTACTO

Mónica Ballaga Croublet

Correo: mballaga@estudiantes.uci.cu

Ciudad de La Habana, Cuba

Susana Reyes Valle

Correo: svalle@estudiantes.uci.cu

Ciudad de La Habana, Cuba

Msc. Rogfel Thompson Martínez

Correo: rthompson@uci.cu

Universidad de las Ciencias Informáticas, Ciudad de la Habana, Cuba

AGRADECIMIENTOS

Mónica Ballaga Croublet

En los agradecimientos casi siempre se cometen injusticias, pues la memoria es a menudo traicionera. Pero aún sabiendo que no existirá una forma para agradecerles, hoy sin embargo, puedo afirmar sin lugar a dudas, que este trabajo no hubiese sido posible sin la ayuda de:

Mis padres que son lo más grande que tengo y que tanto me han apoyado para lograr mis objetivos. A ellos les debo la vida y espero retribuirles lo que me han dado.

Mi tutor Thompson que ha estado presente y dispuesto a ayudarme cuando lo he necesitado.

Mi familia que me ha apoyado en las buenas y las malas, mi más grande agradecimiento.

Mis compañeros, que aunque a muchos no los veré más, no los olvidaré.

La Universidad de las Ciencias Informáticas que me dio la oportunidad de estudiar esta carrera para formarme como ingeniero en tan importante ciencia, lo que me permitirá contribuir al desarrollo tecnológico de nuestra gran nación.

Susana Reyes Valle: *Siempre hay personas que marcan nuestras vidas con solo pequeños detalles. Hoy quiero agradecer a todas esas personas que se han preocupado por mí y porque mi sueño se haya hecho realidad, en especial a:*

Mi mamá y mi papá, que han sido el regalo más grande que la vida me ha dado, gracias por todo lo que han hecho por mí, por apoyarme siempre y convertirme en la persona que soy, a ellos les debo mi vida porque mientras yo estoy aquí estudiando ellos están trabajando muy duro para que no tenga que preocuparme por nada. Los quiero más que a mi propia vida, gracias por existir.

A mis hermanos Michel e Islenis que siempre confiaron en mí y me apoyaron en las buenas y en las malas.

A mis abuelos maternos que son la alegría de mi vida, por estar siempre pendientes de mí y por formar esa familia tan linda de la cual soy parte.

A mis amistades que aunque hayan tenido que convivir con mis peleas, los quiero.

A mi compañera de tesis que aunque acabó con toda la paciencia que tenía me demostró que si se puede, es la mejor.

DEDICATORIA

Mónica Ballaga Croublet

Tengo en la vida dos personas importantes, los que siempre me han apoyado, sin importar lo que piensan. Gracias a ese apoyo he ganado confianza en lo que hago cada día y me he vuelto más independiente.

Mis padres María de los Ángeles y Francisco son los grandes merecedores de todo mi agradecimiento y orgullo. A ellos va dedicado este trabajo, les juro que no los defraudaré. Gracias por todo.

Susana Reyes Valles: *Este trabajo se lo dedico a tres personas importantes en mi vida dos de ellas son mi mamá y mi papá; que me apoyaron siempre y estuvieron presentes en las buenas y las malas, y la tercera persona no se encuentra conmigo en estos momentos pero yo sé que desde el cielo me cuida, está orgulloso de mi y feliz con este triunfo, mi tío Juani que siempre me consintió y me demostró que hay que luchar por lo que se quiere. Gracias por ser parte de mi vida, se merecen este trabajo y mucho más; los amo.*

RESUMEN

Las tecnologías y software desarrollados en el mundo son un producto de la inteligencia y del conocimiento humano, y como producto de este último, no están exentas de errores. Estos errores en los software y tecnologías, conocidos comúnmente como vulnerabilidades, son provocados por malas prácticas, por problemas de seguridad informática cometidos durante el desarrollo; por el uso de otras tecnologías que tienen vulnerabilidades, o por problemas de configuración. Las vulnerabilidades informáticas pueden ser aprovechas por intrusos con la intención de obtener información de un sistema, o adueñarse de él, violando normas de seguridad establecidas.

Un tipo de ataque a las vulnerabilidades del software son las inyecciones de Lenguaje de Consulta Estructurada (SQL, por sus siglas en inglés); las cuales consisten en la inserción o "inyección" de una consulta SQL, a través de los datos de entrada que poseen las aplicaciones, o mediante el Localizador de Recurso Uniforme (URL, por sus siglas en inglés).

En la Universidad de las Ciencias Informáticas se están llevando a cabo investigaciones donde se proponen el uso de técnicas de minería de datos para mitigar estas amenazas. Una de las investigaciones existentes propone una metodología; la cual define 6 procesos necesarios para desarrollar un modelo capaz de filtrar estas amenazas. Estos procesos generan artefactos, los cuales quedan registrados en documentos y son llenados por especialistas que deben tener conocimiento de la metodología. Por la lentitud que ocasiona aprender una metodología y desarrollar sus procesos y artefactos; también debido a las necesidades inminentes de mitigar inyecciones SQL, en la presente investigación se propone el desarrollo de una herramienta de apoyo a esta metodología.

TABLA DE CONTENIDOS

INTRODUCCIÓN

"El único sistema verdaderamente seguro es aquel que se encuentra apagado, encerrado en una caja fuerte de titanio, enterrado en un bloque de hormigón, rodeada de gas nerviosos y vigilado por guardias armados y muy bien pagados, incluso entonces, yo no apostaría mi vida por ello "(Gene Spafford).

En la actualidad la informática está avanzando a pasos agigantados, y con ello el ciberespacio, el cual gestiona casi todo mediante aplicaciones web, ya sea desde una institución, una empresa, o simplemente un usuario con el fin de promover o adquirir cualquier beneficio.

La seguridad de los datos es uno de los pilares de la sociedad informatizada en la que vivimos, el no proteger esta información sería de gran riesgo para las empresas y para los clientes.

En muchos estados existen normas jurídicas que regulan el tratamiento de los datos personales, como por ejemplo en España, donde existe la “Ley Orgánica de Protección de Datos de Carácter Personal“ (1) (LOPD, por sus siglas en español) que tiene por objetivo garantizar y proteger, en lo que concierne al tratamiento de los datos personales, las libertades públicas y los derechos fundamentales de las personas físicas, y especialmente de su honor, intimidad y privacidad personal y familiar. A pesar de que existen leyes como estas, muchas personas dedican a conspirar en contra de la integridad de las aplicaciones web, uno de los ataque que se realizan son mediante las inyecciones SQL. Para todo sitio web esta es una de las vulnerabilidades más devastadora para el negocio, ya que puede conducir a la exposición de toda la información confidencial almacenada en una aplicación de base de datos, incluyendo información útil, como nombres de usuario, contraseñas, direcciones, números de teléfono y detalles de tarjetas de crédito.

Definitivamente las inyecciones SQL son una amenaza creciente a la seguridad de los sistemas Web. Las técnicas actuales para mitigarlas solo son capaces de filtrar inyecciones SQL conocidas. En la Universidad de las Ciencias Informáticas se están llevando a cabo investigaciones donde se proponen el uso de técnicas de minería de datos para mitigar estas amenazas. Una de las investigaciones existentes propone una metodología para desarrollar una base de conocimiento capaz de filtrar estas amenazas.

Esta investigación propone una serie de 6 procesos necesarios para desarrollar un modelo capaz de filtrar inyecciones SQL. Estos procesos generan artefactos, los cuales quedan registrados en documentos. Los artefactos son llenados por especialistas que deben tener conocimiento de la metodología. Por la lentitud que ocasiona aprender una metodología y desarrollar sus procesos y artefactos; también debido a las

necesidades inminentes de mitigar inyecciones SQL, se hace necesario una herramienta de apoyo a esta metodología.

Por lo antes expuesto se plantea como **Problema a resolver**: ¿Cómo contribuir a la mitigación de inyecciones SQL en aplicaciones web?

Como **objeto de estudio** de la presente investigación es: Seguridad Informática en aplicaciones web y el **campo de acción:** La Metodología de mitigación de inyecciones SQL.

Para dar solución al problema expuesto anteriormente se traza como **objetivo general**: Desarrollar una herramienta de apoyo a la Metodología de mitigación de inyecciones SQL. De dicho objetivo general se derivan los siguientes **objetivos específicos**:

- Desarrollar un marco teórico de las estrategias para mitigar inyecciones SQL.
- Definir los requisitos correspondientes a la herramienta de apoyo a la Metodología de mitigación de inyecciones SQL.
- Desarrollar el Modelo de Análisis y Diseño de la herramienta de apoyo a la Metodología de mitigación de inyecciones SQL.
- Implementar una herramienta de apoyo a la Metodología para mitigar inyecciones SQL.
- Validar la herramienta de apoyo a la Metodología para mitigar inyecciones SQL.

Y con estos las siguientes **Tareas Investigativas:**

- Estudio de las estrategias para mitigar las inyecciones SQL.
- Análisis de las técnicas y herramientas para mitigar las inyecciones SQL.
- Análisis de la Metodología de mitigación de inyecciones SQL en aplicaciones web.
- Análisis de los requisitos necesarios para el desarrollo de la herramienta de apoyo a la Metodología de mitigación de inyecciones SQL.
- Descripción de los casos de uso del sistema para el desarrollo de la herramienta de apoyo a la Metodología de mitigación de inyecciones SQL.
- Implementación de la herramienta de apoyo a la Metodología para mitigar las inyecciones SQL.
- Ejecución de las Pruebas de Calidad para la herramienta de apoyo a la Metodología de mitigación de inyecciones SQL.

Los **Métodos de Investigación** que se utilizaron en el desarrollo del trabajo, fueron los siguientes:

Teóricos.

- Analítico-sintético: Se utilizó este método para conocer mejor el desarrollo y funcionamiento de las técnicas de mitigación de las inyecciones SQL. Mediante la información obtenida del estudio del estado del arte, las tecnologías, metodologías, leguajes y herramientas, la utilización de este método será de gran utilidad para conocer así sus ventajas y su correcta utilización para la mitigación de las inyecciones SQL.
- Modelación: Se utilizó este método para realizar los diagramas necesarios en el desarrollo y el mejor entendimiento de la herramienta para el tratamiento de las inyecciones SQL en aplicaciones web.
- Histórico-lógico: Se utilizó este método para realizar un estudio tanto nacional como internacional de la evolución de las herramientas para el tratamiento de las inyecciones SQL.

Este trabajo se encuentra estructurado por 5 capítulos, que mostramos a continuación:

Capítulo 1: Fundamentación Teórica. Se exponen conceptos, técnicas, tecnologías y tendencias de la propuesta presentada.

Capítulo 2: Desarrollo de la Metodología para mitigar inyecciones SQL.

Capítulo 3: Características del sistema. Se ofrece una visión práctica del sistema, los requisitos funcionales y no funcionales, además de una propuesta del sistema.

Capítulo 4: Diseño del sistema. Se presenta una vista interna del sistema, diagrama de clases del diseño.

Capítulo 5: Implementación y Pruebas del sistema, se muestran los elementos utilizados en la implementación del sistema, con todos los artefactos generados y los resultados de las mismas.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

1.1 Introducción

El Lenguaje de Consulta Estructurado (*Structured Query Language*), con la nomenclatura de SQL, es un lenguaje declarativo de acceso a bases de datos relacionales que permite especificar diversos tipos de operaciones sobre las mismas. Concentra características del álgebra y el cálculo relacional permitiendo lanzar consultas con el fin de recuperar información de interés de un "banco de registros", de una forma sencilla y poderosa. Forma parte de los lenguajes de Cuarta Generación (*4GL*).

El hecho de tener un estándar definido por un lenguaje para bases de datos relacionales abre potencialmente el camino a la intercomunicabilidad entre todos los productos que se basan en él. Efectivamente, en general cada productor adopta e implementa en la propia base de datos sólo el corazón del lenguaje SQL (el así llamado Entry level o al máximo el Intermediate level), extendiéndolo de manera individual según la propia visión que cada cual tenga del mundo de las bases de datos ya que estas introducen tipos de valores de campo que no necesariamente están presentes en otras. Sin embargo, existe un conjunto de tipos que están representados en la totalidad de estas bases. SQL permite realizar consultas y funciones de definición, control y gestión a la Base de Datos (2). Las sentencias SQL se clasifican según su finalidad dando origen a tres sub-lenguajes:

- Lenguaje de definición de datos (DDL, por sus siglas en inglés): incluye órdenes para definir, modificar o borrar las tablas en las que se almacenan los datos y de las relaciones entre estas.
- Lenguaje de Control de Datos (DCL, por sus siglas en inglés): contiene elementos útiles para trabajar en un entorno multiusuario, en el que es importante la protección de los datos, la seguridad de las tablas y el establecimiento de restricciones en el acceso, así como elementos para coordinar la compartición de datos por parte de usuarios concurrentes, asegurando que no interfieren unos con otros.
- Lenguaje de Manipulación de Datos (DML, por sus siglas en inglés): permite recuperar los datos almacenados en la base de datos y también incluye órdenes para permitir al usuario actualizar la base de datos añadiendo nuevos datos, suprimiendo datos antiguos o modificando datos previamente almacenados (3).

Se conoce como Inyección SQL al método de infiltración de código intruso que se vale de una vulnerabilidad informática presente en una aplicación en el nivel de validación de las entradas para realizar consultas a una base de datos (4).

El origen de la vulnerabilidad radica en el incorrecto chequeo y/o filtrado de las variables utilizadas en un programa que contiene, o bien genera, código SQL. Es, de hecho, un error de una clase más general de vulnerabilidades que puede ocurrir en cualquier lenguaje de programación o script que esté embebido dentro de otro. Se dice que existe o se produjo una inyección SQL cuando, de alguna manera, se inserta o "inyecta" código SQL invasor dentro del código SQL programado, a fin de alterar el funcionamiento normal del programa y lograr así que se ejecute la porción de código "invasor" incrustado, en la base de datos y así exponer toda la información confidencial que esta contiene. En los últimos años miles de personas se han visto afectados con los diferentes ataques que se han producido a disímiles sitios web vulnerables. Un ejemplo de estos fue lo sucedido el 20 de octubre del 2011 donde ocurrió un ataque masivo de inyecciones SQL debido a que sitios web pobremente configurados bombardean a los visitantes con ataques de malware afectando a 614 000 páginas. Este ataque explota principalmente a los sitios web que corren en el entorno de aplicación de Microsoft ASP.Net. La infección masiva, la cual redirige a los usuarios a sitios que explotan versiones desactualizadas de Java de Oracle, Flash Player de Adobe y varios navegadores, fue divulgada por investigadores de Armorize. En el ataque inicial fueron 180.000 páginas las afectadas y se ha propagado a 613.890 páginas combinadas. La infección inyecta código en los sitios web operados por restaurantes, hospitales, y otros negocios pequeños y planta un enlace invisible en los navegadores de los visitantes hacia otros sitios. Esos sitios a su vez redirigen a varios otros sitios web que incluyen código intensamente ofuscado (5).

Las inyecciones SQL han aumentado según el informe Hacker Intelligence Initiative (HII, por sus siglas en inglés) de Imperva. Este informe muestra cómo se ejecutan los ataques de inyección SQL y el ingenio de los atacantes para esquivar los controles de seguridad (6).

1.1.1 Técnicas para mitigar amenazas de inyecciones SQL

Las técnicas para mitigar las amenazas de inyecciones SQL se agrupan en dos áreas:

- **A nivel del código**, consiste en asegurar el código desde las fases de implementación del proyecto. En esta área, se aplican las técnicas de parametrización de las sentencias, validación de entradas de

información, codificación de la salida de información, canonización y las buenas prácticas de diseño de las aplicaciones.

- **Validación de entradas**

Es el proceso mediante el que se asegura que los datos manipulados en la aplicación son correctos. Lo cual no solo implica que el tipo de datos sea adecuado, sino que esté dentro de un rango válido para la aplicación.

La mayoría de las validaciones de datos se realizan en la introducción de los mismos en la aplicación desde la interfaz de usuario.

- **Codificación de la salida**

Es el proceso de codificar lo que se pasa entre los diferentes módulos o partes de la aplicación. En el contexto de inyección SQL, esto se aplica a los requisitos para codificar, o "citar", contenido que se envía a la base de datos para asegurar que no está tratado de forma inadecuada.

