UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

FACULTAD 6

Trabajo de diploma para optar por el título de Ingeniero en Ciencias Informáticas.

Propuesta de un procedimiento para un desarrollo seguro de los productos del Departamento Geoinformática.

Autora: Yadira Hernández Leyva.

Tutora: Ing. Lisdaynet Gómez Espinosa.

La Habana, junio del 2011

"Año 53 de la Revolución"

DEDICATORIA

A mis padres por su confianza y eterno amor.

A mis hermanos por todo su cariño.

A mis abuelos que siempre estarán en mi corazón.

A mi novio por llenar de alegría mi vida.

AGRADECIMIENTOS

A mi tutora por sus consejos, dedicación y ayuda en la realización de este trabajo.

A mi familia por siempre confiar en mí y brindarme su apoyo incondicional, especialmente a:

Mi mamá por ser mi orgullo y mi ejemplo a seguir.

Mi papá por apoyarme siempre y darme fuerzas para salir adelante.

Mi hermana por compartir mis mayores secretos y ser tan especial conmigo.

Mi hermano por esperar por mí en cada pase y darme todo el cariño del mundo.

Mis abuelos por su amor y dedicación. Por ayudar a mis padres a formar esta persona que soy hoy.

Mis tíos y primos por siempre estar ahí cuando los necesité.

A Francisca por sus consejos y apoyo incondicional durante mi carrera.

A mi novio por estar siempre a mi lado cuando más lo he necesitado.

A la familia de mi novio por apoyarme y brindarme su ayuda incondicional.

A mis amigos que siempre estuvieron presentes de una forma u otra.

A todos los profesores que me han ayudado durante estos cinco años, principalmente al profesor Yunier A. Pimienta por su ayuda incondicional en la realización de la tesis.

Al tribunal por toda su atención y comentarios en cada corte, formando en mí una persona más preparada.

A la Revolución y al Comandante en Jefe Fidel Castro Ruz por crear esta magnífica universidad.

A todas las personas que de una forma u otra ayudaron a que este sueño se hiciera realidad:

Gracias.

DATOS DE CONTACTO

Tutor: Ing. Lisdaynet Gómez Espinosa (lespinosa@uci.cu)

Graduado de Ingeniero en Ciencias Informáticas (UCI), egresado de la UCI en el 2008 con categoría de Instructor. Ha impartido la asignatura Sistemas Operativos y Seguridad Informática, pertenece al Departamento de Sistemas Digitales y trabaja en el proyecto Control de Flotas del Departamento Geoinformática perteneciente al Centro GEySED. Con una experiencia productiva desde el 2006.

DECLARACIÓN DE AUTORÍA

Declaro que soy la única autora del presente trabajo y autorizo a la Universidad de las Ciencias Informáticas (UCI) los derechos patrimoniales del mismo, con carácter exclusivo.

Para que así conste firmo la presente a los ____ días del mes de _____ del año 2011.

_____	_____
Yadira Hernández Leyva	Ing. Lisdaynet Gómez Espinosa
Autora	Tutor

RESUMEN

Los productos desarrollados en el Departamento Geoinformática necesitan que desde fases tempranas a su desarrollo sean introducidos criterios de seguridad para minimizar muchos de los daños que pueden causar las vulnerabilidades, las amenazas y los riesgos que atentan contra su seguridad. Para lograrlo se realizó la propuesta de un procedimiento que será aplicado desde la primera fase del desarrollo. Este estudio se sustenta en las soluciones existentes como el Proyecto Abierto de Seguridad de Aplicaciones Web (OWASP), el Manual de Metodología Abierta de Pruebas de Seguridad (OSSTMM), el Ciclo de Vida de Desarrollo del Software (SDLC) y el proyecto de seguridad informática de la facultad 2 (LABSI). Para el procedimiento se efectuó un análisis del documento de Arquitectura de Software de los proyectos del Departamento Geoinformática (Sistema de Gestión de Datos Geológicos (SGDG) y Plataforma Soberana para el Desarrollo de Sistemas de Información Geográfica (GeneSIG)), evaluando los temas de seguridad que aportan los framework, lenguajes de programación, gestores de bases de datos, etc., utilizados por ambos durante el desarrollo de sus productos. Además, se registraron las vulnerabilidades, las amenazas y los riesgos, así como los principales incidentes a aplicaciones web, sirviendo todo esto de base para la descripción de las fases del procedimiento a desarrollar.

PALABRAS CLAVES

AMENAZAS, INCIDENTES, PROCEDIMIENTO, RIESGOS, SEGURIDAD, VULNERABILIDADES.

ABSTRACT

The products developed in the Geoinformatics Department need since the early phases of its development, to have security criteria to minimize much of the damage that vulnerabilities can cause, threats, and risks that threaten its security. In order to achieve that, it was made the proposal of a procedure that will be applied since the first phase of the development. This study is based on existing solutions such as Open Web Applications Security Project (OWASP), Open Methodology Manual for Security Testing (OMMST), Software Development Life Cycle (SDLC) and the informatics security project from school 2 (LABSI). For the procedure it was made an analysis of the Software Architecture Document from projects in the Geoinformatics Department (Geological Data Management System) and Sovereign Platform for the Development of Geographic Information System (GeneSIG), assessing the security issues that frameworks provide, programming languages, database management systems, etc., used by both during the development of their products. In addition, vulnerabilities, threats, and risks were registered as well as major incidents to web applications, being this all useful as basis for the description of the phases of the procedure to be developed.

KEY WORDS

THREATS, INCIDENTS, PROCEDURE, RISKS, SECURITY, VULNERABILITIES.

ÍNDICE

INTRODUCCIÓN

En el mundo actual por las exigencias de la industria del software, ha sido de suma importancia realizar programas que cuenten con buena calidad y sobre todo que sean seguros.[1] Esto es de vital importancia, ya que a medida que han evolucionado las Tecnologías de la Información y las Comunicaciones (TIC) también han evolucionado los ataques piratas a los medios informáticos. El siglo XXI se está caracterizando por ser un entorno globalizado y altamente competitivo. En este sentido, la seguridad de los productos informáticos que se desarrollan surge como una nueva disciplina, que es un campo necesario y de vital importancia para afrontar con éxito los cambios en el presente milenio.

Hoy día el país se ha visto envuelto en los sucesivos avances tecnológicos que se han desarrollado a lo largo de la historia. Al calor de la batalla de ideas surge la Universidad de las Ciencias Informáticas (UCI), que vincula la docencia e investigación con actividades productivas. En la UCI se desarrollan diversos e importantes software que contribuyen de forma directa al enriquecimiento y desarrollo de las tecnologías, así como a la informatización de las diferentes esferas del país.

