Diseño e Implementación de un Generador y Ejecutor de Planes de Mitigación del Riesgo contra Ataques XSS, basado en OWASP 2013 e ISO/IEC 27002

Trabajo de grado

Cristhiam Gabriel Fernández Ruales
Cristian Daniel Yanza Velasco

Director: MSc. Ember Ubeimar Martínez Flor
Co-Director: MSc. Siler Amador Donado

Universidad del Cauca
Facultad de Ingeniería Electrónica y Telecomunicaciones
Departamento de Sistemas
Línea Investigación Seguridad Informática
Popayán, Septiembre de 2017

Agradecimientos

A Dios por ser mi guía y mi fortaleza en cada paso de mi vida, iluminándome el camino con su protección y bendición, por permitirme llegar hasta donde he llegado, porque hiciste realidad este sueño anhelado y por brindarme una vida llena de aprendizajes, experiencias y sobre todo felicidad. A mi familia y a mis padres Liliana Velasco y Fredy Yanza por infundir en mi la lucha y el deseo de superación, por los valores que me han inculcado, y por haberme dado la oportunidad de tener una excelente educación en el transcurso de mi vida, sobre todo por ser un excelente ejemplo de vida a seguir. A mis directores de tesis Ember Martínez y Siler Amador por sus esfuerzos, dedicación, quienes con sus conocimientos, sus experiencias, su paciencia y su motivación han logrado en mí que pueda terminar mis estudios con éxito. Por haber brindado la oportunidad de desarrollar mi tesis profesional con ustedes y sobre todo por su amistad. A la Universidad del Cauca, por formarme como ingeniero. A los docentes del Departamento de Sistemas por prestarnos su atención, conocimiento y experiencia.

Cristian Daniel

Agradecimiento y dedicatoria especial a mis padres Bertha, Jorge y a mi hermano Damián por todo el apoyo dado en cada paso dado, por su incansable dedicación y entrega para darme oportunidades y herramientas para salir adelante. A mis directores de tesis Ember Martínez y Siler Amador por su guía, esfuerzo y comprensión. A todo el cuerpo docente de la Universidad del Cauca que con sus conocimientos y experiencia enriquecieron mi formación profesional. A todas las personas que me acogieron en esta ciudad y me brindaron amistad y aprecio. A mis compañeros y amigos. A los ingenieros Carlos Ardila y Sandra Buitrón con quienes compartimos lugar de trabajo y sin esperar nada a cambio compartieron su experiencia y consejo.

Cristhiam Gabriel

CONTENIDO

Lista de tablas

Lista de figuras

CAPÍTULO 1

INTRODUCCIÓN

1.1 PLANTEAMIENTO DEL PROBLEMA

El SGSI es la abreviatura utilizada para referirse a un Sistema de Gestión de la Seguridad de la Información. Para Hernández [1], un SGSI representa para una organización el diseño, implantación y mantenimiento de un conjunto de procesos para gestionar eficientemente la accesibilidad de la información, buscando asegurar la confidencialidad, integridad y disponibilidad de los activos de información minimizando a la vez los riesgos de seguridad de la información.

Para Lopez y Ruiz [2] el nivel de seguridad alcanzado por medios técnicos es limitado e insuficiente por sí mismo, por tal motivo, en la gestión efectiva de la seguridad debe tomar parte activa toda la organización, con la gerencia al frente, tomando en consideración también a clientes y proveedores de bienes y servicios. El modelo de gestión de la seguridad debe contemplar unos procedimientos adecuados y la planificación e implantación de controles de seguridad basados en una evaluación de riesgos y en una medición de la eficacia de los mismos.

En el 2007 López y Ruiz [2], sugieren que para la implementación del SGSI con base en la norma ISO/IEC 27001, se utiliza el ciclo continuo PDCA (Plan, Do, Check, Act) en el cual se establece, implementa, utiliza, monitorea, revisa, mantiene y mejora el SGSI, esto conlleva a diferentes pasos como definir el alcance en términos del negocio, la organización, su localización, activos y tecnologías, incluyendo detalle y justificación de cualquier exclusión. Según Enjuto [3], se debe definir qué ámbito debe cubrir, ya que se debe diseñar un SGSI para un alcance reducido, optimizando recursos y resultados. Para López y Ruiz [2], definir una política de seguridad que considere requerimientos legales o contractuales relativos a la seguridad de la información, que además establezca los criterios con los que se va evaluar el riesgo y que esté aprobada por la dirección, así como definir el enfoque de evaluación de riesgos mediante una metodología apropiada para el SGSI y los requerimientos del negocio, lo primordial de la metodología es que los resultados obtenidos sean comparables y repetibles para evitar grados de subjetividad que distorsionen la valoración de los riesgos. En esta sección se menciona el riesgo, que es la posibilidad de sufrir daños o pérdidas, estás pérdidas suelen ser los activos de las empresas o corporaciones, es decir pérdidas de la información almacenada. TechTarget en [4] señala que la amenaza es un componente del riesgo, y se puede considerar como: Un agente de amenaza (Humano o no humano), toma alguna decisión como identificar y explotar una vulnerabilidad que da lugar a algún resultado inesperado o no deseado, es decir pérdida, modificación o divulgación de información, o pérdida de acceso a la información. De acuerdo a lo anterior, La Universidad Nacional de Luján en [5] aporta que la gestión de riesgos se presenta como una actividad clave para el resguardo de los activos de información de una organización y en consecuencia protege la capacidad de cumplir sus principales objetivos. Es un proceso constante que permite a la administración

balancear costos operacionales y económicos causados por la interrupción de las actividades y pérdida de activos, con los costos de las medidas de protección a aplicar sobre los sistemas de información y los datos que dan soporte al funcionamiento de la organización reduciendo los riesgos que presentan los activos de información a niveles aceptables para la misma. El proceso de gestión de riesgos involucra cuatro actividades cíclicas: La identificación de activos y los riesgos a los que están expuestos, el análisis de los riesgos identificados para cada activo, la selección e implantación de controles que reduzcan los riesgos, y el seguimiento, medición y mejora de las medidas implementadas.

Existen numerosas metodologías estandarizadas para la evaluación de riesgos, la organización puede optar por una de ellas, o crear la suya propia. Para López y Spohr en [6], la norma ISO/IEC 27005 es un documento de apoyo que profundiza en directrices sobre la materia como identificar todos aquellos activos de información que tienen algún valor para la organización que están dentro del alcance del SGSI, identificar las amenazas relevantes asociadas a los activos identificados, identificar vulnerabilidades que puedan ser aprovechadas por dichas amenazas, identificar el impacto que podría suponer una pérdida de confidencialidad, integridad y disponibilidad para cada activo.

El estándar ISO/IEC 27005 indica el qué se debe hacer pero no el cómo se debe hacer; como se aborda en los resultados de la evaluación de riesgo realizados por Jurado et al. [7], en el cual se basaron en OWASP (Open Web Application Security Project) 2013 como metodología de medición y evaluación de riesgos. La evaluación del riesgo con OWASP 2013 se realizara con base en el top 10 de los tipos de ataques, entre el más influente es el tipo de ataque XSS (Cross-Site Scripting) el cual es un tipo de inyección de código, en el que las secuencias de comandos maliciosos se inyectan en los sitios web. Según Williams en [8], los ataques XSS ocurren cuando un atacante utiliza una aplicación web para enviar código malicioso, generalmente en forma de script del lado del navegador, a un usuario final diferente, este XSS es un vector de ataque que puede ser utilizado para robar información delicada, secuestrar sesiones de usuario, y comprometer el navegador, subyugando la integridad del sistema como lo comprueban Grossman et al. en [9].

Se puede definir el plan de tratamiento de riesgos, también llamado plan de mitigación del riesgo, ante amenazas como XSS, que según Suarez y Amaya en [10] es una de las actividades que propone la norma ISO/IEC 27001 en la implementación y operación del SGSI, éste plan es un documento en el cual se estipulan acciones apropiadas que se llevarán a cabo ante la ocurrencia de un riesgo, los recursos disponibles en la organización, las responsabilidades de aquellos que asumirán el riesgo y las prioridades con las que se tratará dicho riesgo. El éxito del plan de tratamiento de riesgos radica en aplicar las acciones correctas, aceptar los riesgos con conocimiento y objetividad siempre que satisfagan los criterios de aceptación, evitar riesgos y transferir a otras partes los riesgos asociados. Para llevar a cabo todo este proceso del SGSI el siguiente paso es implantar este plan de tratamiento de riesgo con el fin de alcanzar los objetivos.

Las organizaciones han encontrado en las aplicaciones web una forma efectiva y rápida para gestionar la información, para brindar sus servicios, por eso hoy en día son el blanco de ataques informáticos debido a la facilidad de acceso, y aunque existen guías y estándares que ayudan a gestionar la seguridad de la información, no se evidencian mecanismos automatizados que les ayuden a las organizaciones a identificar y a controlar en tiempo real toda esa serie de ataques maliciosos que sufren las aplicaciones web de

las organizaciones, esto evidentemente es una vulnerabilidad que puede afectar diferentes aspectos como confidencialidad e información corporativa, confidencialidad de los clientes, aspectos financieros, por mencionar algunos. Entre los ataques maliciosos se tiene el ataque XSS (Cross-Site Scripting); en Hostalia [11] se describe a XSS como una vulnerabilidad que muchos desarrolladores web dejan pasar, bien por falta de planificación o por desconocimiento. Esta vulnerabilidad suele aparecer por la falta de mecanismos en el filtrado de los campos de entrada que dispone la web, permitiendo el envío de datos e incluso la ejecución de scripts completos. Este tipo de ataques se realiza sobre aplicaciones que utilicen HTML (normalmente sitios web), el código malicioso utilizado en este tipo de ataques está compuesto por cadena de datos: scripts completos contenidos en enlaces o ejecutados desde formularios vulnerables. Los diversos tipos de ataques que nos podemos encontrar de Cross Site Scripting pueden ser catalogados en dos grandes grupos: XSS persistente o directo, y XSS reflejado o indirecto. XSS Persistente consiste en embeber código HTML peligroso en sitios que lo permitan por medio de etiquetas <script> o <iframe>. Es la más grave de todas ya que el código queda implantado en la web de manera interna y es ejecutado al abrir la aplicación. Dentro de este tipo nos encontramos el subtipo “Local” que aparece por un mal uso de DOM (Modelo de Objetos del Documento) con JavaScript, que permite la apertura de nuevas páginas con código malicioso JavaScript incrustado, afectando el código de la primera página en el sistema local. Estos códigos son ejecutados del lado del cliente, por lo que los filtros utilizados en el servidor no funcionan para este tipo de vulnerabilidades. XSS Reflejado es el tipo de ataque XSS más habitual y consiste en editar los valores que se pasan mediante URL, cambiando el tipo de dato pasado del usuario web, haciendo que ese código insertado se ejecute en dicho sitio. En el 2016, IMPERVA [12] se muestra algunos reportes de estadísticos sobre ataques en los últimos años, dentro de los cuales encontramos el tipo de ataque XSS, y su estadística de la razón de exposición de tráfico malicioso de aplicaciones web, en la que XSS tiene un porcentaje del 84% de aplicaciones expuesta a él, y un 16% de aplicaciones no expuestas.

Actualmente es una obligación de la organización garantizar seguridad de información , frente a cualquier tipo de eventualidad, por tal razón se plantea la necesidad de desarrollar aplicaciones regidas en estándares, que analicen y evalúen riesgos en tiempo real, y que sean capaces de tomar decisiones para controlar el riesgo o disminuirlo hasta donde sea permitido. Para ello se plantea desarrollar un módulo capaz de generar planes de tratamiento de riesgo, que estará encargado de analizar y tomar las decisiones sobre cuáles acciones se deben realizar para mitigar el riesgo, para luego enviar dicho plan al módulo ejecutor, que se encargará de llevar a cabo el plan generado haciendo uso de los dispositivos de seguridad disponibles en la organización.

Es así como se plantea la siguiente pregunta de investigación: **¿Cómo generar y ejecutar un plan de mitigación del riesgo contra ataques XSS basado en la ISO/IEC 27002 y OWASP 2013?**

1.2 **APORTES**

De acuerdo a los trabajos revisados y las brechas identificadas durante el proceso de revisión bibliográfica, la presente propuesta de trabajo de grado busca plantear una solución que permita la generación de planes y a su vez ejecutarlos para mitigar y tratar los riesgos a los que las aplicaciones web están expuestas. El principal aporte de este proyecto es la creación de un prototipo software el cual brindará una gestión sobre la seguridad en las aplicaciones web, generando reportes y acciones correctivas en tiempo real. El trabajo propone una estrategia que se construyó para XSS, pero que puede ser escalada a otros ataques como Inyección, Pérdida de autenticación y gestión de sesiones, Ausencia de control de acceso a las funciones, Falsificación de peticiones en sitios cruzados, entre otros.

El proyecto aporta una base de conocimiento formada a partir del criterio de expertos, que permita controlar ataques informáticos inicialmente de tipo XSS, pero con la posibilidad de ampliar el alcance a cualquier tipo de ataque informático dirigido a las aplicaciones web.

El aporte de este trabajo representa un valor significativo para las organizaciones que poseen aplicaciones web vulnerables, para quienes el costo de corregir dichas vulnerabilidades es demasiado alto, así también para aquellas organizaciones que deseen prevenir ataques futuros para sus activos. También porque permite un monitoreo, un seguimiento, una revisión, una toma de decisiones y una mejora constante del sistema.

1.3 **OBJETIVOS**

1.3.1 **Objetivo general**

Diseñar e implementar un generador y ejecutor de planes de mitigación del riesgo contra ataques XSS, basado en OWASP 2013 e ISO/IEC 27002.

1.3.2 **Objetivos específicos**

- Seleccionar las técnicas para la generación automática del plan de mitigación del riesgo.

- Proponer una representación y especificación del conocimiento requerido para la generación del plan a partir de la medición del riesgo según OWASP 2013 y la norma ISO/IEC 27002.

- Implementar e integrar el módulo generador de planes y el ejecutor al Sistema de Control de Seguridad (SCS 27002/OWASP).

- Verificar y determinar la efectividad del generador y ejecutor de planes a partir de la Guía de Pruebas de OWASP v4.0 para XSS.

1.4 **METODOLOGÍA DE TRABAJO**

En este capítulo se presenta la metodología para el desarrollo de los objetivos propuestos para la ejecución del proyecto el cuál se fundamenta en 3 iteraciones siguiendo las etapas definidas en el *Patrón de Investigación Iterativa* propuesto por Pratt Y Bright [13]. El patrón además de iterativo es incremental y contempla las siguientes etapas o fases:

- *Observación:* Se realizan observaciones de campo sobre la aplicación, en el marco del problema.
- *Identificación:* Con las observaciones efectuadas se identifica el problema.
- *Desarrollo:* Desarrollo de una solución al problema identificado.
- *Pruebas:* Se ejecutan pruebas a la solución desarrollada.

Para este trabajo se incluye una fase inicial que corresponde a la documentación y divulgación de los resultados, el cual conlleva la elaboración de la monografía y artículo de investigación con la información recolectada durante todo el proyecto, y al final se realizará la presentación de los resultados. A continuación se describe como se adaptó el proyecto a cada uno de sus objetivos.

Inicialmente se realizó un estudio sobre las técnicas de generación automática de planes para la mitigación del riesgo, las cuales se evaluaron de acuerdo a diferentes criterios propuestos para seleccionar la que más se ajuste a los requerimientos del proyecto. En el capítulo 3 se tiene los antecedentes de las técnicas de generación de planes automáticos que se lograron identificar. Al final del capítulo 4 se tiene cumplido el primer objetivo específico del trabajo.

En el capítulo 5 se presenta el proceso completo de análisis, identificación y selección de la técnica para la representación y especificación del conocimiento que se requirió para la generación de los planes a partir de la medición de riesgo según OWASP 2013 y la norma ISO/IEC 27002. La técnica seleccionada fue una ontología implementada bajo la metodología de REFSENO y posteriormente traducida a un dominio de planificación. Terminado el capítulo 5 se habrá cumplido con el segundo objetivo específico del proyecto.

Teniendo identificadas y seleccionadas las técnicas de generación de planes automáticos y de representación del conocimiento, se procedió a diseñar el módulo generador de planes y el módulo ejecutor. En el capítulo 6 se muestra todo el proceso que se llevó acabo, primero se realizó el estudio del sistema ya existente para analizar mejoras pertinentes para el proyecto, de acuerdo a las características con que ya se ha desarrollado, al mismo tiempo se realizó el estudio y la selección de los dispositivos de seguridad que se usarán para mitigar el riesgo de ataques XSS de acuerdo a las características y necesidades del sistema, también se tiene en cuenta los dispositivos disponibles con que se cuenta. Luego se modela la arquitectura del sistema y se diseñan los diagramas de clases correspondientes a cada módulo. Una vez analizados los diagramas se procede a desarrollar e implementar ambos módulos, en Java haciendo uso de la plataforma Netbeans. Luego se realizan las respectivas pruebas funcionales de cada módulo. En el capítulo 6 y en los anexos se puede observar fragmentos del código, las

pruebas funcionales ejecutadas y los resultados que se obtuvieron. Al finalizar el capítulo se habrá cumplido con el tercer objetivo específico del proyecto.

En la última fase del proyecto se hizo un estudio de la guía de pruebas para XSS de OWASP V4.0, apoyado de expertos para un mejor entendimiento. En el capítulo 7 se tiene el proceso completo de estas pruebas, en el que se identificaron los casos a realizar, se seleccionan y se genera el plan de pruebas. Luego se ejecuta el plan de pruebas de eficacia para ambos módulos desarrollados. Dentro del capítulo se tienen los resultados obtenidos de estas pruebas. La finalización del capítulo 7 cumple con el cuarto y último objetivo específico del proyecto.

Para finalizar se hace una documentación y divulgación de los resultados, es decir va enfocada en la revisión y análisis de los resultados obtenidos en el proyecto, además de la elaboración del artículo, que se presenta en los anexos.

CAPÍTULO 2

TRABAJOS RELACIONADOS

La investigación sobre la representación y especificación del conocimiento requerido para la generación del plan a partir de la medición del riesgo y sobre la generación de planes automáticos, son la base central del proyecto, como consecuencia, se han publicado numerosas propuestas, con las cuales se hace una comparación y diferenciación al proyecto que se propone en este trabajo.

2.1 REPRESENTACIÓN Y ESPECIFICACIÓN DEL CONOCIMIENTO POR MEDIO DE UNA ONTOLOGÍA

Todo problema es más sencillo resolver si disponemos de conocimiento específico sobre él, por eso surge la necesidad de contar con mecanismos que nos permitan obtener, preservar y transmitir el conocimiento para tratar de obtener una clara, precisa y completa comprensión de la realidad que nos rodea, por lo que cada vez se necesitan mecanismos más compactos que nos permitan conservar el conocimiento que se está manejando. Siguiendo en el contexto tenemos las bases de conocimiento en las que se almacena el conocimiento a través de hechos o relaciones entre diferentes entes, y reglas de inferencia con las cuales se tiene un nivel más avanzado de representación que indican características generales sobre los hechos y que si se cumplen un conjunto de circunstancias se puede inferir algo nuevo. Las ontologías se convirtieron en un área de interés común para la línea de inteligencia artificial, ingeniería del conocimiento, procesamiento del lenguaje natural y representación del conocimiento, entre otros. De manera más reciente, la noción de ontología se ha extendido a áreas tales como: integración inteligente de información desde orígenes heterogéneos, recuperación de información y la gestión del conocimiento, incluso han llegado a abarcar el campo de los servicios Web.

Actualmente la mayoría de trabajos que abordan el problema de representación del conocimiento optan por las ontologías debido a que permiten una formalización que puede dar lugar a la realización de inferencias, y que pueden utilizarse para la representación de modelos reutilizables en dominios distintos que permiten una mejor representación, organización, razonamiento, reutilización y comparación, según lo planteado por Blanco et al. [46]. Uno de los objetivos de una ontología es describir acuerdos ontológicos que sirvan como base para la comunicación entre todos los agentes, lo cual permite disminuir los efectos de la ambigüedad del lenguaje que comúnmente lleva a errores, a falta de entendimiento y a la realización de esfuerzos improductivos.

Por otro lado llevando el enfoque de la ontología a la ingeniería del software, varios autores establecen los beneficios de utilizar las ontologías: Proporcionan una forma de representar y compartir el conocimiento utilizando un vocabulario común, permiten usar

un formato de intercambio de conocimiento, proporcionan un protocolo específico de comunicación y permiten una reutilización del conocimiento.

Las ontologías se componen de:

- **Conceptos:** Son las ideas básicas que se intentan formalizar, pueden ser clases de objetos, métodos, planes, estrategias, procesos de razonamiento.
- **Relaciones:** Representan la interacción y enlace entre los conceptos de un dominio.
- **Funciones:** Son un tipo concreto de relación donde se identifica un elemento mediante el cálculo de una función que considera varios elementos de la ontología.
- **Instancias:** Se utilizan para representar objetos determinados de un concepto.
- **Reglas de restricción o axiomas:** Son teoremas que se declaran sobre relaciones que deben cumplir los elementos de la ontología.

Entre los trabajos más recientes donde sugieren el uso de ontología se encuentran: Prediction and Classification of Web Application Attacks using Vulnerability Ontology [23], Automated Planning and Acting [47], Ontology for attack detection: An intelligent approach to web application security [22], The information systems' security level assessment model based on an ontology and evidential reasoning approach [21] y A Framework to Support the Harmonization between Multiple Models and Standards [48].

Para este trabajo es importante definir e identificar los conceptos y relaciones para la planificación, debido que se necesita un acto mediante el cual el sistema organiza de manera anticipada una situación, evento o acción que ya se sepa que tomará lugar con el objetivo de reducir o mitigar el riesgo que se está analizando. La planificación comprende un análisis de la situación, el establecimiento de objetivos, la formulación de estrategias que permitan alcanzar dichos objetivos, y el desarrollo de planes de acción que señalen como implementar dichas estrategias. Esta planificación permite la organización, coordinación y control de los riesgos y ataques (Para este trabajo ataques Cross-Site Scripting XSS) que llegan al sistema a través de la web.

La planificación es importante debido a que reduce la incertidumbre y minimiza el riesgo, genera eficiencia, y dice cómo y cuándo aplicar el plan con sus respectivas acciones y dispositivos de seguridad teniendo en cuenta los aspectos del tratamiento de riesgos:

- Reducir, seleccionar los controles.
- Transferir.
- Aceptar, se acepta el riesgo según los umbrales que se hayan definido.
- Evitar, terminar la actividad que lo origina.

Los trabajos relacionados que se encontraron en la revisión bibliográfica realizada sobre las representaciones de conocimiento a través de ontologías son:

En 2011, Velásquez et al. [14] presentan los aspectos relevantes de la Ingeniería Ontológica como técnica efectiva para la representación del conocimiento. Iniciando con la conceptualización y el papel que juegan las ontologías dentro de la arquitectura de la web semántica. Presentan metodologías de desarrollo de ontologías más utilizadas, como las diversas herramientas requeridas para acceder al conocimiento almacenado en dichas

ontologías. Este conjunto de propuestas son interesantes, ya que permite dar una visión de las diferentes opciones que existen y así determinar cuál se adapta mejor al proyecto.

En el 2012, Pardo et al. [15] proponen una ontología para apoyar la armonización de modelos múltiples. Este trabajo se ha centrado principalmente en el desarrollo de ontologías para representar elementos claves de dominios particulares, como por ejemplo el modelo CMMI-SW, dominio de ingeniería basada en SWEBOK, entre otros. Como no se encontraron con una ontología que permite la integración de estos conceptos y elementos, deciden llevar a cabo la construcción de su propia ontología, realizando un análisis comparativo de conceptos y terminología de armonización. Dado que esta propuesta de ontología se limita a los dominios presentados anteriormente no es aplicable a nuestro contexto, pero podría servir de referencia para elaborar una ontología propia.

De la misma forma Borst [16], describe una investigación sobre el uso práctico de ontologías para el desarrollo de sistemas de información, en donde se propone una ontología para ser utilizada como una especificación de un sistema de información, que especifica el conocimiento requerido para las tareas que el sistema de información tiene que realizar. Esta investigación propone varias ontologías para diferentes dominios, y explica cómo se pueden especificar y utilizar para hacer ontologías reutilizables, es decir construir ontologías grandes y complejas a partir de las más pequeñas.