- **A nivel de plataforma o aplicación**, que consisten en asegurar el software desplegado. Entre las técnicas que posee se encuentra las de Protección en Tiempo Real.

- **Protección en Tiempo Real**

La protección en tiempo real es una técnica de gran valor para mitigar y prevenir las amenazas de inyecciones SQL conocidas. Reparar directamente el código siempre es la solución ideal; sin embargo, el esfuerzo requerido no siempre es efectivo ni práctico.

Entre las estrategias que asumen la protección en tiempo real para mitigar los problemas de seguridad, debido a las vulnerabilidades de los sistemas a inyecciones SQL, se encuentran los Firewall de Aplicaciones Web (WAF, por sus siglas en inglés), los Filtros de Intercepción y los Firewall de Base de Datos.

- **Firewall de Aplicaciones Web**

Los WAFs, son una tecnología emergente de la seguridad de la información que tiene como objetivo proteger a las aplicaciones Web de los ataques informáticos. Las soluciones WAFs, son capaces de prevenir ataques que para los Firewall de Redes y los Detectores de Intruso, es imposible prevenir.

- **Filtros de Intercepción**

Otra solución de seguridad son los Filtros de Intercepción, que consisten en una serie de módulos independientes, que se pueden encadenar para mejorar el análisis de seguridad en la aplicación. Los Filtros no tienen dependencias explícitas entre ellos, se puede añadir un nuevo filtro, sin dañar el funcionamiento de uno existente. Cada filtro posee una secuencia de patrones que le permiten denegar o permitir la interacción entre el cliente y el servidor Web o aplicación. Esta modularidad hace al sistema de Filtros reusable para todas las aplicaciones.

- **Firewall de Base de Datos**

Una de las estrategias más cercana a los sistemas de almacenamiento de información son los Firewall de Bases de Datos, que es esencialmente un servidor proxy que se encuentra entre la aplicación y la base de datos. La aplicación se conecta al servidor de seguridad de bases de datos y envía la consulta como si se tratara normalmente la acción de conectarse a la base de datos (7).

- **Firewall Semántico de Aplicaciones Web**

El esfuerzo requerido por las técnicas referidas anteriormente no siempre es efectivo ni práctico, ya que no son capaces de mitigar inyecciones SQL no conocidas, por lo que debido a esta debilidad un grupo de investigación de la universidad de Quaid-i-Azam, Islamabad, Paquistán, desarrolla un proyecto de un Firewall Semántico de Aplicaciones Web (SWAF, por sus siglas en inglés). Los investigadores de este proyecto centran su atención en el desarrollo automático de una ontología basada en un texto para el Firewall Semántico de Aplicaciones Web. La potencialidad de esta tecnología radica en el buen desarrollo de la ontología requerida, la que debe variar según el intérprete de la tecnología Web utilizada, y la técnica de Minería de Datos definida para el filtrado de ataques (8).

Para desarrollar una base de conocimiento para un SWAF, que mitigue inyecciones SQL, en una aplicación Web, para un intérprete específico, se hace necesario la utilización de una metodología que ayude a:

- Evaluar los niveles de riesgo a los que está sometida la aplicación Web.

- Definir los elementos necesarios para el desarrollo de la ontología correspondiente al intérprete.
- Definir la base de conocimiento para la mitigación de las inyecciones SQL.
- Definir la técnica de Minería de Datos más adecuada al problema (9).

1.2 Herramientas, tecnologías y metodología utilizadas

1.2.1 Lenguaje de programación.

Python 2.7:

Python es un lenguaje de programación de alto nivel creado por Guido van Rossum a principios de los años 90 cuyo nombre está inspirado en el grupo de cómicos ingleses “Monty Python”. Es un lenguaje con una sintaxis muy limpia y que favorece un código legible. Es interpretado o de script, con tipado dinámico, fuertemente tipado y orientado a objetos. No obstante a pesar de ser un lenguaje interpretado o de script, no le impide que posea muchas de las características de los lenguajes compilados, por lo que se podría decir que es semi interpretado.

Es multiplataforma ya que está disponible en plataformas como UNIX, Solaris, Linux, DOS, Windows, OS/2, Mac OS, etc. lo que posibilita la utilización de librerías específicas de cada plataforma y correr en todos estos sistemas sin grandes cambios. Su sintaxis es simple, clara y sencilla. El tipado dinámico, el gestor de memoria, la gran cantidad de librerías disponibles y la potencia del lenguaje, entre otros, hacen que desarrollar una aplicación en Python sea además de sencillo, muy rápido. (11)

Ofrece gran soporte e integración con otros lenguajes y herramientas. Viene con una biblioteca estándar muy amplia que incluye herramientas matemáticas y dicímiles funcionalidades de gran ayuda desde el más bajo nivel hasta el más alto, facilitándole al programador la implementación de aplicaciones sin la necesidad de recurrir continuamente a bibliotecas externas. Además dispone de una extensa colección de bibliotecas libres disponibles en la mayoría de los repositorios de los sistemas GNU\Linux y es fácil de aprender.

Estas características, junto a la portabilidad de su código, versatibilidad, simplicidad, interactividad, sintaxis clara y legible, productividad, popularidad, Open Source, facilidad, rapidez de aprendizaje y

excelente documentación, lo convierten en un lenguaje muy apropiado y codiciado para numerosas aplicaciones.

Python es un lenguaje muy expresivo, es decir, los programas Python son muy compactos y suelen ser bastante cortos que su equivalente en lenguajes como C. La sintaxis de Python permite la escritura de programas cuya lectura resulta más fácil que para otros lenguajes de programación. Ofrece un entorno interactivo que facilita la realización de pruebas y ayuda a despejar dudas acerca de ciertas características del lenguaje. El entorno de ejecución de Python detecta muchos de los errores de programación que escapan al control de los compiladores y proporciona información muy rica para detectarlos y corregirlos. Posee un amplio juego de estructuras de datos que se pueden manipular de modo sencillo. Además una de las ventajas fundamentales de Python es la gratuidad de su intérprete. (12)

De acuerdo a las características y ventajas que brinda este lenguaje, además de todas las funcionalidades que posee unido a las facilidades de uso, se decidió la utilización de Python para el desarrollo de la aplicación.

1.2.2 Eclipse Helios 3.6

La plataforma Eclipse consiste en un Entorno de Desarrollo Integrado (IDE, Integrated Development Enviroment) abierto y extensible. Un IDE es un programa compuesto por un conjunto de herramientas útiles para un desarrollador de software. Cuenta con una fácil instalación. Como elementos básicos, un IDE cuenta con un editor de código, un compilador/intérprete y un depurador. Eclipse cuenta con numerosas herramientas de desarrollo de software. También da soporte a lenguajes de programación, como son C/C++, Cobol, Fortran, Java, PHP o Python. A la plataforma base de Eclipse se le pueden añadir extensiones (plugins) para extender las funcionalidades de la herramienta. (13)

PyDev es una de las extensiones que existe para eclipse, en este caso permite integrar la plataforma para java un IDE para Python, el cual provee entre otras funcionalidades:

- Completamiento de Código.
- Resaltado de Sintaxis.
- Análisis del Código.
- Refactorizar.
- Debuguear.
- Consola Interactiva.

A través de la investigación realizada para Eclipse, se pudieron apreciar características significativas que demuestran que Eclipse constituye un IDE idóneo para el desarrollo de la aplicación que se desarrolla. Además es importante resaltar su alto nivel de integración con el leguaje a utilizar: Python. Por estas razones y las ventajas que proporciona se decidió la utilización de este entorno de desarrollo integrado. (14)

1.2.3 Qt 4.8

Es una biblioteca multiplataforma ampliamente usada para desarrollar aplicaciones con interfaz gráfica de usuario, así como también para el desarrollo de programas sin interfaz gráfica, como herramientas para la línea de comandos y consolas para servidores. Permite pre visualizar el aspecto final durante la edición al igual que crear y personalizar widgets y diálogos; probando con diferentes estilos y resoluciones. Es posible utilizar Qt con otros lenguajes a través de bindings. Existen bindings de Qt para lenguajes como C#, PHP, Python, y Ruby. (15)

1.2.4 SQLite 0.8

A diferencia de los motores de base de datos convencionales con la arquitectura cliente-servidor, SQLite es independiente, ya que no se comunica con un motor de base de datos sino que las librerías de SQLite pasan a integrar la aplicación. La misma utiliza las funcionalidades de SQLite a través de llamadas simples a sub rutinas y funciones. Esto reduce la latencia en el acceso a la base de datos, debido a que las llamadas a funciones son más eficientes que la comunicación entre procesos. El conjunto de la base de datos (definiciones, tablas, índices, y los propios datos), son guardados como un solo fichero estándar, en la máquina local. A continuación se muestran algunas de las razones que convierten a SQLite como el motor de base de datos idóneo para el desarrollo de la aplicación.

Costo: SQLite es de dominio público, y por tanto, es libre de utilizar para cualquier propósito sin costo y se puede redistribuir libremente.

Tamaño: SQLite tiene una pequeña memoria y una única biblioteca es necesaria para acceder a bases de datos, lo que lo hace ideal para aplicaciones de bases de datos incorporadas.

Rendimiento de base de datos: SQLite realiza operaciones de manera eficiente y es más rápido que MySQL y PostgreSQL.

Portabilidad: Se ejecuta en muchas plataformas y sus bases de datos pueden ser fácilmente portadas sin ninguna configuración o administración.

Estabilidad: SQLite es compatible con ACID, reunión de los cuatro criterios de Atomicidad, Consistencia, Aislamiento y Durabilidad.

SQL: Implementa un gran subconjunto de la ANSI – 92 SQL estándar, incluyendo sub-consultas, generación de usuarios, vistas y triggers.

Interfaces: Cuenta con diferentes interfaces del API, las cuales permiten trabajar con C++, PHP, Perl, Python, Ruby, Tcl, groovy, etc. (16)

1.2.5 SQLAlchemy 0.7

SQLAlchemy es un kit de herramientas SQL para Python y un Mapeo Relacional de objetos (ORM, por sus siglas en inglés) que le da a los desarrolladores todo el poder y flexibilidad de SQL. El trabajo del ORM es justamente leer las tablas, columnas, propiedades y las relaciones entre datos para expresarlos utilizando objetos Python. El ORM estandariza en un sistema configuracional "declarativa", que permite la construcción de clases definidas por usuario en línea con los metadatos de tabla se asignan a, de la misma manera la mayoría de otras herramientas de objetos relacionales proporcionan. Sin embargo, este sistema es totalmente opcional - en su esencia, el ORM considera que la clase definida por el usuario, los metadatos de tabla asociada, y la asignación de los dos para estar completamente separado. A través del uso del mapper () función, cualquier clase de Python arbitrarias se puede asignar a una tabla de base de datos o la vista. Será entonces responsabilidad del ORM de convertir los datos entre el sistema de tipos utilizado en un lenguaje de programación orientado a objetos y el utilizado en una base de datos relacional (17).

```
DATABASE <-----> SQLALCHEMY(ORM) <------> PYTHON
```

1.2.6 Visual Paradigm 8.0

Visual Paradigm es una herramienta de Lenguaje Unificado de Modelado (UML, por sus siglas en inglés) profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. El software de modelado UML ayuda a una rápida construcción de aplicaciones de calidad y a un menor coste. Se decidió utilizar esta herramienta para realizar los diagramas necesarios en el desarrollo y el mejor entendimiento de la aplicación. Permite dibujar distintos tipos de diagramas y generar documentación. (18)

1.2.7 UML

Es un lenguaje para especificar, visualizar, construir y documentar los artefactos de los sistemas de software, así como también el modelado de negocios. Se encuentra definido por varios niveles, adecuado para los modelos orientados a objetos.

UML da la destreza necesaria para la creación de sistemas de software que se encuentren bien diseñados, que sean robustos y de fácil mantenimiento, debido a que da la habilidad para analizar y diseñar un sistema desde la perspectiva de los objetos. (19)

1.2.8 Marco de Trabajo de Análisis de Riesgos de Debilidades Comunes 0.8

Para la evaluación de los riesgos se propone utilizar el sistema de Análisis de Riesgos CWRAF (Marco de Trabajo de Análisis de Riesgos de Debilidades Comunes, por su significado en español) en su versión 0.8, debido a que en cada proyecto es necesario determinar cuál es el riesgo asociado a las inyecciones SQL, lo que serviría como toma de decisión para aplicar o no la base de conocimiento que genera la Metodología.

Provee un medio para el desarrollador y el consumidor de software, para priorizar las debilidades de sus sistemas que son relevantes para su negocio, su misión y sus tecnologías desplegadas. CWRAF apoya a las personas en el razonamiento y comunicación sobre la importancia relativa de diferentes debilidades, ayuda a desarrolladores y a usuarios, introducir más software seguros en sus entornos operacionales. CWRAF, provee un marco de trabajo para la evaluación de debilidades en una forma consistente, flexible y abierta, acorde al contexto para varios dominios de negocio.

CWRAF:

- Incluye un mecanismo para evaluar riesgos de las debilidades.
- Soporta selección y priorización de debilidades relevantes y personalización de necesidades específicas al negocio, o a la misión de la institución en cuestión.
- En conjunto con el Sistema de Evaluación de Debilidades Comunes (CWSS, por sus siglas en inglés), puede ser usada por los usuarios para determinar las debilidades más importantes en su negocio. (20)

1.2.9 Metodología.

Proceso Unificado Racional (RUP, por sus siglas en inglés):

Una metodología es una colección de procedimientos, técnicas, herramientas y documentos auxiliares que ayudan a los desarrolladores de software en sus esfuerzos por implementar nuevos sistemas de información. Una metodología está formada por fases, cada una de las cuales se puede dividir en sub-fases, que guiarán a los desarrolladores de sistemas a elegir las técnicas más apropiadas en cada momento del proyecto y también a planificarlo, gestionarlo, controlarlo y evaluarlo. (21)

RUP

La metodología de desarrollo seleccionada es RUP y se caracteriza por ser:

- **Dirigido por casos de uso:** Los casos de uso describen los requisitos funcionales del sistema desde la perspectiva del usuario y se usan para determinar el alcance de cada iteración y el contenido de trabajo de cada persona del equipo de desarrollo.
- **Centrado en la arquitectura:** La arquitectura permite ganar control sobre el proyecto para manejar su complejidad y controlar su integridad. Hace posible la reutilización a gran escala y provee una base para la gestión del proyecto.
- **Iterativo e incremental:** Se divide en 4 fases: Inicio, Elaboración, Construcción y Transición, y cada una de ellas se divide en iteraciones. Cada iteración realizada añade funcionalidades al producto de software o mejora las existentes. Además en cada iteración se trabaja en un número de disciplinas haciendo énfasis en algunas de ellas. Las disciplinas propuestas por RUP son: Modelado del negocio, Requisitos, Análisis y Diseño, Implementación, Pruebas, entre otras. (22)

Se definió RUP como metodología pues constituye un estándar en el desarrollo del software actualmente, de forma que se adapta a un amplio rango de proyectos y organizaciones. Representa una guía en el desarrollo de proyectos que requieren seguimiento y control. Al estar dividido en fases brinda una mayor organización del trabajo. Es la metodología más utilizada para el análisis, diseño, implementación y documentación de sistemas orientados a objetos. Permite además obtener un producto con calidad.

Se considera que la utilización de RUP puede ser de suma importancia para el desarrollo de la aplicación de forma eficiente y con alta productividad debido a que define qué se tiene que hacer, cómo,quién y cuándo lo hace en cada momento del proceso de desarrollo.

También es política del departamento de Seguridad Informática del centro de Telemática la utilización de esta metodología de desarrollo para el proceso de implementación de todos los productos de software. (23)

1.3 Conclusiones

Durante la realización de este capítulo se hizo un estudio sobre las inyecciones SQL, las técnicas y metodologías existentes para su tratamiento. Por último se justificó la selección de las metodologías, tecnologías y herramientas necesarias para desarrollar la herramienta de apoyo a la Metodología de mitigación de inyecciones SQL.

CAPÍTULO 2: DESARROLLO DE LA MÉTODOLOGÍA PARA MITIGAR INYECCIONES SQL.