En el Departamento Geoinformática perteneciente a la Facultad 6, se desarrollan Sistemas de Información Geográfica (SIG) bajo tecnologías *OpenSource*[2], cumpliendo con los estándares *OpenGIS*[3]. En el mismo se implementan aplicaciones informáticas para la toma de decisiones empresariales basado en SIG, soluciones sobre bases de datos espaciales, dispositivos móviles, así como servicios de mapas y administración de metadatos geográficos. Además, agrupa proyectos dedicados a la conceptualización y desarrollo de soluciones informáticas para la geología y la minería, con temáticas de base de datos referativas, portales de entidades geológicas y tratamiento de información asociada al patrimonio geológico.

A partir de una investigación realizada se verificó que el plan de seguridad que llevan a cabo los proyectos de este departamento, en su mayoría, son los desarrollados por estudiantes de cuarto año

[1] Libre de vulnerabilidades.

[2] Software distribuido y desarrollado libremente.

[3] Cuyo fin es la definición de estándares abiertos dentro de los Sistemas de Información Geográfica (SIG).

como proyecto de curso. El mismo es para proteger los activos (datos, software y hardware), pues no cuentan con un grupo de medidas o documento formal que respalde la seguridad de los productos durante su desarrollo. Se deja todo a la improvisación o experiencia personal de los desarrolladores, lo que trae consigo que los productos que se realicen no tengan la mayor seguridad posible. Provocando que las debilidades y amenazas incidan con mayor fuerza sobre estos, afectando de manera directa o indirecta la producción del departamento.

Atendiendo a la situación problemática antes abordada se plantea como **problema a resolver,** ¿Cómo mejorar el proceso de desarrollo de software en el Departamento Geoinformática para que los productos obtenidos cumplan los objetivos de la seguridad informática?

Quedando definido como **objeto de estudio,** el proceso de gestión de la seguridad informática en el desarrollo de aplicaciones, enmarcado en el **campo de acción**, procedimientos para mejorar la seguridad informática en el desarrollo de aplicaciones del Departamento Geoinformática.

En tanto, se propone como **objetivo general,** plantear un procedimiento que especifique cómo desarrollar productos en el Departamento Geoinformática, obedeciendo los objetivos de la seguridad informática.

Quedando definido como **idea a defender** que, con un procedimiento que especifique cómo desarrollar productos en el Departamento Geoinformática, se contribuirá a que los mismos cumplan los objetivos de la seguridad informática.

Para darle cumplimiento al objetivo general se han propuesto las siguientes **Tareas de investigación:**

1. Elaborar el Diseño Teórico y Metodológico de la Investigación.
2. Evaluar el estado del arte de las soluciones informáticas que garantizan la seguridad de las aplicaciones.
3. Caracterizar la seguridad informática en el desarrollo del software.
4. Evaluar los riesgos informáticos de los proyectos del Departamento Geoinformática.

5. Enumerar las vulnerabilidades que presentan las herramientas y metodologías definidas en la arquitectura de los proyectos del Departamento Geoinformática.
6. Plantear un procedimiento para desarrollar software seguro en el Departamento Geoinformática.
7. Validar el procedimiento planteado, utilizando técnicas de recopilación de información.

Para una correcta comprensión y realización de la investigación se hace uso de diferentes métodos científicos, los cuales señalan el procedimiento necesario para llevar a cabo la investigación:

Métodos teóricos:

- Análisis Histórico-Lógico: Permitió realizar un análisis de las principales soluciones existentes utilizadas para el desarrollo seguro de las aplicaciones.
- Analítico-Sintético: Permitió el estudio de los conceptos relacionados con este tema, así como los documentos existentes de los proyectos inscritos en el Departamento Geoinformática, los cuales servirán de base en la investigación.

Métodos empíricos:

- Entrevista: Permitió realizar entrevistas a los líderes del proyecto de seguridad informática de la facultad 2 (Anexo1), obteniendo información acerca de lo que hacen en el mismo. Además, a los líderes y desarrolladores de los proyectos del Departamento Geoinformática (Anexo2).
- Encuesta: Permitió la validación de los resultados de la investigación realizada, mediante el criterio de expertos (Anexo 3).

El desarrollo de esta investigación se encuentra estructurado en 3 capítulos:

Capítulo 1: Fundamentación Teórica: Se hace referencia a los conceptos fundamentales que sustentan la base de los capítulos restantes. Además, se realiza una descripción general del objeto de estudio, así como un análisis de algunas de las soluciones existentes para el desarrollo de software seguro en el mundo y en la universidad.

Capítulo 2: Análisis de los riesgos: Se realiza un estudio profundo de la documentación que tributa a determinar los riesgos a partir del análisis e identificación de las amenazas y vulnerabilidades de los productos existentes en el Departamento Geoinformática.

Capítulo 3: Propuesta de solución: Se describen los pasos que se establecen por cada fase del procedimiento.

Capítulo 1: Fundamentación Teórica

1.1- Introducción

En el presente capítulo se abordarán los conceptos asociados al dominio del problema. Además, se realizará una descripción general del objeto de estudio, así como el análisis de algunas de las soluciones existentes que permiten el desarrollo de software seguro.

1.2- Conceptos asociados al dominio del problema

En el avance de la investigación se determinan un conjunto de conceptos importantes para el completo dominio y comprensión del tema, entre estos se encuentra el de seguridad informática. **¿Qué se entiende por *seguridad informática*?** Es la capacidad de las redes o de los sistemas de información para resistir, con un determinado nivel de confianza, los accidentes o acciones malintencionadas que comprometan la disponibilidad, integridad y confidencialidad de los datos almacenados o transmitidos y de los servicios que dichas redes y sistemas ofrecen o hacen accesibles. **¿Qué se define como *sistema de información*?** Cualquier sistema o producto destinado a almacenar, procesar o transmitir información (López Crespo 2006).

Al trabajar la seguridad informática con tres objetivos confidencialidad, integridad y disponibilidad se hace necesario conocer sus definiciones. **¿A qué se refiere la *integridad de la información*?** Se refiere a la característica que previene contra la modificación o destrucción no autorizada de activos del dominio. **¿Qué asegura la *disponibilidad de la información*?** Asegura de que los usuarios autorizados tienen acceso cuando lo requieran a la información y sus activos asociados. **¿A qué se refiere la *confidencialidad de la información*?** Se refiere al aseguramiento de que la información es accesible sólo para aquellos autorizados a tener acceso (López Crespo 2006).

Otro de los conceptos asociados a este tema es el de ***riesgo*,** siendo este la probabilidad de que una amenaza se materialice, utilizando vulnerabilidades existentes de un activo o un grupo de activos, generándole pérdidas o daños (Sena y Tenzer 2004). Se entiende como ***vulnerabilidad*** a la potencialidad o posibilidad de ocurrencia de la materialización de una amenaza sobre dicho activo. **¿Qué son las *amenazas*?** Son eventos que pueden desencadenar un incidente en la organización, produciendo daños materiales o pérdidas inmateriales en sus activos (López Crespo 2006). **¿Qué se entiende por *incidente de seguridad de la información*?** Es una o varias acciones inesperadas y

no deseadas que tengan una gran probabilidad de atentar contra la seguridad de la información (Comerciales 2007).