Gawich et al. [17] presentan la definición de ontología y una especificación explícita de la conceptualización que implica la exploración de conceptos y relaciones en el dominio de interés. Este artículo presenta metodologías de construcción de ontologías como Uschold y King, entre otras. Este artículo no propone una comparación con metodologías más relevantes actualmente, sin embargo se utiliza esta información para tener bases de comparación con las otras metodologías que se han decidido seleccionar.

El objetivo de Fonou y Huisman [18], es proporcionar una dirección para la aplicación de metodologías de construcción de ontologías existentes en los procesos de desarrollo de la Web Semántica de modelos de ontología específicos del dominio del gobierno electrónico (e-government). El trabajo se enfoca más en sistemas electrónicos del gobierno, pero sin embargo es de gran interés, ya que el marco y las técnicas empleadas para desarrollar los modelos de ontología semántica podrían repetirse en otros dominios del conocimiento para construir ontologías.

En el 2010, Álvares y Martínez [19], presentan una metodología para la creación de ontologías taxonómicas haciendo uso de una serie de métricas y la posibilidad de realizar consultas, de varios tipos sobre dichas ontologías, y sobre otras ontologías remotas accesibles vía http. Además el sistema incorpora una API para la visualización de cualquiera de las ontologías descritas anteriormente en Protégé que es un producto para editar ontologías. Es oportuna esta información para tener una idea clara de la implementación de las ontologías por medio de una herramienta, pero este trabajo solo muestra una parte de la ontología, no va más allá en cuanto a su creación de reglas de inferencia y la adaptación a código java, donde serán ejecutadas dichas reglas.

De la misma forma, Corcho et al. [20] en 2010 revisan y comparan las principales metodologías, herramientas y lenguajes para la construcción de ontologías que han sido reportadas en la literatura, así como las principales relaciones entre ellas. La tecnología

ontológica es hoy en día lo suficientemente madura: muchas metodologías, herramientas e idiomas ya están disponibles. El trabajo a futuro en este campo debe orientarse hacia la creación de un banco de trabajo integrado común para los desarrolladores de ontologías para facilitar el desarrollo, el intercambio, la evaluación, la evolución y la gestión de ontologías, para proporcionar apoyo metodológico para estas tareas y traducciones, y desde diferentes lenguajes ontológicos. Este banco de trabajo no debe crearse desde cero, sino integrar los componentes tecnológicos actualmente disponibles.

En el 2015, Solic et al. [21] presentan un modelo que puede ser la base para una nueva solución de evaluación de seguridad de sistemas de información. Esta solución es capaz de cubrir una amplia gama de todos los posibles problemas de seguridad de la información. El modelo propuesto se basa en una ontología de OWL para la base de conocimiento, utiliza un algoritmo de Razonamiento Probatorio Mejorado para cálculos matemáticos y posee un simple algoritmo de reflejo del agente inteligente como elemento de apoyo a la decisión. Las propiedades de este modelo sobrevienen de las propiedades de sus elementos constructivos. La base de conocimiento que se está construyendo sobre ontología OWL es un elemento importante del modelo. Puede proporcionar alta flexibilidad y aplicabilidad a diferentes sistemas de información y organizaciones empresariales; Actualmente para estar al día con respecto a los problemas de seguridad y nuevas amenazas; teniendo en cuenta todos los aspectos posibles relacionados con cuestiones de seguridad, por ejemplo, problemas de seguridad de red, software y hardware, influencia humana, políticas de seguridad y planes de recuperación ante desastres, el algoritmo de Razonamiento Probatorio Mejorado se basa en la teoría Dumpster-Shafer y es adecuado para cálculos con juicios subjetivos de expertos que combinan calificaciones cualitativas con evaluaciones cuantitativas. En este trabajo, explican cómo conectar y utilizar cada uno de los elementos constructivos del modelo para obtener resultados de evaluación de la seguridad de la información, además, realizan un estudio de caso con el modelo propuesto en una organización de pequeñas empresas. Para probar el modelo, también usan el método de evaluación de riesgo cualitativo estándar en la misma organización empresarial para comparar resultados cualitativos. El modelo presentado podría alcanzar su objetivo si se desarrollara en una herramienta de software integrada con una base de conocimientos ontológicos bien definida y actualizada.

El objetivo en el trabajo de Razzaq et al. [22] es demostrar cómo una metodología de ingeniería ontológica puede aplicarse sistemáticamente para diseñar y evaluar estos sistemas de seguridad. Se muestra un modelo ontológico detallado que satisface el trabajo generalizado de las aplicaciones web, los protocolos de comunicación subyacentes y los ataques. Más específicamente, el modelo ontológico propuesto capta el contexto, no sólo puede detectar ataques de especificación de protocolo HTTP, sino que también ayuda a concentrarse sólo en porciones específicas de la solicitud y respuesta donde es posible un script malicioso. El modelo también captura el contexto de ataques importantes, las diversas tecnologías utilizadas por los hackers, la fuente, el objetivo y las vulnerabilidades explotadas por el ataque, el impacto en los componentes del sistema y los controles para la mitigación. Para evaluar la calidad del modelo propuesto se incluye un conjunto completo de métricas para la evaluación de la ontología, que incluye corrección, exactitud, consistencia, solidez, orientación de la tarea, exhaustividad, amplitud, reutilización, claridad, integridad, eficiencia y expresividad. El modelo propuesto se clasificó bien frente a las métricas mencionadas anteriormente, además un prototipo de sistema de detección de ataques basado en el modelo mostró un mejor desempeño en la

detección y baja tasa de falsos positivos. Este modelo es muy cercano a la solución que propone en este proyecto, pero no contempla la metodología de OWASP 2013, los dispositivos de seguridad, y los estados para el tratamiento del riesgo.

Por otro lado, Salini y Shenbagam [23], se proponen un enfoque para las defensas efectivas contra los ataques a nivel de aplicación. El sistema propuesto es un sistema basado en la ontología que puede predecir y clasificar los ataques de aplicaciones web, almacena de manera efectiva información de amenazas, vulnerabilidades y ataques. Los ataques se pueden predecir mediante el análisis de la vulnerabilidad y las amenazas, se clasifican en función del nivel de severidad de los ataques a los objetivos de seguridad, además, el sistema también proporciona sugerencias para la prevención y contramedida a los ataques previstos, ayudando así a los desarrolladores en el desarrollo de aplicaciones web seguras. Los resultados fueron prometedores en comparación con el método convencional de base de conocimiento. Esta información proporcionada es vital para el desarrollo del proyecto, debido a que se asemeja, a lo que se desea construir, pero no contempla la metodología OWASP para la medición de riesgos.

En el ámbito local, Jurado et al. [7] en el 2015 desarrollaron un sistema capaz de detectar ataques informáticos de tipo XSS haciendo uso de técnicas de inteligencia artificial, para luego medir el nivel de riesgo al que está expuesta una organización con dicho ataque, para posteriormente tomar acciones predefinidas y contrarrestar el impacto negativo en los activos de información. Las acciones del sistema se basaban en el uso de un cortafuegos y un servidor de notificaciones a los administradores del sistema, por tal motivo proponen como trabajo futuro la utilización de técnicas de representación del conocimiento para la correcta toma de decisiones por parte del control.

2.2 TÉCNICAS PARA LA GENERACIÓN AUTOMÁTICA DE PLANES DE MITIGACIÓN DEL RIESGO

El conocimiento corresponde a un insumo fundamental para la resolución de problemas y la planificación automática; ésta última se utiliza para situaciones en las que se requiere un comportamiento estratégico y razonado complejo. En el 2004 Aktolga et al. [31] afirman que para solucionar un problema se deben tener en cuenta las técnicas de búsqueda estándar. La planificación suele considerarse como un término genérico de resolución de problemas, ya que se ocupa de la búsqueda en un nivel abstracto. La resolución de problemas suele estar más relacionada con la ejecución del plan, mientras que la planeación está involucrada con la generación del plan. Esto se debe al hecho de que los sistemas de resolución de problemas suelen estar diseñados para resolver una tarea específica y, por lo tanto, están restringidos. Sin embargo, los sistemas modernos de planificación son capaces de lidiar con problemas mucho más difíciles y problemas inesperados que surgen durante el proceso de planificación. Dado que el alcance de los problemas tratados en la planificación es mucho más amplio, un planificador se ve como el productor o generador de la solución, y un sistema de resolución de problemas sólo "demuestra" una solución específica. La planificación es el proceso de calcular varios pasos de un procedimiento de resolución de problemas antes de ejecutar cualquiera de ellos. Un problema de planificación en Inteligencia Artificial, según Torres [32] se puede definir como un problema de búsqueda que requiere encontrar una secuencia eficiente de

acciones para conducir un sistema desde un estado inicial a un estado objetivo. Por lo tanto el objetivo del campo de la planificación en Inteligencia Artificial es construir algoritmos capaces de encontrar una secuencia de acciones que resuelvan un problema planteado.

Según Alechina [33], existen dos maneras en cuanto a la generación de planes: la primera, planificación automática basada en un dominio y la segunda, por medio de búsquedas. La principal diferencia entre la búsqueda y planificación es la representación de los estados, en búsqueda los estados son representados como una sola entidad (que puede ser un objeto bastante complejo), pero en la planificación, los estados tienen representaciones estructuradas (conjunto de propiedades) que son utilizadas por el algoritmo de planificación.

En la Tabla 1, tomada de Alechina [33], se observa una comparación entre planificación y búsqueda.

	BÚSQUEDA	**PLANEACIÓN**
Estados	Estructura de datos	Sentencias lógicas
Acciones	Código	Precondiciones / Resultados
Objetivo	Código	Sentencia lógica (Conjunción)
Plan	Secuencia de So	Restricciones sobre las acciones

Tabla 1. Comparación entre planificación y búsqueda.

Planificar es importante tanto para los individuos, comunidades y organizaciones. En una organización la planificación es utilizada de tres formas. Primero, como medio para proteger la organización de posibles amenazas que surgen de un entorno peligroso y cambiante. Segundo, como forma de mejorar la situación actual global de la organización. Tercero, como forma de organizar y coordinar a todos los miembros de la organización en persecución de unos objetivos comunes. Para García et al. [34], éste triple enfoque de la planificación y la dificultad del problema convierten a la planificación automática en un área de conocimiento dentro del ámbito de la Inteligencia Artificial de gran interés para todas las personas, y organizaciones involucradas en la investigación.

En general, la planificación es el proceso de controlar un grupo de tareas para completar lo que se describe en el dominio del problema para lograr ciertos objetivos. Dependiendo de cuan complejo es el dominio, aumenta el número de estados requeridos para completar las tareas, en Inteligencia Artificial cuando se enfrenta a un problema, actuar racionalmente es importante para hacer un plan eficiente para superar ese problema. La planificación comienza a partir de cierto punto, que se llama el estado inicial, y termina en el estado objetivo, que es el destino. La planificación según Fakhtehyavari y Vaziri [35], es el proceso de elegir ciertas acciones considerando el efecto de la acción, empleando estrategias, priorizando tareas y cambiándolas para hacer un plan razonable capaz de alcanzar la meta. Se han desarrollado planificadores automáticos en una gran variedad de contextos como en la robótica, operaciones de evacuación y rescate, entre otros. Este éxito de la planificación automática para resolver problemas reales ha impulsado a los expertos en este tema a desarrollar planificadores más amigables, y también la continua aparición de nuevas técnicas y teorías.

Según Alechina et al. [33], los sistemas de planificación hacen lo siguiente:

- Explorar las acciones y la representación de objetivos para permitir una selección.
- Dividir y conquistar por sub-objetivos.
- Requisitos para la construcción secuencial de soluciones.

2.2.1 **Problemas de planificación**

Los problemas de planificación pueden ser simples o complejos, pero todos los problemas tienen al menos una solución. En el caso de soluciones múltiples, no todas son la respuesta óptima al problema, pero al menos una es la respuesta razonable. De hecho, la solución en general es el conjunto de acciones que se eligen para ser realizadas en los diferentes estados para conducirnos al estado objetivo. Para resolver problemas de planificación existen varias alternativas disponibles que varían en complejidad. Por ejemplo, una forma posible para el planificador es tomar un camino al azar y luego averiguar si este es el que puede cumplir el resultado deseado. Si la respuesta es no, entonces intentará otra que parezca más prometedora con el conocimiento de sus intentos anteriores. Como resultado, la siguiente opción podría ser al menos una solución razonable o, como máximo, la respuesta óptima al problema. La otra alternativa posible propuesta por Fakhtehyavari y Vaziri [35] es la de una búsqueda de fuerza bruta donde se examinan todas las posibilidades que conducen a la selección de la solución óptima. Este enfoque parece ser fácil, pero en realidad lleva mucho tiempo pasar por todas las soluciones posibles. Una tercera posibilidad es definir un conjunto de reglas que sirvan como guía para encontrar la solución del problema planteado. La última pero no menos importante posibilidad es el uso de una función heurística que selecciona entre varias posibles soluciones la respuesta óptima al problema.

Un problema de planificación se formula en base a tres elementos (Figura 1), tomada de García et al. [34]:

- Una representación de las variables y los objetos del contexto, así como de un conjunto de acciones que podemos aplicar sobre ellos y cuyos efectos alteran el entorno. Estos elementos definen el dominio de planificación.
- Una representación del estado en el que se encontrarán los objetos del dominio en el entorno antes de que empecemos la ejecución de nuestro plan llamado estado inicial.
- Una representación de la meta u objetivo que deseamos cumplir tras la ejecución de nuestro plan.

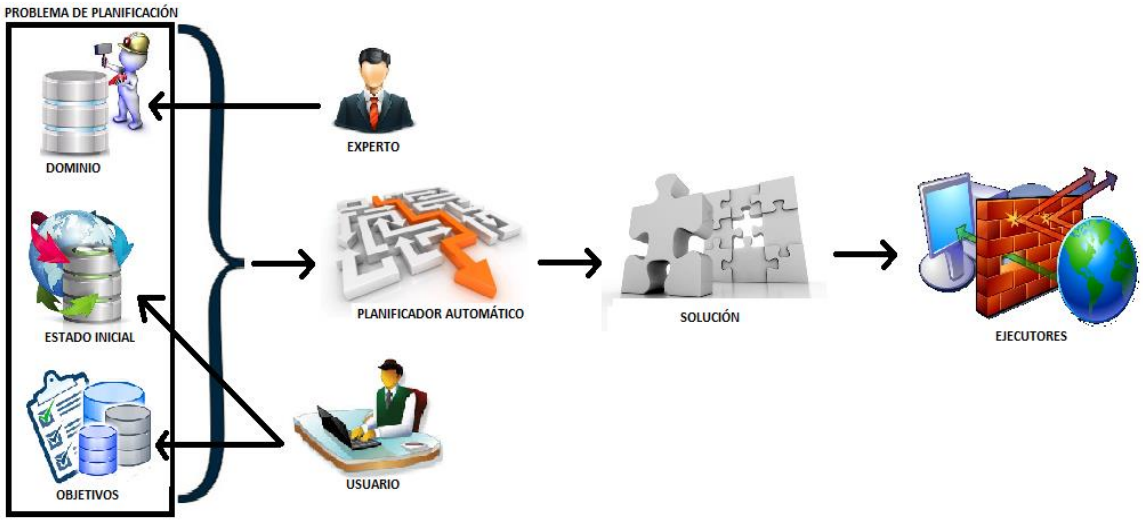

Figura 1. Partes de un problema de planificación.

Los trabajos relacionados que se lograron encontrar a través de la revisión bibliográfica sobre los planificadores automáticos son:

En el 2003, Hoffmann [24] propone un lenguaje de planificación basado en STRIPS (Stanford Research Institute Problem Solver) para una planificación exitosa, porque STRIPS es una de las técnicas algorítmicas más populares, la cual se basa en guiar la búsqueda por una función heurística basada en las tareas de planificación. Este artículo presenta el planeador Metric-FF como uno de los más populares y eficientes según la IPC (International Planning Competition), sin embargo usa un lenguaje de planificación con limitaciones, porque restringe el tipo de estados meta que pueden especificarse a una conjunción de literales positivos, no soporta acciones paralelas (cuando múltiples acciones se aplican simultáneamente al mismo estado) y su expresividad puede no ser completa para problemas específicos.

Por otro lado, Adriá [25] muestra una vista general de la planificación automática, presenta soluciones muy específicas con modelos e interfaces gráficas de usuario (GUI) para aplicar esta planificación y se enfocan en el lenguaje de planificación PDDL (Planning Domain Definition Lannguage). Es importante esta solución para tomarla como guía en este proyecto, debido que da pasos muy específicos para desarrollar un buen planificador.

En el trabajo de Guzman [26] se presenta los diferentes enfoques en que se aplican los sistemas de planificación en la generación automática de programas de control para sistemas de manufacturas y se ilustra en la planificación STRIPS. La solución propuesta tiene sus limitaciones con el lenguaje, debido a que es poco genérico.

En 2009, Raimondi et al. [27] presenta una metodología y una herramienta para el problema de probar y verificar que un dominio de planificación definido en PDDL satisface un conjunto de requisitos, muestran como los casos de prueba pueden traducirse en

objetivos de planificación adicionales. Para automatizar este proceso introducen PDVer, un plug-in Eclipse para la generación automática de código PDDL a partir de requisitos expresados en LTL (Linear Temporal Logic). Esta herramienta no cuenta con la suficiente expresividad que ofrece PDDL en sus últimas versiones.

Posteriormente, en el 2014 Wotawa y Bozic [28] realizan una evaluación de los planificadores basados en STRIPS y PDDL para realizar pruebas de seguridad automatizadas, en este caso, el dominio de planificación se construyó formalizando patrones de ataques conocidos, sin embargo, la adaptación de los modelos de ataque no es fácil y requiere un esfuerzo significativo. Con el fin de facilitar el modelado, sugieren representar los ataques como una secuencia de acciones conocidas que tienen que llevarse a cabo para tener éxito, cada acción tiene algunas condiciones previas y algunos efectos, por lo tanto son capaces de representar las pruebas en este contexto como un problema de planificación donde el objetivo es romper la aplicación bajo prueba. En el artículo se discute el enfoque de las pruebas de planificación propuestas presentando algunos resultados experimentales poco satisfactorios, esto se debe a que el número de planes generados debería ser reducido cuando se usan diferentes técnicas de planificación. Esta planificación no se hace de manera automática debido a sus múltiples planes generados, y además no cuenta con los conceptos de la medición del riesgo según OWASP y los controles de la norma ISO/IEC 27002.

En el 2015 Bozic y Wotawa [29] proponen que las pruebas de seguridad siguen desempeñando un papel importante en la ingeniería de software, porque debido a su latencia, las fallas de seguridad en las aplicaciones web siempre corren el riesgo de ser explotadas. Para evitar posibles daños, las medidas de prevención apropiadas deben incorporarse en el tiempo y en el mejor de los casos durante el ciclo de desarrollo del software. En este trabajo, contribuyen a este objetivo y presentan la herramienta PURITY, del inglés Planning-based secURITY testing tool, para probar aplicaciones web, la cual ejecuta casos de prueba contra un sitio web determinado, detecta si el sitio web es vulnerable frente a algunas de las vulnerabilidades más comunes, es decir, las inyecciones de SQL y las secuencias de comandos entre sitios (XSS). El objetivo es parecerse a una actividad maliciosa siguiendo secuencias típicas de acciones potencialmente conducentes a un estado vulnerable, la ejecución de la prueba procede automáticamente. A diferencia de otras herramientas de prueba de penetración, PURITY se basa en la planificación, los casos de prueba se obtienen de un plan, que a su vez se genera a partir de valores iniciales específicos y acciones dadas, estas últimas están destinadas a imitar las acciones habitualmente realizadas por un atacante, además PURITY también permite al usuario configurar parámetros de entrada y pruebas a un sitio web de manera manual. Este artículo presenta partes que se pueden aplicar para el proyecto, sin embargo, todavía requiere mejorar incluyendo características adicionales, por ejemplo, agregando más acciones en la especificación de dominio y aumentando la configurabilidad permitiendo una mayor intervención manual.

Plch [30] en el 2011 realiza un estudio del lenguaje de definición de dominio de planificación (PDDL) para las definiciones de dominios y problemas de planificación, ya que los investigadores y diseñadores suelen cometer errores semánticos y de sintaxis debido a la complejidad del lenguaje, al mismo tiempo, es difícil leer y trabajar con documentos más grandes en PDDL, por ello se ha desarrollado una herramienta llamada PDDL Studio, cuyo objetivo es ayudar en la creación e inspección de documentos PDDL,

y las características principales del editor son: 1) analizador de PDDL capaz de localizar sintaxis y errores semánticos, 2) resaltado de sintaxis de PDDL, 3) finalización de código sensible al contexto y sugerencias - similar a IntelliSense de Microsoft para lenguajes declarativos, 4) colapso de código, 5) Gestión de documentos PDDL, 6) integración del planificador. El editor PDDL también cuenta con una herramienta PDDL Parser, que se puede utilizar como analizador independiente para otros proyectos. Las pautas son esenciales, sin embargo no está dentro del contexto de este proyecto, se debe adaptar. Por otra parte, es limitada en cuanto a la expresividad que ofrece la versión más reciente de PDDL, y esto podría causar problemas futuros, ya que estaría muy limitado para algunas acciones.

CAPÍTULO 3

SELECCIÓN DEL LENGUAJE DE PLANIFICACION

3.1 CRITERIOS Y SELECCIÓN DEL LENGUAJE DE PLANIFICACIÓN

Un sistema de planificación inteligente entrega una serie de pasos, representados por operadores, que se deben ejecutar para ir desde un estado inicial hasta una meta (estado meta o lista de tareas cumplidas). Para ello el problema del mundo real debe ser llevado al ambiente planificador y determinar el dominio del mismo. Se presentan dos enfoques: *Planificadores Dependientes del Dominio* y *Planificadores Independientes del Dominio.*

Según Duque et al. [36], en muchos planificadores los operadores son atómicos y pueden actuar directamente, mientras que en otros, como el caso de la planificación jerárquica, se permiten operadores abstractos que pueden ser descompuestos en un grupo de pasos que forman un plan que implanta el operador.

3.1.1 Planificadores dependientes del dominio

Usan heurísticas del dominio específico para controlar las operaciones del planificador, por tal motivo, desarrollar un planificador automático específico para cada uno de los posibles dominios de aplicación de esta tecnología sería una labor ardua y costosa. Además, cuando sea necesario adaptar el planificador para resolver un tipo de problema distinto o bien cuando las condiciones del dominio para el cual el planificador fue diseñado cambiase habría que rehacer el planificador completamente, es por ello que estos tipos de planificadores no se usan en la actualidad, debido a la gran evolución e investigación en el área de los planificadores automáticos en la Inteligencia Artificial. Algunas de las áreas que se abarcaron para crear planificadores dependientes del dominio son: Robótica, Sistemas de manejo de emergencias (inundaciones), Operaciones de evacuación y rescate, Incendios forestales, Gestión de procesos empresariales, Sistemas de manufacturación, o incluso en misiones espaciales. Aunque este enfoque es pionero en cuanto planificadores, al paso del tiempo se fue deteriorando debido a su costo en cada cambio en el ámbito del problema, y porque estaba muy ligado a una sola área. Debido a esto surgió la necesidad de crear un nuevo enfoque más general, es cuando surgieron los planificadores independientes del dominio.

3.1.2 Planificadores independientes del dominio

En este enfoque la representación del conocimiento y los algoritmos se espera que trabajen bien en una amplia gama de situaciones, según Fakhtehyavari y Vaziri [35]. Debido a las restricciones encontradas en el ítem 3.2.1, desde los inicios de la planificación se han tratado de desarrollar algoritmos de planificación independientes del dominio de aplicación. Esta separación requiere que, por un lado dispongamos de unos modelos generales que nos permitan almacenar la información del contexto con el que estamos trabajando y por otro lado diseñamos los algoritmos que operen sobre estas

estructuras. El algoritmo de planificación deberá ser capaz de comprender el modelo de representación de la información y ser capaz de razonar sobre ella, pero sin estar especializado en el dominio en concreto. El problema de planificación consiste en alcanzar el objetivo, partiendo del estado inicial, utilizando acciones y objetos definidos en el dominio. Estas representaciones deben tener una sintaxis y una semántica bien definida además de ser lo suficientemente expresivas para poder resolver una gran cantidad de problemas en diversos dominios de aplicación, es por ello que para el proyecto nos centraremos en los planificadores independientes del dominio de aplicación, ya que lo que se pretende es que el sistema funcione correctamente y sea adaptable para aplicaciones de cualquier área o ámbito, es decir que el sistema se podrá integrar fácilmente, y no se deberá hacer una reestructuración, que genere más costos y esfuerzo.