2.1 Introducción

En este capítulo se definirán las fases necesarias para desarrollar un filtro de inyecciones SQL con técnicas de Minería de Datos. Se desarrollará cada fase de la metodología propuestas, así como los roles y artefactos necesarios.

2.2 Metodología para Mitigar Inyecciones SQL

La metodología define procedimientos y métodos para guiar el proceso de desarrollo. En la propuesta que se realiza, se tiene en cuenta una combinación de los sistemas de evaluaciones de riesgos con la metodología de Minería de Datos CRISP-DM (Chapman y otros, 2007). En cada una de las fases se van a generar artefactos, los cuales son las entradas para las próximas fases.

Un artefacto no es más que una pieza de información tangible que es creada, modificada y usada por los trabajadores al realizar actividades; representa un área de responsabilidad (Jacobson y otros, 1999). Los artefactos definidos se describirán en cada fase donde sean utilizados.

Para desarrollar la metodología propuesta es necesario que se cuente con conocimientos en el área de la Minería de Datos y de Seguridad Informática. Para esto, la metodología define dos roles: el Especialista de Minería de Datos y el Especialista de Seguridad Informática. Los roles no son más que el comportamiento específico de una entidad que participa en un contexto particular (Jacobson y otros, 1999).

Especialista de Minería de Datos: tiene la responsabilidad de llevar a cabo los procesos definidos por la metodología, guiar el proceso de Minería de Datos y dar cumplimiento satisfactorio al modelo realizado.

Especialista de Seguridad Informática: tiene la responsabilidad de llevar a cabo los procesos definidos por la metodología, guía el proceso de Mitigación de Riesgos y dar cumplimiento satisfactorio al filtrado de inyecciones SQL. Debe tener conocimiento sobre las inyecciones SQL.

Los procesos de Minería de Datos y Mitigación de Riesgos, son líneas entrelazadas en la metodología, que requieren del trabajo conjunto del especialista de Minería de Datos y del especialista de Seguridad Informática. Definen las fases y los elementos necesarios para llevarlas a cabo.

Para esta metodología se han definido seis fases que se explican a continuación:

1) **Comprensión del Negocio**. Tiene como objetivo recopilar información sobre los objetivos, funcionamiento y tecnologías utilizadas en la aplicación Web.

2) **Análisis de Riesgos del Negocio**. Tiene como objetivo verificar el nivel de riesgo, en cuanto a inyecciones SQL, de la aplicación Web; argumentar la necesidad del firewall de aplicaciones Web y del de Base de Datos.

3) **Colección y Preparación de los Datos**. Tiene como objetivo recolectar y preparar los datos necesarios para el proceso de aprendizaje del modelo para filtrar inyecciones SQL. Estos datos dependerán del servidor de aplicaciones utilizado y del gestor de bases de datos.

4) **Modelado**. Tiene como objetivo definir los procesos y las técnicas de Minería de Datos que más se ajustan a la solución del problema.

5) **Evaluación**. Tiene como objetivo verificar el buen funcionamiento del modelo definido.

6) **Despliegue**. Tiene como objetivo desplegar el modelo en un firewall de aplicaciones Web y en uno de Bases de Datos. Se verifica el funcionamiento del modelo en un entorno real.

Esta metodología, para un correcto desarrollo, aplica el flujo de trabajo que se ilustra en la figura 1.

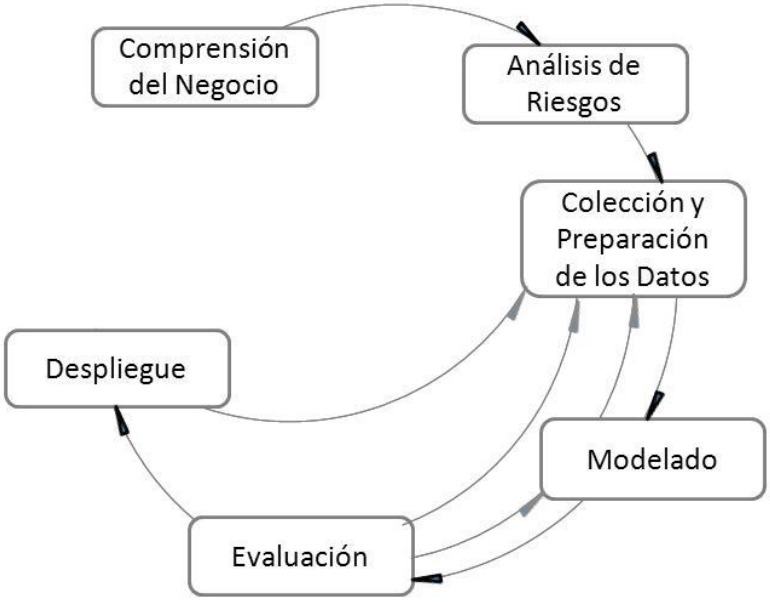


Figura 1: Diagrama de flujo de la metodología.

2.2.1 Comprensión del Negocio

El principal objetivo en esta fase es entender, desde una perspectiva de negocio, lo que se quiere lograr. Un mal entendimiento de los objetivos deseados, produce respuestas correctas a preguntas incorrectas, o erróneas. Esto pudiera llevar a un fracaso en el alcance de los objetivos perseguidos.

Los especialistas que pondrán en práctica la metodología propuesta, tienen esclarecido cuáles son los objetivos perseguidos: mitigar inyecciones SQL. La metodología guía el proceso de cómo se va a lograr realizar la mitigación de esta amenaza. En este caso, es necesario que los especialistas comprendan el negocio de la aplicación Web, ya que no tienen por qué haber sido miembros del equipo de desarrollo del sistema, o pertenecer a la entidad propietaria del sitio. Es importante para otras fases siguientes, que se registren las tecnologías con que se desarrollaron el sistema Web. Si es posible, sería de suma importancia obtener la documentación de toda la información generada durante el proceso de desarrollo del sistema.

Con lo expresado en el párrafo anterior, se hace necesaria la creación de un artefacto denominado “Plan de Desarrollo”, que registre:

- Los objetivos del sistema Web.
- Los lenguajes, tecnologías y *Frameworks* utilizados en su desarrollo.
- Medios asignados para el desarrollo.
- Identificación de bases de datos donde se encuentren registros de inyecciones SQL.

Este artefacto es la salida de esta fase y la entrada a la siguiente.

En esta fase la responsabilidad de ambos especialista se encuentra al mismo nivel, ya que en ella se sientan las bases del desarrollo del modelo de filtrado de inyecciones SQL. La recogida de la información puede realizarse mediante entrevistas a los especialistas de las tecnologías de la entidad correspondiente.

2.2.2 Análisis de Riesgos del Negocio

Para la evaluación de los riesgos se propone utilizar el sistema de Análisis de Riesgos CWRAF en su versión 0.8 (MITRE, 2011). Para lograr que esta información permanezca persistente, se define el artefacto “Análisis de Riesgos del Negocio”. La mayor responsabilidad sobre este proceso, la desempeñará el especialista de seguridad informática.

Este se hace necesario porque cada negocio o empresa, tiene diferentes prioridades, diferentes entornos de amenazas y diferentes tolerancia a riesgos. Para resolver esta problemática CWRAF, define los conceptos que se describen a continuación.

Primer paso: definir el Dominio del Negocio.

El Dominio del Negocio, típicamente cubre todos los tipos de industria o sectores industriales. Deben contener procesos que sean controlados mediante *software* y requieran algún grado de seguridad en sus operaciones. Ejemplo de dominios de negocio son: Comercio electrónico, Salud Pública, Energía, Telecomunicaciones. Deben poseer algún *software* para poder hacerse un Análisis de Riesgos con CWRAF.

Segundo paso: definición del Grupo Tecnológico y Modelos Tecnológicos (MT).

Existen Modelos Tecnológicos que pueden ser usados en varios dominios de negocios. También hay debilidades inherentes a Modelos Tecnológicos, ejemplo: las inyecciones SQL son inherentes a los modelos basados en la Web. En la tabla 1 se muestran algunos ejemplos para un buen entendimiento de los Grupos Tecnológicos y los Modelos Tecnológicos.

Tabla 1: Ejemplos de Grupos Tecnológicos y sus Modelos Tecnológicos correspondientes.

Grupo Tecnológico	Modelos Tecnológicos (MT)
Tecnologías Web	Navegadores Web, Servidores Web, Aplicaciones y Servicios basados en la Web.
Sistemas de Control Industrial	Navegadores Web, Servidores Web, Aplicaciones y Servicios
Sistemas Operativos	Sistema Operativo de Propósito General, Virtualización de SO, Sistema de Operaciones en Tiempo Real, kernel (núcleo de un sistema operativo).

Tercer paso: definición del Escenario de Interacción.

El Escenario de Interacción provee una formalización de dirigida a definir un particular entorno del dominio del negocio. Esto incluye los roles que juegan los *software* dentro del entorno, y las prioridades de la organización en cuanto a la seguridad de estos.

Los escenarios de interacción le permiten al CWRAF soportar diversos contextos, los cuales pueden tener diferentes requerimientos para priorizar las debilidades de los software.

Cuarto paso: definición de los Valores del Contexto del Negocio (VCN).

Un parte importante de los Escenarios de Interacción, son los Valores del Contexto del Negocio (VCN). Estos constan de tres elementos fundamentales:

- Objetivo de seguridad a tener cuenta para garantizar la seguridad de un sistema. Ejemplos: Integridad, Confidencialidad y Disponibilidad.
- Una descripción general de las evaluaciones e interacciones de los modelos relevantes para la seguridad, que son concernientes al dominio de negocio.
- Prioridades de seguridad del Dominio del Negocio con respecto a los potenciales suceso que pueden ocurrir, debido a un ataque exitoso.

Quinto paso: Unión de los valores del negocio y las debilidades.

Para realizar el análisis del impacto de las debilidades en CWRAF, son necesarios los siguientes elementos:

- Impacto: el tipo de Técnica de Impacto a considerar.

- Importancia: es un valor entre 0 y 10 que cuantifica el impacto de cualquier debilidad que puede ser explotable, basado en una capa. También se le refiere como el subvalor.
- Explicación: es una explicación asociada al impacto del negocio, si la debilidad correspondiente ha sido explotada.

Sexto paso: Toma de decisiones.

Después que se adquirió la priorización de mitigar las debilidades. Los especialistas en conjunto con los directivos de la entidad, deben definir la necesidad o no de la implantación de un Firewall Semántico de Aplicación Web y de Base de Datos.

2.2.3

2.2.4 Colección y Preparación de los Datos

Para esta metodología, la comprensión de los datos no lleva una carga relevante, ya que se cuenta con un especialista de seguridad informática y este debe conocer sobre el formato de las inyecciones SQL.

Colección de Datos

Como requisito a la colección de los registros de inyecciones SQL, se encuentran las tecnologías Web utilizadas en el desarrollo del sistema. Esto permitirá, localizar inyecciones SQL propias para estas tecnologías. La recolección de estos datos se puede realizar a través de las bases de datos de vulnerabilidades.

Preparación de Datos.

Para el proceso de preparación de los datos, es necesario, que de todos los registros obtenidos se seleccionen los más adecuados para el análisis. Los criterios incluyen la importancia a los objetivos de la Minería de Datos, la calidad, y las restricciones técnicas como límites sobre el volumen de datos o los tipos de datos.

Pudiera ser necesario llevar a cabo una limpieza de datos. Elevar la calidad de los datos al nivel requerido por las técnicas de análisis seleccionadas. Esto puede implicar la selección de los subconjuntos de datos limpios, la inserción de datos por defectos adecuados, o técnicas más ambiciosas tales como: la estimación de datos faltantes mediante modelado.

Como salida de esta fase, estarían los registros de datos listos para iniciar el proceso de modelado. Es necesario elaborar un documento donde se registren los datos modificados, por si es necesario deshacer algún cambio el proceso sea más sencillo. En este documento, también se debe guardar los nombres de las bases de datos de donde se obtuvieron los registros de inyecciones SQL.

2.2.5 Modelado

Esta fase se desarrollará en tres procesos fundamentales:

- Identificación de los términos.
- Selección de la técnica de modelado.
- Construcción del modelo.

Identificación de los términos

Debido a que una inyección SQL, no es más que una cadena de caracteres, se hace necesario identificar los términos de la sentencia para poder confeccionar el modelo correspondiente. Para la confección de la relación de términos, serán de mucha utilidad las tecnologías utilizadas para el desarrollo de la Web. A través de los intérpretes de las tecnologías Web, se puede identificar los términos y definir grupos o clases. Para llevar a cabo este proceso se desarrollan dos tareas: Construcción del glosario de términos y Selección de variables.

En la construcción del glosario de términos, se identifican todos los términos que tienen relación con las inyecciones SQL y las tecnologías utilizadas por los clientes. Posteriormente, de estos términos se seleccionan los más apropiados como variables para el proceso de filtrado de inyecciones SQL.

Selección de la técnica de modelado.

El objetivo de este modelo es lograr filtrar inyecciones SQL con técnicas de minería de datos. Como primer paso del modelado es necesario identificar las técnicas de minería de datos más adecuadas para solucionar el problema. Esto permitirá que dada las técnicas se identifiquen las herramientas que las contengan. La identificación de las técnicas eficaces para la resolución de este problema pudiera ser basada en una comparativa, y de ellas, definir la que más se acomoda a las condiciones de recursos.

Construcción del modelo.

Ejecutar la herramienta de minería seleccionada, sobre el conjunto de datos preparados para crear el filtro de inyecciones SQL.

Como salida de esta fase, además del modelo, es necesaria la confección de un artefacto llamado “Procesos del Modelado”, donde se escriba de forma explícita el glosario de términos definido por las tecnologías utilizadas en el desarrollo del sistema, y las técnicas y procesos de Minería de Datos definidas para la realización del modelo.

2.2.6 Evaluación

Este paso evalúa el grado al que el modelo responde a los objetivos de negocio. Una opción de evaluación es probar el modelo sobre una aplicación de prueba. La aplicación de pruebas puede consistir en suministrarle al modelo, inyecciones SQL conocidas, y que no estaban en su base de conocimiento. De esta forma, se puede evaluar la efectividad del modelo en función de los porcientos alcanzado por el filtrado de inyecciones SQL. También es necesario probar sentencias comunes para el sistema, así se logra validar que el modelo permite el funcionamiento normal de este.

Esta fase genera el artefacto “Evaluación”, donde se registra el proceso de evaluación del modelo definido. Al concluir esta fase, se pudiera pretender una mejora en el modelo definido. Esta metodología facilitaría ese proceso. Se definiría que los especialistas puedan empezar nuevamente desde la fase de Colección y preparación de los datos, siembre que no hayan ocurridos cambios en el negocio ni en las tecnologías utilizadas en el sistema.

2.2.7 Despliegue

Las intenciones de esta fase de despliegue, es que este modelo obtenido, para filtrar de inyecciones SQL, sea incorporado a un firewall de aplicaciones Web o de base de datos, convirtiéndolos en un firewall semántico. Es necesario realizar varias pruebas del funcionamiento del modelo desplegado con el firewall. Todos los procesos desarrollados en la fase de despliegue, deben ser recogidos en un artefacto llamado “Despliegue”.

Concluido el desarrollo de todas las fases de esta metodología, se pudiera pretender una revisión de todos sus procesos, para perfeccionar y mejorar el modelo obtenido, así como todos los artefactos generados en la aplicación de la metodología.

2.3 Conclusiones

En este capítulo se definió cada una de las fases correspondiente a la Metodología de Mitigación de Inyecciones SQL. Esta metodología tiene la relevancia de conducir a la evaluación de la importancia, para un sistema, de mitigar inyecciones SQL; de conducir el proceso para el desarrollo de un modelo, que filtre inyecciones SQL; y desplegar este modelo en un firewall de aplicaciones Web.

CAPÍTULO 3: CARACTERÍSTICAS DEL SISTEMA

3.1 Introducción

Luego de haber analizado en el capítulo anterior las herramientas y tecnologías para el desarrollo de la solución con el objetivo de establecer una visión general de lo que el sistema debe hacer, en el presente capítulo se describen las características del software, concebidas previamente luego de una modelación del entorno en que se desarrolla el mismo. Además se abordarán todos los aspectos relacionados con el dominio de la solución y se enfocarán los procesos involucrados en el campo de acción realizando un análisis de la ejecución del mismo. También se presentan los requisitos funcionales y no funcionales, indispensables en el proceso de desarrollo del software.