1.3- Descripción general del objeto de estudio

El proceso de gestión de la seguridad informática en el desarrollo de aplicaciones es indispensable en una organización. En este sentido se debe gestionar eficientemente la accesibilidad a la información, respaldando la confidencialidad, disponibilidad e integridad de la misma. Se debe minimizar a su vez los riesgos y vulnerabilidades que puedan existir en las aplicaciones, logrando incrementar la eficacia, reducir los plazos de entrega del servicio y mejorar la calidad. Todo ello encaminado a alcanzar los objetivos establecidos.

En el proceso de gestión hay que tener en cuenta la relación entre el proveedor y el cliente, ya que existe interacción entre ambos. Una de las ventajas que presenta dicho proceso es que todos los productos que se desarrollan en las organizaciones tienen altas probabilidades de incurrir en errores, por lo que una correcta aplicación permitirá mejorar la seguridad y a su vez la calidad de los productos.

La seguridad, en la informática como en otras áreas, se basa en la protección de sus activos, pero no solo de estos, sino también de las aplicaciones durante su desarrollo. Las mismas deben ser protegidas en todo momento de las modificaciones y acceso de personal no autorizado, haciendo cumplir los tres objetivos de la seguridad informática: integridad, disponibilidad y confidencialidad. De esta manera se lograrán aplicaciones más robustas y resistentes a ataques.

1.4- Análisis de las soluciones existentes

1.4.1- OWASP

El Proyecto Abierto de Seguridad de Aplicaciones Web (OWASP), es una comunidad dedicada a permitir a las organizaciones realizar el desarrollo, adquisición y mantenimiento de aplicaciones fiables. Todas las herramientas, documentos, foros y delegaciones del OWASP son libres y abiertos a cualquiera interesado en mejorar la seguridad de las aplicaciones (OWASP 2008).

OWASP produce muchos tipos de materiales con un desarrollo colaborativo y abierto. Además, dicha fundación es una entidad que asegura el mantenimiento del proyecto a largo plazo y gestiona las

Conferencias de Seguridad en Aplicaciones en todo el mundo. Los proyectos del OWASP cubren muchos aspectos de la seguridad en aplicaciones, ayudando de esta forma a las organizaciones a mejorar su capacidad de producir código seguro (OWASP 2008).

Este marco de pruebas consta de las siguientes fases:

- 1 Antes de empezar el desarrollo

1.1 Revisión de Políticas y Estándares: Asegurar que las políticas, documentación y estándares adecuados están implementados. La documentación es extremadamente importante, ya que brinda al equipo de desarrollo políticas y directrices a seguir. Ninguna política o estándar puede cubrir todas las situaciones con las que se enfrentará un equipo de desarrollo. Documentando las incidencias comunes y predecibles, habrá menos decisiones que afrontar durante el mismo.

1.2 Desarrollo de Métricas y Criterios de Medición (Asegurar la Trazabilidad): Antes de empezar el desarrollo, es necesario planificar el programa de medición. Definir los criterios que deben ser medidos proporciona visibilidad de los defectos tanto en el proceso como en el producto. Es algo esencial definir las métricas antes de comenzar el desarrollo, ya que puede haber necesidad de modificar el proceso para poder capturar los datos necesarios.

- 2 Durante la definición de requisitos y el diseño

2.1 Revisión de los Requisitos de Seguridad: Los requisitos de seguridad definen cómo funciona una aplicación desde una perspectiva de la seguridad. En este sentido es indispensable que sean probados y comprobar si existe deficiencias en las definiciones. A la hora de buscar inconsistencias en los requisitos hay que tener en cuenta mecanismos de seguridad, entre los que se encuentran la Confidencialidad de los Datos, Integridad, Autenticación y Autorización.

2.2 Revisión del Diseño y de la Arquitectura: Las aplicaciones deberían tener una arquitectura y un diseño documentado. Es indispensable comprobar estos elementos para asegurar que el diseño y la arquitectura imponen un nivel de seguridad adecuado. Identificar fallos de seguridad en la fase de diseño, no es solo efectivo por los costes a la hora de identificar errores, sino que también puede ser la fase más efectiva para realizar cambios.

2.3 Creación y Revisión de Modelos UML: Una vez completados el diseño y la arquitectura, es necesario construir modelos que describan cómo funciona la aplicación. Los modelos se emplean para conformar junto a los diseñadores de sistemas una comprensión exacta de cómo funciona la aplicación. Si se descubre alguna vulnerabilidad, se le debería transmitir al arquitecto del sistema para buscar soluciones alternativas.

2.4 Creación y Revisión de Modelos de Amenazas: Con las revisiones del diseño, la arquitectura y los modelos explicando cómo funciona el sistema exactamente, es hora de acometer un modelado de amenazas. Es preciso analizar el diseño y la arquitectura para asegurarse de que esas amenazas son mitigadas, aceptadas por el negocio o asignadas a terceros. Cuando las amenazas identificadas no tienen estrategias de mitigación, es necesario revisar el diseño y la arquitectura con los arquitectos de los sistemas para modificar el diseño.

- 3 Durante el desarrollo

3.1 Inspección del Código por Fases: El equipo de seguridad debería realizar una inspección del código por fases con los desarrolladores y, en algunos casos, con los arquitectos del sistema. El propósito de dicha inspección es entender el flujo de programación a un alto nivel, su esquema, la lógica y la estructura del código que conforma la aplicación. La inspección del código por fases permite además al equipo de revisión de código obtener una comprensión general del código fuente y facilita a los desarrolladores explicar el por qué se han desarrollado ciertos elementos de un modo en particular.

3.2 Revisiones de Código: Con una buena comprensión de cómo está estructurado el código y por qué ciertas cosas han sido programadas, el probador puede examinar el código real en busca de defectos de seguridad.

- 4 Durante el despliegue

4.1 Pruebas de Penetración sobre la Aplicación: Tras haber comprobado los requisitos, analizado el diseño y realizado la revisión del código, debería asumirse que se han identificado todas las incidencias. Las pruebas de penetración de la aplicación después de que hayan sido implementadas proporcionan una última comprobación, lo cual permite asegurarse de que no se ha olvidado nada.

4.2 Comprobación de Gestión de Configuraciones: La prueba de intrusión de la aplicación debería incluir la comprobación de cómo se implementó su infraestructura. Aunque la aplicación puede ser segura, un pequeño detalle de la configuración podría estar en una etapa de instalación por defecto y ser vulnerable a explotación.

- 5 Operación y mantenimiento

5.1 Ejecución de Revisiones de la Gestión Operativa: Debe existir un proceso que detalle cómo es gestionada la sección operativa de la aplicación y su infraestructura.

5.2 Ejecución de Comprobaciones Periódicas de Mantenimiento: Deberían realizarse comprobaciones de mantenimientos mensuales o trimestrales sobre la aplicación e infraestructura, para asegurar que no se han introducido nuevos riesgos de seguridad y que el nivel de seguridad sigue intacto.