Dentro de este enfoque se tiene algunos lenguajes de planificación, de los cuales se opta por 3 lenguajes para evaluar por medio de unos criterios para la selección del más adecuado para el proyecto. Los 3 lenguajes son: STRIPS (Stanford Research Institute Problem Solver), usado en Aktolga et al. [31] y Guzmán [26]; PDDL (The Planning Domain Definition Language), usado en García et al. [34] y Ghallab et al. [38], y ADL (Action Description Language), usado en Fakhtehyavari [35].

3.1.3 **Lenguajes de planificación**

La discusión propuesta por Russell y Norving [37] sugiere que la representación de los problemas de planificación, estados, acciones y objetivos, deberían permitir que los algoritmos de planificación aprovecharan la estructura lógica del problema. La clave es encontrar un lenguaje que sea lo suficientemente expresivo como para describir una amplia variedad de problemas, pero lo suficiente restrictivo para permitir que los algoritmos eficientes operen sobre él. En esta parte se describirán los tres lenguajes seleccionados (STRIPS, PDDL, ADL), y a través de unos criterios de selección se tomara la decisión de cuál es el lenguaje de planificación adecuado para el proyecto. Cabe aclarar que la selección de estos 3 lenguajes se hizo por medio de la revisión bibliográfica sobre los lenguajes más utilizados en la actualidad como se puede evidenciar en el trabajo de García et al. [34], Alechina [33], Fakhtehyavari y Viziri [35], Duque et al. [36], Russell y Norving [37], Aktolga et al. [31], Raimondi et al. [27], Ghallab et al. [38] y Guzmán [26].

3.1.3.1 **Lenguaje STRIPS**

En el 2004, Aktolga [31] presenta el lenguaje STRIPS (Stanford Research Institute Problem Solver), en español Solucionador de problemas del instituto de investigación de Stanford, por Fikes y Nilson en 1971, el cual se usa comúnmente en la planificación para representar estados, metas y acciones. La Closed World Assumption (CWA) sostiene que, cualquier estado o relación que no esté definido explícitamente deber ser falso. Los estados y metas están representados como conjunciones de literales positivos. Los operadores se utilizan como acciones instanciadas para transformar estados a otros. En otras palabras con STRIPS apareció el concepto de acción como elemento de representación de cambio. La Asunción de STRIPS presupone que el valor de verdad de un predicado no es afectado por la ejecución de una acción a no ser que aparezca explícitamente como efecto de la misma.

García et al. [34] describen que la representación de STRIPS de estados y acciones se basa en la lógica de primer orden. En lógica de primer orden, un predicado se describe como una tupla de la forma (cabeza a1, a2,…, an) donde cabeza es el nombre del predicado y a1, a2,…, an es un número indeterminado de argumentos que pueden ser variables sin instanciar, variables instanciadas o constantes. En la representación de acciones de STRIPS, cada acción se describe a partir de tres componentes:

- Lista de precondición: es una lista de predicados que deben ser ciertos en un estado para que pueda aplicarse una acción.
- Lista de adición: es una lista de predicados nuevos que se hacen ciertos tras la aplicación de la acción.
- Lista de supresión: es una lista de predicados antiguos que dejan de ser ciertos tras la aplicación de la acción.

Esta representación de acciones tuvo gran acogida por los investigadores en planificación y fue usada como la base de la mayoría de las técnicas de planificación automática desde principios de los años 70 como ABSTRIPS propuesta por Sacerdoti [39], NONLIN propuesta por Tate [40] o TWEAK propuesta por Sacerdoti [41] hasta los años 90 como PRODIGY propuesta por Veloso et al. [42].

El STRIPS es un planificador por regresión, es decir dada la descripción de un problema de planificación a partir de estados iniciales, estados objetivos y un conjunto de operaciones o acciones disponibles, el enfrenta su solución como un proceso de búsqueda en espacio de estados, comenzando por el estado meta hasta el estado inicial. Esta técnica que construye el plan al revés está basada en la regresión de objetivos a través de la descripción de una acción. El lenguaje STRIPS permite representar una cantidad de dominios y problemas bastante compleja, pero su expresividad en algunos casos puede no ser suficiente. Por ello a lo largo del tiempo han ido apareciendo lenguajes sucesores de STRIPS que aumentan su expresividad.

3.1.3.2 **Lenguaje ADL**

En el 2015 Bozic y Wotawa [29] presentan el lenguaje ADL (Action Definition Language), en español Lenguaje de Descripción de Acciones. ADL se basa en STRIPS y en cierto sentido es una versión actualizada de STRIPS. Fue acuñado en 1987 y es básicamente un lenguaje de planificación automatizado que se utiliza activamente en el área robótica. El lenguaje de descripción de la acción encuentra una condición específica que conduce a un conjunto de acciones que prometen llevarnos a la meta. Cada acción tiene condiciones previas que deben cumplirse para permitir que la acción comience a realizarse. El rendimiento de la acción causa algunos efectos al "ambiente" para modificarlo. El medio ambiente puede ser identificado por estados particulares, que están satisfechos o no.

ADL soluciona la falta de expresividad de STRIPS al ser una ampliación del lenguaje, el cual, según García et al. [34] incluía las siguientes mejoras en cuanto a expresividad:

- Posibilidad de incluir literales negados como objetivo del plan así como en las precondiciones de las acciones

- Objetos del dominio y variables en fuertemente tipadas, de esta forma los objetos de dominio se pueden unificar con las variables sin instanciar de un literal en una precondición
- Cláusulas disyuntivas, cuantificadas universalmente y existencialmente en las precondiciones de una acción
- Inclusión de un operador de igualdad expresado como un literal que permite la comparación de variables dentro de las precondiciones de una acción
- Efectos condicionados que sólo aplican cuando el estado del mundo cumple con una determinada condición. Los efectos condicionales permiten resumir varias acciones con una estructura de precondiciones y efectos muy similar en una sola.

Obviamente a medida que crece el grado de expresividad del lenguaje de representación también crece la complejidad de los algoritmos de planificación que lo procesan.

3.1.3.3 **Lenguaje PDDL**

García et al. [34] presentan el lenguaje PDDL (Planning Domain Description Language), en español el Lenguaje de Definición del Dominio de Planeación. Este lenguaje es considerado un estándar. PDDL nace en 1998 por McDermott como lenguaje para describir los problemas de la IPC de ese mismo año. La IPC (International Planning Competition), la competición internacional de planificadores es un evento bianual, que se realiza junto a la Conferencia Internacional de Planificación y Secuenciación CAPS (*International Conference of Planning and Schudeling*). Los objetivos de esta competición son proporcionar una base para el debate y la comparación de los sistemas de planificación, justo en la frontera de las capacidades de los planificadores actuales para proponer nuevas direcciones de investigación. Además, se proporciona un conjunto de problemas con una representación común que permiten la comparación y evaluación de los distintos sistemas de planificación.

Hay diferentes versiones de PDDL disponibles como lo describe Fakhtehyavari y Vazari [35], se encuentra PDDL 1.1 a la más reciente versión llamada PDDL 3.1. PDDL se usa ampliamente para codificar los dominios problemáticos y expresar los problemas de planificación temporal. Consiste en *objetos*, *predicados* que son los atributos de los objetos, *acciones* que son capaces de aplicar cambios, *especificación de objetivos* que debe ser verdadera para lograr el plan y finalmente estado inicial que es el comienzo del estado. El dominio contiene estados y acciones, mientras que la definición contiene el estado inicial, los objetos y la especificación de objetivos. PDDL es un lenguaje muy expresivo, de hecho hay muchos planificadores que no pueden soportarlo plenamente, por ello incorpora un mecanismo de requerimientos sobre el planificador, como una serie de flags, que el planificador debe soportar para operar sobre el dominio. En la versión 1.0 de PDDL, propuesta por Ghallab et al. [38], su expresividad era muy similar a la de ADL con la incorporación de la posibilidad de definir dominios jerárquicos. Afortunadamente PDDL es un lenguaje vivo que va evolucionando con cada edición de la IPC. En la edición del 2002 se presentó una nueva versión del lenguaje la 2.1, propuesta por Long y Fox [43] que incorporaba importantes novedades. PDDL 2.1 se organizaba en 4 niveles con expresividad creciente. En el nivel 1 tenía la misma expresividad que PDDL 1.0. En el nivel 2 se incluye una mayor expresividad para el tratamiento de expresiones numéricas o *fluents.* En el nivel 3 se introduce temporización en las acciones, con precondiciones y

efectos *at-start, at-end* u *overall.* En el nivel 4 permite la definición de acciones con efectos continuos en el tiempo. La última versión oficial de PDDL es la 3.1, propuesta por Gerevini y Long [44], presentada en la IPC del 2006, esta versión recoge la posibilidad de introducir preferencias de usuario y restricciones en el dominio de planificación.

3.1.4 **Criterios seleccionados para la evaluación de los lenguajes de planificación**

En esta sección, se toman de algunos criterios propuestos por Parker et al. [45], los cuales fueron previamente estudiados y adaptados al proyecto, así mismo se realizan reuniones con los expertos en el tema, para definir los criterios realmente importantes que fueran útiles para la selección del lenguaje de planificación que mejor se adaptó al proyecto, luego se realizó una comparación entre los lenguajes STRIPS y ADL, para ver las ventajas y desventajas entre ambos, debido a que el lenguaje PDDL posee una gran ventaja que es la de compartir características de los dos lenguajes nombrados anteriormente, por último se presenta una comparación entre los 3 lenguajes de planificación, donde evaluaremos cada criterio, se analiza las ventajas y desventajas de cada lenguaje para llegar a la selección correcta.

3.1.4.1 **Criterios y cuadro de comparación de los lenguajes de planificación**

Los criterios seleccionaron fueron establecieron gracias a una minuciosa revisión bibliográfica. Ahora, con los criterios definidos se pretende asignar una escala representativa para asignar el nivel de importancia como se muestra en la Tabla 2, adaptada de Sacerdoti [39].

ESCALA REPRESENTATIVA	
Menos importante	1
Moderadamente importante	3
Fuertemente importante	5
Muy fuertemente importante	7
Esencial o críticamente importante	9

Tabla 2. Escala representativa.

A continuación se presentan los criterios seleccionados y se le asigna el valor de la escala representativa de cada uno, éstos criterios fueron usaron para evaluar el lenguaje de planeación que mejor se adaptó al proyecto, así mismo se asigna el valor correspondiente a cada una de las ponderaciones que tiene cada criterio, el cual tiene un intervalo de 1 a 9, donde 9 es el mejor de los casos, 5 el caso promedio y 1 el peor caso.

- **Expresividad:** Una decisión importante es la expresividad de la representación del conocimiento. Cuanto más expresiva, es decir, algo es más fácil y más compacto. Dentro del lenguaje de planificación este término es fundamental para poder realizar la mayor parte de acciones, hechos, entre otros. Este criterio tiene las siguientes ponderaciones (Baja (1), Media (5), Alta (9)). Su valor en la escala representativa es 9.
- **Soporte con Java:** Si el lenguaje cuenta con soporte para implementarlo con el lenguaje Java, debido a que el proyecto está desarrollado en este lenguaje, es muy

importante que el lenguaje de planificación pueda escribirse en código Java para una fácil integración a lo desarrollado en los demás módulos. Este criterio tiene las siguientes ponderaciones (No soporta (1), Soporta (9)). Su valor en la escala representativa es 9.

- **Última modificación:** Se utiliza este criterio para ver su última renovación y así poder comparar que tan actualizadas están los lenguajes de planificación. Para este caso el valor de la ponderación es 9 si está recientemente actualizado el lenguaje y 1 si no es muy reciente su actualización. Su valor en la escala representativa es 1.
- **Soporte y documentación:** Se refiere a la documentación que tiene cada lenguaje de planificación, si existe buena información sobre cada uno de ellos, y si esta información es completa y entendible. Este criterio tiene las siguientes ponderaciones (Malo (1), Bueno (5), Excelente (9)). Su valor en la escala representativa es 7.
- **Lenguaje completo:** Si el lenguaje de planificación abarca todas las necesidades del proyecto, si ofrece un conjunto de características completas para cubrir todos los requisitos, es decir, posee todos los componentes, estructuras, expresiones necesarias. Este criterio tiene las siguientes ponderaciones (Incompleto (1), Completo (9)). Su valor en la escala representativa es 7.
- **Facilidad de ejecución:** Este criterio se refiere a qué tan fácil es de entender y seguir todas sus características o expresividades que proponen, es decir si son simples, comprensibles y sencillas para su realización y ejecución. Este criterio tiene las siguientes ponderaciones (Difícil (1), Medio (5), Fácil (9)). Su valor en la escala representativa es 5.
- **Proyectos reales:** Este criterio se refiere en que tan óptimo es el lenguaje, o que tan eficiente es para proyectos reales y más complejos que puedan ser necesarios abordar. Este criterio tiene las siguientes ponderaciones (No es suficiente (1), Suficiente (9)). Su valor en la escala representativa es 3.

Una vez que se tienen definidos los criterios, se procede a evaluar cada uno de los lenguajes. En la Tabla 3 se observa las comparaciones realizadas de los tres lenguajes según cada criterio.

LENGUAJE / CRITERIO	STRIPS	ADL	PDDL
SOPORTE CON JAVA	Soporta	No soporta	Soporta
EXPRESIVIDAD	Baja	Media	Alta
LENGUAJE COMPLETO	Incompleto	Completo	Completo
SOPORTE Y DOCUMENTACIÓN	Excelente	Bueno	Excelente
FACILIDAD DE EJECUCIÓN	Fácil	Fácil	Fácil
PROYECTOS REALES	No es suficiente	Suficiente	Suficiente
ÚLTIMA MODIFICACIÓN	1971	1987	2006

Tabla 3. Criterios para la selección de un lenguaje de planificación.

Se hace una valoración ponderada de los lenguajes de planificación automática, haciendo uso de los valores asignados en la descripción de cada criterio de selección.

En la Tabla 4 se muestra los valores cuantitativos para cada lenguaje de planificación y el valor de ponderado para cada criterio de selección.

CRITERIO (Ponderación)	STRIPS	ADL	PDDL
SOPORTE CON JAVA (9)	9	1	9
EXPRESIVIDAD (9)	1	5	9
LENGUAJE COMPLETO (7)	1	9	9
SOPORTE Y DOCUMENTACIÓN (7)	9	5	9
FACILIDAD DE EJECUCIÓN (5)	9	9	9
PROYECTOS REALES (3)	1	9	9
ÚLTIMA MODIFICACIÓN (1)	1	1	9

Tabla 4. Valoración ponderada de los lenguajes de planificación.

En la Tabla 5 se muestra el valor correspondiente del ponderado de cada criterio normalizado por el valor del factor asignado. Estos valores se hicieron de la siguiente manera: se toma el valor de ponderación de cada criterio de selección y se divide entre 44 que es la suma total de todas las ponderaciones, luego ese valor se multiplica por el valor de cada uno de los factores de cada lenguaje, y finalmente se suman los valores resultantes para cada lenguaje, obteniendo el valor total de su evaluación.

CRITERIO	STRIPS	ADL	PDDL
SOPORTE CON JAVA	1,841	0,205	1,841
EXPRESIVIDAD	0,205	1,023	1,841
LENGUAJE COMPLETO	0,159	1,432	1,432
SOPORTE Y DOCUMENTACIÓN	1,432	0,796	1,432
FACILIDAD DE EJECUCIÓN	1,023	1,023	1,023
PROYECTOS REALES	0,068	0,614	0,614
ÚLTIMA MODIFICACIÓN	0,023	0,023	0,205
TOTAL VALOR	**4,751**	**5,116**	**8,388**

Tabla 5. Valores correspondientes de los ponderados de los criterios por los ponderados de los factores.

Analizando la Tabla 5, permite concluir que el lenguaje de planeación que mejor se adapta para el proyecto es PDDL. Algunas razones extras que se tuvo en cuenta para esta selección es que PDDL es un lenguaje común para la competición mundial de planificadores IPC, también se selecciona PDDL debido a que es un estándar ampliamente usado en la comunidad de planificación en Inteligencia Artificial. Cabe decir que se trabaja con la versión 3.0 porque no hay planificadores que soporte la especificación 3.1 completa, si no subconjuntos de ella.

Adriá et al. [25] presentan otras razones por la que se ha seleccionado PDDL 3.0, el cual engloba las características de las versiones precedentes y aporta nuevas funcionalidades:

- **Acciones con duración:** Las acciones ya no son instantáneas sino que requieren determinado tiempo para su ejecución, esto implica que las precondiciones y efectos tengan asociado un punto de ejecución en el que o bien se debe cumplir la condición

o se produce el efecto, este punto puede ser el inicio o final de la acción para los efectos y los mismos o durante toda la acción para las precondiciones.

- **Expresiones y variables numéricas:** Además de contar con los predicados con valores verdadero/falso, PDDL añade la posibilidad de utilizar variables con valores numéricos pudiendo representar las precondiciones por ejemplo en forma de umbrales.
- **Métricas:** Permiten definir el uso de funciones de optimización como una combinación lineal de uno o varios factores del problema y así obtener una medida de cuan buena es la solución obtenida y especificar un grado de calidad deseado. Por ejemplo se puede pedir que se solucione el problema en menos de *n* acciones.
- **Ventanas temporales:** Se utilizan para representar eventos externos al problema cuyo valor y momento serán conocidos por el planificador, por ejemplo que un recurso está disponible en un horario determinado.

Con estas aportaciones PDDL permite que el modelado de problemas sea más cercano a la realidad y a la vez más fácil su representación, aportando funcionalidad no solo a la forma de conseguir el plan sino también elementos para establecer criterios de calidad de las soluciones.

3.1.5 **Algoritmo de planificación**

El planificador utiliza el modelo de planificación basada en estados ya que las características del dominio hacen que en este modelo resulte sencilla su representación y control. Para este proyecto se utilizó el algoritmo de planificación automática Metric-FF debido a que en la literatura es una de las más usadas y recomendadas y ocupó el primer puesto en la competición IPC (International Planning Competition), además que va muy ligada al lenguaje de planificación PDDL, el cual fue seleccionó para este proyecto. Hoffmann [24] describe que Metric-FF es una extensión del planificador FF (Fast Forward) el cual está implementado en C. Ha participado en los dominios numéricos del III Concurso Internacional de Planificación, demostrando un desempeño muy competitivo. Como se dijo, Metric-FF trata con PDDL 2.1 nivel 2, combinado con ADL. Aparte de las características ADL bien conocidas, esto permite un número finito de variables numéricas de estado (funciones en los números racionales, aplicadas a las tuplas de los objetos presentes en la tarea a manejar). En cualquier punto de una fórmula de condición (pre, efecto o meta) donde se permite un átomo lógico, también permitimos ahora una restricción numérica, es decir, para una comparación ("<", "<=", = ","> = ", Or"> ") entre dos expresiones sobre los números racionales y las variables numéricas, utilizando los operadores" + "," - "," * "y" /". En cualquier punto donde se permite un efecto lógico, ahora también se permite un efecto numérico que afecta al valor de una variable numérica (lado izquierdo) por una expresión numérica (lado derecho), según uno de los operadores ": = "," + = ", O" - = "(el idioma original también permite los operadores de asignación" * = "y" / = ").

Naturalmente, Metric-FF hereda las principales ideas utilizadas en FF (Fast Forward). La búsqueda sigue siendo una variación de hill-climbing (Escalada) en el espacio de todos los estados alcanzables, y la evaluación heurística todavía se hace mediante la resolución de una tarea en cada estado de búsqueda, utilizando un algoritmo de estilo Graphplan (es decir, para resolver las tareas). Los planes resultantes todavía informan a la búsqueda por

caminos de una estimación de distancia de meta, el número de pasos en el plan, así como por caminos de una estimación de cuales acciones son las más usadas, respuesta. Hoffmann [24] sugiere que las condiciones en el caso puramente lógico prefieren valores "superiores" de las variables proposicionales: las condiciones negativas se compilan como un pre-proceso y, por lo tanto, siempre es preferible tener más hechos proposicionales verdaderos. La observación explotada en Metric-FF es que la misma metodología puede aplicarse en el contexto numérico, al menos en un subconjunto del lenguaje. La tarea es pre-procesada de tal manera que todas las restricciones numéricas son monotónicas, es decir, para cualquier restricción “con”, si “con” es verdadera en un estado “S” entonces “con” es verdadera en cualquier estado “S” 'donde, para todas las variables x, x (S') > = X (S). Para lograr la propiedad de la monotonicidad, se necesita, en las restricciones y efectos numéricos, expresiones que son monótonas en todas las variables. En la implementación actual, nos limitamos a expresiones lineales que obviamente tienen esta propiedad. El sistema Metric-FF se implementa para hacer frente a lo que se llama tareas lineales.

En conclusión, se usa PDDL como lenguaje para la definición del dominio de planificación ya que se adapta a los requerimientos del proyecto, así mismo, se eligió el algoritmo de planificación Metric-FF para inferir las acciones de los planes porque es el más usado de acuerdo a la revisión bibliográfica realizada.

CAPÍTULO 4

REPRESENTACIÓN Y ESPECIFICACIÓN DEL CONOCIMIENTO PARA EL CONTROL DE SEGURIDAD

4.1 METODOLOGÍA PARA EL DESARROLLO DE LA ONTOLOGÍA

En el 2009, Ohgren [49] propone que una ontología es una especificación explícita formal de una conceptualización compartida de un dominio o ámbito específico. Para Borst [16] los conceptos y limitaciones de la ontología deben ser definidos explícitamente y la conceptualización es un modelo abstracto de algún fenómeno con los conceptos pertinentes identificados de ese fenómeno.

Las ontologías son formales, lo que significa que no son ambiguas para evitar tergiversaciones en los conceptos. La especificación usando un lenguaje formal permite el razonamiento; las ontologías utilizan especificaciones explícitas para hacer suposiciones del dominio para el razonamiento y para apoyar la comprensión del dominio. Por otro lado las ontologías tiene un aspecto importante el cual es limitarse a un área específica de la aplicación (dominio) para ser manejable, esto lleva a una conceptualizaciones comunes o compartidas que permiten el agrupamiento de conocimientos. Las ontologías resultan muy útiles para facilitar el razonamiento automático, es decir, sin intervención humana. Partiendo de unas reglas de inferencia, un motor de razonamiento puede usar los datos de las ontologías para inferir conclusiones de ellos, si establecemos estas reglas las máquinas pueden hacer deducciones.

Mizoguchi [50] afirma que el desarrollo de una ontología no es una tarea fácil, se requiere de habilidades y sigue siendo un arte más que una tecnología. Vizcaíno et al [51] proponen que una metodología sofisticada ayuda al desarrollo de una ontología, aunque estas metodologías no se han madurado lo suficiente, hay una gran variedad de ellas disponibles. En general, las metodologías dan una serie de pautas de cómo se debe llevar a cabo las actividades identificadas en el proceso de desarrollo de la ontología, qué tipo de técnicas son las más adecuadas en cada actividad y cuáles son los productos o artefactos de salida. Ahora con el fin de sistematizar la implementación de la ontología se decide usar una metodología. En la literatura se proponen diferentes metodologías entre las cuales se tuvo en cuenta una serie de criterios para una selección más acotada de las mejores metodologías existentes hasta la actualidad, en las que se realizó una tabla de comparación entre las metodologías seleccionadas para el desarrollo de la ontología. Las metodologías que se estudiaron para este proyecto son: La metodología de Uschold y King (ENTERPRISE), la metodología de Grüninger y Fox (TOVE), la metodología REFSENO (A Representation Formalism of Software Engineering Ontologies), METHONTOLOGY, y Ontology Development 101: A Guide to Creating Your First Ontology.

Se realizó un estudio minucioso entre estas metodologías con unos criterios que se establecieron como se muestra en la Tabla 6. Ahora, con los criterios definidos se pretende asignar una escala representativa para asignar el nivel de importancia como se mostró en la Tabla 2 del capítulo 4.

A continuación se presentan los criterios de selección y su escala representativa que fueron usados para evaluar la metodología de desarrollo de ontologías que mejor se adaptó al proyecto, así mismo se asigna el valor correspondiente a cada una de las ponderaciones que tiene cada criterio, el cual tiene un intervalo de 1 a 9, donde 9 es el mejor de los casos, 5 el caso promedio y 1 el peor caso.