3.1.1 Objeto de automatización

Con el objetivo de facilitar la aplicación de la Metodología de mitigación de inyecciones SQL, se propone la realización de una Herramienta de apoyo a dicha metodología. Esta herramienta debe brindar la posibilidad de desarrollar procesos que generan artefactos, los cuales quedan registrados en documentos que muestran un reporte con los resultados.

3.2 Modelo de Dominio

Para el mejor entendimiento de los conceptos manejados en la Metodología se realiza el Modelo de Dominio, el cual captura los tipos de objetos más importantes en el contexto del sistema, ayuda a comprender los conceptos que utilizan los usuarios y con los que deberá trabajar la aplicación. Los objetos del dominio representan los eventos que suceden en el entorno que trabaja el sistema.

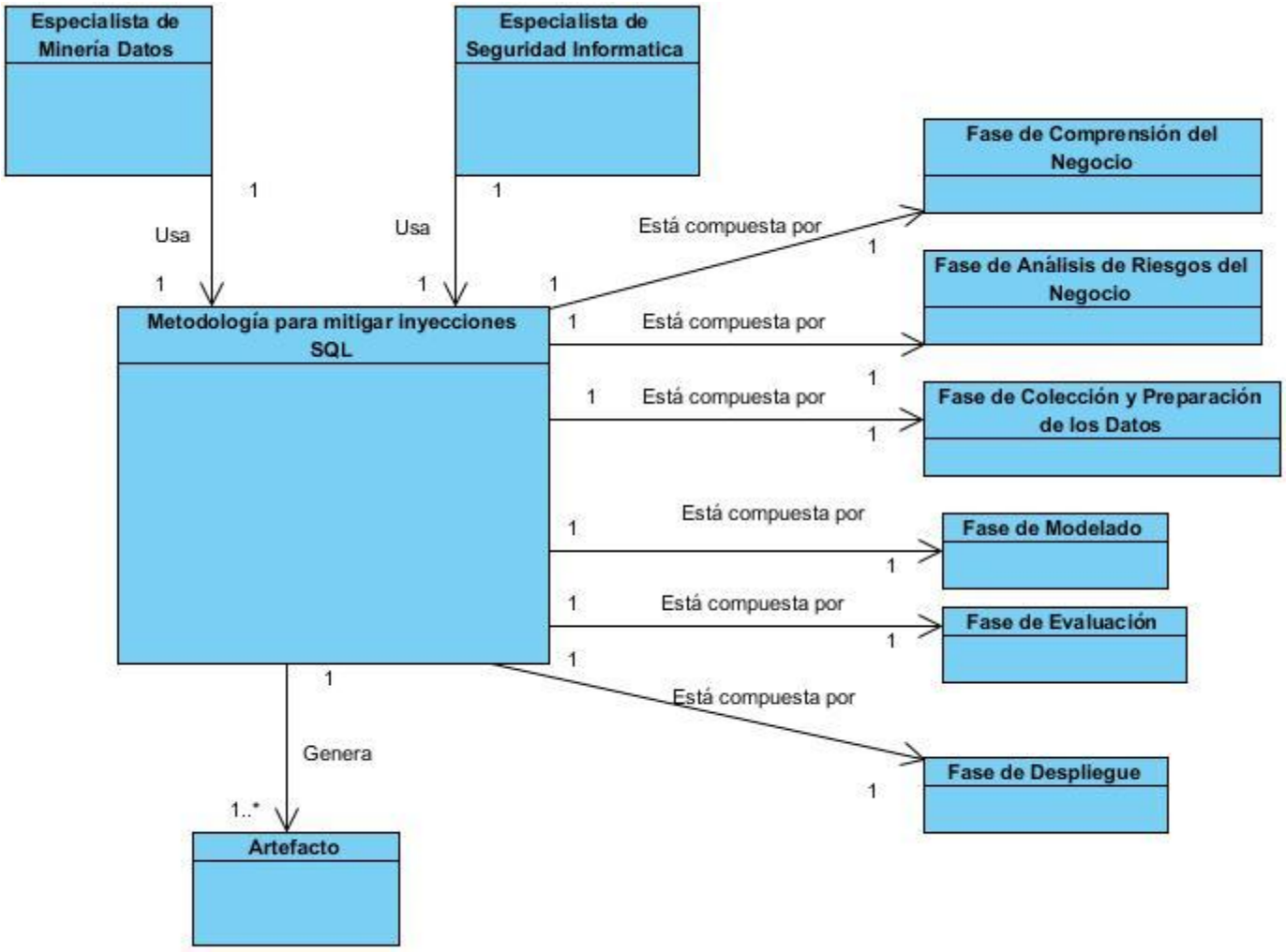


Figura 2: Diagrama de dominio.

3.2.1 Conceptos

Especialista de Minería de Datos: Persona con conocimiento suficiente capaz de llevar a cabo los procesos definidos por la metodología y guía el proceso de Minería de Datos.

Especialista de Seguridad Informática: Persona con conocimiento suficiente capaz de llevar a cabo los procesos definidos por la metodología y guía el proceso de Mitigación de Riesgos.

Metodología para mitigar inyecciones SQL: Metodología para elevar los niveles de seguridad informática en aplicaciones Web mediante el análisis de inyecciones SQL con el empleo de técnicas de minería de datos.

Fase de Comprensión del Negocio: Tiene como objetivo recopilar información sobre los objetivos, funcionamiento y tecnologías utilizadas en la aplicación Web.

Fase de Análisis de Riesgos del Negocio: Tiene como objetivo verificar el nivel de riesgo, en cuanto a inyecciones SQL, de la aplicación Web; argumentar la necesidad del firewall de aplicaciones Web y del de Base de Datos.

Fase de Colección y Preparación de los Datos: Tiene como objetivo recolectar y preparar los datos necesarios para el proceso de aprendizaje del modelo para filtrar inyecciones SQL. Estos datos dependerán del servidor de aplicaciones utilizado y del gestor de bases de datos.

Fase de Modelado: Tiene como objetivo definir los procesos y las técnicas de Minería de Datos que más se ajustan a la solución del problema.

Fase de Evaluación: Tiene como objetivo verificar el buen funcionamiento del modelo definido.

Fase de Despliegue: Tiene como objetivo desplegar el modelo en un firewall de aplicaciones Web y en uno de Bases de Datos. Se verifica el funcionamiento del modelo en un entorno real.

Artefacto: Un artefacto no es más que una pieza de información tangible que es creada, modificada y usada por los trabajadores al realizar actividades.

Para el desarrollo de la aplicación es necesario el levantamiento de requisitos. Estos describen el comportamiento del sistema a desarrollar, donde se incluyen casos de uso los cuales puntualizan todas las interacciones futuras que el usuario tendrá con el producto desarrollar. Los requisitos se clasifican funcionales y no funcionales.

3.3 Definición de los requerimientos funcionales

Los requisitos funcionales no son más que condiciones o capacidades funcionales que el sistema debe tener. Definen actividades internas del software, como manipulación de datos y flujos de información (24). A continuación se muestran los requisitos funcionales de la Herramienta:

RF1: Crear Proyecto: Permite adicionar proyectos a la herramienta de apoyo a la metodología.

1.1: Permite insertar los siguientes datos:

1.1.1: Nombre del proyecto.

1.1.2: Fecha de inicio del proyecto.

1.1.3: Nombre del especialista en Minería de datos.

1.1.4: Nombre del especialista en Seguridad Informática.

1.1.5: Nombre del Supervisor.

RF2: Eliminar Proyecto: Permite eliminar proyectos existentes en la herramienta de apoyo a la metodología.

RF3: Modificar Proyecto: Permite modificar proyectos existentes en la herramienta de apoyo a la metodología.

3.1: Permite modificar los siguientes datos:

3.1.1: Nombre del proyecto.

3.1.2: Fecha de inicio del proyecto.

3.1.3: Nombre del especialista en Minería de datos.

3.1.4: Nombre del especialista en Seguridad Informática.

3.1.5: Nombre del Supervisor.

RF4: Registrar información del sistema**:** Tiene como objetivo recopilar información sobre los objetivos, funcionamiento y tecnologías utilizadas en la aplicación Web.

4.1: Permite registrar los siguientes datos del sistema:

4.1.1: Nombre del sistema.

4.1.2: Tecnologías que utiliza.

4.1.3: Nombre del propietario.

4.1.4: Objetivo del negocio.

4.1.5: Recursos asignados al desarrollo del proyecto.

RF5: Definir el Dominio del Negocio**:** Permite registrar el objeto social de la empresa, compañía, o institución para la cual se desarrolló el proyecto. Ejemplos: Financiero, Salud Pública, Servicio de Emergencia, Telecomunicaciones.

5.1: Permite insertar el siguiente dato:

5.1.1: Dominio del Negocio.

RF6: Definir el Grupo Tecnológico: Permite definir la interrelación de tecnologías que juntas proveen determinadas características, la cual es usada para solucionar un problema determinado existente en un proyecto. Como ejemplo se incluye: las tecnologías Web, almacenamiento de datos, sistemas en tiempo real, sistema operativos.

6.1: Permite insertar el siguiente dato:

6.1.1: Grupo Tecnológico.

RF7: Definir los Modelos Tecnológicos (MT): Permite definir los componentes, sistemas o arquitectura, que son usados para soportar la misión de una organización en particular existente en un proyecto.

7.1: Permite insertar el siguiente dato:

7.1.1: Modelos Tecnológicos.

RF8: Definir el Escenario de Interacción: Es un escenario donde se relaciona el Modelo Tecnológico aplicado para dar cumplimiento a una función del Dominio del Negocio correspondiente.

8.1: Permite insertar los siguientes datos:

8.1.1: Nombre

RF9: Definir los Valores del Contexto del Negocio (VCN): Permite definir una descripción de las evaluaciones relevantes de seguridad sobre un Escenario de Interacción, combinado con las prioridades de seguridad del Dominio del Negocio. Los Valores del Contexto del Negocio forman un puente entre la seguridad concerniente al Dominio del Negocio de potenciales debilidades que tiene el sistema para el cual se aplica la metodología.

9.1: Permite insertar los siguientes datos:

9.1.1: Objetivos de Seguridad a tener cuenta para garantizar la seguridad de un sistema. Ejemplos: Integridad, Confidencialidad y Disponibilidad.

9.1.2: Descripción General de las evaluaciones e interacciones de los modelos relevantes para la seguridad.

9.1.3: Prioridad de Seguridad del Dominio del Negocio con respecto a los potenciales sucesos que pueden ocurrir.

RF10: Unir los valores del negocio y las debilidades: Permite definir la unión de los valores del negocio y las debilidades del sistema.

10.1: Permite insertar los siguientes datos:

10.1.1: Técnica de impacto a considerar.

10.1.2: Importancia del impacto de cualquier debilidad que puede ser explotable.

10.1.3: Explicación asociada al impacto del negocio.

RF11: Definir fuente de Datos**:** Permite definir cuáles son las fuentes de datos para la recolección y preparación del proceso de aprendizaje del modelo para filtrar inyecciones SQL.

11.1: Permite insertar los siguientes datos:

11.1.1: Nombre de la fuente de datos

11.1.2: Fecha en que se toman los datos.

11.1.3: Número de registros que tiene para esa tecnología.

11.1.4: URL de donde se tomó esa fuente de datos.

RF12: Adicionar términos: Tiene como objetivo adicionar los términos que serán utilizados para la confeccionar el modelo.

12.1: Permite adicionar los siguientes datos:

12.1.1: Términos de las tecnologías de desarrollo Web.

12.1.2: Descripción general de los términos definidos.

RF13: Eliminar términos: Tiene como objetivo eliminar los términos seleccionados para confeccionar el modelo.

RF14: Modificar términos: Tiene como objetivo modificar los términos seleccionados para confeccionar el modelo.

14.1: Permite modificar los siguientes datos:

14.1.1: Términos de las tecnologías de desarrollo Web.

14.1.3: Descripción general de los términos definidos.

RF15: Adicionar Técnica de Modelado: Permite adicionar la técnica de modelado a emplear para el desarrollo del modelo.

15.1: Permite adicionar los siguientes datos:

15.1.1: Nombre de las técnicas de modelado.

15.1.2: Gráfica de la aplicación de la tecnología de modelado.

RF16: Eliminar Técnica de Modelado: Permite eliminar la técnica de modelado escogida.

RF17: Modificar Técnica de Modelado: Permite modificar la técnica de modelado escogida.

17.1: Permite modificar los siguientes datos:

17.1.1: Nombre de las técnicas de modelado.

17.1.2: Gráfica de la aplicación de la tecnología de modelado.

RF18: Adicionar herramienta de Minería de Datos: Permite adicionar la herramienta de Minería de Datos a utilizar.

18.1: Permite adicionar los siguientes datos:

18.1.1: Nombre de la herramienta de Minería de Datos.

18.1.2: Versión de la herramienta.

RF19: Eliminar herramienta de Minería de Datos: Permite eliminar la herramienta de Minería de Datos escogida.

RF20: Modificar herramienta de Minería de Datos: Permite modificar la herramienta de Minería de Datos escogida.

20.1: Permite modificar los siguientes datos:

20.1.1: Nombre de la herramienta de Minería de Datos.

20.1.2: Versión de la herramienta.

RF21: Evaluar mediante la Matriz de Confusión: Tiene como objetivo indicar si el sistema está confundiendo dos clases.

21.1: Permite adicionar los siguientes datos:

21.1.1: TP: Verdadero Positivo.

21.1.2: FP: Falso Positivo.

21.1.3: TN: Verdadero Negativo.

21.1.4: FN Falso Negativo.

RF22: Adicionar datos del despliegue: Tiene como objetivo adicionar los componentes necesarios para realizar el despliegue del proyecto.

22.1: Permite adicionar:

22.1.1: Tecnología sobre la cual se va a aplicar la base de conocimiento.

22.1.2: Descripción de la tecnología adicionada.

RF23: Exportar información a un documento PDF del proyecto completo: Tiene como objetivo registrar en un documento PDF la información del proyecto completo.

3.4 Definición de los requerimientos no funcionales

Los requerimientos no funcionales son propiedades o cualidades que el producto debe tener. Estas propiedades constituyen las características que hacen al producto atractivo, usable, rápido o confiable. En muchos casos dichos requisitos son fundamentales en el éxito del producto (25). Los requerimientos no

funcionales para el desarrollo de la Herramienta de apoyo a la Metodología de Mitigación de Inyecciones SQL son:

- **Requisitos de apariencia o interfaz externa.**
  - El sistema contará con una interfaz sencilla, se evitaran las barras de desplazamiento, tanto verticales como horizontales.
  - Se desarrollará un menú que facilitará el acceso a las distintas funcionalidades del sistema.
- **Requisitos de software.**
  - Se requiere que en la computadora se encuentre instalado el intérprete de Python.
- **Requisitos de portabilidad.**
  - La herramienta deberá ser multiplataforma ya que los sistemas web se desarrollan en distintas condiciones.
- **Requisitos de ayuda y documentación.**
  - La herramienta debe contar con un manual de usuarios que contenga las instrucciones de tipo paso a paso, para entender el funcionamiento de esta, así como un listado de definiciones para términos y acrónimos del mismo.

3.5 Modelo de Casos de Uso del sistema

3.5.1 Actores del sistema

Tabla 2: Descripción de los actores del sistema.

Nombre	Justificación
Usuario	El actor Usuario representa una persona que tiene la responsabilidad de llevar a cabo los procesos definidos por la metodología, guiar los procesos de Minería de Datos y de Mitigación de Riesgos, y dar cumplimiento satisfactorio al modelo realizado y al filtrado de inyecciones SQL. Debe tener conocimiento sobre las inyecciones SQL.