5.3 Verificación del Control de Cambios: Después de que cada cambio haya sido aprobado, es vital como parte del proceso de gestión de cambios, que este sea comprobado para asegurar que el nivel de seguridad no haya sido afectado.

1.4.2- SDLC

El Ciclo de Vida de Desarrollo del Software (SDLC), es un proceso bien conocido por los desarrolladores, el cual es necesario para evitar problemas de seguridad recurrentes en una aplicación, desarrollando estándares, políticas y guías de uso que funcionen dentro de la metodología de desarrollo. El modelado de amenazas y otras técnicas deberían ser empleados para ayudar a asignar los recursos apropiados en aquellas partes de un sistema más expuestas al riesgo (OWASP 2008).

La integración de la seguridad en cada fase del SDLC, permite un enfoque integral a la seguridad de aplicaciones que se apoya en los procedimientos ya existentes en la organización. Entre las fases que lo componen se encuentran la de diseño, desarrollo e implementación, donde pueden encontrarse problemas relacionados con el Sistema de Autenticación (OWASP 2008).

SDLC es utilizado por los analistas de sistemas para desarrollar sistemas de información, incluyendo requisitos, validación, entrenamiento y propiedad del usuario con la investigación, el análisis, diseño,

puesta en práctica y el mantenimiento. Además, se conoce como el desarrollo de los sistemas de información o desarrollo del uso, dando lugar a un sistema de alta calidad que resuelva o exceda expectativas del cliente, dentro de valoraciones de tiempo y costes. También es barato mantener y rentable (WorldLingo 2010).

Esta metodología consta de las siguientes fases:

1- Requerimientos de recopilación y análisis: En esta fase se realiza la recogida de información y análisis de las necesidades del usuario.

2- Diseño del sistema: Se crea una estructura de la muestra de todo el proyecto y se reúnen los datos necesarios.

3- Desarrollo: Se realiza la codificación de todo el proyecto. Los códigos se generan con facilidad si un diseño adecuado se realiza en la etapa anterior. De acuerdo a las necesidades de la aplicación, se selecciona el lenguaje de programación.

4- Las pruebas del sistema: Después de la generación de códigos, se realizan las pruebas de todos los módulos, los cuales están integrados entre sí y se seleccionan las herramientas de pruebas adecuadas para la comprobación de errores.

5- Operaciones y mantenimiento: En la etapa final de SDLC, el software desarrollado se da a los usuarios. El mantenimiento es necesario después de que el desarrollo de un proyecto es exitoso. Es evidente que los cambios se producen una vez que el proyecto sea entregado al usuario final. Los desarrolladores deben desarrollar el proyecto de tal forma que es adaptable a los cambios. La principal operación del software no debe verse afectado por esos cambios.

1.4.3- OSSTMM

El Manual de la Metodología Abierta de Prueba de Seguridad (OSSTMM) reúne las diversas pruebas y métricas utilizadas por los profesionales durante las auditorías de seguridad. El OSSTMM intenta ser el manual de referencia del profesional y se encuentra en constante evolución. Se centra en los detalles técnicos de los elementos que deben ser probados (Ramos 2010).

Además, tiene como objetivo crear un solo método para realizar pruebas de seguridad en profundidad. Factores como las credenciales del auditor, el tamaño de la compañía de seguridad, las finanzas o el respaldo de ventas, impactan en la escala y complejidad de esta prueba. En el mismo no se encontrará ninguna recomendación para cumplir con la metodología como si se tratase de un diagrama de flujo (Vincent Herzog 2003).

Se trata de un conjunto de pasos que deben ser realizados una y otra vez durante la realización de una prueba. Lo más importante en esta metodología es que las diferentes pruebas son evaluadas y realizadas donde sean aplicables, hasta que se obtengan los resultados esperados dentro de un período de tiempo determinado. Solo así el probador habrá ejecutado la prueba en conformidad con el modelo OSSTMM (Vincent Herzog 2003).

1.4.4- LABSI

Actualmente en la Facultad 2 perteneciente a la UCI, existe un proyecto relacionado con la seguridad de los productos durante su desarrollo (LABSI). En el mismo se realizan las pruebas de seguridad una vez terminada la aplicación, logrando así identificar las vulnerabilidades, alcanzando obtener mejoras significativas en la seguridad del software desarrollado.

1.5- Conclusiones

Después de un análisis de las soluciones existentes se evidencia que no se puede utilizar solamente una de ellas sin tener en cuenta las demás, por lo que es recomendable explotar las ventajas que brindan e incorporarlas en algún punto del desarrollo. Por tanto se concluye que para alcanzar un desarrollo seguro hay que estar actualizado con las herramientas y documentos aportados por OWASP. Al desarrollar estándares, políticas y guías de uso se debe tener en cuenta lo planteado en el SDLC. Para establecer las diversas pruebas y métricas a utilizar se ha de consultar el OSSTMM y una vez terminado el producto y antes de desplegarlo, se ha de pasar por las pruebas que realiza LABSI para validar su seguridad.

Capítulo 2: Análisis de los riesgos

2.1- Introducción

En el presente capítulo se abordarán desde el punto de vista de la seguridad informática algunas de las herramientas, lenguajes de programación, framework, entre otros componentes descritos en el documento “Arquitectura de Software” de los proyectos existentes en el Departamento Geoinformática (Sistemas de Gestión de Datos Geológicos (SGDG) y Plataforma soberana para el desarrollo de Sistemas de Información Geográfica (GeneSIG)). Además, se realizará un estudio de los riesgos detallados en el documento “Plan Mitigación de Riesgos” de los proyectos SGDG y GeneSIG para analizar e identificar amenazas y vulnerabilidades.

2.2- Elementos comunes en el desarrollo de los productos SGDG y GeneSIG

A partir de un estudio realizado sobre el documento de arquitectura de los productos SGDG y GeneSIG, a continuación se relacionan los temas de seguridad que tratan algunas de las herramientas, lenguajes de programación, framework, entre otros componentes que son utilizados por ambos proyectos durante el desarrollo de sus productos.

2.2.1- Lenguaje de programación PHP

PHP es un lenguaje de programación fácil de aprender y los programadores lo aprenden como manera de agregar interactividad a sus sitios web, pero muchos olvidan los aspectos de seguridad que deben ser tenidos en cuenta al implementar las aplicaciones. A veces no se piensa en el daño que puede sufrir un sitio web hasta que ya es demasiado tarde. (PatrocinarTuFuncion.com 2006). A continuación se relacionan algunos de los problemas más comunes de seguridad y cómo evitarlos:

- Nunca confiar en los usuarios: Nunca se debe confiar en que los usuarios van a mandar los datos que en verdad se esperan. Además, los problemas pueden presentarse fácilmente debido a que este haga algo intencionalmente.
- Variables globales: La opción “register_globals” se puede fijar en php.ini, lo cual permite que se utilicen variables globales. En este sentido hay que asegurarse de que se utilizan solamente las variables que se han fijado explícitamente.
- Mensajes de error: Son una herramienta muy útil para los programadores y hackers. En PHP para evitarlos, se puede utilizar .htaccess o php.ini, fijando “error_reporting” a “0”.