- **Tamaño del proyecto:** Este criterio se refiere a qué tan grande es el proyecto que se va a realizar, y si la metodología es óptima para el tamaño del proyecto que se tiene. Para este caso se tiene un proyecto pequeño en cuanto a la ontología ya que no necesita muchos conceptos en ella. Este criterio tiene las siguientes ponderaciones (Excelente (9), Bueno (5), Malo (1)). Su valor en la escala representativa es 7.
- **Soporte y documentación:** Se refiere a la documentación que tiene cada metodología, si existe buena información sobre cada una de ellas, y si esa información es completa y entendible. Este criterio tiene las siguientes ponderaciones (Excelente (9), Bueno (5), Malo (1)). Su valor en la escala representativa es 9.
- **Relevancia:** Este criterio va enfocado en el estudio de las metodologías más buscadas, utilizadas, calificadas en artículos para el desarrollo de metodologías, las más relevantes, las que están en el top de metodologías por el momento. Este criterio tiene las siguientes ponderaciones (Relevante (9), No relevante (1)). Su valor en la escala representativa es 9.
- **Tiempo de ejecución:** Se refiere a que tan rápida es la metodología para llevarse a cabo, es decir que tan ágil es parar llevar el desarrollo de la ontología. Se analiza todas las fases de cada metodología y se compara con las demás para saber cuál es preferible para sacar rápidamente la ontología desarrollada, también analizamos el número de artefactos que propone la metodología y que tan relevantes son esos artefactos para el desarrollo de la ontología. Este criterio tiene las siguientes ponderaciones (Rápido (9), Mediano (5), Lento (1)). Su valor en la escala representativa es 5.
- **Completitud:** Si la metodología abarca todas las necesidades para desarrollar una ontología, es completa en cuanto a sus fases para poder desarrollar una ontología, es decir que tiene todos sus elementos, partes o aspectos se realizan totalmente, no deja fases inconclusas. Este criterio tiene las siguientes ponderaciones (Completo (9), Incompleto (1)). Su valor en la escala representativa es 9.
- **Facilidad de ejecución:** Este criterio se refiere a que tan fácil es de entender y seguir la metodología para el desarrollo de la ontología, es decir si las fases que proponen son simples, comprensibles y sencillas para su realización o ejecución. Este criterio tiene las siguientes ponderaciones (Fácil (9), Medio (5), Difícil (1)). Su valor en la escala representativa es 5.
- **Última modificación:** Se utiliza este criterio para ver su última renovación y así poder comparar que tan actualizadas están las metodologías. Para este caso el valor de la ponderación es 9 si está recientemente actualizado el lenguaje y 1 si no es muy reciente su actualización. Su valor en la escala representativa es 1.

- **Nivel de formalidad:** Este criterio permite saber que tan rigurosas son estas metodologías, es decir si esas metodologías no permiten ser flexibles con algunas fases, lo cual nos pueden restringir en algunos casos que serían más factibles para el desarrollo de la ontología. Esta formalidad se refiere también que tan guiada es la metodología, aunque esta formalidad implica rigor, también se puede ser riguroso informalmente. Este criterio tiene las siguientes ponderaciones (Semiformal (9), Formal (5), Conceptual (1)). Su valor en la escala representativa es 3.

Una vez que se tienen definidos los criterios, se procede a evaluar cada uno de los lenguajes. En la Tabla 6 se observa las comparaciones realizadas de los tres lenguajes según cada criterio. Con el análisis de la Tabla 6 se puede deducir que la metodología que mejor se adapta a este trabajo es REFSENO, ya que obtuvo mejores resultados en comparación con las otras metodologías.

METODOLOGÍA / CRITERIO	USCHOLD Y KING	GRÜNINGER Y FOX	REFSENO	METHONTOLOGY	ONTOLOGY DEVELOPMENT 101
Completitud	Completo	Completo	Completo	Completo	Incompleto
Relevancia	Relevante	Relevante	Relevante	Relevante	No Relevante
Soporte/documentación	Bueno	Bueno	Excelente	Excelente	Excelente
Tamaño proyecto	Excelente	Bueno	Excelente	Excelente	Bueno
Facilidad ejecución	Fácil	Medio	Fácil	Fácil	Medio
Tiempo ejecución	Rápido	Lento	Rápido	Lento	Lento
Nivel formalidad	Conceptual, Semi/Formal	Formal	Semiformal	Semiformal	Semiformal
Última modificación	1998	1995	1998	1999	2001

Tabla 6. Criterios de comparación entre las metodologías para el desarrollo de ontologías.

Luego, se realizó una valoración ponderada de las metodologías para el desarrollo de ontologías, haciendo uso de los valores asignados en la descripción de cada criterio de selección. En la Tabla 7 se muestran los valores cuantitativos asignados a cada metodología y los valores ponderados para cada criterio de selección.

CRITERIO (Ponderación)	USCHOLD Y KING	GRÜNINGER Y FOX	REFSENO	METHONTOLOGY	ONTOLOGY DEVELOPMENT 101
Completitud(9)	9	9	9	9	1
Relevancia(9)	9	9	9	9	1
Soporte/documenta (9)	5	5	9	9	9
Tamaño proyecto(7)	9	5	9	9	5
Facilidad ejecución(5)	9	5	9	9	5
Tiempo ejecución (5)	9	1	9	1	1
Nivel de formalidad (3)	5	5	9	9	9
Última modificación (1)	9	1	9	9	9

Tabla 7. Valoración ponderada de las metodologías para el desarrollo de ontologías.

En la Tabla 8 se muestra el valor correspondiente del ponderado de cada criterio normalizado por el valor del factor asignado. Estos valores se hicieron de la siguiente manera: se toma el valor de ponderación de cada criterio de selección y se divide entre 48 que es la suma total de todas las ponderaciones, luego ese valor se multiplica por el valor de cada uno de los factores de cada metodología, y finalmente se suman los valores resultantes para cada metodología, obteniendo el valor total de su evaluación.

METODOLOGÍA / CRITERIO	USCHOLD Y KING	GRÜNINGER Y FOX	REFSENO	METHONTOLOGY	ONTOLOGY DEVELOPMENT 101
Completitud	1,688	1,688	1,688	1,688	1,688
Relevancia	1,688	1,688	1,688	1,688	1,688
Soporte y documentación	0,938	0,938	1,688	0,938	0,938
Tamaño del proyecto	1,313	0,729	1,313	1,313	0,729
Facilidad de ejecución	0,938	0,521	0,938	0,938	0,521
Tiempo de ejecución	0,938	0,104	0,938	0,938	0,104
Nivel de formalidad	0,313	0,313	0,563	0,313	0,313
Última modificación	0,188	0,021	0,188	0,188	0,021
Total Valor	**8,004**	**6,002**	**9,000**	**8,004**	**6,002**

Tabla 8. Valores ponderados de los criterios para cada metodología.

Ahora, analizando la información de la Tabla 9 se observa que hay muchos procesos que las metodologías no proponen o no describen porque no son relevantes o importantes para el desarrollo de la ontología. Si se analiza cada concepto o criterio estudiado se mira que rigurosidad y que completitud tiene cada metodología para llevar a cabo el desarrollo de las ontologías, y esto permite tener una claridad de cual se adapta mejor al proyecto, en este caso un proyecto no tan grande. Si observamos la tabla nos damos cuenta que el menos riguroso y adaptable es REFSENO.

METODOLOGÍA / CRITERIO			USCHOLD Y KING	GRÜNINGER Y FOX	REFSENO	METHONTOLOGY	ONTOLOGY DEVELOPMENT 101
Procesos de gestión del proyecto	**Iniciación de proyecto**		No Propone	No Propone	No Propone	No Propone	No Propone
	Monitoreo y control del proyecto		No Propone	No Propone	No Propone	Propone	No Propone
	Gestión de calidad de la ontología		No Propone	No Propone	No Propone	No Propone	No Propone
Procesos orientados al desarrollo de la ontología	**Procesos del pre-desarrollo**	**Exploración de conceptos**	No Propone	No Propone	No Propone	No Propone	No Propone
		Asignación del sistema	No Propone	No Propone	No Propone	No Propone	No Propone
	Procesos de desarrollo	**Requisitos**	Propone	Propone	Propone	Describe En Detalle	Propone
		Diseño	No Propone	Describe	Propone	Describe En Detalle	Propone
		Implementación	Propone	Describe	Propone	Describe En Detalle	Propone
	Procesos post-desarrollo	**Instalación**	No Propone	No Propone	No Propone	No Propone	No Propone
		Operación	No Propone	No Propone	No Propone	No Propone	No Propone
		Soporte	No Propone	No Propone	No Propone	No Propone	No Propone
		Mantenimiento	No Propone	No Propone	Propone	Propone	No Propone
		Retiro	No Propone	No Propone	No Propone	No Propone	No Propone
Procesos integrales	**Adquisición de conocimientos**		Propone	Propone	Propone	Describe En Detalle	Propone
	Verificación y validación		Propone	Propone	Describe	Describe En Detalle	No Propone
	Gestión de la configuración de la ontología		No Propone	No Propone	No Propone	Describe En Detalle	No Propone
	Documentación		Propone	Propone	No Propone	Describe En Detalle	No Propone
	Capacitación		No Propone	No Propone	No Propone	No Propone	No Propone

Tabla 9. Criterios orientados al proceso para comparación de metodologías para el desarrollo de ontologías.

Finalmente, se optó por utilizar la metodología REFSENO debido a los criterios analizados en la Tabla 8 y Tabla 9. Por último se tiene una conclusión de por qué REFSENO es la metodología que se adapta a nuestro proyecto, según Vizcaíno et al. [51] y Pardo et al. [48]:

- Se basa en una adaptación de la metodología METHONTOLOGY, que es ampliamente utilizado para la definición de ontologías en diferentes campos o áreas.
- REFSENO fue diseñado particularmente como una especialización de METHONTOLOGY, aunque este caso fue diseñado para el desarrollo de ontologías

en ingeniería de software por medio de construcciones para describir los conceptos, atributos y relaciones entre ellas.

- REFSENO hace una distinción entre los diferentes niveles de conocimiento: conceptual y específico del contexto, en cambio los enfoques citados anteriormente representan un nivel de abstracción más elevado.
- La metodología propone diferentes técnicas para revisar la consistencia de la ontología y, además, tiene métodos para controlar la consistencia de las instancias en un nivel de implementación, característica que otras metodologías no consideran.
- Facilita el uso de razonamiento basado en casos ya que considera el peso de cada atributo lo cual ayuda a calcular funciones de similitud.

En el 2012, Pardo et al. proponen las siguientes etapas para el desarrollo de una ontología con REFSENO:

- Planificación.
- Especificación de los requisitos de la ontología.
- Conceptualización, es la ontología propiamente dicha (equivalente al diseño en un sistema software).
- Implementación utilizando una herramienta informática.

4.2 **DEFINICIÓN DE TERMINOS, ATRIBUTOS Y RELACIONES**

Una vez definida la metodología REFSENO para el proyecto se continuó con la aplicación de ella. Se presenta la especificación de requisitos de la ontología que se va a desarrollar, ver la Tabla 10.

CONCEPTO	VALOR
DOMINIO	Diseño e Implementación de un Generador y Ejecutor de Planes de Mitigación del Riesgo contra Ataques XSS, basado en OWASP 2013 e ISO/IEC 27002.
AUTORES	Cristian Daniel Yanza Velasco, Cristhiam Fernandez Ruales.
PROPÓSITO	Integración de los conceptos relacionados con los planes de mitigación de riesgo contra ataques XSS, OWASP 2013 e ISO/IEC 27002.
NIVEL FORMALIDAD	Semiformal (Diagramas de clases UML y tablas y texto REFSENO).
ALCANCE	La siguiente ontología se ha especificado: La Ontología Ontology_thesis (agrupa todos los conceptos, con sus relaciones, instancias y atributos de conceptos).
FUENTES CONOCIMIENTO	Mirar Tabla 11.

Tabla 10. Especificación de requisitos de la ontología.

La ontología se ha representado mediante un diagrama de clases representado en UML y tablas. Para Pardo et al. [48], el diagrama de clases se complementa con una representación textual semiformal basada en el formalismo de REFSENO. En las tablas se colocaron los elementos más relevantes para el proyecto.

ID	FUENTE
1	Towards the automatic and optimal selection of risk treatments for business processes using a constraint programming approach: (Varela-Vaca et al, 2013).
2	Vulnerability Management: (Foreman, 2010).
3	Ontology for attack detection: An intelligent approach to web application security: (Razzaq, 2014a).
4	Semantic security against web application attacks: (Razzaq, 2014b).
5	Incorporating attacker capabilities in risk estimation and mitigation: (Ben Othmane et al, 2015).
6	Current state of research on cross-site scripting (XSS) - A systematic literature review: (Hydara, 2015).

Tabla 11. Fuentes utilizadas.

Una vez definida la especificación de los requisitos el siguiente paso en la metodología fue identificar los conceptos con sus definiciones claras y concisas, que identifique claramente las relaciones entre ellos. Las definiciones precisas de los conceptos incluidos en la ontología se presenta en la Tabla 12, la cual se organiza de la siguiente manera: Las columnas uno y dos muestran el concepto o término, y su súper concepto respectivamente, mientras que la columna tres muestra la definición del término, y la columna cuatro muestra la fuente de la que el término ha sido adoptado o adaptado.

Algunos de los valores utilizados en la cuarta columna son:

- Nuevo [término], si el término tiene un nuevo significado en esta propuesta.
- Tomado de [fuente], si el término se ha definido a partir de una fuente.

CONCEPTO	SUPERCONCEPTO	DESCRIPCION	FUENTE
Type attack	Concept	Forma en que se emite el ataque, está categorizado de acuerdo al top 10 de OWASP. Para nuestro trabajo solo se abarcará el ataque XSS (A3 OWASP 2013)	Nuevo
Xss	Type attack	Es un tipo de ataque que se encuentra en el puesto 3 del top 10 de OWASP, este ataque ocurre cuando un atacante utiliza una aplicación web para enviar código malicioso, generalmente en forma de un script del lado del navegador, a un usuario final.	Nuevo
Dom	Xss	Es un tipo de ataque XSS, DOM (Modelo de objetos del documento) con JavaScript, que permite la apertura de nuevas páginas con código malicioso JavaScript incrustado, afectando el código de la primera página en el sistema local, estos códigos son ejecutados del lado del cliente, porque los filtros utilizados en el servidor no funcionan para este tipo de vulnerabilidades.	Nuevo
Persistent	Xss	Es un tipo de ataque XSS, en el cual consiste en embeber código HTML peligroso en sitios que lo permitan por medio de etiquetas <script> o <iframe>. Es la más grave de todas ya que el código se queda implantado en la web de manera interna y es ejecutado al abrir la aplicación web.	Nuevo

CONCEPTO	SUPERCONCEPTO	DESCRIPCION	FUENTE
No persistent	Xss	Es un tipo de ataque XSS más habitual y consiste en editar los valores que se pasan mediante URL, cambiando el tipo de dato pasado del usuario a la web, haciendo que ese código insertado se ejecute en dicho sitio.	Nuevo
Risk	Concept	Combinación de la probabilidad de un evento y la consecuencia del mismo. (ISO/IEC 27000)	Tomado de OWASP
Likelihood	Concept	Cantidad matemática que representa la posibilidad de que ocurra un evento.	Tomado de OWASP
Impact	Concept	Cambio adverso en el nivel de los objetivos de negocio. (ISO/IEC 27000)	Tomado de OWASP
Threat agent	Likelihood	Individuo o grupo que puede representar una amenaza a los bienes de la organización. (OWASP)	Tomado de OWASP
Vulnerability factors	Likelihood	Asociado a las características de la vulnerabilidad, es decir son las medidas de sus factores.	Tomado de OWASP
Technical impact	Impact	Impacto o consecuencias a la parte de infraestructura como resultado de un ataque satisfactorio a los sistemas de una organización. (OWASP)	Tomado de OWASP
Business impact	Impact	Consecuencias en el negocio como resultado de un ataque satisfactorio. (OWASP)	Tomado de OWASP
Plan	Concept	1. Parte del ciclo de Deming (PDCA) 2. Conjunto de acciones que se generan para mitigar el riesgo.	Nuevo
Action	Concept	Tarea parte de un plan que se ejecutará con el fin de mitigar el riesgo.	Nuevo
Security devices	Concept	Elementos hardware, software u organizacionales que aseguran los bienes de una organización.	Nuevo
Protocol	Concept	Conjunto de reglas y estructura de mensajes que se aplican a una comunicación.	Nuevo
Http message structure	Concept	Forma en la cual se organizan las partes de un mensaje.	Tomado de Razzaq A et al., 2014.
Http	Protocol	Protocolo de comunicación perteneciente a la capa de transporte del modelo OSI, encargado de la trasferencia de información de la web.	Tomado de Razzaq A et al., 2014.
Http response	Http	Después de recibir e interpretar un mensaje de solicitud, el servidor responde con un mensaje de respuesta HTTP: un Status-line, cero o más campos de cabeceras de respuesta seguido de CRLF, una línea vacía (indica el final del campo de cabecera), un message-body opcional.	Tomado de Razzaq A et al., 2014.

CONCEPTO	SUPERCONCEPTO	DESCRIPCION	FUENTE
Http request	Http	Un cliente de HTTP envía una solicitud HTTP a un servidor en la forma de un mensaje de solicitud que incluye el siguiente formato: un Request-line, cero o más campos de cabeceras seguido de CRLF, una línea vacía (es decir, una línea con anda anterior a la CRLF indicando el final de los campos de cabecera) y opcionalmente un message-body	Tomado de Razzaq A et al., 2014.
Control	Concept	Es un medio de gestión del riesgo, incluyendo políticas, procedimiento, directrices, prácticas o estructuras organizativas, que pueden ser administrativas, técnicas, de gestión o de naturaleza legal. También se utiliza como sinónimo de salvaguardia o de contramedida.	Nuevo
State	Concept	Este permite decidir el tratamiento de riesgos, aceptar riesgo residual. Se deriva en 4 estados: Reducir, Transferir, Aceptar y Evitar.	Nuevo
Monitor	Concept	El monitoreo del sistema es utilizado para verificar la efectividad de los controles y planes adoptados y para verificar la conformidad del modelo. Mantiene y mejora el sistema a través del seguimiento y la evaluación del desempeño contra la política y los objetivos de la organización, e informando de los resultados.	Nuevo

Tabla 12. Glosario de conceptos.

Luego, se describieron las relaciones de los conceptos nombrados en la tabla anterior, como se ve en la Tabla 13, la cual se organiza de la siguiente manera: En la columna uno se muestra el nombre de la relación, en la columna dos se muestra los dos conceptos que están implicados en la relación, y en la columna tres se da una pequeña descripción de cómo es la cardinalidad y como se debe leer esta relación.

NOMBRE	CONCEPTOS	DESCRIPCIONES
has a_RL	Risk – Likelihood	Un riesgo tiene una o muchas probabilidades. Una probabilidad se usa para calcular uno o muchos riesgos.
has a_RI	Risk – Impact	Un riesgo tiene uno o muchos impactos. Un impacto se usa para calcular uno o muchos riesgos.
has a_LT	Likelihood – Threat agent	Una probabilidad tiene un valor de agente de amenazas. Un agente de amenaza se usa para calcular uno o muchos riesgos.
has a_LVf	Likelihood – Vulnerability factors	Una probabilidad tiene un valor de factores de vulnerabilidad. Un factor de vulnerabilidad se usa para calcular uno o muchas probabilidades.
has a_ITi	Impact – Technical impact	Un impacto tiene un valor de impacto técnico. Un impacto técnico se usa para calcular uno o muchos impactos.
has a_IBi	Impact – Bbusiness impact	Un impacto tiene un valor de impacto de negocio. Un impacto de negocio se usa para calcular uno o muchos impactos.

NOMBRE	CONCEPTOS	DESCRIPCIONES
using a_TaP	Type attack – Protocol	Un tipo de ataque usa uno o muchos protocolos. Un protocolo es usado por uno o muchos tipos de ataque.
has a_HM	Http – Http message structure	Un Http tiene una sola estructura de mensaje Http. Una estructura de mensaje Http es usada por un Http.
generates a_TaR	Type attack – Risk	Un tipo de ataque genera uno o muchos riesgos. Un riesgo es generado por uno o muchos tipos de ataque.
has a_PlA	Plan – Action	Un plan tiene una o varias acciones. Una acción es usada por uno o varios planes.
executed by_ASd	Action – Security devices	Una acción utiliza un dispositivo de seguridad. Un dispositivo de seguridad es utilizado por una o varias acciones.
using a_PlSd	Plan – Security devices	Un plan usa uno o varios dispositivos de seguridad. Un dispositivo de seguridad es usado por uno o muchos planes.
generates a_SPl	State – Plan	Un estado genera un plan. Un plan es generado por uno o muchos estados.
feeds a_MoS	Monitor - State	Un monitor alimenta un estado. Un estado es alimentado por un monitor.
generates a_SC	State – Control	El estado genera un control. El control es generado por un estado.
is associated_CTa	Control – Type attack	Un control está asociado a uno o más tipos de ataques. Un tipo de ataque se asocia a uno o más controles.
is associated_CR	Control – Risk	Un control está asociado a uno o más riesgos. Un riesgo se asocia a uno o más controles.

Tabla 13. Tabla de interrelaciones.

En la Tabla 14 se tiene las relaciones inversas de los conceptos.

NOMBRE	CONCEPTOS	DESCRIPCIONES
is part of_LR	Likelihood – Risk	Un riesgo tiene una o muchas probabilidades. Una probabilidad se usa para calcular uno o muchos riesgos.
is part of_IR	Impact – Risk	Un riesgo tiene uno o muchos impactos. Un impacto se usa para calcular uno o muchos riesgos.
is part of_TL	Threat agent - Likelihood	Una probabilidad tiene un valor de agente de amenazas. Un agente de amenaza se usa para calcular uno o muchos riesgos.
is part of_VfL	Vulnerability Factors - likelihood	Una probabilidad tiene un valor de factores de vulnerabilidad. Un factor de vulnerabilidad se usa para calcular uno o muchas probabilidades.
is part of_TiI	Technical impact – Impact	Un impacto tiene un valor de impacto técnico. Un impacto técnico se usa para calcular uno o muchos impactos.
is part of_BiI	Business impact - Impact	Un impacto tiene un valor de impacto de negocio. Un impacto de negocio se usa para calcular uno o muchos impactos.
is used by_PTa	Protocol - Type attack	Un tipo de ataque usa uno o muchos protocolos. Un protocolo es usado por uno o muchos tipos de ataque.

NOMBRE	CONCEPTOS	DESCRIPCIONES
is part of_MH	Http message structure - Http	Un Http tiene una sola estructura de mensaje. Una estructura de mensaje es usada por un Http.
is generated by_RTa	Risk – Type attack	Un tipo de ataque genera uno o muchos riesgos. Un riesgo es generado por uno o muchos tipos de ataque.
is part of_APl	Action - Plan	Un plan tiene una o varias acciones. Una acción es usada por uno o varios planes.
running an_SdA	Security devices – Action	Una acción utiliza un dispositivo de seguridad. Un dispositivo de seguridad es utilizado por una o varias acciones.
is used by_SdPl	Security devices – Plan	Un plan usa uno o varios dispositivos de seguridad. Un dispositivo de seguridad es usado por uno o muchos planes.
is generated by_PlS	Plan – State	Un estado genera un plan. Un plan es generado por uno o muchos estados.
is fed by_SMo	State - Monitor	Un monitor alimenta un estado. Un estado es alimentado por un monitor.
is generated by_CS	Control – State	El estado genera un control. El control es generado por un estado.
has been associated_TaC	Type attack – Control	Un control está asociado a uno o más tipos de ataques. Un tipo de ataque se asocia a uno o más controles.
has been associated_RC	Risk – Control	Un control está asociado a uno o más riesgos. Un riesgo se asocia a uno o más controles.

Tabla 14. Interrelaciones inversas.

4.3 CONCEPTUALIZACIÓN DE LA ONTOLOGÍA

Una vez definidos todos los conceptos y las relaciones entre ellos, y además ya evaluados y aceptados por expertos en el tema, se decide hacer un modelo software de los conceptos identificados y sus relaciones, haciendo uso de la herramienta StarUML (Figura 2). Este modelo contiene los conceptos con sus respectivas relaciones, además de sus atributos, y cardinalidades.