3.5.2 Diagrama de Casos de Uso del sistema

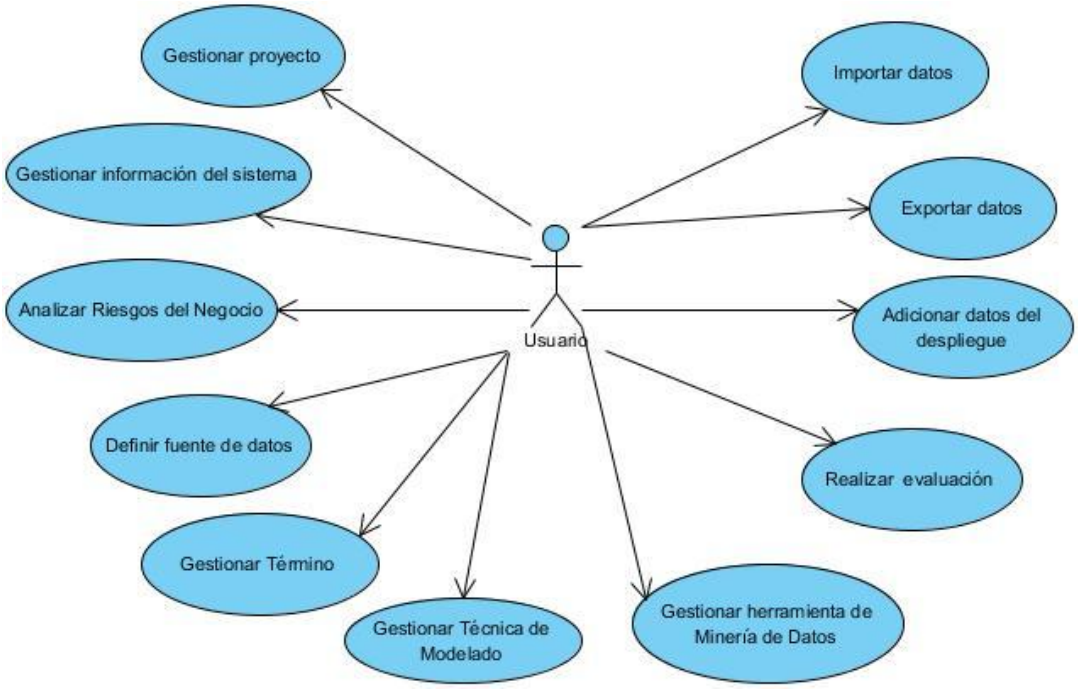


Figura 3: Diagrama de Casos de Uso del sistema.

3.5.3 Descripción de Casos de Uso del sistema

Tabla 3: Descripción del Caso de Uso Gestionar proyecto.

Caso de Uso:	Gestionar proyecto
Actor	Usuario: Gestiona los proyectos a los que se les va a aplicar la Metodología de mitigación de inyecciones SQL.
Resumen	En este caso de uso se brinda la funcionalidad de crear un nuevo proyecto donde se deberá registrar su nombre, fecha de inicio, nombre del especialista en Minería de datos, nombre del especialista en Seguridad Informática y nombre del Supervisor, y se podrá eliminar y modificar un proyecto.
Objetivo	El objetivo de este caso de uso es gestionar los proyectos a los que se les va a aplicar la Metodología de mitigación de inyecciones SQL.

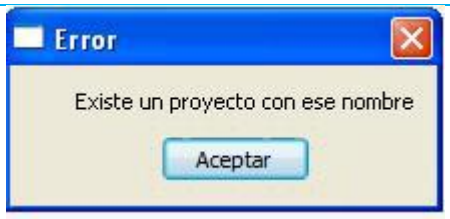
<table>
<tr><td>Precondiciones:</td><td colspan="2">Para modificar la información de los parámetros del proyecto o para eliminar un proyecto debe existir al menos un proyecto creado.</td></tr>
<tr><td>Referencias:</td><td colspan="2">RF1, RF2, RF3</td></tr>
<tr><td>Prioridad:</td><td colspan="2">Crítico</td></tr>
<tr><td colspan="3">Flujo Normal de los Eventos</td></tr>
<tr><td colspan="2">Acción del Actor:</td><td>Respuesta del Sistema:</td></tr>
<tr><td colspan="2">1. Indica que desea “Crear proyecto”.</td><td>2. Brinda la posibilidad de registrar los datos relativos al proyecto:
Nombre del proyecto.
Fecha de inicio del proyecto.
Nombre del especialista en Minería de datos.
Nombre del especialista en Seguridad Informática.
Nombre del Supervisor.
Y las opciones:
Aceptar.
Cancelar.</td></tr>
<tr><td colspan="2">3. Introduce los datos requeridos.</td><td></td></tr>
<tr><td colspan="2">4. Indica que desea guardar los cambios</td><td>5. Verifica que no existan campos obligatorios vacíos.</td></tr>
<tr><td colspan="2"></td><td>6. Verifica que no exista un proyecto con el mismo nombre.</td></tr>
<tr><td colspan="2"></td><td>7. Almacena la información.</td></tr>
<tr><td colspan="3">Prototipo de Interfaz</td></tr>
</table>

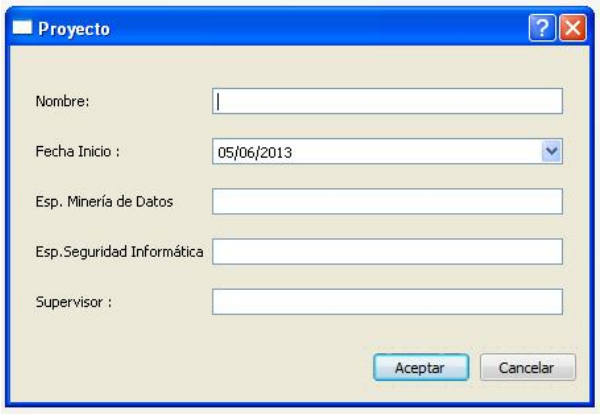
Flujo Alterno 2a "Opción Cancelar".	
Acción del Actor:	**Respuesta del Sistema:**
2a1. Indica que desea cancelar.	2a2. Cierra la interfaz Crear proyecto.
Flujos Alternos 5a "Campos obligatorios vacíos"	
Acción del Actor:	**Respuesta del Sistema:**
	5a1. Muestra un mensaje de error "Debe completar los campos vacíos".

Prototipo de Interfaz

Flujo Alterno 6a "Existe un proyecto con ese nombre"	
Acción del Actor:	**Respuesta del Sistema:**
	6a1. Muestra un mensaje de error "Existe un proyecto con ese nombre".

Prototipo de Interfaz

<table>
<tr><td colspan="2"> 
</td></tr>
<tr><td colspan="2">Sección “Modificar proyecto”.</td></tr>
<tr><td>Acción del Actor:</td><td>Respuesta del Sistema:</td></tr>
<tr><td>1. Indica que desea “Modificar proyecto”.</td><td>2. Brinda la posibilidad de seleccionar el proyecto que desea modificar.</td></tr>
<tr><td>3. Indica el proyecto al cual se le va a modificar la información.</td><td>4. Brinda la posibilidad de modificar los datos relativos al proyecto:
Nombre del proyecto.
Fecha de inicio del proyecto.
Nombre del especialista en Minería de datos.
Nombre del especialista en Seguridad Informática.
Nombre del Supervisor.
Y las opciones
Aceptar.
Cancelar.</td></tr>
<tr><td>5. Modifica la información de los parámetros deseados.</td><td></td></tr>
<tr><td>6. Indica que desea guardar los cambios.</td><td>7. Verifica que no existan campos vacíos.</td></tr>
<tr><td></td><td>8. Si el usuario modificó el nombre del proyecto, verifica que el nuevo nombre no coincida con el nombre de los demás proyectos que están almacenados.</td></tr>
<tr><td></td><td>9. Guarda las actualizaciones realizadas.</td></tr>
<tr><td colspan="2">Prototipo de Interfaz</td></tr>
</table>

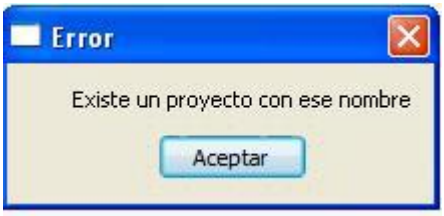


Flujo Alterno 4a "Opción Cancelar".	
Acción del Actor:	**Respuesta del Sistema:**
4a1. Indica que desea cancelar.	4a2. Cierra la interfaz Proyecto.
Flujo Alterno 7a "Campos vacíos"	
	7a1. Muestra un mensaje de error "Debe completar los campos vacíos".

Prototipo de Interfaz

Flujo Alterno 8a " Existe un proyecto con ese nombre "	
	8a1. Muestra un mensaje de error "Existe un proyecto con ese nombre".

Prototipo de Interfaz



Sección "Eliminar proyecto".	
Acción del Actor:	**Respuesta del Sistema:**

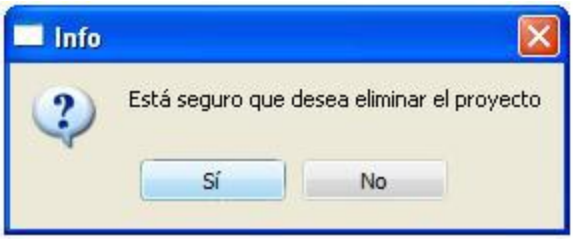
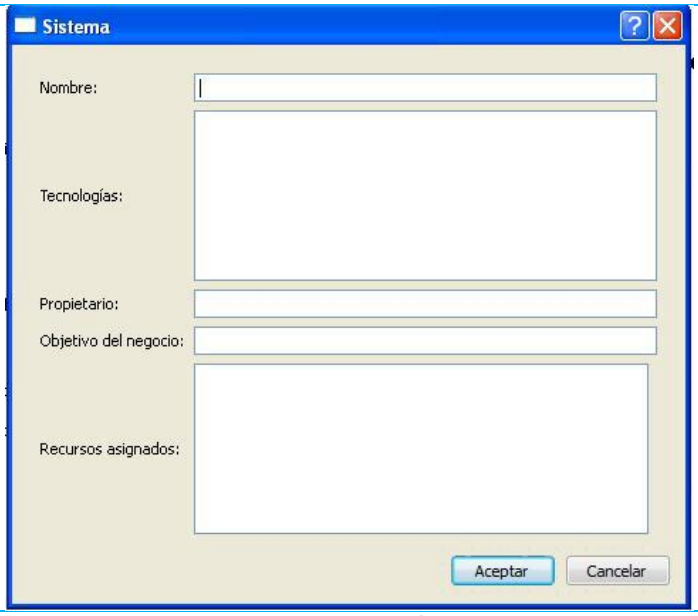
1. Indica que desea “Eliminar proyecto”.	2. Brinda la posibilidad de seleccionar el proyecto que desea eliminar. Y la opción: • Eliminar
3. Indica el proyecto que desea eliminar.	4. Muestra un mensaje de confirmación: “Está seguro que desea eliminar el proyecto” Y las opciones Si y No
5. Indica que desea eliminar el proyecto seleccionado.	6. Almacena los nuevos cambios.
Prototipo de Interfaz 	
Flujos Alternos 5a Opción “No”	
Acción del Actor:	**Respuesta del Sistema:**
5a1. Indica que no desea eliminar el proyecto	5a.2. Cancela la opción.

Tabla 4: Descripción del Caso de Uso Registrar información del sistema.

Caso de Uso:	Registrar información del sistema
Actor	Usuario: Recopilar información sobre los objetivos, funcionamiento y tecnologías utilizadas en la aplicación Web.
Resumen	En este caso de uso se brinda la funcionalidad de recopilar información sobre los objetivos, funcionamiento y tecnologías utilizadas en la aplicación Web.
Objetivo	El objetivo de este caso de uso es recopilar información sobre los objetivos, funcionamiento y tecnologías utilizadas en la aplicación Web.

<table>
<tr><td>Precondiciones:</td><td colspan="2">Para registrar información del sistema debe existir al menos un proyecto creado.</td></tr>
<tr><td>Referencias:</td><td colspan="2">RF4</td></tr>
<tr><td>Prioridad:</td><td colspan="2">Crítico</td></tr>
<tr><td colspan="3">Flujo Normal de los Eventos</td></tr>
<tr><td colspan="2">Acción del Actor:</td><td>Respuesta del Sistema:</td></tr>
<tr><td colspan="2">1. Indica que desea “Registrar información del sistema” en la interfaz.</td><td>2. Brinda la posibilidad de registrar los datos relativos al sistema:
Nombre del sistema.
Tecnologías que utiliza.
Nombre del propietario.
Objetivo del negocio.
Recursos asignados al desarrollo del proyecto.
Y las opciones:
Aceptar.
Cancelar.</td></tr>
<tr><td colspan="2">3. Introduce los datos requeridos.</td><td></td></tr>
<tr><td colspan="2">4. Indica que desea guardar los cambios.</td><td>5. Verifica que no existan campos obligatorios vacíos.</td></tr>
<tr><td colspan="2"></td><td>6. Almacena la información.</td></tr>
<tr><td colspan="3">Prototipo de Interfaz</td></tr>
</table>



Flujo Alterno 2a "Opción Cancelar".

Acción del Actor:	**Respuesta del Sistema:**
2a1. Indica que desea cancelar.	2a2. Cierra la interfaz Sistema.

Flujo Alterno 5a "Campos obligatorios vacíos"

	5a1. Muestra un mensaje de error "Debe completar los campos vacíos".

Prototipo de Interfaz

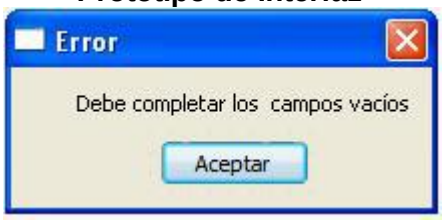


Tabla 5: Descripción del Caso de Uso Analizar Riesgos del Negocio.

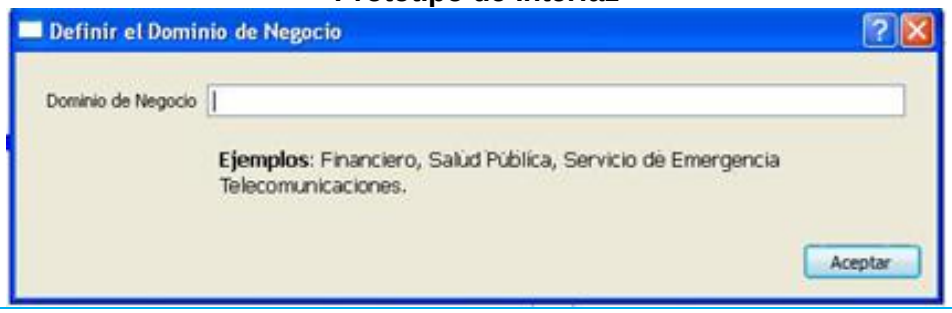
Caso de Uso:	Analizar Riesgos del Negocio
Actor	Usuario: Analiza los riesgos del negocio a los que está expuesto el

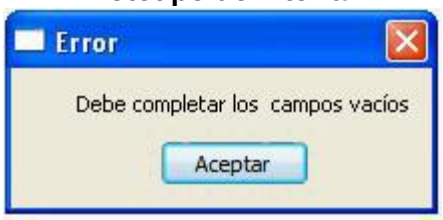
	proyecto.
Resumen	En este caso de uso se brinda la funcionalidad de definir el dominio del negocio**,** definir el grupo tecnológico, los modelos tecnológicos, el escenario de Interacción, los valores del contexto del Negocio y de unir los valores del negocio y las debilidades.
Objetivo	El objetivo de este caso de uso es analizar los riesgos del negocio a los que está expuesto el proyecto.
Precondiciones:	Para analizar los riesgos del negocio de un proyecto es necesario que este haya sido creado.
Referencias:	RF5, RF6, RF7, RF8, RF9, RF10
Prioridad:	Crítico

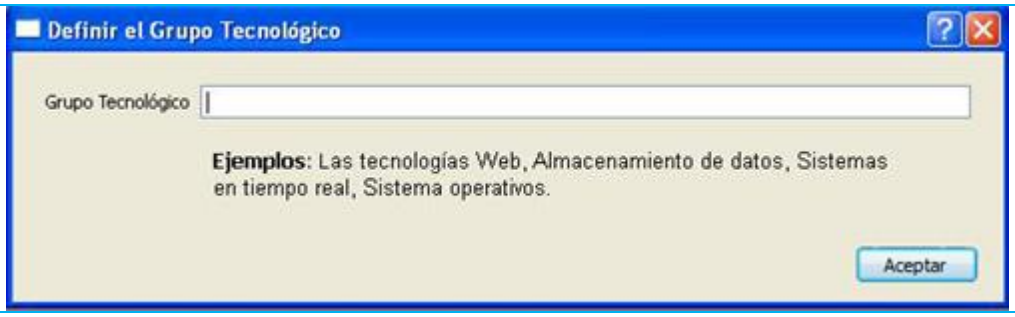
Flujo Normal de los Eventos

Acción del Actor:	**Respuesta del Sistema:**
1. Selecciona la opción “Definir el Dominio de Negocio”.	2. Brinda la posibilidad de registrar los datos relativos al dominio de Negocio, proponiéndole un ejemplo con lo que debe poner (Ejemplos: Financiero, Salud Pública, Servicio de Emergencia, Telecomunicaciones.) Y la opción: ▪ Aceptar.
3. Introduce la información requerida para definir el dominio de negocio.	
4. Indica que desea guardar los cambios.	5. Verifica el campo a llenar no esté vacío.
	6. Almacena la información.