Una de las ventajas más grandes de PHP es la facilidad con la cual puede comunicarse con las bases de datos. Muchos usuarios hacen uso excesivo de esto y grandes sitios confían en las bases de datos para funcionar. Sin embargo, con esa ventaja hay problemas suficientemente grandes en la seguridad a los que se tendrá que hacer frente. Uno de estos problemas es cuando un usuario utiliza un fallo para poder atacar directamente al servidor de base de datos con sentencias SQL (Alvarez 2002).

En PHP el archivo seguridad.php, se encargará de dotar seguridad a toda la aplicación de acceso restringido. La técnica que se utilizará es incluirlo al principio de todas las páginas que se quiera que permitan un acceso restringido. El módulo de seguridad, incluido al principio de cada archivo, realizará las comprobaciones oportunas y actuará permitiendo ver el archivo o denegando su visualización dependiendo de dichas comprobaciones (Alvarez 2002).

Dependiendo del nivel de seguridad que se desee implementar, la creación de este archivo puede ser más o menos complicada. Lo único que se hará será recuperar la variable de sesión donde se guarda si ese usuario ha sido autentificado o no (Alvarez 2002). Luego se comprueba esa variable para saber si se ha autentificado el usuario o no, realizando las siguientes acciones:

- Si no se había autentificado, se redirige a la página que tiene el formulario de autentificación. Además, se sale del script PHP, con lo que la página deja de ejecutarse y el resto no se verá.
- Si se había autentificado, se seguiría ejecutando la página con el contenido que correspondiese. No hay que olvidar que este archivo de seguridad se va a ejecutar como un include al principio de todos los archivos de la aplicación restringida, lo que significa que, si no se hace nada, se seguiría mostrando la página donde este archivo está incluido.

2.2.2- Framework Symfony

La seguridad es uno de los pilares de Symfony. La posibilidad de ejecutar una acción puede ser restringida a usuarios con ciertos privilegios. Las herramientas proporcionadas por Symfony para este propósito permiten la creación de aplicaciones seguras, en las que los usuarios necesitan estar autenticados antes de acceder a alguna característica o parte de la aplicación. Añadir esta seguridad a una aplicación requiere dos pasos: declarar los requerimientos de seguridad para cada acción y autenticar a los usuarios con privilegios para que puedan acceder de manera segura. Antes de ser

ejecutada, cada acción pasa por un filtro especial que verifica si el usuario actual tiene privilegios de acceder a la acción requerida (Zaninotto y Potencier 2009).

En Symfony, los privilegios están compuestos por dos partes:

- Las acciones seguras requieren que los usuarios estén autenticados.
- Las credenciales son privilegios de seguridad agrupados bajo un nombre y que permiten organizar la seguridad en grupos.

Para restringir el acceso a una acción se crea y se edita un archivo de configuración YAML llamado security.yml en el directorio config del módulo. En este archivo, se pueden especificar los requerimientos de seguridad que los usuarios deberán satisfacer para cada acción o para todas las acciones. Las mismas no incluyen restricciones de seguridad por defecto, así que cuando no existe el archivo security.yml o no se indica ninguna acción en este, todas las acciones son accesibles por todos los usuarios. En caso de existir este archivo, Symfony busca por el nombre de la acción y si existe, verifica que se satisfagan los requerimientos de seguridad (Zaninotto y Potencier 2009). Lo que sucede cuando un usuario trata de acceder a una acción restringida depende de sus credenciales:

- Si el usuario está autenticado y tiene las credenciales apropiadas, entonces la acción se ejecuta.
- Si el usuario no está autenticado, es redireccionado a la acción de login.
- Si el usuario está autenticado pero no posee las credenciales apropiadas, será redirigido a la acción segura por defecto.

2.2.3- Servidor de aplicaciones Apache

A veces, cuando se dispone de un servidor web propio se cometen errores que ponen en peligro las páginas que se están alojando. Por norma general, si los conocimientos de gestión y administración de un servidor web son pocos, se recomienda utilizar aquellos más sencillos, siendo capaces de controlar todo y no cometer errores. Evidentemente apache es uno de los más grandes e importantes pero no por ello el más seguro. Si se quiere utilizar este servidor web se debe tener en cuenta algunas nociones de seguridad (Emilio 2009). Cuando un servidor apache recibe una petición de una

página web, antes de devolver el resultado, lleva a cabo varias acciones para verificar que la petición está autorizada (Cuenca 2006). Las acciones pueden ser agrupadas en tres tipos:

- Autentificación: Puede estar gestionada por distintos módulos, dependiendo de la forma de implementación. Si se decide llevarla a cabo gestionando ficheros con listas de usuarios y contraseñas (encriptadas), se deberá utilizar el módulo mod_auth. Sin embargo, en caso de decidir llevarla a cabo mediante base de datos, se deberá utilizar los módulos mod_auth_dbm. Si el usuario no está autenticado, es redireccionado a la acción de login.
- Autorización: Es gestionada o bien mediante la directiva <directory> en el fichero principal de configuración, o bien mediante la configuración de la carpeta a través de ficheros .htaccess.
- Control de Acceso: Se puede llevar a cabo mediante las directivas <directory><files>y <location>, o a través del fichero de configuración .htaccess para controlar una carpeta específica.

2.2.4- Gestor de Base de Datos PostgreSQL

La seguridad en PostgreSQL se materializa en tres aspectos:

- Seguridad en la manipulación de los ficheros.
- Seguridad en los accesos de los clientes.
- Definición de los privilegios para acceder a los objetos de la base de datos a los usuarios.

Todos los ficheros almacenados en la base de datos están protegidos contra escritura por cualquier cuenta que no sea la del usuario del gestor. Este usuario es el único que puede leer, escribir y ejecutar sin restricción. Es importante poder definir desde qué equipos se pueden conectar a la base de datos, así como los usuarios y las bases de datos a las cuales se pueden conectar. PostgreSQL por lo general utiliza el puerto 5432, el cual no debería ser accesible desde lugares no confiables. A cada usuario de Postgres se le asigna un nombre de usuario (opcionalmente) y una contraseña. Por defecto, los usuarios no tienen permiso de escritura a base de datos que no hayan creado (Torrealba 2010).

Las acciones que se pueden realizar en cada momento vienen condicionadas por los permisos del usuario que se conecte a la base de datos. Todos los objetos (tablas, vistas y secuencias) tienen un propietario, que es la persona que lo creó y que puede establecer permisos en el objeto. Los

permisos se componen de un nombre de usuario o grupo y un conjunto de derechos. Por otra parte se puede agregar que el sistema lleva automáticamente la cuenta de todas las operaciones realizadas por los usuarios sobre los datos. Esto se realiza mediante el registro de auditoría, el cual es un archivo o base de datos especial que mantiene el control de acceso a la base de datos, con el objetivo de saber qué o quién realizó una determinada modificación y en qué momento (Torrealba 2010).