Este modelo nos permite ver de una manera fácil y rápida de cómo se van a relacionar los conceptos y también que atributos tienen, nos permite observar de una forma más específica y clara como va estar interrelacionada la ontología que se va a crear. Una vez que ya se tiene completamente término este modelo se procede a crear la ontología en alguna herramienta que sea adecuada para ello.

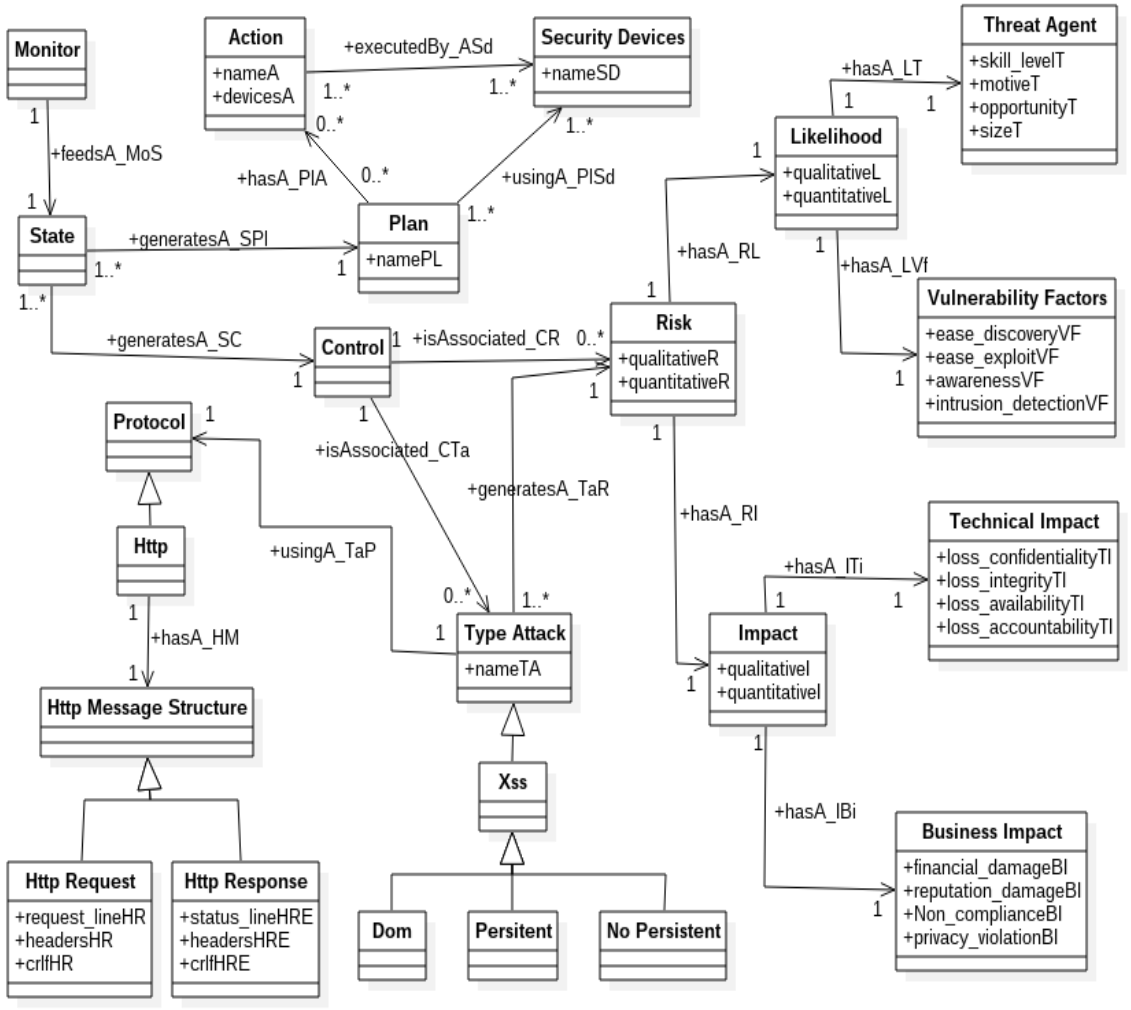

Figura 2. Diagrama de UML de la ontología.

4.4 **IMPLEMENTACIÓN DE LA ONTOLOGÍA**

Así como son variadas las metodologías de desarrollo de ontologías, también son variadas las herramientas relacionadas con las ontologías. Las ontologías requieren de un lenguaje lógico y formal para ser expresadas. Se han desarrollado numerosos lenguajes para este fin, algunos basados en la lógica de predicados y otros basados en frames (taxonomías de clases y atributos), que tienen un mayor poder expresivo, pero menor poder de inferencia; e incluso existen lenguajes orientados al razonamiento [46]. Con respecto a las herramientas de desarrollo de ontologías, éstas han mejorado enormemente desde la creación de los primeros entornos, que sirven para la construcción de nuevas ontologías o bien para la reutilización de las existentes, destacan entre sus funcionalidades la edición y consulta, así como la exportación e importación de

ontologías, la visualización en diversos formatos gráficos. Estas herramientas han sido creadas para integrar la tecnología de la ontología en los sistemas de información actuales, y se construyen como entornos integrados robustos que proporcionan soporte tecnológico a la mayor parte de las actividades de la ontología en el ciclo de vida, tienen arquitecturas extensibles, basadas en componentes donde los nuevos módulos se pueden agregar fácilmente para proporcionar una mayor funcionalidad al entorno según Pardo et al. [15]. Entre las herramientas de desarrollo de ontologías se encuentran Ontolingua Server, OdonEdit, WebODE, Protégé, Hozo, entre otras.

Tras evaluar las herramientas anteriores, se ha llegado a la conclusión de que la más recomendable para este trabajo es la herramienta Protégé en su versión 5.0.0 que implementa el lenguaje OWL (Ontology Web Language) para el modelado de ontologías basadas en Frames. La misma ha sido desarrollada por la Universidad de Stanford y se utiliza para el desarrollo de Ontologías y Sistemas basados en el conocimiento por medio de una interfaz de usuario que facilita la creación de estructuras de frames con clases, slots e instancias de una forma integrada.

Se escogió Protégé por las siguientes razones:

- Es un editor gratuito y de código abierto para construir ontologías y representar el conocimiento.
- Detrás de la herramienta hay una gran comunidad de desarrolladores y usuarios universitarios, empresariales y gubernamentales.
- Está escrito en Java, para nuestro proyecto es importante porque la arquitectura y la aplicación que se está realizando está desarrollada en este mismo lenguaje, lo que facilita más la integrabilidad con la herramienta; y la programación es orientada a objetos.
- Modelo de conocimiento extensible para permitir a los usuarios redefinir las representaciones primitivas.
- Un formato de archivo de salida personalizable para adaptarse a cualquier lenguaje formal.
- Interfaz de usuario personalizable.
- Potente arquitectura plug-in para permitir la integración con otras aplicaciones.

También se toma en cuenta la opinión de expertos en el tema sobre que herramienta era mejor para nuestro proyecto, donde coincidieron con la herramienta Protégé, al igual que en estos trabajos donde recomiendan su uso, entre los trabajos más relevantes se encuentran: Construcción de un modelo conceptual para gramáticas formales y máquinas abstractas con ontologías usando Protégé [46]; Ontology for attack detection: An intelligent approach to web application security [22]; A Framework to Support the Harmonization between Multiple Models and Standards [48]; Methodologies and methods for building ontologies [47]; Integración de técnicas avanzadas de web semántica para diseño y visualización de estructuras de conocimiento [19]; A Methodology for Ontology Building [20]; Semantic-Driven e-Government: Application of Uschold and King Ontology Building Methodology for SemanticOntologyModels Development [18]; Ontologies: a technical of knowledge representation [14].

En la Figura 3 se muestra la ontología implementada en Protégé 5.0.0.

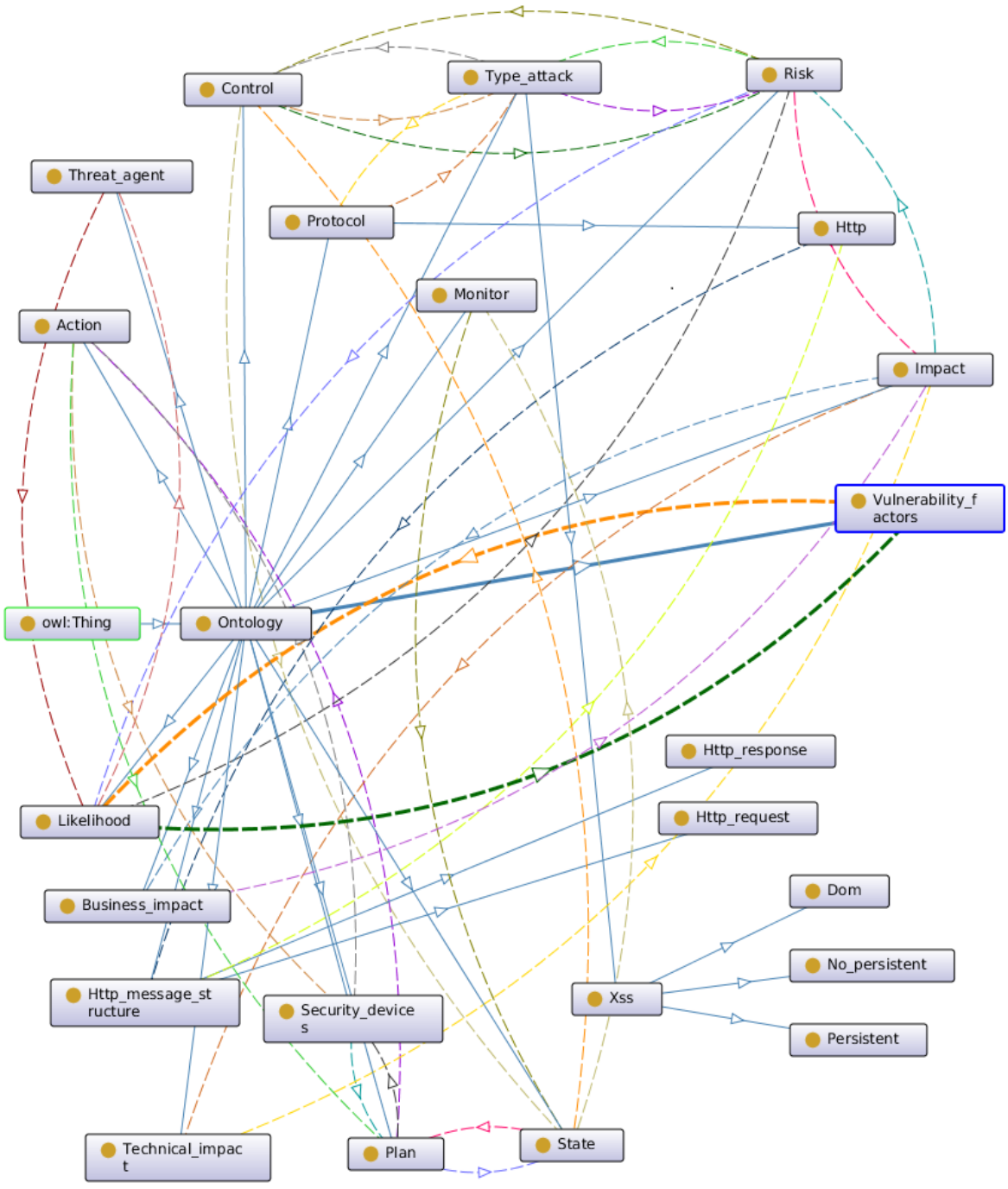

Figura 3 Implementación en Protégé**.**

4.5 **AJUSTE Y CALIBRACIÓN**

Este proceso es evolutivo en todas sus fases, inicialmente se crea una tabla de conceptos como se muestra en la Tabla 15, pero que se fue refinando al paso del tiempo, cada vez que se hacen evaluaciones y validaciones con los expertos.

Como todo proyecto software, no son totalmente correctos en sus primeras iteraciones, se deben ir mejorando, esto permite corrección de errores, mejorar el rendimiento, mejoras funcionales y eficiencia y fiabilidad, en este trabajo se crearon las tablas y el modelo propuesto por REFSENO, que cada vez que se validaban se iban eliminando o agregando conceptos, ya que en un análisis más profundo se observaban ambigüedades, y conceptos que no aplicaban para el proyecto. Una vez definido y calibrado totalmente las tablas y el modelo, ya se procede a implementar la ontología en la herramienta Protégé, así logrando el éxito de su desarrollo.

A continuación mostraremos que cambios surgieron alrededor de todo el trabajo: Inicialmente como lo indica la metodología de REFSENO se creó la tabla de conceptos como se muestra en la Tabla 15, en la cual se observan muchos conceptos que no se encuentran en la Tabla 12, que es la versión completa y refinada del proyecto, también se agregaron nuevos conceptos. Estos cambios fueron necesarios para tener una ontología más óptima y más adaptable para el proyecto, se tuvo muchos cambios en cuanto a conceptos, por lo que por transitividad cambio la tabla de las relaciones, el proceso de los cambios se observa a continuación desde la tabla inicial, hasta llegar a la tabla final.

CONCEPTO	SUPERCONCEPTO	DESCRIPCION	FUENTE
Type attack	Concepto	Forma en que se emite el ataque, está categorizado de acuerdo al top 10 de OWASP. Para nuestro trabajo solo se abarcará el ataque XSS (A3 OWASP 2013)	Nuevo
Vulnerability	Concepto	Debilidad de un bien que puede ser explotada por una amenaza. (ISO/IEC 2700)	Tomado de OWASP
Risk	Concepto	Combinación de la probabilidad de un evento y la consecuencia del mismo. (ISO/IEC 27000)	Tomado de OWASP
Likelihood	Concepto	Cantidad matemática que representa la posibilidad de que ocurra un evento.	Tomado de OWASP
Impact	Concepto	Cambio adverso en el nivel de los objetivos de negocio. (ISO/IEC 27000)	Tomado de OWASP
Threat agent	Probabilidad	Individuo o grupo que puede representar una amenaza a los bienes de la organización. (OWASP)	Tomado de OWASP
Vulnerability factors	Probabilidad	Asociado a las características de la vulnerabilidad, es decir son las medidas de sus factores.	Tomado de OWASP
Technical impact	Impacto	Impacto o consecuencias a la parte de infraestructura como resultado de un ataque satisfactorio a los sistemas de una organización. (OWASP)	Tomado de OWASP
Business impact	Impacto	Consecuencias en el negocio como resultado de un ataque satisfactorio. (OWASP)	Tomado de OWASP
Plan	Concepto	1. Parte del ciclo de Deming (PDCA) 2. Conjunto de acciones que se generan para mitigar el riesgo.	Nuevo

CONCEPTO	SUPERCONCEPT	DESCRIPCION	FUENTE
Action	Concepto	Tarea parte de un plan que se ejecutará con el fin de mitigar el riesgo.	Nuevo
Security devices	Concepto	Elementos hardware, software u organizacionales que aseguran los bienes de una organización.	Nuevo
Protocol	Concepto	Conjunto de reglas y estructura de mensajes que se aplican a una comunicación.	Tomado de Razzaq A et al., 2014.
Http message structure	Concepto	Forma en la cual se organizan las partes de un mensaje.	Tomado de Razzaq A et al., 2014.
Request line	Concepto	Vista general de la petición, con los métodos de la petición.	Tomado de Razzaq A et al., 2014.
Payload	Concepto	Datos que se envían como cuerpo de una petición	Tomado de Razzaq A et al., 2014.
Request header	Concepto	Meta datos de la comunicación, comúnmente contiene información de quien realiza la petición.	Tomado de Razzaq A et al., 2014.
Url	Concepto	Localizador uniforme de recursos. Cadena de caracteres que indica la ruta para acceder a un recurso de un sistema.	Tomado de Razzaq A et al., 2014.
Query string	Concepto	Cadena de caracteres con pares de datos (key=value) que se envían como información adicional en una petición.	Tomado de Razzaq A et al., 2014.
Parameter cookie	Concepto	Trozo de información que se produce como resultado de la comunicación entre un servidor y un cliente, la cual le indica al servidor la actividad previa del usuario.	Tomado de Razzaq A et al., 2014.
Tag	Concepto	Etiqueta del lenguaje de marcas de hipertexto que indica un elemento de una página web.	Tomado de Razzaq A et al., 2014.
Malicious input	Concepto	Cadena de caracteres con código malicioso de scripting que puede ser ejecutado en los clientes o en el servidor, comúnmente es enviada a través de los controles de formulario de una página web.	Tomado de Razzaq A et al., 2014.
Parameter value	Concepto	El valor que toma una variable que puede derivar de una query string, tag o una cookie.	Tomado de Razzaq A et al., 2014.

CONCEPTO	SUPERCONCEPT	DESCRIPCION	FUENTE
Http	Protocolo	Protocolo de comunicación perteneciente a la capa de transporte del modelo OSI, encargado de la trasferencia de información de la web.	Tomado de Razzaq A et al., 2014.
Html	Concepto	Lenguaje con el cuál se define la estructura de la información en las páginas web.	Tomado de Razzaq A et al., 2014.
Control	Objetivo control	Es un medio de gestión del riesgo, incluyendo políticas, procedimiento, directrices, prácticas o estructuras organizativas, que pueden ser administrativas, técnicas, de gestión o de naturaleza legal. También se utiliza como sinónimo de salvaguardia o de contramedida.	Nuevo
Control objective	Dominio	Declaración que describe lo que se quiere lograr como resultado de los controles de ejecución.	Nuevo
Domain	Concepto	Área organizacional o técnica donde se aplican los controles.	Nuevo

Tabla 15. Glosario de conceptos versión 1.0.

La tabla de conceptos de la Tabla 15 fue refinada una vez más generando los conceptos que se muestran a continuación en la Tabla 16.

CONCEPTO	SUPERCONCEPTO	DESCRIPCION	FUENTE
Type attack	Concepto	Forma en que se emite el ataque, está categorizado de acuerdo al top 10 de OWASP. Para nuestro trabajo solo se abarcará el ataque XSS (A3 OWASP 2013)	Nuevo
Risk	Concepto	Combinación de la probabilidad de un evento y la consecuencia del mismo. (ISO/IEC 27000)	Tomado de OWASP
Likelihood	Concepto	Cantidad matemática que representa la posibilidad de que ocurra un evento.	Tomado de OWASP
Impact	Concepto	Cambio adverso en el nivel de los objetivos de negocio. (ISO/IEC 27000)	Tomado de OWASP
Threat agent	Probabilidad	Individuo o grupo que puede representar una amenaza a los bienes de la organización. (OWASP)	Tomado de OWASP
Vulnerability factors	Probabilidad	Asociado a las características de la vulnerabilidad, es decir son las medidas de sus factores.	Tomado de OWASP
Technical impact	Impacto	Impacto o consecuencias a la parte de infraestructura como resultado de un ataque satisfactorio a los sistemas de una organización. (OWASP)	Tomado de OWASP
Business impact	Impacto	Consecuencias en el negocio como resultado de un ataque satisfactorio. (OWASP)	Tomado de OWASP
Plan	Concepto	1. Parte del ciclo de Deming (PDCA) 2. Conjunto de acciones que se generan para mitigar el riesgo.	Nuevo
Action	Concepto	Tarea parte de un plan que se ejecutará con el fin de mitigar el riesgo.	Nuevo
Security devices	Concepto	Elementos hardware, software u organizacionales que aseguran los bienes de una organización.	Nuevo

CONCEPTO	SUPERCONCEPTO	DESCRIPCION	FUENTE
Protocol	Concepto	Conjunto de reglas y estructura de mensajes que se aplican a una comunicación.	Tomado de Razzaq A et al., 2014.
Http message structure	Concepto	Forma en la cual se organizan las partes de un mensaje.	Tomado de Razzaq A et al., 2014.
Http	Protocolo	Protocolo de comunicación perteneciente a la capa de transporte del modelo OSI, encargado de la trasferencia de información de la web.	Tomado de Razzaq A et al., 2014.

Tabla 16. Glosario de conceptos versión 1.1.

Después de esta versión de la tabla de conceptos se logró refinar completamente la tabla y la versión final se observó anteriormente en este capítulo en la Tabla 12.

A continuación, en la Tabla 17 se muestra el proceso evolutivo de las relaciones de cada concepto:

NOMBRE	CONCEPTOS	DESCRIPCIONES
explotada por un_VTa	Vulnerability – Tipe attack	Una vulnerabilidad puede ser explotada por uno o muchos tipos de ataque. Un tipo de ataque explota uno o muchas vulnerabilidades
genera un_VR	Vulnerability – Risk	Una vulnerabilidad genera una medida de riesgo. Un riesgo es generado por una o muchas vulnerabilidades
explota una	Tipe attack – Vulnerability	Un tipo de ataque explota una vulnerabilidad. Una vulnerabilidad es explotada por uno o muchos tipos de ataques.
tiene una_RP	Risk – Likelihood	Un riesgo tiene una o muchas probabilidades. Una probabilidad se usa para calcular uno o muchos riesgos.
tiene un_RI	Risk – Impact	Un riesgo tiene uno o muchos impactos. Un impacto se usa para calcular uno o muchos riesgos.
usa un_TaPr	Tipe attack – Protocol	Un tipo de ataque usa uno o muchos protocolos. Un protocolo es usado por uno o muchos tipos de ataque.
tiene una_HEm	Http – Message structure	Un Http tiene una sola estructura de mensaje. Una estructura de mensaje es usada por un Http.
tiene parte de_EmRe	Message structure – Request	Una estructura de mensaje tiene parte de un request. Un request es parte de una estructura de mensaje.
tiene un_ReRl	Request – Request line	Un request tiene un request line. Un request line es usado por un request.
tiene un_ReRh	Request – Request header	Un request tiene un request header. Un request header es usado por un request.
tiene un_RePa	Request – Payload	Un request tiene un payload. Un payload es usado por un request.

NOMBRE	CONCEPTOS	DESCRIPCIONES
contiene una_RlU	Request line – Url	Un request line contiene una URL. Una URL es contenida por un request line.
contiene una_UQs	Url – Query string	Una URL contiene una o varias query string. Una query string es contenida por una URL.
contiene una_RhPc	Request header – Parameter cookie	Un request header contiene una cookie. Una cookie es usada por un request header.
usa un_PaHm	Payload – Html	Un payload usa un HTML. Un HTML es usado por un payload.
tiene un_HmT	Html – Tag	Un HTML tiene uno o varios tag. Un tag es usado por un HTML.
contiene un_QsPv	Query string – Parameter value	Un query string contiene un parameter value. Un parameter value es contenido por un query string.
contiene un_PcPv	Parameter cookie – Parameter value	Una cookie contiene un parameter value. Un parameter value es contenido por una cookies.
contiene un_TPv	Tag – Parameter value	Un tag contiene uno o varios parameter value. Un parameter value es contenido en uno o varios tag.
es infectado por una_PvEma	Parameter value – Malicious input	Un parameter value es infectado por una entrada maliciosa. Una entrada maliciosa infecta uno o varios parameter value.
explota una_EmaV	Malicious input – Vulnerability	Una entrada maliciosa explota una o varias vulnerabilidades. Una vulnerabilidad es explotada por una o varias entradas maliciosas.
disminuye una_CV	Control – Vulnerability	Un control disminuye una o varias vulnerabilidades. Una vulnerabilidad es disminuida por uno o varios controles.
usa un_CPl	Control – Plan	Un control usa un o varios planes. Un plan es usado por un control.
tiene una_PlA	Plan – Action	Un plan tiene una o varias acciones. Una acción es usada por uno o varios planes.
utiliza un_ADs	Action – Security devices	Una acción utiliza un dispositivo de seguridad. Un dispositivo de seguridad es utilizado por una o varias acciones.
mitiga un_CR	Control – Risk	Un control mitiga uno o varios riesgos. Un riesgo es mitigado por uno o varios controles.
contiene un_RhPv	Request header – Parameter value	Un request header contiene un parameter value. Un parameter value es contenido por un request header.

Tabla 17. Interrelaciones versión 1.0.

Las interrelaciones también fueron refinadas y se muestran a continuación en la Tabla 18.