Prototipo de Interfaz



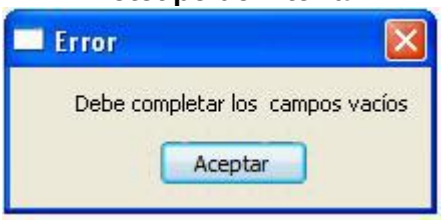
<table>
<tr><th colspan="2">Flujo Alterno 5a ”Campo obligatorio vacío”</th></tr>
<tr><td></td><td>5a1. Muestra un mensaje de error “Debe completar los campos vacíos”.</td></tr>
<tr><td colspan="2">Prototipo de Interfaz
 
</td></tr>
<tr><th colspan="2">Sección “Definir el Grupo Tecnológico”.</th></tr>
<tr><th>Acción del Actor:</th><th>Respuesta del Sistema:</th></tr>
<tr><td>1. Se habilita la opción “Definir el Grupo Tecnológico”.</td><td>2. Brinda la posibilidad de registrar los datos relativos al Grupo Tecnológico. Proponiéndole un ejemplo con lo que debe poner (Ejemplos: las tecnologías Web, almacenamiento de datos, sistemas en tiempo real, sistema operativos.).
Y la opción:
▪ Aceptar</td></tr>
<tr><td>3. Introduce la información requerida para definir el grupo tecnológico.</td><td></td></tr>
<tr><td>4. Indica que desea guardar los cambios.</td><td>5. Verifica que el campo a llenar no esté vacío.</td></tr>
<tr><td></td><td>6. Almacena la información.</td></tr>
<tr><td colspan="2">Prototipo de Interfaz</td></tr>
</table>



Flujo Alterno 5a ”Campo obligatorio vacío”

	5a1. Muestra un mensaje de error “Debe completar los campos vacíos”.

Prototipo de Interfaz



Sección “Definir los Modelos Tecnológicos”.

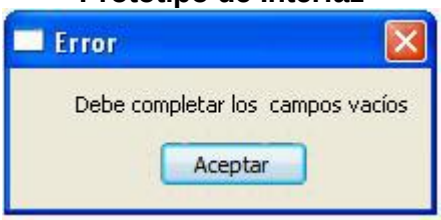
Acción del Actor:	**Respuesta del Sistema:**
1. Se habilita la opción “Modelos Tecnológicos”.	2. Brinda la posibilidad de registrar los datos relativos a los componentes, sistemas o arquitectura, que son usados para soportar la misión de una organización en particular existente en un proyecto. Y la opción: ▪ Aceptar
3. Introduce la información requerida para definir los modelos tecnológicos.	
4. Indica que desea guardar los cambios.	5. Verifica que el campo a llenar no esté vacío.
	6. Almacena la información.

Prototipo de Interfaz

Flujo Alterno 5a ”Campo obligatorio vacío”

	5a1. Muestra un mensaje de error “Debe completar los campos vacíos”.

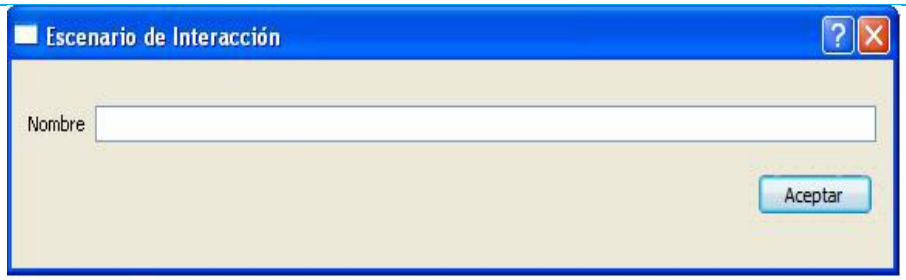
Prototipo de Interfaz



Sección “Definir el Escenario de Interacción”.

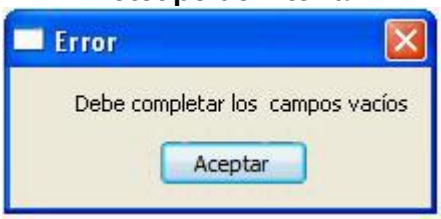
1. Se habilita la opción “Definir el Escenario de Interacción”.	2. Brinda la posibilidad de registrar los datos relativos al escenario de interacción: ▪ Nombre. Y la opción: ▪ Aceptar.
3. Introduce la información requerida para definir el escenario de interacción.	
4. Indica que desea guardar los cambios.	5. Verifica que el campo a llenar no esté vacío.
	6. Almacena la información.

Prototipo de Interfaz



Flujo Alterno 5a ”Campo obligatorio vacío”	
	5a1. Muestra un mensaje de error “Debe completar los campos vacíos”.

Prototipo de Interfaz



Sección “Definir los Valores del Contexto del Negocio (VCN)”.	
Acción del Actor:	**Respuesta del Sistema:**
1. Se habilita la opción “Definir los Valores del Contexto del Negocio (VCN)”.	2. Brinda la posibilidad de registrar los datos relativos a los valores del contexto del negocio: ▪ Objetivos de Seguridad: Son los elementos a tener en cuenta para garantizar la seguridad de un sistema. ▪ Una descripción general de las evaluaciones e interacciones de los modelos relevantes para la seguridad, que son concernientes al dominio de negocio. ▪ Prioridades de seguridad del Dominio del Negocio con respecto a los potenciales suceso que pueden

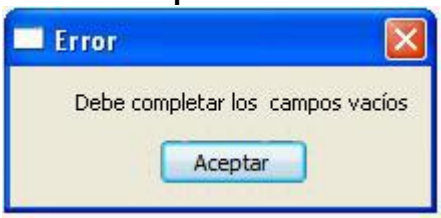
	ocurrir, debido a un ataque exitoso. Y las opciones: ▪ Aceptar
3. Introduce la información requerida para definir el escenario de interacción.	
4. Indica que desea guardar los cambios.	5. Verifica que los campos obligatorios a llenar no estén vacíos.
	6. Almacena la información.

Prototipo de Interfaz

Flujo Alterno 5a "Campos obligatorios vacíos"

	5a1. Muestra un mensaje de error "Debe completar los campos vacíos".

Prototipo de Interfaz

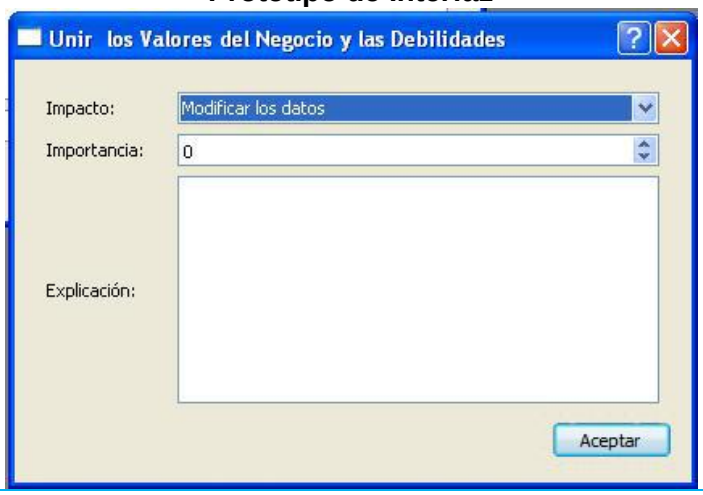


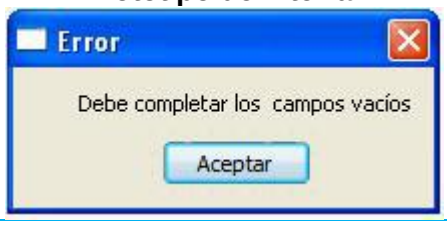
Sección "Unir los valores del negocio y las debilidades".

Acción del Actor:	**Respuesta del Sistema:**
1. Se habilita la opción "Unir los valores del negocio y las debilidades".	2. Brinda la posibilidad de registrar los datos relativos para poder unir los valores del

	negocio y las debilidades: ▪ Impacto: El tipo de Técnica de Impacto a considerar. ▪ Importancia: Es un valor entre 0 y 10 que cuantifica el impacto de cualquier debilidad que puede ser explotable, basado en una capa. También se le refiere como el subvalor. ▪ Explicación: Es una explicación asociada al impacto del negocio, si la debilidad correspondiente ha sido explotada. Y las opciones: ▪ Aceptar
3. Introduce la información requerida para unir los valores del negocio y las debilidades.	
4. Indica que desea guardar los cambios.	5. Verifica que los campos obligatorios a llenar no estén vacíos.
	6. Almacena la información.

Prototipo de Interfaz



Flujo Alterno 7a ”Campos obligatorios vacíos”	
	7a1. Muestra un mensaje de error “Debe introducir los datos para poder Unir los valores del negocio y las debilidades”.
Prototipo de Interfaz 	

3.6 Conclusiones

En este capítulo se arribó a la propuesta de la solución a través de la modelación del dominio y el planteamiento de los requisitos funcionales y no funcionales, también se mostró una representación visual de las funcionalidades a través del Diagrama de Casos de Uso del Sistema, por lo que se obtiene como resultado la base necesaria para la realización del diseño de la herramienta.

CAPÍTULO 4: DISEÑO E IMPLEMENTACIÓN DEL SISTEMA

4.1 Introducción

Luego de analizar en el capítulo anterior los requerimientos funcionales y casos de uso del sistema se procede a analizar si es posible dar una solución que satisfaga los requisitos significativos de la arquitectura. En el presente capítulo se estarán abordando los aspectos relacionados con la construcción de la solución propuesta así como las principales características de los patrones de diseño a utilizar, además del patrón arquitectónico empleado. También se elaborarán los diagramas de clases del diseño del sistema y luego se realizarán las pruebas a la solución presentada. El análisis y diseño contribuyen a una arquitectura sólida y estable que soporte las funcionalidades del sistema correlacionadas a requerimientos funcionales y crean un punto de partida para las actividades de implementación.

4.2 Arquitectura de Software

Las técnicas metodológicas desarrolladas con el fin de facilitar la programación se engloban dentro de la llamada Arquitectura de Software o Arquitectura lógica. Se refiere a un grupo de abstracciones y patrones que brindan un esquema de referencia útil para guiar el desarrollo de software dentro de un sistema informático. Así, los programadores, diseñadores, ingenieros y analistas pueden trabajar bajo una línea común que les posibilite la compatibilidad necesaria para lograr el objetivo deseado.

4.3 Patrón Arquitectónico

El patrón arquitectónico aplicado es el de n capas, pues el sistema está estructurado específicamente en 3 capas: presentación, negocio y acceso a datos, donde el objetivo principal es separar los diferentes aspectos del desarrollo, es decir, las cuestiones de presentación, lógica de negocio y acceso a datos, permitiendo que un cambio en una de las capas no genere cambios en las demás.

La arquitectura del sistema puede verse representada en la siguiente figura, en la capa de presentación se encuentran las clases que tienen la responsabilidad de visualizar el contenido al usuario, la capa de negocio incluye las clases que implementan la lógica del negocio y la capa de acceso a datos contiene las clases que se relacionan con los ficheros y permiten la manipulación de los datos. La comunicación entre

las capas se realiza a través de interfaces, lo que permite la organización de la implementación y la estandarización del código.

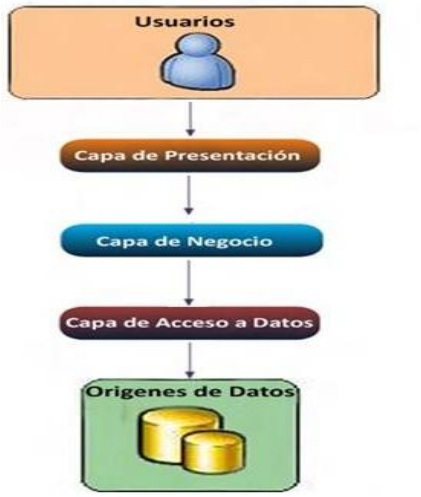


Figura 4: Arquitectura del Sistema.

4.4 Patrones de Diseño

Los patrones de diseño son soluciones de diseño a los problemas recurrentes en la construcción de software. Son a menudo mal interpretados como aplicables sólo a la programación en los grandes sistemas, pero en realidad, se pueden aplicar a la solución de problemas en la programación pequeña como en la implementación de estructuras de datos o algoritmos simples. Los patrones de diseño se pueden combinar en los componentes que resuelven grandes problemas.

Para la construcción del modelo de diseño del sistema fueron utilizados los patrones GRASP (Patrones de asignación de responsabilidades) y GoF (Grupo de los cuatro), ambos dirigidos al desarrollo de sistemas orientados a objetos.

4.4.1 Patrones GRASP

GRASP es un acrónimo de General Responsibility Assignment Software Patterns (Patrones Generales de Software para Asignar Responsabilidades, por sus siglas en inglés).

Los patrones GRASP describen los principios fundamentales del diseño de objetos y la asignación de responsabilidades, expresados como patrones.

4.4.1.1 Experto

Problema: Se necesita lograr que cada clase cumpla con la responsabilidad que le corresponde.

Solución: Asignar una responsabilidad a la clase que tiene la información necesaria para cumplirla.

Implementación: Las clases controladoras de la capa de presentación y las clases de implementación de la capa de negocio cuentan con la información necesaria para cumplir cada una las responsabilidades que le corresponden, por ejemplo la clase **`MainWindow`** se encarga de gestionar toda la información del sistema.

4.4.1.2 Alta Cohesión

Problema: Se necesita lograr que las clases trabajen en su misma área de aplicación.

Solución: Este patrón es una medida de cuán relacionadas y enfocadas están las responsabilidades de una clase. Caracteriza a las clases que están estrechamente relacionadas y consiste en colaborar con otros objetos para compartir el esfuerzo si la tarea a realizar es grande.

Implementación: Este patrón es aplicable en el sistema puesto que las clases tienen pocos métodos y estas se encuentran relacionadas, por ejemplo **`DialogProyecto, DialogSistema`**.

4.4.1.3 Bajo Acoplamiento

Problema: Se necesita lograr una escasa dependencia entre clases.
Solución: Realizar un diseño de clases independientes que puedan soportar los cambios de una manera fácil y permitan la reutilización. Acoplamiento bajo significa que una clase no depende de muchas clases.

Implementación: Al aplicar el patrón arquitectónico n capas se asegura el bajo acoplamiento, de forma tal que las cuestiones de presentación, negocio y acceso a datos están totalmente desligadas, propiciando que un cambio en una capa no afecte a las demás capas inferiores, esto se logra estableciendo la comunicación entre las capas mediante interfaces.

4.4.2 Patrones GoF

Los patrones GoF (Grupo de los cuatro), se agrupan en tres categorías atendiendo las funciones que realizan:

- **Creacionales:** Abarcan los procesos de creación de objetos.
- **Estructurales**: Tratan con la composición de las clases y objetos.

- **De comportamiento:** Caracterizan el modo en que las clases u objetos interactúan y distribuyen responsabilidades.

4.4.2.1 Singleton

Este patrón consiste en crear una única instancia de un objeto para una aplicación y la creación de un mecanismo de acceso global a dicha instancia. Se utiliza por ejemplo en la clase DialogProyecto que representa una única instancia de modo que es accesible desde múltiples objetos del sistema.