Para asegurar la integridad de los datos, PostgreSQL posee medidas de seguridad que impiden que se introduzcan datos erróneos, lo cual puede suceder tanto por motivos físicos (defectos de hardware, actualización incompleta ocasionada por causas externas), como de operación (introducción de datos incoherentes). Realiza un aseguramiento de la información, enviando mensajes de error o advertencias cuando se viola alguna de las restricciones, a su vez puede crear reglas de integridad particulares para casos específicos (Torrealba 2010).

2.2.5- Control de versiones Subversion

El trabajo con Subversion se puede realizar con un cliente y un servidor o varios clientes y un servidor compartiendo códigos en forma local, en la red de una empresa o hacia lugares remotos con conectividad a Internet. Para ser esto posible Subversion utiliza distintos protocolos (svn, http, svn+ssh, https y file), siendo de muy buena práctica utilizarlo aunque sea una sola persona dedicada a la programación o un grupo de trabajo. Trabajar con Subversion permite respaldar y recuperar revisiones anteriores de códigos desde un avanzado IDE de desarrollo (Eclipse, Visual Studio), explorador de archivos o mediante líneas de comando en distintas plataformas (Linux, Windows, Mac, Unix) (Collins-Sussman, Fitzpatrick y Michael Pilato 2004).

2.2.5.1- ¿Cómo funciona?

El corazón de Subversion es su repositorio central guardando la información de los archivos en estructura de árbol (los directorios también son tratados como archivos). Las lecturas y escrituras de la información son a través de un sistema cliente-servidor. Cuando una persona realiza una lectura de un archivo, siempre obtiene la última versión, pero en cualquier momento puede consultar revisión de la información anterior. Subversion genera de todos los archivos una copia de seguridad visible y otra

oculta en forma automática, generando doble respaldo (Collins-Sussman, Fitzpatrick y Michael Pilato 2004).

El modelo que utiliza Subversion es chequear, copiar una nueva actualización, modificar, mezclar y subir los códigos al repositorio central. Cada desarrollador puede realizar los cambios que estime conveniente sin entorpecer el trabajo del otro. Para coordinar que en el repositorio siempre se encuentre la última versión "se debe primero verificar si existen cambios en él". Lo ideal es que el grupo de trabajo asigne a cada desarrollador un módulo distinto, evitando la generación de conflicto a la hora de que dos personas trabajen con el mismo archivo (Collins-Sussman, Fitzpatrick y Michael Pilato 2004).

Subversion es de gran utilidad a la hora de mezclar códigos, pero siempre debe haber una persona única dentro del proyecto que sea el responsable de que esto suceda de la manera correcta. Para evitar conflictos siempre debe existir una buena comunicación entre los desarrolladores. Si ocurre un error o se eliminaron archivos en las últimas revisiones, se puede estar tranquilo porque ningún código se pierde, solo puede darse el caso que no sea visible en la última versión dentro del repositorio actual (Collins-Sussman, Fitzpatrick y Michael Pilato 2004).

Una vez analizadas las herramientas que se utilizan en el desarrollo de los productos, se dialogó con los líderes de los proyectos SGDG y GeneSIG los temas de seguridad con que disponen dichas herramientas, detectándose como brecha de seguridad:

- Los proyectos SGDG y GeneSIG no realizan los registros de auditorías que el Gestor de Base de Datos PostgreSQL mantiene para controlar el acceso a la base de datos.

2.3- Amenazas identificadas en el desarrollo de los proyectos

1. Pérdida o destrucción de la información.
2. Publicación de información no autorizada.
3. Robos de tecnologías informáticas.
4. Rotura de los medios informáticos.
5. Fallas del fluido eléctrico.
6. Fallas en el hardware.

7. Fallas de aplicaciones.
8. Accesos no autorizados.
9. Falta de calificación del RRHH.
10. Planificación de actividades imprevistas por parte de la universidad.
11. Afectación del proceso productivo.
12. Desconocimiento del área a informatizar.

2.4- Vulnerabilidades identificadas en el desarrollo de los proyectos SGDG y GeneSIG

2.4.1- SGDG

1. La información que se conoce del negocio no es lo más abarcadora posible.
2. Atrasos en los cursos de capacitación necesarios tanto para los analistas como para los desarrolladores.
3. Falta de conocimiento acerca de la tecnología definida en la arquitectura.
4. Falta de compromiso del equipo de desarrollo.
5. Modificaciones no controladas en el plan de trabajo.
6. Escasos medios informáticos para trabajar.
7. Insuficientes medidas de control de acceso.

2.4.2- GeneSIG

1. Equipo de proyecto con escasa o ninguna experiencia en proyectos similares.
2. Falta de métricas, estándares y políticas de seguridad.
3. Falta de revisiones técnicas, procedimientos de pruebas y casos de prueba.
4. Seguimiento incorrecto de los riesgos en el proyecto.
5. Extravío de documentos o archivos importantes para el proyecto.
6. Cambios imprevistos en los requerimientos durante las diferentes etapas del proyecto.
7. Cambios en los miembros del equipo.
8. Cambios en las condiciones de trabajo.
9. Fechas de compromisos inestables.
10. Falta de motivación en el proyecto.
11. Pocos equipos de desarrollo.
12. Cambios imprevistos en las actividades del cronograma de trabajo.

2.5- Riesgos detectados en el desarrollo de los proyectos

Al estudiar el concepto de riesgo se puede ver la estrecha relación que existe entre vulnerabilidad y amenaza con este término. En este sentido se debe mantener un control estricto sobre las amenazas que afectan y las vulnerabilidades que presentan los productos, protegiendo así de esta forma la seguridad de las aplicaciones. A partir de este análisis a continuación se relacionan los riesgos existentes en los proyectos:

1. Pérdida de tiempo de trabajo.
2. Atraso en el cronograma del proyecto.
3. Interrupción del proceso de desarrollo.
4. Incumplimiento del plan de producción del proyecto.
5. Incumplimiento de las tareas del equipo.
6. Errores en el producto final.
7. Producto inseguro.
8. Retraso en los plazos de entrega planificados.
9. Falta de calidad en el producto final.
10. Mala organización del equipo.
11. Falta de aprovechamiento de las ventajas que brindan en cuanto a la seguridad los componentes utilizados en el desarrollo del producto.
12. Desconocimiento del equipo de las herramientas a utilizar en el desarrollo del proyecto.

2.6- Principales incidentes a aplicaciones web

1. Fallas en la implementación de protocolos: El programador junto con el encargado de la seguridad informática, debe analizar los posibles requerimientos de seguridad necesarios para que la aplicación funcione sobre un ambiente de red que brinde mayores niveles de seguridad y control de tráfico.

2. Desbordamientos y chequeos de sintaxis: Dos elementos importantes en la revisión y evaluación del software. Por un lado la evaluación de los desbordamientos bien sea de memoria o de variables específicas dentro de un programa y por otro lado, la verificación de buen uso de los comandos o palabras reservadas en el lenguaje de programación, que permitan al programador un uso adecuado

y eficiente de las estructuras. Si este aspecto no se considera con el rigor necesario, se estará comprometiendo la integridad del ambiente de ejecución de la aplicación.