NOMBRE	CONCEPTOS	DESCRIPCIONES
has a_RL	Risk – Likelihood	Un riesgo tiene una o muchas probabilidades. Una probabilidad se usa para calcular uno o muchos riesgos.
has a_RI	Risk – Impact	Un riesgo tiene uno o muchos impactos. Un impacto se usa para calcular uno o muchos riesgos.
has a_LT	Likelihood – Threat agent	Una probabilidad tiene un valor de agente de amenazas. Un agente de amenaza se usa para calcular uno o muchos riesgos.
has a_LVf	Likelihood – Vulnerability factors	Una probabilidad tiene un valor de factores de vulnerabilidad. Un factor de vulnerabilidad se usa para calcular uno o muchas probabilidades.
has a_ITi	Impact – Technical impact	Un impacto tiene un valor de impacto técnico. Un impacto técnico se usa para calcular uno o muchos impactos.
has a_Ibi	Impact – Business impact	Un impacto tiene un valor de impacto de negocio. Un impacto de negocio se usa para calcular uno o muchos impactos.
using a_TaP	Type attack – Protocol	Un tipo de ataque usa uno o muchos protocolos. Un protocolo es usado por uno o muchos tipos de ataque.
has a_HM	Http – Message structure	Un Http tiene una sola estructura de mensaje. Una estructura de mensaje es usada por un Http.
generates a_TaR	Type attack – Risk	Un tipo de ataque genera uno o muchos riesgos. Un riesgo es generado por uno o muchos tipos de ataque.
has a_PlA	Plan – Action	Un plan tiene una o varias acciones. Una acción es usada por uno o varios planes.
executed by_ASd	Action – Security devices	Una acción utiliza un dispositivo de seguridad. Un dispositivo de seguridad es utilizado por una o varias acciones.
using a_PlSd	Plan – Security devices	Un plan usa uno o varios dispositivos de seguridad. Un dispositivo de seguridad es usado por uno o muchos planes.
mitigated by_RPl	Risk – Plan	Un riesgo es mitigado por un plan. Un plan mitiga uno o muchos riesgos.
is controlled by_TaPl	Type attack – Plan	Un tipo de ataque es controlado por un plan. Un plan controla uno o muchos tipos de ataques.

Tabla 18. Interrelaciones versión 1.1.

A continuación, en la Tabla 19 se muestran las interrelaciones inversas para la versión 1.1 de la tabla de relaciones.

NOMBRE	CONCEPTOS	DESCRIPCIONES
has a_RL	Risk – Likelihood	Una vulnerabilidad puede ser explotada por uno o muchos tipos de ataque. Un tipo de ataque explota uno o muchas vulnerabilidades
has a_RI	Risk – Impact	Una vulnerabilidad genera una medida de riesgo. Un riesgo es generado por una o muchas vulnerabilidades
has a_LT	Likelihood – Threat agent	Un tipo de ataque explota una vulnerabilidad. Una vulnerabilidad es explotada por uno o muchos tipos de ataques.
has a_LVf	Likelihood – Vulnerability factors	Un riesgo tiene una o muchas probabilidades. Una probabilidad se usa para calcular uno o muchos riesgos.
has a_ITi	Impact – Technical impact	Un riesgo tiene uno o muchos impactos. Un impacto se usa para calcular uno o muchos riesgos.
has a_Ibi	Impact – Business impact	Un tipo de ataque usa uno o muchos protocolos. Un protocolo es usado por uno o muchos tipos de ataque.
using a_TaP	Type attack – Protocol	Un Http tiene una sola estructura de mensaje. Una estructura de mensaje es usada por un Http.
has a_HM	Http – Message structure	Una estructura de mensaje tiene parte de un request. Un request es parte de una estructura de mensaje.
generates a_TaR	Type attack – Risk	Un request tiene un request line. Un request line es usado por un request.
has a_PlA	Plan – Sction	Un request tiene un request header. Un request header es usado por un request.
executed by_ASd	Action – Security devices	Un request tiene un payload. Un payload es usado por un request.
using a_PlSd	Plan – Security devices	Un request line contiene una URL. Una URL es contenida por un request line.
mitigated by_RPl	Risk – Plan	Una URL contiene una o varias query string. Una query string es contenida por una URL.
is controlled by_TaPl	Type attack – Plan	Un request header contiene una cookie. Una cookie es usada por un request header.

Tabla 19. Interrelaciones inversas versión 1.1.

Después de esta versión de la tabla de relaciones se logra refinar completamente los conceptos e interrelaciones directas e inversas de la ontología, para finalmente contar con 14 conceptos de los cuales 4 son nuevos, 3 fueron tomados de Razzaq et al. [22] y 7 fueron abstraídos de OWASP. El número de relaciones de la versión 1.1 cuenta con 14 relaciones refinadas. El proceso de refinamiento de conceptos y relaciones permite que la ontología se aproxime a la representación del conocimiento necesario para el planificador del Sistema Control de Seguridad, además de unificar los conceptos existentes en OWASP, la norma ISO/IEC 27002 y la ontología propuesta por Razzaq et al. [22].

En conclusión, se seleccionó la ontología como forma de representación y especificación del conocimiento necesario para la elaboración de planes de mitigación del riesgo basados en OWASP 2013 y la norma ISO/IEC 27002.

CAPÍTULO 5

IMPLEMENTACIÓN DE LOS MÓDULOS GENERADOR Y EJECUTOR DE PLANES

En este capítulo se presenta los diferentes modelos para la representación del sistema completo, del módulo generador de planes y el módulo ejecutor de planes, entre los modelos se encuentra diagrama de clases, diagrama de actividades, estructura de red y la arquitectura, que permite generar una visión más clara del sistema. En este capítulo también se muestra la implementación de cada módulo mencionado anteriormente usando la metodología de desarrollo XP (eXtreme Programming).

5.1 METODOLOGÍA DE DESARROLLO

Extreme programming es una metodología para el desarrollo ágil de software, permitiendo enfocar los esfuerzos en crear un producto software que cubra las necesidades inmediatas. Se seleccionó XP como metodología de desarrollo de este proyecto ya que permite trabajar de forma iterativa e incremental, generando módulos de forma rápida y ordenada.

5.1.1 Ciclo de vida de XP

El proyecto consistió en construir los módulos planificador y ejecutor del Sistema Control de Seguridad ISO/IEC 27002 – OWASP, los cuales permitieran inferir un plan de mitigación del riesgo basado en la norma ISO/IEC 27002 y la metodología de medición del riesgo OWASP 2013.

El desarrollo estuvo guiado por entregas de avances funcionales en periodos cortos lo que permitió obtener una retroalimentación del funcionamiento y necesidades del Sistema Control de Seguridad.

5.1.1.1 Fase de exploración

Se estudió la arquitectura y tecnologías usadas en el Sistema Control de Seguridad ISO/IEC 27002 – OWASP, permitiendo determinar el lenguaje usado (JAVA) y el funcionamiento general del sistema. De la misma forma, se estudiaron diferentes librerías que permitieran realizar planificación automática basadas en PDDL y conexión remota con dispositivos a través de protocolo SSH (Secure Shell) que se ajustaron al lenguaje de programación. La librería seleccionada para la planificación fue PDDL4J desarrollada por Pellier [58] en 2011. La librería seleccionada para la conexión a través de SSH fue Jsch.

De acuerdo a lo anterior, se creó un documento de requerimientos que detalla el comportamiento y restricciones de los módulos desarrollados. En el Anexo 6 se presenta el documento de requisitos.

5.1.1.2 **Fase de planeación**

Se definieron los requisitos funcionales y no funcionales de los módulos, los cuales fueron desarrollados en 3 iteraciones: la primera correspondió al desarrollo del módulo de planificación automática basada en PDDL, la segunda en el desarrollo del módulo ejecutor junto con la conexión SSH y la tercera iteración correspondió al desarrollo de un módulo administrador de dispositivos que permite determinar el estado de cada dispositivo de seguridad conectado.

La Tabla 20 se muestra las iteraciones realizadas durante el desarrollo de los módulos junto con los responsables.

NO. DE ITERACIONES	FUNCIONALIDAD	RESPONSABLE
1	Módulo Planificador	Cristian Yanza, Cristhiam Fernández
2	Módulo Ejecutor y conexión SSH	Cristian Yanza, Cristhiam Fernández
3	Módulo administrador de dispositivos	Cristian Yanza, Cristhiam Fernández

Tabla 20. Iteraciones del ciclo de desarrollo

Debido a que los requisitos de los módulos desarrollados fueron definidos de acuerdo a los requerimientos del Sistema Control de Seguridad ISO/IEC 27002 – OWASP, se decidió obviar las historias de usuario.

5.1.1.3 **Fase de producción**

Se diseñaron las pruebas unitarias en primera instancia, como lo dice la metodología, se diseñaron diagramas de actividades y de clases para modelar el comportamiento del sistema y la codificación se realizó en parejas, por los autores del presente trabajo: Cristian Daniel Yanza Velasco y Cristhiam Gabriel Fernández.

5.1.1.4 **Fase de muerte**

En la fase final del desarrollo se evaluó que los módulos desarrollados cumplieran con los requisitos del Sistema Control de Seguridad ISO/IEC 27002 – OWASP, y se generó la documentación de cada módulo de acuerdo a las pruebas realizadas.

5.2 **ARQUITECTURA DEL SISTEMA**

Para este trabajo, inicialmente se propone una arquitectura que representa todo el sistema del control con sus componentes, el cual permite una previa verificación de que todos los elementos esenciales o importantes se encuentran identificados y representados correctamente.

En la Figura 4, se observa un diagrama que muestra la arquitectura del sistema, en la que se puede detallar claramente los elementos que conforman el sistema, se observa internamente como se conforman los módulos y sus relaciones entre ellos, además del tipo de protocolo que usan para comunicarse. La representación de la arquitectura está adaptada a la propuesta que se realizó en el trabajo de Jurado et al. [7], pero para este trabajo se hace una profundización y una mejora en el módulo generador de planes y en el módulo ejecutor de planes.

La mejora del módulo planificador consistió en implementar la planificación de forma automática basada en una representación del conocimiento, puesto que, el módulo del Sistema Control de Seguridad contenía acciones estáticas en su programación, lo que lo hacía poco escalable en acciones y a dispositivos de seguridad se refiere. Con la mejora, es posible agregar nuevos dispositivos de seguridad de forma sencilla a partir de un archivo XML (eXtensible Markup Language) y nuevas acciones lo que enriquece el dominio de planificación.

Por otro lado, la mejora del ejecutor consiste en establecer una conexión segura a través del protocolo SSH, para que los mensajes viajen cifrados a través de la red, además de la capacidad de informar del estado de cada dispositivo al dominio de planificación, permitiendo generar acciones alternativas ante un dispositivo inactivo.

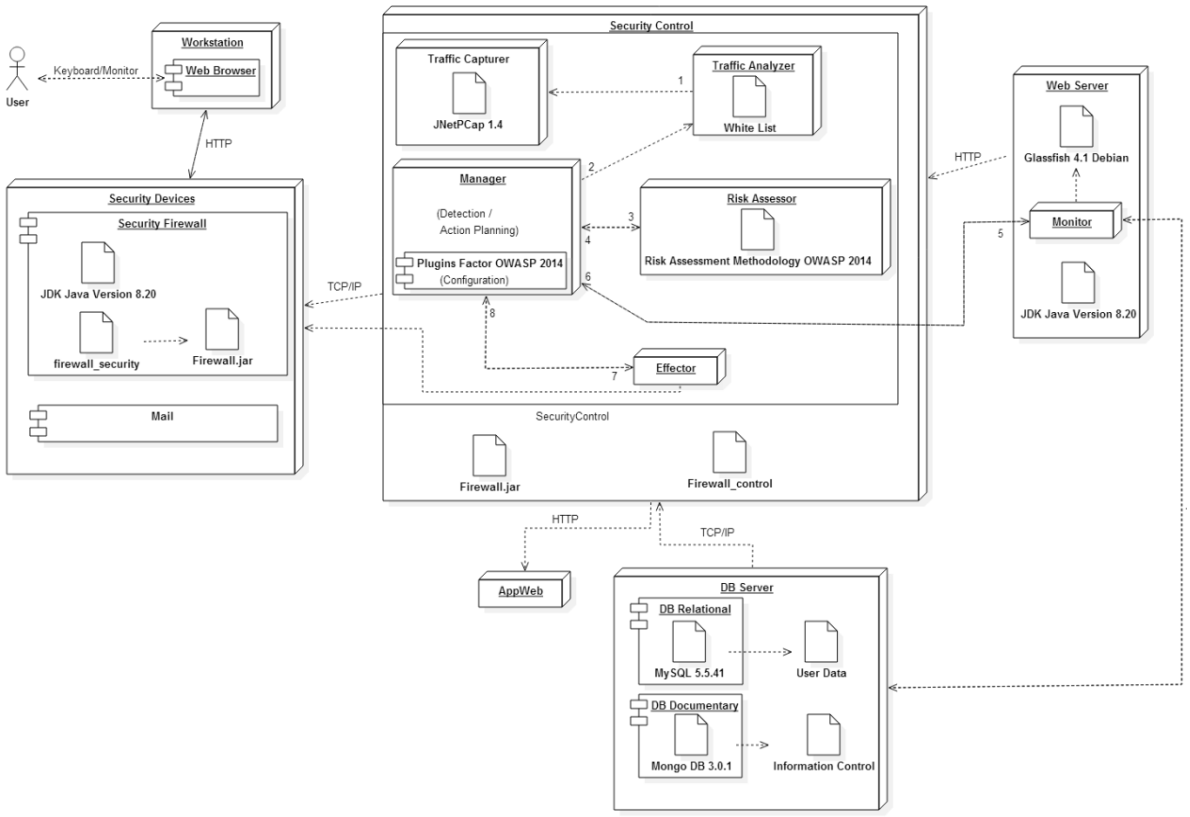

Figura 4 Arquitectura del sistema.

A continuación, en la Figura 5 se muestra la estructura completa del Sistema Control de Seguridad ISO/IEC 27002-OWASP, la cual comprende una red privada con una conexión a internet a través de la cual llegan las solicitudes a la aplicación web a proteger, una MZ (zona militarizada, solo visible a la red interna) que aloja el servidor de base de datos que aloja el registro de actividades del Sistema. Una DMZ (Zona Desmilitarizada, accesible a través de internet) que aloja una aplicación cuya función es la de mostrar la actividad del Sistema. Un servidor que aloja el Sistema Control de Seguridad y a través de la cual pasa casa solicitud para ser analizada, evaluada y procesada. Por último, la aplicación web vulnerable, por fuera de la red del sistema y solo alcanzable a través del control de seguridad. Esta representación muestra como está conectado el sistema en cuanto a sus componentes de red. La arquitectura de red muestra las especificaciones de hardware que debe tener cada servidor requerido para la implantación del sistema. Entre las especificaciones se encuentra la CPU, tarjeta RAM, disco duro, tarjetas de red. En la figura se puede ver las direcciones IP de cada servidor con sus respectivas interfaces, además de los servidores que se encuentran en la red militarizada (MZ) y en la red desmilitarizada (DMZ).

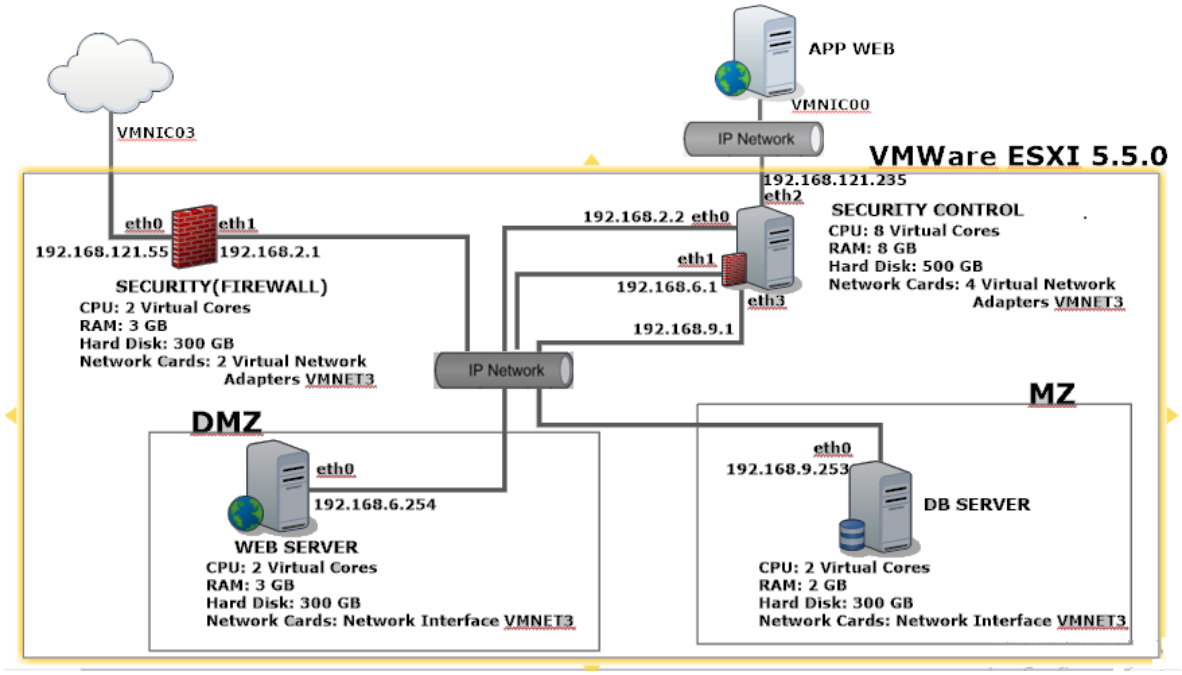

Figura 5 Estructura de red del sistema.

Una vez que se tiene la arquitectura y la estructura de red del sistema, se realiza una representación del proceso o funcionamiento del sistema, modelado a través de un diagrama de actividades. En la Figura 6 se puede visualizar el diagrama de actividades general del sistema.

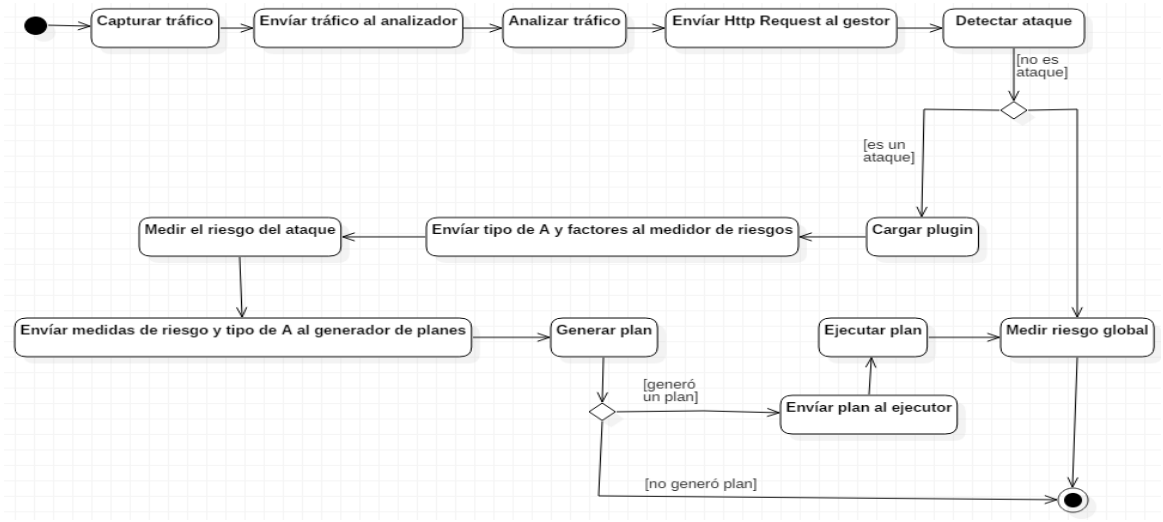

Figura 6 Diagrama de actividad del sistema.

A continuación, tomado de Jurado et al. [7] se hace una descripción general de cada uno de los componentes de la arquitectura y la descripción que se seleccionaron para el desarrollo.

- **Módulo Capturador de Tráfico (Traffic Capturer):** Este módulo es el encargado de capturar el tráfico de red (paquete), que va dirigido hacia una dirección IP y puerto que se le especifique. El tráfico de red que cumple las condiciones definidas anteriormente es enviado al módulo Analizador de Tráfico.
- **Módulo Analizar de Tráfico (Traffic Analyzer):** Este módulo es el encargado de obtener la cabecera HTTP, de los elementos enviados por el módulo Capturador de Tráfico y obtener la URL, payload e IP de origen para enviarlos al módulo Gestor.
- **Módulo Gestor (Manager):** Este módulo es el encargado de determinar si la URL enviada por el módulo Analizador de Tráfico, contiene un ataque del OWASP Top Ten 2013, mantener la comunicación entre los módulos Evaluador de Riesgo, Monitor, y el Efector de Decisiones, tomar una decisión si existe un riesgo por un ataque que pertenezca al OWASP Top 10 - 2013, además debe administrar los Plug-Ins construidos en base al OWASP Top 10 - 2013.
- **Módulo Evaluador de Riesgo (Risk Assessor):** Este módulo con los datos recibidos del Gestor, se encarga de generar una calificación en una de las cinco opciones definidas por OWASP (Crítico, Alto, Medio, Bajo, Nota) y le entrega la valoración del riesgo al Gestor.
- **Módulo Monitor (Monitor):** Este módulo recibe del gestor la información de la URL, el A detectado, el valor del riesgo calculado a partir del ataque y la decisión tomada y los almacena. También evalúa el estado actual de control contra los estados anteriores y le envía el resultado de esa evaluación al Gestor.
- **Módulo Efector de Decisiones (Effector):** Este módulo es el encargado de recibir del Gestor (La categoría de riesgo, el A de OWASP y calificación de riesgos) y ejecute todas las acciones proporcionas por el módulo Gestor buscando disminuir el riesgo en la Aplicación Web, mediante dispositivos de seguridad como el Firewall, IDS, IPS, Proxy y entre otros que estén disponibles.

En este trabajo se propone dos módulos (Módulo generador y ejecutor de planes) para una mejora del sistema Control de Seguridad propuesto por Jurado et al. [7], a continuación se presentan la descripción de los módulos.

5.3 **MÓDULO GENERADOR DE PLANES**

Este módulo es el encargado de generar o crear un plan para mitigar el riesgo, cuando se ha detectado un ataque hacia el sistema. Para la generación de este plan es necesario que el módulo reciba las medidas de riesgo según OWASP, el tipo de ataque o tipo de A y los requerimientos que se necesitan para una ejecución satisfactoria. El módulo tiene conexión o relación directa con el Módulo Medidor de Riesgo, Módulo Gestor y con el Módulo Ejecutor.

Dentro de las funciones del módulo generador de planes se encuentran:

- Consultar los dispositivos de seguridad disponibles en el inventario.
- Consultar la ontología para obtener el dominio que será usado en el planificador.
- Generar el problema para el planificador.

- Cargar el planificador y hacer uso de él.
- Construir el plan.
- Enviar el plan generado al ejecutor.

Se usa un diagrama de actividades como se muestra en la Figura 7, para representar el proceso del Módulo Generador de Planes.

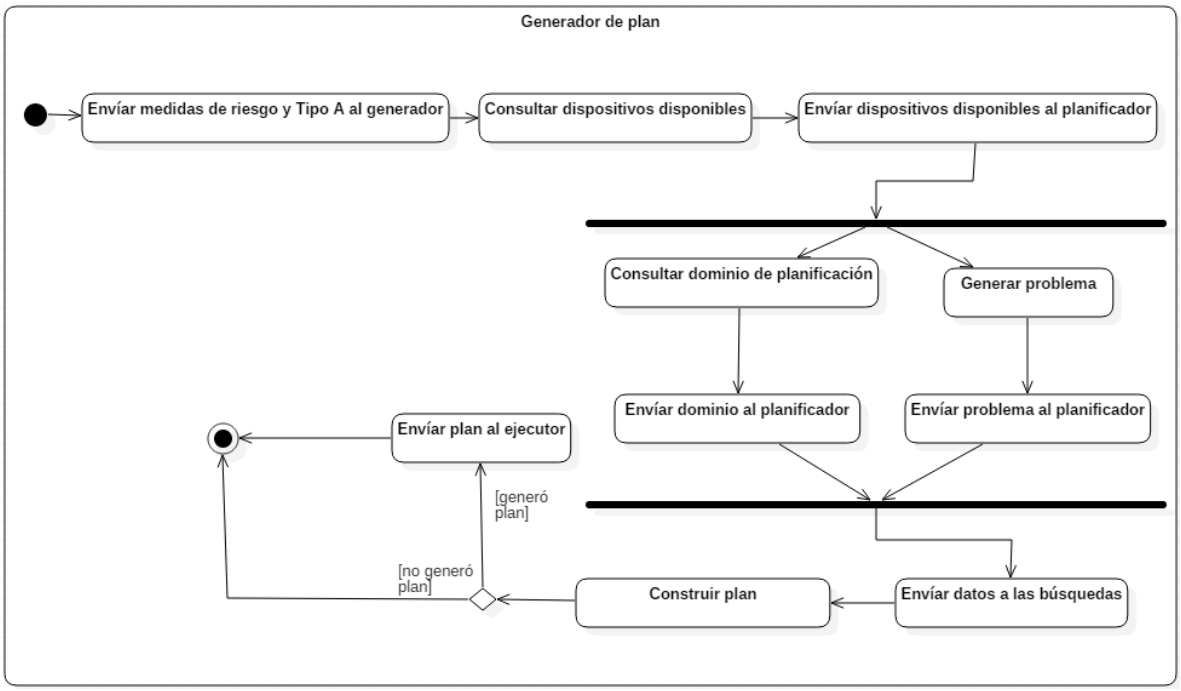

Figura 7 Diagrama de Actividades del Módulo Generador de Planes.