4.5 Diagramas de clases del Diseño

Los diagramas de clases del Diseño describen gráficamente las especificaciones de las clases del software y contienen las clases, atributos, métodos y dependencias existentes entre ellas. A continuación se presentan los Diagramas de Clases del Diseño de los casos de uso críticos del sistema. Donde las clases que comienzan ui_nombre son las interfaces que permiten la comunicación entre las capas; mientras que las que comienzan con Dialog (nombre) son las clases que implementan la lógica y el comportamiento de los métodos.

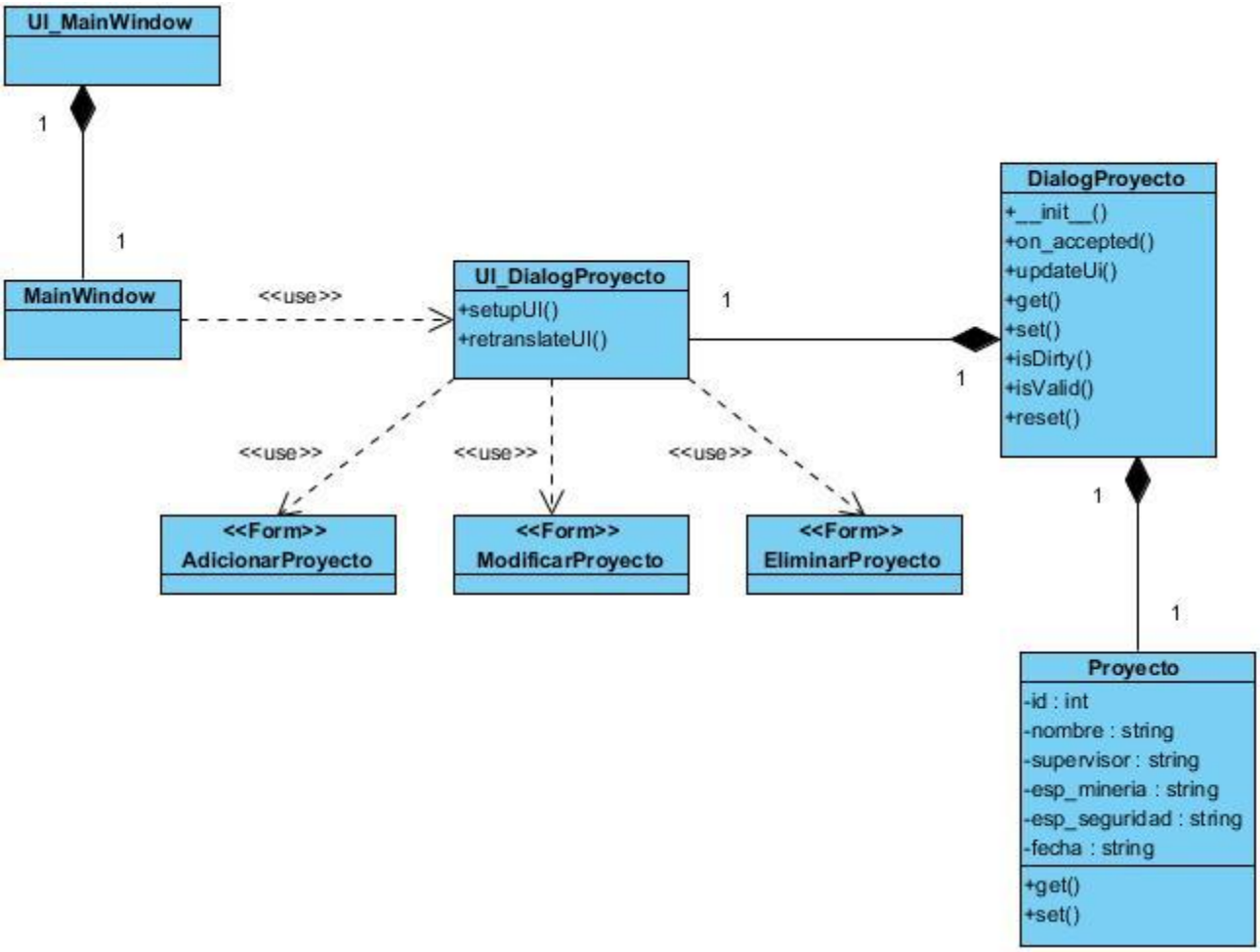


Figura 5: Diagrama de Clases “Gestionar Proyecto”.

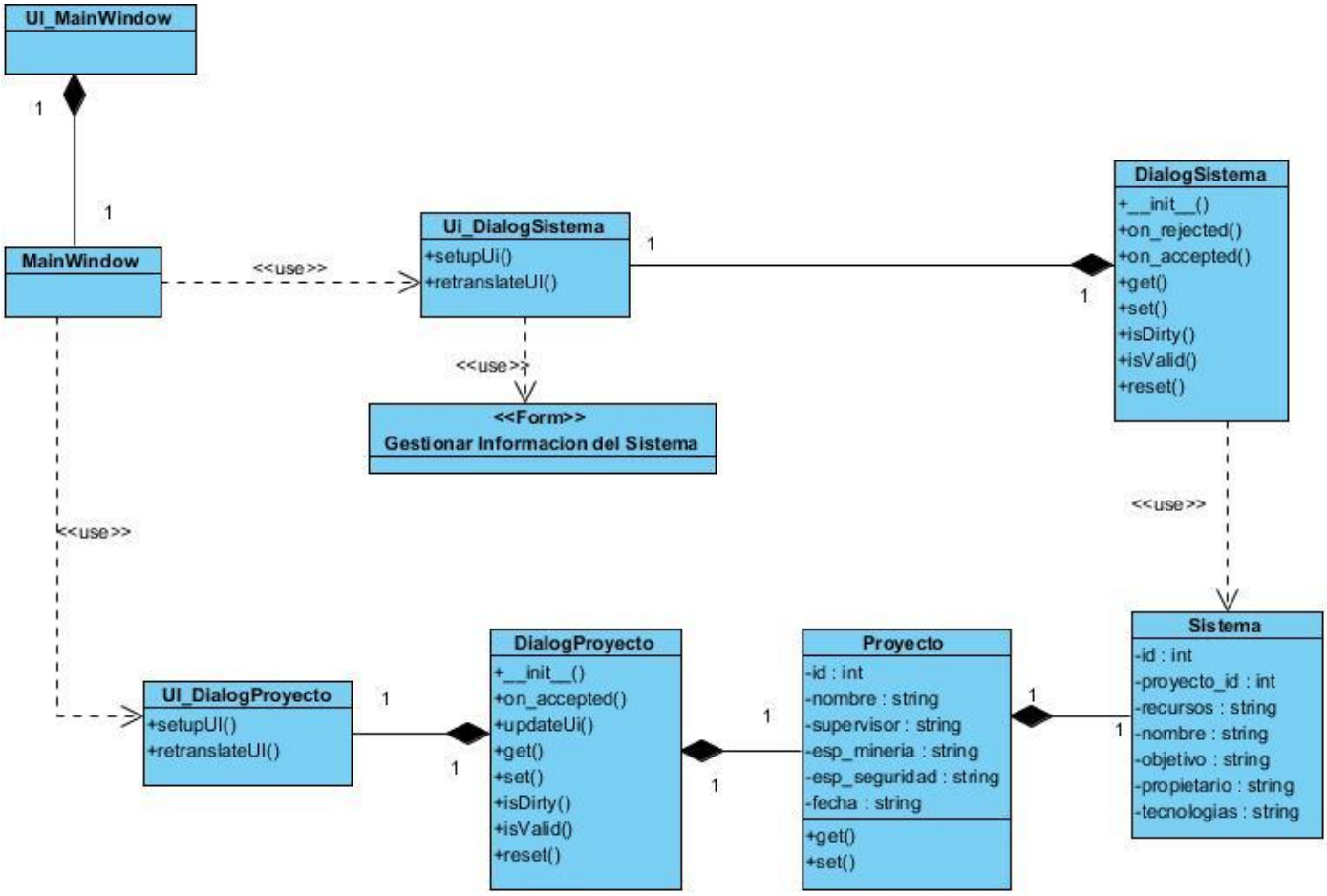


Figura 6: Diagrama de Clases “Gestionar información del sistema”.

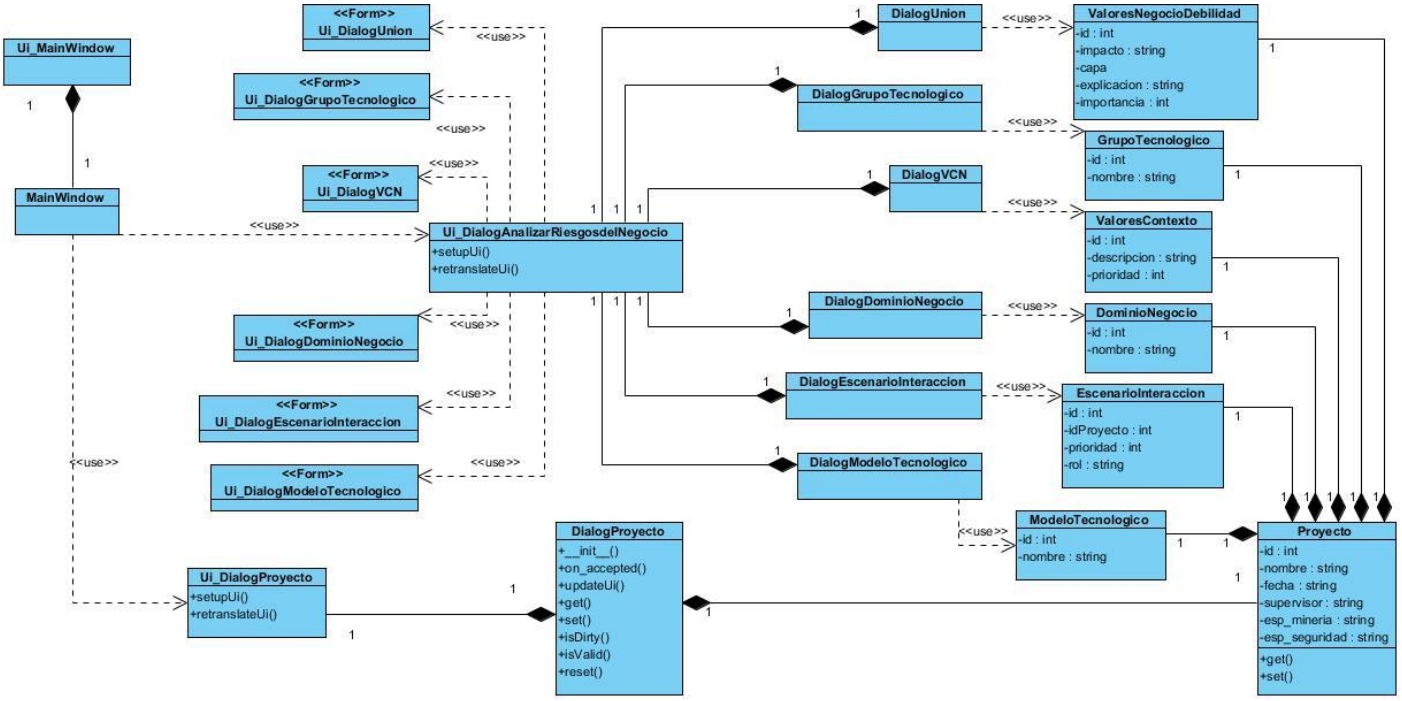


Figura 7: Diagrama de Clases “Analizar Riesgos del Negocio”.

4.6 Modelo físico de datos (Modelo de datos)

El Modelo de Datos representa la base conceptual para diseñar aplicaciones que hacen un uso intensivo de datos, así como la base formal para las herramientas y técnicas empleadas en el desarrollo y uso de sistemas de información, es decir, un Modelo de Datos es la estructura o representación física de las tablas de la Base de Datos. El modelo físico de datos se desarrolló a partir de la base del conjunto de clases persistentes y sus asociaciones en el Modelo de Diseño.

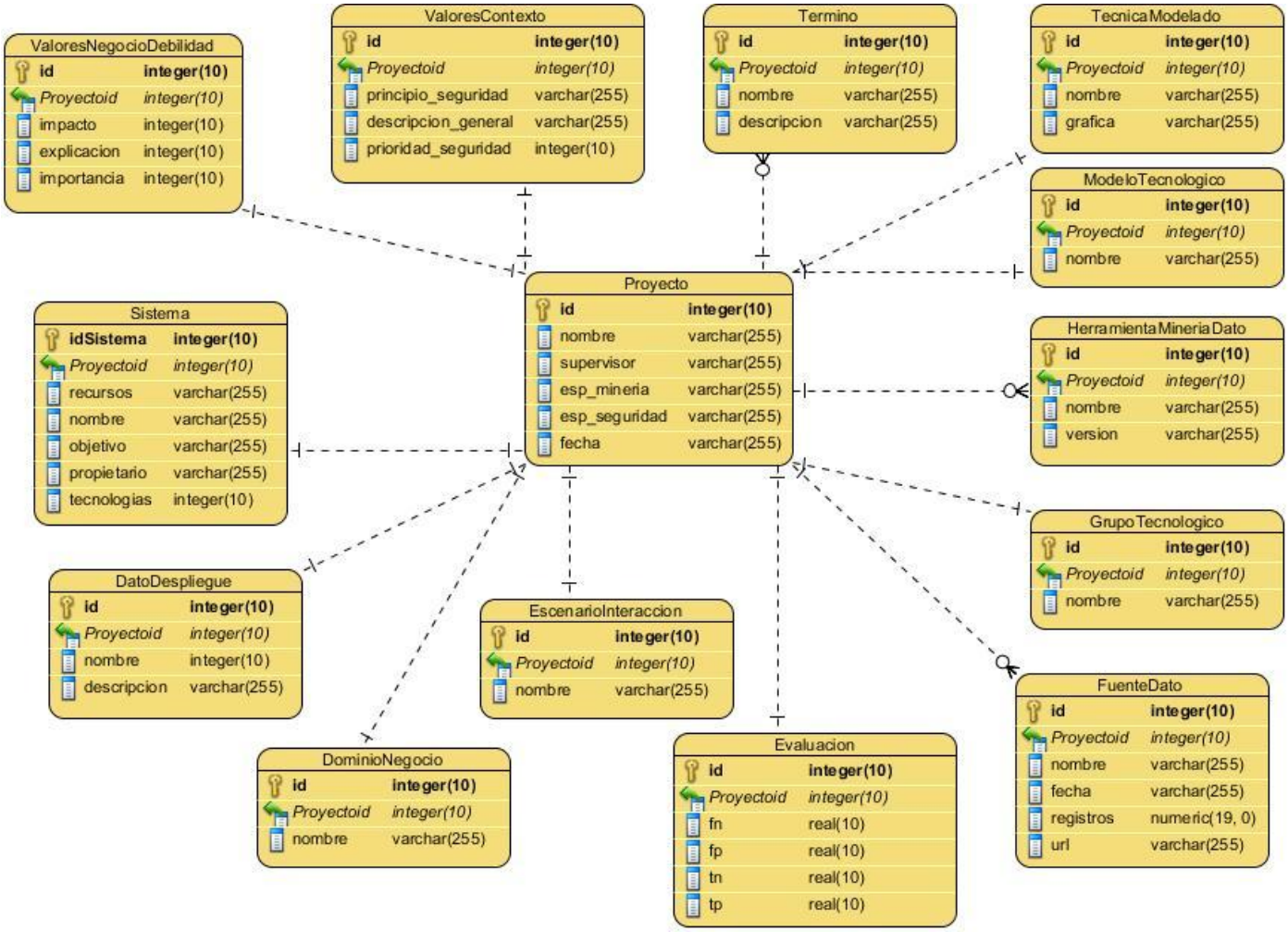


Figura 8: Modelo entidad- relación de la Base de Datos.

4.7 Conclusiones

En este capítulo se ha presentado un estudio relacionado con el diseño de la aplicación que sirven de base fundamental en la implementación. Los diagramas y especificaciones de diseño que se proponen constituyen una guía que puede ser fácilmente comprendida por los desarrolladores con el objetivo de implementar la aplicación que se ha diseñado.