3. Cambios en el ambiente de ejecución: Los parches, los cambios en la configuración y variables de entorno alrededor de las aplicaciones son elementos críticos para mantener una ejecución adecuada y controlada de las rutinas y acciones previstas en el software. Al descuidar este aspecto, es probable involucrar efectos de borde o condiciones de excepción no previstas que comprometan no solamente un módulo de la aplicación sino el sistema de información mismo.

4. Invocaciones no controladas: Inadecuado manejo de errores o excepciones en las aplicaciones o exceso de privilegios de ejecución, los cuales se manifiestan en comportamientos inesperados del software que generalmente ofrecen mayores privilegios o accesos adicionales a la información del sistema. En este sentido, el control adecuado de interrupciones, mensajes de error y entorno de ejecución de los programas se vuelve crítico al ser estos elementos los que definen la interacción del software con el usuario final y su relación con el entorno de ejecución.

5. Bypass a bajo nivel: Las implicaciones de esta fuente de vulnerabilidades hace referencia al aseguramiento que la aplicación debe tener al ser invocada o ejecutada en un ambiente computacional seguro. El programador debe fortalecer y asegurar una manera autorizada de ingreso a la aplicación por parte del usuario, estableciendo mecanismos de monitoreo y control que velen porque esto se cumpla. El sobrepasar un control de acceso a un objeto, bien sea a través de permisos deficientemente otorgados, artificios que interrumpan la normal ejecución (contraseñas de BIOS) o por la manipulación de la memoria de ejecución de la aplicación, constituye un atentado directo contra la confiabilidad e integridad del software.

6. Convenientes pero peligrosas características del diseño del software: Esta fuente de vulnerabilidad presenta funcionalidades que son deseables en el software para aumentar la versatilidad de uso de las aplicaciones. Entre estas se encuentran herramientas de depuración o debugging, conexiones remotas en puertos especiales, entre otras, las cuales ofrecen importantes elementos a los programadores y usuarios, pero que generalmente abren posibilidades de ingresos no autorizados que comprometen la integridad del sistema.

7. Cross-site scripting (XSS): Hoy día es muy común encontrar sitios y usuarios afectados por este tipo de agresión. Para ejecutarla basta nada más con manejar un poco de las etiquetas HTML y algún lenguaje de Scripting en sitios que no están protegidos contra este tipo de ataque (Araya Ramos 2006).

8. Inyección de SQL: Consiste en modificar el comportamiento de las consultas mediante la introducción de parámetros no deseados en los campos a los que tiene acceso el usuario. Este tipo de errores puede permitir a usuarios malintencionados acceder a datos a los que de otro modo no tendrían acceso y, en el peor de los casos, modificar el comportamiento de las aplicaciones (Manivesa 1997).

2.7- Conclusiones

En el presente capítulo se realizó un análisis de los temas de seguridad que tratan los componentes utilizados en el desarrollo de los productos, para eliminar brechas de seguridad que pudieran ser introducidas a las aplicaciones. Además, se logró identificar las amenazas, vulnerabilidades y riesgos que afectan a los productos del Departamento Geoinformática para un mejor control de los mismos. Se realizó un estudio de los principales incidentes a aplicaciones web para ser eliminados como posibles amenazas, teniendo todo esto en cuenta para la descripción de las fases del procedimiento a desarrollar.

Capítulo 3: Propuesta de solución

3.1- Introducción

En el presente capítulo se describirá el procedimiento desglosado por fases que deberá cumplir el Departamento Geoinformática para obtener productos seguros. En las mismas se detallarán cuáles son los aspectos de seguridad que deben ser tratados durante el desarrollo.

3.2- Pasos a ejecutar en cada fase

Al conceptualizar el proyecto se ha de establecer quién estará al frente de la seguridad informática y las actividades que lo involucren.

3.2.1- Fase inicial

3.2.1.1- Garantizar el conocimiento previo del equipo de trabajo sobre el producto a realizar

1. Hacer reuniones organizativas iniciales. Establecer pautas de seguridad con líderes y clientes.

2. Desarrollar políticas y estándares necesarios.

3. Caracterizar a los desarrolladores, teniendo en cuenta cursos de capacitación vencidos, roles desempeñados y experiencia en el tipo de producto a desarrollar.

4. Capacitar a los desarrolladores teniendo en cuenta las necesidades detectadas.

3.2.1.2- Causas de incidentes de seguridad

1. Buscar y analizar vulnerabilidades.

2. Analizar las amenazas.

3.2.1.3- Gestión de Riesgo

1. Detectar los riesgos que afecten el desarrollo del producto.

3.2.1.4- Acceso a los documentos

1. Subir al repositorio los documentos aportados por los clientes y el proyecto.

2. A estos documentos solo tendrán acceso los usuarios autorizados.

3.2.1.5- Acceso de los usuarios

1. Establecer los permisos en correspondencia con el rol y responsabilidad de cada usuario.

3.2.1.6- Analizar las características del producto en cuanto a la seguridad

1. Proteger la información contra accesos no autorizados utilizando mecanismos de validación que lo garantice.

2. Usar mecanismos de encriptación de los datos que por cuestiones de seguridad no deben viajar al servidor en texto plano.

3.2.1.7- Seguridad en los componentes del desarrollo

1. En la definición de la arquitectura se han de incorporar los elementos de seguridad que manejan las herramientas descritas en el documento “Arquitectura de Software” del proyecto.

3.2.2- Fase intermedia

3.2.2.1- Controlar las características del producto en cuanto a la seguridad

1. Controlar durante el desarrollo de los productos el cumplimiento de los requisitos no funcionales de seguridad establecidos.

3.2.2.2- Desarrollar la documentación necesaria

1. Documentar el diseño.

2. Construir modelos que describan el funcionamiento de la aplicación.

- El responsable de seguridad debe informar al arquitecto si se detecta alguna vulnerabilidad.
- En caso de que no exista estrategia de mitigación para amenazas detectadas, se ha de actualizar el documento Gestión de Riesgo y Plan de Seguridad Informática.

- Según lo regido en los documentos antes mencionados, el diseño debe ser modificado por el arquitecto.

3.2.2.3- Utilizar los elementos de seguridad

1. Identificar fallos de seguridad en el diseño.

2. Estudio del documento de arquitectura.

3. Realizar las auditorías correspondientes al Gestor de Base de Datos “PostgreSQL” para mantener el control de acceso a la base de datos.

4. Desactivar register_globals en PHP (archivos php.ini o .htaccess). Activarlo solo en caso de que se vaya a ser uso de variables globales.