En la Figura 7, se describe la secuencia de pasos para el Generador de planes, el cual tiene por objetivo crear un plan con acciones para la mitigación del riesgo, teniendo en cuenta los niveles de medición de los riesgos planteados por la metodología OWASP. El proceso inicia cuando el modulo gestor envía un archivo XML con las medidas de riesgo, el tipo de ataque o tipo de A del top 10 de ataques según OWASP 2013 y los requerimientos necesarios al módulo generador de planes. El modulo generador de planes consulta al módulo de dispositivos de seguridad para conocer los dispositivos con los que se cuenta actualmente y el generador pueda establecer únicamente las acciones posibles. Una vez obtenida la lista de dispositivos se utiliza la ontología como guía para identificar los conceptos más relevantes que permita generar el dominio y el problema. El dominio hace referencia a la forma de representar el conocimiento en lenguaje PDDL usado como entrada al planificador. El problema es el estado inicial del sistema, las condiciones y el objetivo de planificación al cual se desea llegar o alcanzar. El generador de planes envía el dominio y el problema al planificador a través de dos archivos en formato PDDL. El planificador toma estos dos archivos y los ejecuta con sus diferentes búsquedas (las búsquedas que tiene el planificador son búsquedas heurísticas que tiene el planificador en su librería), las cuales se encargan de encontrar el mejor camino para lograr el objetivo, es decir entrega la acción o acciones que permiten llegar al objetivo

propuesto. El generador toma la acción o conjunto de acciones identificadas para construir el plan. Finalmente si el generador crea un plan de mitigación para el riesgo actual entonces este es enviado al ejecutor de planes, pero en caso que no se genere algún plan, entonces la solicitud será válida o permitida y podrá seguir con su ejecución normal.

En esta parte se centra en como la adquisición del conocimiento se traduce de manera que el planificador pueda hacer uso de ella. Se propuso el uso de ontología para modelar el dominio de planificación. En concreto se utilizó la ontología para representar el conocimiento de carácter estático: el vocabulario del dominio, los tipos y relaciones de las entidades o conceptos. Una vez con los conceptos y todo el conocimiento definido se procede a realizar el dominio y el problema de planificación traduciendo la ontología a lenguaje PDDL. Esta aproximación tiene dos ventajas fundamentales: aprovechamos la expresividad de las DLs para modelar dominios complejos, y las herramientas de gestión de ontologías (editores, razonadores, repositorios, entre otros) de la web semántica.

Para generar las acciones a través de la ontología para implementarla en PDDL, se traduce manualmente el lenguaje OWL (ontología en Protégé). Se pasan los conceptos identificados en el proceso de la ontología al lenguaje PDDL. Esta traducción del conocimiento es parcial, debido a que no es necesario traducir toda la ontología para llevar acabo nuestro proyecto. Además el resultado de esta traducción fue revisada por expertos, para la verificación del conocimiento, y saber cuál era la mejor manera de representarlo.

```
(define (domain xss_actions)
   (:requirements :strips :typing :equality :adl)
   (:types risk security_device - object)
   (:predicates
      (critical_risk ?x - risk)
      (high_risk ?x - risk)
      (medium_risk ?x - risk)
      (low_risk ?x - risk)
      (note_risk ?x - risk)
      (available_firewall ?x - security_device)
      (available_ids ?x - security_device)
      (available_sms ?x - security_device)
      (available_email ?x - security_device)
      (available_send_email_action)
      (available_send_sms_action)
      (available_add_signature_action)
   )

   (:action Redirect
      :parameters (?r - risk ?sd - security_device)
      :precondition (and (medium_risk ?r) (available_firewall ?sd))
      :effect (and (available_send_email_action))
   )

   (:action Block
      :parameters (?r - risk ?sd - security_device)
      :precondition (and (or (high_risk ?r) (critical_risk ?r) ) (available_firewall ?sd))
      :effect (and (available_send_sms_action))
   )

   (:action Send_email
      :parameters (?r - risk ?sd - security_device)
      :precondition (and (available_send_email_action) (medium_risk ?r)          (available_email ?sd))
      :effect (and (available_add_signature_action) (not (available_send_email_action)))
   )

   (:action Send_sms
      :parameters (?r - risk ?sd - security_device)
      :precondition (and (available_send_sms_action) (or (high_risk ?r) (critical_risk ?r)) (available_sms
?sd))
      :effect (and (available_add_signature_action) (not (available_send_sms_action)))
   )

   (:action Add_signature
      :parameters (?r - risk ?sd - security_device)
      :precondition (and (available_add_signature_action) (available_ids ?sd))
      :effect (and (low_risk ?r) (not (available_add_signature_action)))
   )
)
```

Figura 8 Dominio del planificador.

En la Figura 8, se muestra el archivo del Dominio con la traducción a PDDL con base a los conceptos identificados en la Ontología. Estos conceptos, y sus atributos fueron transformados a lenguaje PDDL de la siguiente manera, y se explicará que significa en cada una de las opciones mostradas.

PDDL tiene los siguientes componentes de planificación como lo explica Doherty [56]:

- **Objects:** Cosas del mundo que nos interesa, en este caso de las acciones sobre ataques XSS.
- **Predicates:** Propiedades de los objetos que nos interesan, puede ser verdadera o falsa.
- **Initial State:** El estado del mundo en que comenzamos.
- **Goal specification:** Cosas que queremos que sean verdad.
- **Actions/ Operators:** Formas de cambiar el estado del mundo.

El componente dominio (*domain*), para este caso xss_actions, es una cadena que identifica al dominio de planificación y además es la manera como se relaciona con el problema como se verá más adelante. Debido a que PDDL es un lenguaje muy general y además la mayoría de los planificadores solo admiten un subconjunto, los dominios pueden declarar requisitos, entre los más utilizados se encuentran: strips (El subconjunto más básico de PDDL usa solamente STRIPS), equality (El dominio utiliza el predicado =, que significa igualdad), typing, adl (El dominio utiliza parte o todo de ADL, es decir disyunciones, cuantificadores en precondiciones y objetivos, efectos cuantificadores y condicionales). En este caso para los requisitos (*requirements*) se utiliza las 4 opciones nombradas anteriormente, por el uso necesario para crear las acciones o reglas.

El componente tipos (*types),* es para declarar los tipos de parámetro y objetos. Si los tipos se van a utilizar en el dominio, lo primero que se debe hacer es declararlos en los requisitos, y luego los nombres de tipo tienen que ser declarados antes de ser utilizados. Usualmente se hace antes de la declaración de predicados. Los types que se declararon en este caso son risk y security_devices de tipo object. El componente de predicados (*predicates*), tiene como función especificar el número de argumentos que debe tener el predicado, es decir los nombres de los parámetros no importan (siempre y cuando sean distintos). En este caso se realizaron varios predicados, en los cuales se encuentran dos maneras diferentes, uno con un argumento y otro sin argumentos (trabaja como especie de booleano para poder activar las reglas que son necesarias para el plan que se esté realizando).

Se crean cada una de las acciones que son necesarias para crear los diferentes planes, para ello se define el nombre que va a tener cada acción, para el caso en particular se usa el block, redirect, send_email, send_sms y add_signature. La lista de parámetros (*parameters*), es simplemente la lista de las variables en las que opera la regla en particular, es decir sus argumentos, utilizando la sintaxis de escritura asignada anteriormente en los predicados (*predicates).* La precondición (*precondition*), es que debe satisfacerse algo antes de que se ejecute la acción, si no se especifica alguna precondición entonces la acción siempre es ejecutable. El efecto (*effect*), son los cambios en el cual la acción impone al estado actual del mundo. Estos efectos pueden ser cuantificados y condicionales. Los efectos permiten llegar al objetivo propuesto en el problema de planificación, y así lograr la generación del plan.

Se genera el problema de planificación para poder determinar el plan que mejor se acomode a las necesidades. A continuación en la Figura 9, se muestra el archivo del Problema traducido al lenguaje PDDL.

```
(define (problem xss_problem)
   (:domain xss_actions)
   (:objects risk0 - risk firewall email sms ids - security_device)
   (:init
      (medium_risk risk0)
      (available_firewall firewall)
      (available_email email)
      (available_sms sms)
      (available_ids ids)
   )
   (:goal
      (or (low_risk risk0) (note_risk risk0))
   )
)
```

Figura 9 Archivo del Problema de planificación.

Un problema de planificación es aquello que el planificador busca resolver, y se define con respecto a un dominio de planificación, el cual es el conocimiento necesario sobre cada elemento del sistema, los estados de cada uno y las consecuencias de llegar a uno de éstos estados. La definición del Problema especifica dos cosas: el estado inicial y el objetivo a alcanzar. El archivo del Problema utiliza los siguientes componentes: problem, domain, objects, init, goal.

El componente problema (*problem*), es una cadena que identifica la tarea de planificación, para este caso se le asigna el nombre de xss_problem. El componente dominio (*domain*), recibe el nombre del dominio que se ha creado anteriormente (en este caso xss_actions), para poder asociar las acciones de ese dominio en específico a las condiciones del problema y poder llegar al objetivo. El componente objetos (*objects*), si está presente, describe los objetos que existen en este problema en específico o en la situación inicial. El estado inicial (*init*), es la lista de todos los átomos o hechos que son verdaderos, para este caso son los valores del análisis del riesgo, es decir, las mediciones del OWASP que se generaron en un proceso anterior. Además los dispositivos que se tienen habilitados también entran como estado inicial. El objetivo (*goal*), de una definición de problema incluye la solución a un problema, para este caso el riesgo es de medición baja o nota, según lo especificado en OWASP. El objetivo es la meta que se desea llegar para resolver el problema propuesto y que se resuelve a partir de una acción o una serie de acciones.

Para la implementación del generador de planes se usa la librería creada por Damien Pellier [55], llamada PDDL4J, que es de código abierto bajo la licencia LGPL cuyo objetivo es facilitar el desarrollo de herramientas JAVA para la planificación automatizada basada en el lenguaje PDDL. La librería contiene un analizador de PDDL 3.1 con las clases que necesita para manipular estos conceptos y un conjunto de heurísticas clásicas ya implementadas (h_ff, h_max, entre otras).

Otro elemento fundamental en el generador de planes es el *parser,* que es una clase que convierte los datos que entrega el control en un problema de planificación (es una cadena en lenguaje PDDL), y a su vez convierte el resultado del planificador en un arreglo de acciones que pueda manejar el control para enviarlo al módulo ejecutor de planes, que es el que ejecuta el plan generado.

5.4 **MÓDULO EJECUTOR DE PLANES**

Este módulo es el encargado de ejecutar el plan que envió el Módulo Generador de Planes. Para la ejecución de este plan es necesario que el módulo reciba los scripts de cada acción que conforman el plan. El módulo tiene conexión o relación directa con el Módulo Generador de Planes y el Módulo Medidor de Riesgo.

Dentro de las funciones del módulo ejecutor de planes se encuentran:

- Establecer conexión al servidor de cada uno de los dispositivos de seguridad a través de SSH.
- Verificar conexión al servidor y estados de los dispositivos.
- Ejecutar scripts de las acciones del plan.
- Enviar notificaciones de los estados de ejecución, es decir si se efectuaron o no las operaciones anteriores.

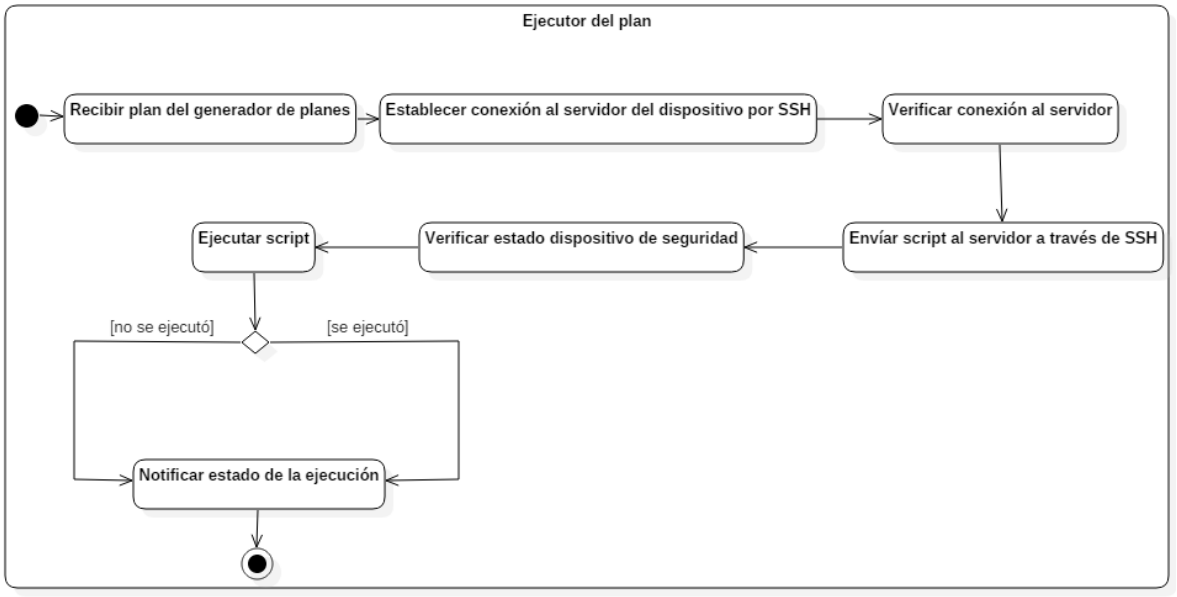

Figura 10 Diagrama de Actividades del Módulo Ejecutor de Planes.

En la Figura 10, se describe la secuencia de pasos para el Ejecutor de planes, el cual tiene como objetivo ejecutar las acciones del plan. El proceso inicia cuando el módulo generador envía el plan al módulo ejecutor a través de una lista de objetos (donde cada objeto es una acción). El ejecutor establece una conexión al servidor de dispositivos de

seguridad a través de SSH la primera vez que arranca el Control de Seguridad, para verificar los dispositivos de seguridad disponibles. Se establece una conexión por SHH con el servidor donde se encuentra el dispositivo de seguridad necesario para ejecutar una acción del plan. Esta comunicación permite establecer un seguimiento seguro entre ambos módulos. Este proceso se lleva a cabo para una de las acciones que posee el plan recibido.

Para la implementación de este módulo se hizo necesario separarlo del módulo gestor, esto para su independencia y escalabilidad. Su desarrollo resulta fácil porque él recibe el plan con sus respectivas acciones como parámetro proveniente del módulo generador de planes, este establece una conexión SSH, con el servidor de dispositivos para una comunicación segura, por ello fue necesario diseñar dos módulos extras, uno de administrador de dispositivos y otro de clientes SSH. Una vez la conexión segura este establecida, el ejecutor de planes envía cada una de las acciones del plan a los dispositivos correspondientes para que sea ejecutado, la comunicación entre ellos jamás se pierde, es decir el ejecutor siempre mantiene una comunicación con el servidor de dispositivos, que estará notificando ante cualquier evento, así llevar una buena trazabilidad.

5.4.1 **Módulo administrador de dispositivos**

Este módulo es el encargado de gestionar los dispositivos de seguridad con los que cuenta el Control de Seguridad, facilitando la adición y configuración de nuevos dispositivos a través de un archivo XML. Una de las funciones de este módulo consiste en proporcionar a los desarrolladores las acciones disponibles para cada dispositivo de seguridad y determinar si un dispositivo está correctamente configurado o activo.

También permite verificar si la conexión es correcta, es decir si están los dispositivos disponibles en el sistema, y si la conexión de comunicación segura a través de SSH se ha realizado con éxito.

5.4.2 **Módulo cliente SSH**

Este módulo es el encargado de asegurar la comunicación con los dispositivos de seguridad a través de un canal cifrado con una conexión SSH. Se utiliza tanto en el ejecutor de planes con el módulo de dispositivos para establecer la comunicación entre ellos, y poder verificar la disponibilidad de los dispositivos, necesarios para generar el plan de mitigación contra el riesgo que se esté manejando en ese momento.

En conclusión se desarrolló un generador y ejecutor de planes con la capacidad de elaborar y ejecutar acciones estructuradas de mitigación del riesgo para aplicaciones web teniendo en cuenta la metodología de medición de riesgo OWASP 2013 y los controles de seguridad de la norma ISO 27002 para ataques Cross Site Script.

CAPÍTULO 6

EVALUACIÓN

En este capítulo se desarrolló la fase de evaluación de acuerdo a la metodología para el desarrollo de los objetivos propuestos para la ejecución del proyecto el cuál se fundamenta en 3 iteraciones siguiendo las etapas definidas en el Patrón de Investigación Iterativa propuesto por Pratt [13]. En la fase final de la metodología se realizaron pruebas unitarias y de integración al prototipo, con el fin de descubrir defectos o problemas, tanto en los módulos individualmente como el sistema funcionando en su totalidad.

Los diseños de casos de prueba se desarrollan con el enfoque de caja negra, con el fin de demostrar que las funciones del prototipo son operativas, que las entradas se están correctamente implementadas y que las salidas son las esperadas o adecuadas.

De los tipos de métodos de caja negra se usó el de partición equivalente, mediante el cual se busca un conjunto de estados válidos o inválidos para las entradas y análisis de valores límites. Este método se usó para seleccionar valores límites de las entradas y salidas, con el fin de aumentar la eficiencia de la prueba.

6.1 PRUEBAS UNITARIAS

Las pruebas unitarias o casos de pruebas unitarias se realizaron con la idea de evaluar cada módulo desarrollado en el proyecto de manera independiente, esto verificando así la lógica y el funcionamiento interno de cada módulo. Las pruebas se realizaron para los tres módulos implementados: Módulo generador de planes, Módulo ejecutor de planes y el Módulo de dispositivos de seguridad.

Las pruebas unitarias son realizadas a través de una batería de pruebas diseñada, y los resultados se registran en un formato para llevar el control y hacer una verificación de la correcta ejecución del sistema, y si los resultados son los esperados.

En la Tabla 21 se observa el formato del enfoque de la prueba unitaria, donde se tiene la respectiva descripción de la prueba, el elemento que se va evaluar, la categoría de la prueba, los requerimientos, las condiciones, y sus características. En la Tabla 22 se tiene el resultado de las diferentes pruebas unitarias ejecutadas. También existe el formato de la batería de pruebas, es decir, cuáles fueron las entradas en el generador de planes para la ejecución de este módulo. Para ver el diseño completo de las pruebas unitarias ver el Anexo 4, Pruebas Unitarias.

Identificación:	GE - MGP - PU – 01 (v.01)
Descripción	
Elemento: Generador de planes (pertenece al Gestor)	Categoría de Prueba: Unitaria
Enfoque de Prueba: Caja Negra	

Requerimiento a probar: **RF-01.01.** El GESTOR debe contener los siguientes elementos:

> Un módulo de PLANEACIÓN contra el ataque, llamado GENERADOR DE PLANES.
> Infiere acciones de respuesta basadas en el dominio de planificación para suministrarlas al ejecutor de planes. Ese listado de acciones debe configurarse en el dominio de planificación.

Clases de Equivalencia Identificadas:

Condición	**Clases características**
Elaboración de Planes	{ Plan para responder ataque con calificación de riesgo CRITICO Y (HABILITADO (Firewall)) } **1.1**
	{ Plan para responder ataque con calificación de riesgo CRITICO Y (HABILITADO (Sms)) } **1.2**
	{ Plan para responder ataque con calificación de riesgo ALTO Y (HABILITADO (Firewall)) } **1.3**
	{ Plan para responder ataque con calificación de riesgo ALTO Y (HABILITADO (Sms)) } **1.4**
	{ Plan para responder ataque con calificación de riesgo MEDIO Y (HABILITADO (Firewall)) } **1.5**
	{ Plan para responder ataque con calificación de riesgo MEDIO Y (HABILITADO (Email)) }**1.6**
	{ No plan para responder ataque con calificación de riesgo BAJO o riesgo NOTA }**1.7**

Tabla 21. Formato de prueba unitaria.

Clases características	Número Prueba	Descripción de la Entrada	Ejemplar de la Entrada	Clases Cubiertas	Salida Esperada (ver convenciones en la página siguiente)	Salida Obtenida
1.1	1	ataque con calificación CRITICO Y (HABILITADO (Firewall)) }	Riesgo: CRÍTICO Firewall: Habilitado SMS: Inhabilitado IDS: Inhabilitado Email: Habilitado	**1.1**	Plan contra ataque de nivel CRITICO, con la acción de BLOQUEAR	Plan Generado: Plan contra ataque de nivel CRITICO, con la acción de BLOQUEAR
1.3	2	ataque con calificación ALTO Y (HABILITADO (Firewall)) }	Riesgo: ALTO Firewall: Habilitado SMS: Inhabilitado IDS: Inhabilitado Email: Habilitado	**1.3**	Plan contra ataque de nivel ALTO, con la acción de BLOQUEAR	Plan Generado: Plan contra ataque de nivel CRITICO, con la acción de BLOQUEAR
1.5, 1.6	3	ataque con calificación MEDIO Y (HABILITADO (Firewall)) (HABLITADO (email)) }	Riesgo: MEDIO Firewall: Habilitado SMS: Inhabilitado IDS: Inhabilitado Email: Habilitado	**1.5 y 1.6**	Plan contra ataque de nivel MEDIO, con la acción de REDIRECCIONAR y la acción de ENVIAR CORREO	Plan Generado: Plan contra ataque de nivel MEDIO, con la acción de REDIRECCIONAR y la acción de ENVIAR CORREO
1.7	4	ataque con calificación BAJO	Riesgo: BAJO Firewall: Habilitado SMS: Inhabilitado IDS: Inhabilitado Email: Habilitado	**1.7**	No debe generar ningún plan	Plan generado: NO GENERÓ PLAN
1.7	5	ataque con calificación NOTA	Riesgo: NOTA Firewall: Habilitado SMS: Inhabilitado IDS: Inhabilitado Email: Habilitado	**1.7**	No debe generar ningún plan	Plan generado: NO GENERÓ PLAN

Tabla 22. Resultados de la prueba unitaria

El resultado de la prueba unitaria permite evidenciar que el Modulo Generador de Planes y el Modulo Ejecutor de Planes cumplen con los requerimientos funcionales, ya que los resultados obtenidos fueron los esperados.

6.2 **PRUEBAS DE INTEGRACIÓN**

Las pruebas de integración se realizaron para comprobar que los módulos funcionen en conjunto. Con las Pruebas de Integración, se pudo asegurar el correcto funcionamiento del sistema, probando que todos los elementos unitarios que componen el sistema, funcionan probándolos en grupo. Se diseñaron pruebas de integración para los módulos desarrollados en este proyecto y los que ya estaban diseñados en el trabajo de Jurado et al. [7]. sólo se integraban los módulos que pasaban con éxito las pruebas unitarias. En la Tabla 23 se muestran los resultados de las pruebas de integración.

	Número Prueba	**Ejemplar de la Entrada**	**Clases Cubiertas**	**Salida Esperada**	**Salida Obtenida**
Clases características	1	A3 en URL con código embebido: /dvwa/vulnerabilities/xss_r/?name=%3Cp/onmouseover=javascript:alert(1);%20%3EKCF%3C/p%3E	1	A3 detectado	A3 detectado
	2	A3 en URL con etiqueta HTML /dvwa/vulnerabilities/xss_r/?name=%27%27%3E%3Cscript%3Ealert(1)%3C/script%3E	1	A3 detectado	A3 detectado
Clases NO características	3	Tráfico sospechoso que no es A3: /dvwa/login.php?username=1' or '1' = 1'))/*&password;=foo	2	NO ES A3	NO ES A3

Tabla 23. Resultados de la prueba de integración

El resultado de las pruebas de integración donde se unieron todos los módulos al Control de Seguridad permite comprobar que el sistema cumple con todas las funciones establecidas y requeridas para un correcto funcionamiento.