CAPÍTULO 5: IMPLEMENTACIÓN Y PRUEBA DEL SISTEMA

5.1 Introducción

En este capítulo se explicará todo lo relacionado con la fase de implementación y prueba del sistema, mostrando el Diagrama de Componentes. Además se presentan los tipos de pruebas realizadas y los resultados de cada una de ellas para validar la solución.

5.2 Diagrama de Componentes

Un Diagrama de Componentes modela los aspectos físicos del sistema (ejecutables, tablas, librerías y documentos) y sus relaciones; mostrando las organizaciones y dependencias lógicas entre los componentes del software los cuales representan los tipos de elementos software que entran en la fabricación de las aplicaciones.

A continuación el Diagrama de Componentes del sistema:

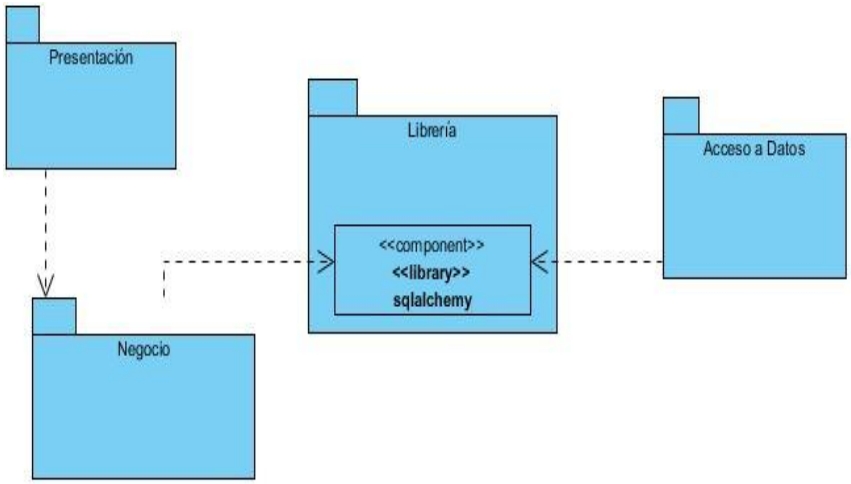


Figura 9: Diagrama de Paquetes y Librerías.

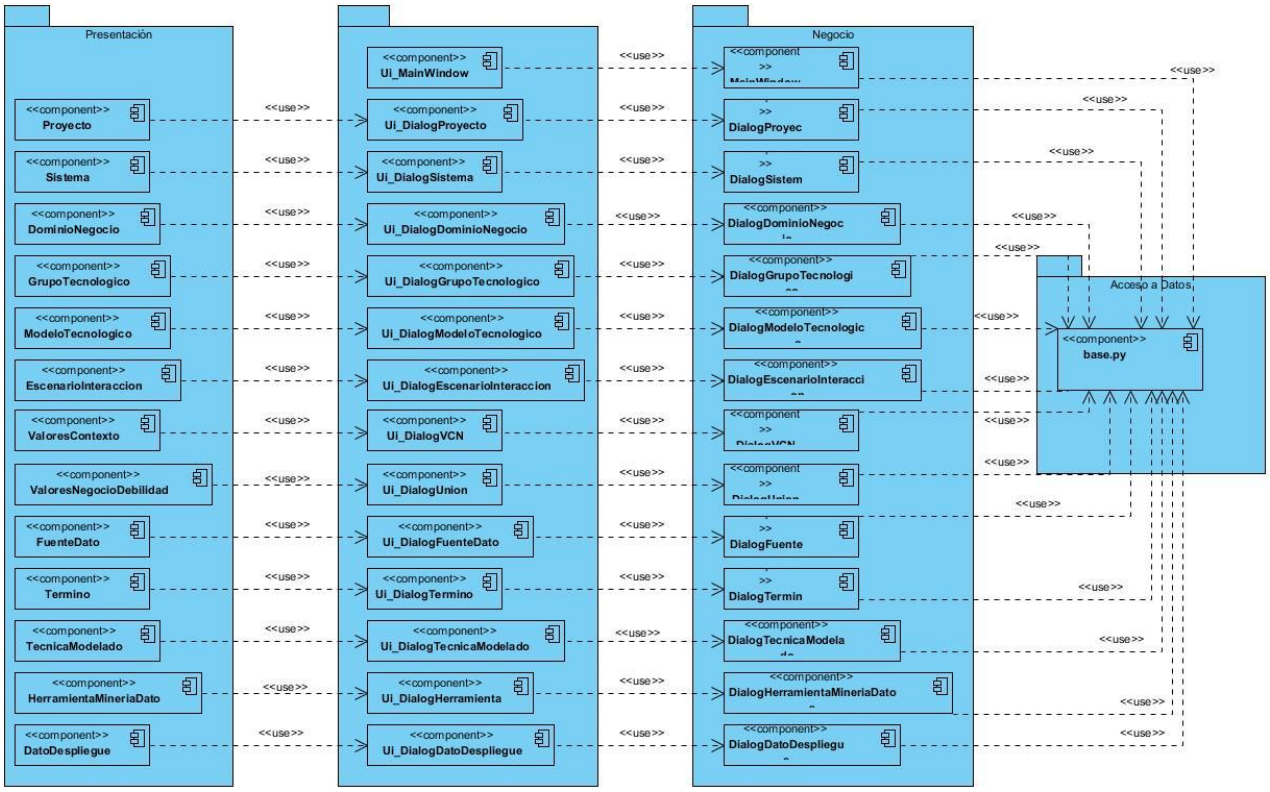


Figura 10: Diagrama de Componente del código.

Descripción de los componentes del sistema

- **Componentes del paquete Presentación:** clases que se encargan de manipular la información procedente de las interfaces de usuario.
- **Componentes del paquete Negocio:** clases que implementan la lógica y el comportamiento de los métodos.
- **base:** clase que se encarga de manipular los datos procedentes de los ficheros.
- **Sqlalchemy:** traduce todo a lenguaje Python para interactuar con la DB.

5.3 Pruebas al software

Las pruebas se realizan con la intención de descubrir y documentar errores, verificando la interacción de componentes y que todos los requisitos se han implementado correctamente para poder dar una indicación de calidad. También identifica y asegura que los defectos identificados se han rectificado antes

de entregar la aplicación al cliente. La ejecución de las pruebas permite verificar que la aplicación cumple con los requisitos funcionales y no funcionales especificados en la etapa de diseño de la solución.

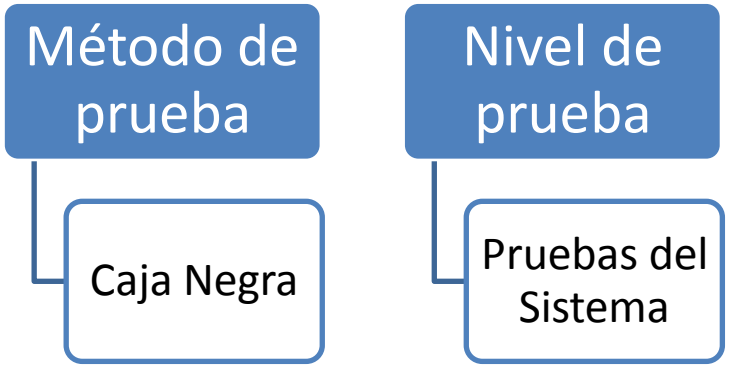


Figura 11: Método y nivel de prueba utilizado.

Método de Caja Negra

El método de caja negra incluye las pruebas que se llevan a cabo sobre la interfaz del software. O sea, los casos de prueba pretenden demostrar que las funciones del software son operativas, que la entrada se acepta de forma adecuada y que se produce un resultado correcto, así como que la integridad de la información externa se mantiene. Verifican las especificaciones funcionales y no consideran la estructura interna del programa.

Pruebas del Sistema

Para verificar que se hayan integrado adecuadamente todos los elementos del sistema y que las operaciones apropiadas funcionen como un todo se realizó las pruebas del sistema. Es en esta prueba donde se buscan los defectos globales dados por la mala integración y que impiden una buena aceptación en la decisión del cliente.

A continuación se presentan los resultados de las Pruebas correspondientes al Caso de Uso “Gestionar Proyecto”.

Las no conformidades registradas se clasifican en altas (errores en la interpretación de los procesos de la entidad y de funcionalidad), medias (errores de terminología y de diseño de interface) y bajas (errores de redacción y ortografía).

En el siguiente escenario de pruebas se realizaron dos iteraciones; donde en la primera iteración se obtuvieron un total de 9 No conformidades mientras que en la segunda.

Tabla 6: Características del escenario de pruebas.

	Microprocesador	RAM	Disco Duro	Sistema Operativo
PC del Cliente	Celeron 2.6 GHz	1 GB	150 GB	Ubuntu

Tabla 7: No Conformidades.

No Conformidad	Clasificación
Cuando se elimina un proyecto no se elimina toda la información de las clases relacionadas con él.	Media
Los atributos mostrados en el sistema presentan faltas de ortografía.	Baja
Los mensajes de confirmación presentan faltas de ortografía.	Baja
Se pueden crear dos proyectos con el mismo nombre.	Alta
Cuando se crea un proyecto y después vas a crear otro no limpia los campos.	Media
Después que eliminas los proyectos siguen cargados en la Vista.	Alta
Cuando se adiciona un proyecto no se muestra toda su información.	Alta
Cuando se adiciona un proyecto no se sabe cuándo se termina el proceso.	Alta
La fuente de datos no cuenta con un nombre que informe el origen de los datos	Media

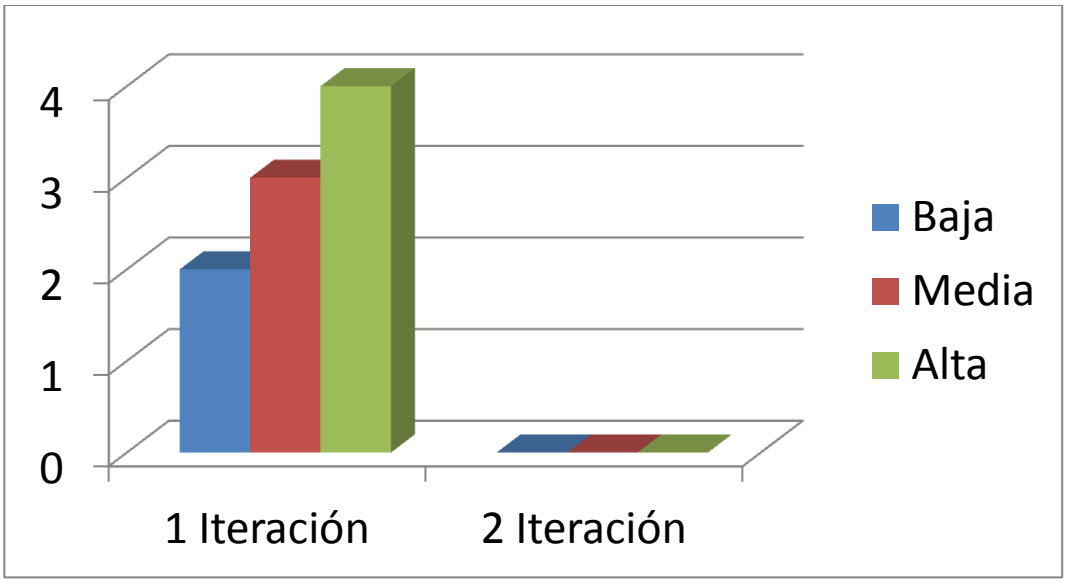


Figura 12: No Conformidades.

5.4 Conclusiones

En este capítulo se confeccionaron los Diagramas de Componentes mostrando la relación entre los principales componentes del sistema. Además se desarrollaron las Pruebas del Sistema dando solución a cada una de las no conformidades encontradas, validando así las funcionalidades implementadas. Finaliza de esta forma la implementación y validación de la Herramienta, con lo que se da cumplimiento al objetivo general trazado inicialmente.

CONCLUSIONES

Con la realización del trabajo de diploma Herramienta de Apoyo a la Metodología de Mitigación de Inyecciones SQL, se cumplió con los principales objetivos planteados realizando una herramienta capaz de contribuir al desarrollo de la Metodología de Mitigación de Inyecciones SQL.

Se estructuraron cinco capítulos en los cuales se evidencia todo el proceso de desarrollo de la herramienta, realizando un estudio detallado del problema existente, estado del arte y se sentaron las bases para la implementación de la herramienta, además se hace una selección de la metodología, herramientas y lenguaje para su desarrollo. También se realizó la Prueba del Sistema, que permitió validar la solución propuesta.

BIBLIOGRAFÍA

1. *Ley Orgánica de Protección de Datos de Carácter Personal.* España : s.n., 1999. 1.

2. HTMLPOINT.com. *HTMLPOINT.com.* [En línea] 1997-2006. http://www.htmlpoint.com/sql/sql_04.htm.

3. Introdución al SQL. *Introdución al SQL.* [En línea] Jesús Vegas, Abril de 1998. [Citado el: 9 de Enero de 2013.] http://www.infor.uva.es/~jvegas/cursos/bd/sqlplus/sqlplus.html.

4. [En línea] http://msdn.microsoft.com/es-es/library/ms161953%28v=sql.105%29.aspx.

5. [En línea] http://www.seguridad.unam.mx/noticias/?noti=4951.

6. [En línea] http://www.xombra.com/index.php?do=noticias&nota=6492&t=.

7. GREENSQL. [En línea] 2011. greensql.org.

8. **Islamabad.** *Web Application Security Solution.* 2010.

9. —. *Web Application Security Solution.* 2010.

10. **THOMPSON, ROGFEL.** *METODOLOGÍA PARA ELEVAR LOS NIVELES DE SEGURIDAD INFORMÁTICA EN APLICACIONES WEB MEDIANTE EL ANÁLISIS DE INYECCIONES SQL CON EL EMPLEO DE TÉCNICAS DE MINERÍA DE DATOS.* 2012.

11. **Duque, Raúl Gonzáles.** *Python para todos.* España : s.n.

12. **Marzal, Andrés y García, Isabel.** *Introducción a la programación en Python.*

13. [En línea] http://curso-sobre.berlios.de/introsobre/2.0.1/sobre.html/eclipse.html.

14. [En línea] http://pydev.org/.

15. Introducción a Qt. [En línea] http://www.zonaqt.com/book/export/html/80.

16. SQLite. [En línea] [Citado el: 26 de Febrero de 2013.] http://www.sqlite.org/copyright.html.

17. SQLAlchemy. [En línea] http://www.sqlalchemy.org/.

18. [En línea] http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_%28M%C3%8D%29_14720_p/.

19. [En línea] http://www.docirs.cl/uml.htm.

20. **MITRE.** CWRAF Common Weakness Risk Analysis Framework (CWRAF). [En línea] 2011. [Citado el: 26 de Febrero de 2013.] http://cwe.mitre.org/cwraf/.

21. **Aragón, Adys.** *Diseño e Implementación del módulo Evaluación.* La Habana : s.n., 2012.

22. **Pérez, Isaías Carrillo.** *Metodología de Desarrollo de Software.*

23. **Vazquez, Yosbel Morales y Bedoya, Bárbara Daisy.** *Software para la intercomunicación de redes aisladas. Módulo correo electrónico v2.0.* La Habana : s.n., 2012.

24. **Pressman, Roger S.** *Ingeniería del Software: un enfoque práctico.* s.l. : McGraw-Hill/Interamericana de Espana, S.A., 1988. 8476152221, 9788476152225.

25. **Pressman, Roger S.** *Ingeniería del Software: un enfoque práctico.* s.l. : McGraw-Hill/Interamericana de Espana, S.A., 1988. 8476152221, 9788476152225.

26. **Duque, Raúl González.** *Python para todos.*

27. Mundo geek. *Mundo geek.* [En línea] http://mundogeek.net/archivos/2006/06/05/easyeclipse/.

28. [En línea] http://www.rational.com.ar/herramientas/rup.html.

RECOMENDACIONES

A continuación se listan las recomendaciones en vistas de posibles mejoras a la herramienta:

- Implementar nuevas funcionalidades que permitan importar un proyecto.
- Implementar nuevas funcionalidades que permitan construir un modelo capaz de filtrar inyecciones SQL.