En la Tabla 24 se observa el formato del enfoque de la prueba de integración, donde se tiene la respectiva descripción de la prueba, la identificación, la categoría de la prueba, los requerimientos, las condiciones, y sus características. También existe el formato de la batería de pruebas, es decir, las entradas en el sistema para su ejecución. Para ver el diseño completo de las pruebas de integración ver el Anexo 4, Pruebas de Integración.

<table>
<tr><td>Identificación:</td><td colspan="2">Integración-GE-PLUGIN-EVALUADOR_(v.01)</td></tr>
<tr><td colspan="3">Descripción</td></tr>
<tr><td colspan="2">Elemento: Detector (Pertenece al Gestor)</td><td>Categoría de Prueba: Integración</td></tr>
<tr><td colspan="2">Enfoque de Prueba: Caja Negra</td><td></td></tr>
<tr><td colspan="3">Requerimiento a probar:

RF-02.03. El Gestor debe tener protocolo de comunicación establecido con el Repositorio de Plug-ins para recuperar la valoración correspondiente a los factores que apliquen a la Categoría de Riesgo respectiva.

RF-02.04. El Gestor debe tener protocolo de comunicación establecido con el Evaluador de Riesgos, para enviarle los parámetros indicados en el Requerimiento RF-02.03.

RF-02.05. El Gestor debe tener protocolo de comunicación establecido con el Evaluador de Riesgos, para recibir la calificación general del riesgo en forma cualitativa, de acuerdo con la escala definida por OWASP.

RF-02.05. El Gestor debe tener una comunicación establecido a través de SSH con el Módulo Ejecutor y el Módulo de Dispositivos de Seguridad, esto para poder saber disponibilidad de dispositivos a la hora de ejecutar una acción del plan generado.

Clases de Equivalencia Identificadas:

<table>
<tr><th>Condición</th><th>Clases características</th></tr>
<tr><td>Cargar factores del plug-in para enviar al Gestor y este a su vez al evaluador para que calcule la calificación del riesgo

Factores de calificación del plug-in consistentes con la calificación recibida desde el evaluador</td><td>{ Ataque tipo A3 con Calificación de Factores para riesgo CRÍTICO} 1.1

{ Ataque tipo A3 con Calificación de Factores para riesgo ALTO} 1.2

{ Ataque tipo A3 con Calificación de Factores para riesgo MEDIO} 1.3

{ Ataque tipo A3 con Calificación de Factores para riesgo BAJO} 1.4

{ Ataque tipo A3 con Calificación de Factores para riesgo NOTA} 1.5</td></tr>
</table>
</td></tr>
</table>

Tabla 24. Formato de prueba de integración.

6.3 **GUIA DE PRUEBAS OWASP**

La Open Web Application Security Project (OWASP) [59], es una organización que se enfoca en el mejoramiento de la seguridad del software generando una serie de guías de desarrollo, pruebas y de revisión de las aplicaciones web [60]; de este conjunto de guías usaremos la Guía de Pruebas de OWASP en su versión 4 [61], especialmente la sección referente a ataques XSS para verificar y evaluar la efectividad del planificador y el ejecutor.

6.3.1 **Ambiente de pruebas**

El ambiente de pruebas consiste en una instancia de la aplicación web DVWA (Damn Vulnerable Web Application), un firewall y un servidor de correos como dispositivos de seguridad configurados en el administrador de dispositivos, como se vio en la Figura 8.

6.3.2 **Objetivos de la prueba**

El objetivo general de la prueba es determinar el nivel de efectividad del planificador y el ejecutor de planes desarrollados.

6.3.3 **Alcance de la prueba**

La prueba cubre únicamente a los numerales 4.8.1 Pruebas para Cross Site Scripting Reflejado propuestas por OWASP [61] y 4.8.2 Pruebas para Cross Site Scripting Almacenado del apartado Pruebas de validación de entradas correspondiente a la Guía de Pruebas de OWASP v4, todo esto únicamente sobre la aplicación DVWA. La dinámica de las pruebas será caja gris debido a que se evaluará solo lo concerniente a entrada de datos desde la aplicación por lo cual se supone que se tiene acceso al vector de vulnerable a ataques XSS. La prueba se limitará a 500 ataques de tipo XSS de diversos niveles de dificultad y de acuerdo a los planes generados por el planificador y la ejecución de éstos en los distintos dispositivos de seguridad.

6.3.4 **Pruebas de validación de entradas**

La prueba de validación de entradas consiste en manipular los parámetros de entrada de datos de la aplicación, bien sea a través de formularios, parámetros a través de la URL o por medio del payload de la solicitud HTTP.

En la Tabla 25 se describe el proceso de pruebas propuesto por OWASP [61] para ataques XSS.

ID Vulnerabilidad	Nombre	Severidad	Cantidad
INPUTVAL-01	Prueba de XSS Almacenado y Reflejado	Alta	500

Descripción: La prueba se realizó a través de una herramienta de pruebas usando 6 datasets distribuidos de la siguiente forma:
D1, D2, D3, D4: cada dataset con 100 solicitudes con código malicioso XSS que fueron descargados del sitio web www.xxsed.com que recopila una base de datos de ataques XSS realizados a diferentes sitios web, estos ataques se adquirieron haciendo uso de un script desarrollado en Python que capturó los ataques y los ajustó para ser ejecutados en la aplicación DVWA. Cada dataset se ejecutará con una dirección IP diferente para simular un ataque masivo.
D5, D6: con 100 solicitudes no maliciosas cada uno

Clasificación de ataque: A3 (XSS)

Herramientas: xsser

Resultado esperado: De la presente prueba se espera que al ejecutar cada dataset se genere un único plan el cual debe ser ejecutado en los dispositivos de seguridad existentes (firewall y servidor de correo) con lo cual se debe detener el ataque quedando los demás registros del dataset sin alcanzar su objetivo de explotación.

Tabla 25. Descripción de la prueba

6.3.5 **Resultados**

En la Tabla 26, se puede ver el resultado de las pruebas realizadas según la Guía de Pruebas de OWASP, en la cual se presenta el número de planes generados desde el módulo planificador, los planes ejecutados en los dispositivos de seguridad y los ataques detenidos y exitosos.

DATASET	PLANES GENERADOS	PLANES EJECUTADOS	ATAQUES DETENIDOS	ATAQUES EXITOSOS
D1	1	1	73	26
D2	2	2	75	23
D3	1	1	78	21
D4	1	1	73	27
D5	0	0	0	0
D6	0	0	0	0

Tabla 26. Resultados de la pruena basada en la Guía de Pruebas OWASP v4

Para el dataset D2 se ejecutaron 2 planes simultáneamente, lo cual se debe al módulo capturador de tráfico del Sistema Control de Seguridad que no retiene las solicitudes sino que crea una copia para poder ser analizada y generar el plan, con lo cual debido al tiempo tomado para generar y ejecutar el plan el capturador puede encolar más solicitudes para análisis.

Así mismo, para los datasets D1, D2, D3 y D4 se lograron llevar a cabo un promedio de 24,5% de la totalidad de los ataques realizados, esto se debe en primera medida al módulo capturador del tráfico que no retiene las solicitudes; por otra parte el módulo de detección de ataques XSS no es capaz de detectar el 100% de los ataques con lo que se da un alto número de falsos negativos que corresponde a ataques codificados precisamente para evadir los distintos dispositivos de seguridad.

Por otro lado, con los datasets D5 y D6 correspondientes a solicitudes no maliciosas el control de seguridad no encontró nada sospechoso dejando pasar la totalidad de solicitudes.

En el 2010, Perdomo et al. [62], proponen el cálculo de la efectividad como la sumatoria del valor real de los indicadores propuestos, dividido por el valor máximo de los indicadores como se muestra a continuación.

$$Efectividad = \sum_{i=1}^{n} \frac{Valor\ real\ del\ indicado}{Valor\ máximo\ de\ los\ indicadores}$$

Tomando en cuenta que no existe precedente de un sistema capaz de crear planes en base a conocimiento representado como dominio de planificación de acuerdo a una medición de riesgo que la batería de pruebas es única, se propone como indicador de efectividad el número de ataques detenidos por el sistema. De acuerdo a esto, la efectividad del planificador y el ejecutor integrados al Sistema Control de Seguridad es de 74,75.

Con la ejecución de las pruebas unitarias y de integración se pudo comprobar la funcionalidad de la implementación e implantación del módulo generador y ejecutor de planes del sistema. Además se pudo evidenciar la efectividad del sistema, ya que las salidas obtenidas fueron las mismas que las salidas esperadas para el dataset de pruebas. En cuanto a eficiencia, las pruebas del planificador arrojaron que el tiempo que toma la búsqueda de un plan depende del tamaño del dominio, en nuestro caso al ser un dominio pequeño en el cual solo se especifican acciones para el tipo de ataque XSS obtuvimos un tiempo promedio de 0.1 segundos.

CAPÍTULO 7

CONCLUSIONES Y TRABAJO FUTURO

7.1 CONCLUSIONES

Con el desarrollo del módulo generador de planes de mitigación de riesgo según OWASP 2013 y los controles de seguridad sugeridos por la norma ISO/IEC 27002 se logró obtener las acciones adecuadas para reducir el nivel de riesgo del sistema ante ataques de tipo XSS en todos sus niveles.

El desarrollo de la ontología facilitó la interoperabilidad semántica entre los componentes del Control de Seguridad al proveer los conceptos necesarios para la implementación de estrategias para ataques diferentes a XSS.

El uso de PDDL como el lenguaje para la especificación del dominio de planificación permitió tener una mayor expresividad en la definición de las acciones del plan, así como la facilidad de agregar nuevos estados y acciones al dominio.

La implementación de un administrador de dispositivos para la gestión de la configuración permitió una comunicación segura, a través de un medio cifrado para cada dispositivo de seguridad, y además tener control sobre el estado de los dispositivos de seguridad y el resultado de la ejecución de los planes de tratamiento de riesgo para conocer las acciones disponibles para la planificación.

Al usar la Guía de Pruebas OWASP v4 para evaluar la efectividad del generador de planes y del ejecutor se determinó que la implementación propuesta en conjunto con los demás módulos del Sistema Control de Seguridad tiene un nivel de efectividad del 75,5; el cual servirá como punto de comparación para evaluaciones futuras.

El diseño del generador y ejecutor de planes permitió escalar a nivel de dispositivos de seguridad, ataques, dominios, lenguajes de definición de dominio y sistema de representación del conocimiento permitiendo que otras investigaciones puedan incorporar nuevas o variaciones en las estrategias de planificación.

Uno de los problemas encontrados durante la traducción de los conceptos definidos en la ontología al lenguaje PDDL fue la pérdida de algunos conceptos debido a la naturaleza de los dominios de planificación, los cuales corresponden a conocimiento de mundo cerrado, mientras que la ontología es una representación de conocimiento de mundo abierto.

La medición del nivel de efectividad del sistema desarrollado se vio afectado por la interdependencia del planificador con los módulos capturador y detector de tráfico del Sistema Control de Seguridad, debido a que estos últimos no capturaron y detectaron la totalidad de ataques de las pruebas realizadas.

7.2 **RECOMENDACIONES Y TRABAJO FUTURO**

En el presente proyecto de investigación, se propuso usar una traducción al lenguaje PDDL desde los conceptos y relaciones identificados en la ontología para la planificación. Es posible que los resultados mejoren toda la expresividad de estos formalismos para representar tanto el vocabulario del dominio como el estado del sistema, si se utiliza la lógica descriptiva para la planificación.

El desarrollo del proyecto se enfoca en el tipo de ataque XSS del top 10 de OWASP 2013, se propone ampliar el dominio de conocimiento en la ontología y en la planificación con otros ataques y sus Plug-ins correspondientes.

Aumentar los dispositivos de seguridad implementando las interfaces adecuadas para la comunicación con el Control de Seguridad y esto permita una mayor gama de acciones para la generación de planes.

Se sugiere probar en un entorno real el Control de Seguridad con las mejoras propuestas en este trabajo.

BIBLIOGRAFÍA

[1] H. Hernández, "Plan de implementación de la ISO/IEC 27001:2005", Universitat Oberta de Catalunya, 2014.

[2] ISO2700, "Sistema de Gestión de la Seguridad de la Información (SGSI)", 2007. [En línea]. Obtenido el 04-Feb-2016 de: http://goo.gl/RmnMSl.

[3] J. Enjuto, "Definiendo el alcance del SGSI", Seguridad y Gestión, 2006.

[4] TechTarget. Gestión de riesgos de seguridad de la información: Comprensión de los componentes [en línea]. < https://goo.gl/OFb2hE > [citado en 18 de enero de 2017].

[5] UNIVERSIDAD NACIONAL DE LUJÁN. Riesgo vs. Seguridad de la Información [en línea]. < https://goo.gl/aoAJJu > [citado en 18 de enero de 2017].

[6] A. López Neira and J. Ruiz Spohr, "Gestión de seguridad de la información", ISO2700, 2012. [En línea]. Obtenido el 04-Feb-2016 de http://goo.gl/3qkIwj.

[7] J. Jurado Narváez, D. Penagos Mosquera, S. Donado, E. Martínez Flor, C. Ardila Albarracín and S. Garcés, "Control Inteligente para el Servicio Crítico de un Sistema de Información en Línea Enmarcado en un Dominio de la ISO/IEC 27002", Pregrado, Universidad del Cauca, 2015.

[8] J. Williams, "Cross-site Scripting (XSS)", OWASP, 2006. [En línea]. Obtenido el 04-Feb-2016 de https://goo.gl/D55Tu2.

[9] J. Grossman, R. Hansen, P. Petkov, A. Rager and S. Forgie, XSS Attacks: Cross site scripting Exploits and Defense. Burlington, Mass: Syngress, 2007, p.11.

[10] L. Suarez Sierra and C. Amaya Tarazona, "Lección 24: Plan de Tratamiento de riesgos", Universidad Nacional Abierta y a Distancia, 2013. [En línea]. Obtenido el 04-Feb-2016 de http://goo.gl/FOYIFJ.

[11] HOSTALIA. Qué es y cómo funciona un ataque Cross-Site Scripting: XSS ATTACKS [en línea]. < https://goo.gl/0F6csq > [citado en 18 de enero de 2017].

[12] IMPERVA. 2015 Web Application Attack Report (WAAR) [en línea]. <https://www.imperva.com/docs/HII_Web_Application_Attack_Report_Ed6.pdf> [citado en 18 de enero de 2017].

[13] Kevin S. Pratt, Hr. Bright, "Design Patterns for Research Methods: Iterative Field Research", Texas A&M University, 2009.

[14] T. Velasquez, M. Puentes and J. Guzmán, "Ontologies: a technical of knowledge representation". Avances En Sistemas E Informatica, 8(2), 2011–2016.

[15] C. Pardo, F. Pino, F. García, M. Piattini and M. Baldassarre, An ontology for the harmonization of multiple standards and models. Computer Standards and Interfaces, 34(1), 48–59. 2012.

[16] W.N. Borst, Construction of Engineering Ontologies. International Journal of Human-Computer Studies (Vol. 46). 1997.

[17] M. Gawich, A. Badr, A. Hegazy and H. Ismail, "A Methodology for Ontology Building". International Journal of Computer Applications, 56(2), 975–8887. 2012.

[18] J. Fonou Dombeu and M. Huisman, "Semantic-Driven e-Government: Application of Uschold and King Ontology Building Methodology for Semantic Ontology Models Development". International Journal of Web & Semantic Technology, 2(4), 111–20. 2011.

[19] M. Alvarez and R. Martínez, "Integración de Técnicas avanzadas de Web Semántica para Diseño y Visualización de Estructuras de Conocimiento". 2010.

Trabajo de grado (Ingeniero en Informática). Universidad de Murcia. Facultad de Informática. Ingeniería en Informática.
[20] O. Corcho, M. Fernández-López and A. Gómez-Pérez, "Methodologies, tools and languages for building ontologies. Where is their meeting point? Data and Knowledge Engineering", 46(1), 41–64. 2003.
[21] K. Solic, H. Ocevcic and M. Golub, "The information systems' security level assessment model based on an ontology and evidential reasoning approach". Computers & Security, 55, 100–112. 2015.
[22] A. Razzaq, Z. Anwar, H. Ahmad, K. Latif and F. Munir, "Ontology for attack detection: An intelligent approach to web application security". Computers & Security, 45, 124–146. 2014.
[23] P. Salini and J. Shenbagam, "Prediction and Classification of Web Application Attacks using Vulnerability Ontology", 116(21), 42–47. 2015.
[24] J. Hoffmann, The metric-FF planning system: Translating "ignoring delete lists" to numeric state variables. Journal of Artificial Intelligence Research, 20, 291–341. 2003.
[25] V. Adriá, E. Onainda de la Rivaherrera and A. Garrido, "Diseño e implementación de un planificador para entornos de e-learning". Tesis de maestría en Inteligencia Artificial (Ingeniero en Informática). Universidad Politecnica de Valencia. Facultad de Informática. Departamento de Sistemas Informáticos y Computación.
[26] J. Guzmán, U. D. Francisco José de Caldas and U. N. Colombia, "Aplicación del planificador strips para la construcción de un programa de control", 1–5.
[27] F. Raimondi, C. Pecheur and G. Brat, "PDVer , a Tool to Verify PDDL Planning Domains. Artificial Intelligence", 76–82. 2009.
[28] F. Wotawa, and F. Bozic, "Plan It! Automated Security Testing Based on Planning", 48–62. 2014.
[29] J. Bozic, and Wotawa, "PURITY: A Planning-based secURITY Testing Tool". Proceedings - 2015 IEEE International Conference on Software Quality, Reliability and Security-Companion, QRS-C 2015, 46–55. 2015.
[30] T. Plch, M. Chomut, C. Brom and R. Barták, "Inspect , Edit and Debug PDDL Documents : Simply and Efficiently with PDDL Studio", 2–5. 2011.
[31] E. Aktolga, U. Schmid and H. Gust. 2004, A Java planner for Blocksworld problems, Trabajo de grado. Universidad de Osnabrueck.
[32] L. Torres, Problemas. Universidad Nacional de Colombia, 2010
[33] N. Alechina, Planning and Search, Classical Planning. Universidad de Nottingham, 2014, pág. 1–34.
[34] O. García Pérez, L. Castillo and J. Fernández, "Planificación en dominios temporales usando técnicas HTN". Tesis doctoral. Universidad del Granada. Departamento de Ciencias de la Computación e Inteligencia Artificial.
[35] G. Fakhtehyavari, Y. Vaziri. 2012, Master of Science Thesis in the Programme Intelligent System Design. Automated Planners and Planning analysis with Durative Actions. Clambers University of technology. Department of Applied Information Technology.
[36] N. Duque, D. Ovalle and J. Jiménez, "Modelo Adaptativo para Cursos Virtuales basado en Técnicas de Planificación Inteligente". Avances En Sistemas E Informatica, 4(1), 39–46. 2007.
[37] S. Russell and P. Norving, Artificial Intelligence: A Modern Approach. Prentice Hall, 2002, Chapter 11

[38] M. Ghallab, A. Howe, C. Knoblock, D. McDermott, A. Ram, M. Veloso, D. Weld and D. Wilkins, "PDDL - The Planning Domain Definition Language". AIPS-98 Planning Competition Committee. Yale Center for Computational Vision and Control. Tech Report CVC TR-98-003/DCS TR-1165, Oct. 1998.
[39] D. Sacerdoti, “Planning in a hierarchy of abstraction spaces”, Artificial Intelligence, tomo 5, págs. 115–135, 1974.
[40] A. Tate, “Generating project networks”, en IJCAI 1977, págs. 888–893, 1977.
[41] D. Sacerdoti, “The nonlinear nature of plans”, en IJCAI 1975, págs. 206–214, 1975.
[42] M. Veloso, J. Carbonell, A. Pérez, D. Borrajo, E. Fink and J. Blythe, “Integrating planning and learning: The PRODIGY architecture”, Journal of Experimental and Theoretical Artificial intelligence, tomo 1, No-7, 1995.
[43] D. Long and M. Fox, "PDDL2.1: An Extension to PDDL for Expressing Temporal Planning Domains", 2003, Journal of Artificial Intelligence Research, 20, Pg: 61, 124.
[44] A. Gerevini and D. Long, "Plan Constraints and Preferences in PDDL3", 2006, ICAPS Workshop on Soft Constraints and Preferences in Planning.
[45] K. Parker, J. Chao, T. Ottaway and J. Chang, "A formal language selection process for introductory programming courses". Journal of Information Technology Education, 5, 133–151. 2006.
[46] C. Blanco, J. Lasheras, R. Valencia-garcía, E. Fernández-medina, A. Toval and M. Piattini, "Ontologías de Seguridad : Revisión sistemática y comparativa", Informe Técnico UCLM-TSI-003, Jul. 2008.
[47] M. Ghallab, D. Nau and P. Traverso, "Automated planning and acting". Cambridge University Press. 2016.
[48] C. Pardo, F. García, F. Pino and M. Piattini, "A Framework to Support the Harmonization between Multiple Models and Standards". 2012. Tesis Doctoral. University of Castilla-La Mancha. Institute of Information Technologies and Systems, Spain.
[49] A. Ohgren, "Towards an Ontology Development Methodology for Small and Medium-sized Enterprises", 2009. Tesis. Jönköping University. School of Engineering. Department of Computer and Electrical Engineering.
[50] R. Mizoguchi, "Part 2: Ontology development, tools and languages". Organization, 22(1), 1–27. 2004.
[51] A. Vizcaíno, J. Soto, F. García, F. Ruiz and M. Piattini, "Aplicando gestion del conocimiento en el proceso de mantenimiento del software". Inteligencia Artificial, 10(31), 91–98. 2006.
[52] M. Cardenas, M. Marciszack, J. Castillo and J. Vázquez, "Construcción de un modelo conceptual para gramáticas formales y máquinas abstractas con ontologías usando Protégé". Universidad Técnologica Nacional.
[53] J. Guzmán, M. López and I. Durley, "Metodologías y métodos para la construcción de ontologías", (50), 133–140. 2012.
[54] A. Sanchéz Ruiz-Granados, P. González Calero, B. Díaz Agudo, "Una Aproximación Ontológica al Modelado de Conocimiento en los Dominios de Planificación". 2010. Tesis Doctoral. Universidad Complutense de Madrid, Facultad de Informática, Departamento de Ingeniería de Software e Inteligencia Artificial.
[55] D. Pellier. Pellierd/pddl4j library [en línea]. <https://github.com/pellierd/pddl4j> [citado en 31 de marzo de 2017].
[56] P. Doherty, Writing Planning Domains and Problems in PDDL [en línea]. < https://goo.gl/7w0RpU > [citado en 8 de marzo de 2017].

[57] M. Ghallab, A. Howe, C. Knoblock, D. McDermott, A. Ram, M. Veloso, D. Weld and D. Wilkins, "PDDL - The Planning Domain Definition Language". AIPS-98 Planning Competition Committee. Yale Center for Computational Vision and Control. Tech Report CVC TR-98-003/DCS TR-1165, Oct. 1998.

[58] D. Pellier, “PDDL4J”, [en linea] http://sourceforge.net/projects/pdd4j/, [Citada en 31 de marzo de 2017]

[59] Open Web Application Security Project OWASP, Development Guide. [en línea]. <https://goo.gl/0Qm2b> [Citado el 31 de marzo de 2017]

[60] Open Web Application Security Project OWASP, Proactive Controls. [en línea]. <https://goo.gl/8GUPQc> [Citado el 31 de marzo de 2017]

[61] Open Web Application Security Project OWASP, Testing Guide. [en línea]. <https://goo.gl/YEvaD> [Citada el 31 de marzo de 2017]

[62] C. Perdomo, N. Arsenio, R. Acosta, O. Nilda. Sistema para el cálculo de la efectividad y la eficiencia del proceso de integración de la gestión de la ciencia, la innovación tecnológica y el medio ambiente a escala territorial. Ciencia y sociedad, Instituto Tecnológico de Santo Domingo, República Dominicana.