3.2.2.4- Validar entrada y salida de datos

1. Validar cada uno de los campos en donde el usuario pueda o le sea requerido el ingreso de datos.

3.2.2.5- Causas de incidentes de seguridad

1. Analizar y buscar nuevas vulnerabilidades.

2. Buscar nuevas amenazas.

3.2.2.6- Gestión de Riesgo

1. Detectar los riesgos que afecten el desarrollo del producto.

3.2.3- Fase final

3.2.3.1- Validar el sistema

1. Validar las incompatibilidades con otros ambientes de ejecución.

3.2.3.2- Incidentes de seguridad al sistema

1. Aplicar las pruebas de instrusión a la aplicación.

2. Realizar comprobaciones de mantenimiento mensuales.

3. Efectuar comprobaciones después de cada cambio, verificando así que el nivel de seguridad no haya sido afectado.

3.2.3.3- Comprobar las características del producto en cuanto a la seguridad

1. Comprobar el cumplimiento de los requisitos no funcionales de seguridad establecidos.

3.2.3.4- Despliegue

1. Garantizar montar la aplicación en un servidor configurado de forma segura para evitar las inyecciones SQL y ataques directos a la base de datos.

3.3- Descripción del procedimiento

Entrada	Fase	Actividad	Tarea	Resp. o Rol	Salida
Ficha técnica del proyecto realizada. Roles y responsabilidades (RRHH).	Inicial	Garantizar el conocimiento previo del equipo de trabajo sobre el producto a realizar.	1. Hacer reuniones organizativas iniciales. Establecer pautas de seguridad con líderes y clientes.	Líder de proyecto.	Acta de las reuniones. Pautas de seguridad.
			2. Desarrollar políticas y estándares necesarios.	Asesor de Seguridad Informática (SI).	Políticas y estándares.
			3. Caracterizar a los desarrolladores teniendo en cuenta cursos de	Subdirector de Formación.	Encuesta sobre los conocimientos del RRHH. Resultados de

			capacitación vencidos, roles desempeñados y experiencia en el tipo de producto a desarrollar.		la encuesta. Documento de caracterización de los desarrolladores.
			4. Capacitar a los desarrolladores teniendo en cuenta las necesidades detectadas.	Subdirector de Formación.	Plan de capacitación (Cursos Optativos).
Pautas de seguridad.	Inicial	Causas de incidentes de seguridad.	1. Buscar y analizar vulnerabilidades.	Asesor de SI.	Documento de Vulnerabilidades y Amenazas.
			2. Analizar las amenazas.	Asesor de SI.	
Cronograma de trabajo. Documento de Vulnerabilidades y Amenazas.	Inicial	Gestión de Riesgo.	1. Detectar los riesgos que afecten el desarrollo del producto.	Asesor de SI.	Documento Gestión de Riesgo.
Documentación del proyecto.	Inicial	Acceso a los documentos.	1. Subir al repositorio los documentos aportados por los clientes y el proyecto.	Líder de proyecto.	Repositorio actualizado.

			2. A estos documentos sólo tendrán acceso los usuarios autorizados.	Gestión y configuración.	Listas de control de acceso (ACLs).
Listas de control de acceso (ACLs).	Inicial	Acceso de los usuarios.	1. Establecer los permisos en correspondencia con el rol y responsabilidad de cada usuario.	Gestión y configuración.	Backups (Generados por el Subversion).
Pautas de seguridad.	Inicial	Analizar las características del producto en cuanto a la seguridad.	1. Proteger la información contra accesos no autorizados utilizando mecanismos de validación que lo garantice.	Analista principal.	Especificación de requisitos de seguridad para el producto.
			2. Usar mecanismos de encriptación de los datos que por cuestiones de seguridad no deben viajar al servidor en texto plano.	Asesor de SI.	Clasificación de la información. Selección del mecanismo de encriptación para los datos.

Documento de Arquitectura de Software.	Inicial	Seguridad en los componentes del desarrollo.	1. En la definición de la arquitectura se han de incorporar los elementos de seguridad que manejan las herramientas.	Asesor de SI y Arquitecto.	Documento de Arquitectura de Software actualizado.

Entrada	**Fase**	**Actividad**	**Tarea**	**Resp. o Rol**	**Salida**
Especificación de requisitos de seguridad para el producto.	Intermedia	Controlar las características del producto en cuanto a la seguridad.	1. Controlar durante el desarrollo de los productos el cumplimiento de los requisitos no funcionales de seguridad establecidos.	Asesor de SI.	Cumplimiento de los requisitos no funcionales de seguridad establecidos.
Modelo de análisis.	Intermedia		1. Documentar el diseño.	Diseñador principal.	Modelo de diseño.
			2. Construir modelos que describan el funcionamiento de la aplicación.	Diseñador principal.	Prototipo de interfaz.
Documento de Arquitectura de Software	Intermedia	Utilizar los elementos de	1. Identificar fallos de seguridad en el	Diseñador principal.	Diseño de la aplicación

actualizado. Archivos del servidor de aplicaciones.		seguridad.	diseño.		actualizado.
			2. Estudio del documento de arquitectura.	Líder de los implementadores y Administrador de Base de Datos.	Código de seguridad. Registro de auditoría.
			3. Realizar las auditorías correspondientes al Gestor de Base de Datos “PostgreSQL”.	Administrador de Base de Datos.	Registro de auditoría actualizado.
			4. Desactivar register_globals en PHP. Activarlo sólo en caso de que se vaya a ser uso de variables globales.	Líder de los implementadores.	Archivos del servidor de aplicaciones actualizado.
Modelo de Casos de Uso del Sistema.	Intermedia	Validar entrada y salida de datos.	1. Validar cada uno de los campos en donde el usuario pueda o le sea requerido el ingreso de datos.	Líder de los implementadores.	Código validado.

Documento de Vulnerabilidades y Amenazas.	Intermedia	Causas de incidentes de seguridad.	1. Analizar y buscar nuevas vulnerabilidades.	Asesor de SI.	Documento de Vulnerabilidades y Amenazas actualizado.
			2. Buscar nuevas amenazas.	Asesor de SI.	
Documento de Vulnerabilidades y Amenazas actualizado. Cronograma de trabajo. Documento Gestión de Riesgo.	Intermedia	Gestión de Riesgo.	1. Detectar los riesgos que afecten el desarrollo del producto.	Asesor de SI.	Documento Gestión de Riesgo actualizado.

Entrada	**Fase**	**Actividad**	**Tarea**	**Resp. o Rol**	**Salida**
Sistema.	Final	Validar el sistema.	1. Validar las incompatibilidades con otros ambientes de ejecución.	Líder de los implementadores.	Sistema validado.
Plan de pruebas. Casos de pruebas basados en requisitos de	Final	Incidentes de seguridad al sistema.	1. Aplicar las pruebas de instrusión a la aplicación.	Verificador y Asesor de SI.	No conformidades. Registro de evaluaciones del proyecto.

seguridad. Plan de gestión de configuración.			2. Realizar comprobaciones de mantenimiento mensuales.	Gestión de configuración y cambio y Revisor Técnico.	Documento de Revisión.
			3. Efectuar comprobaciones después de cada cambio, verificando así que el nivel de seguridad no haya sido afectado.	Gestión de configuración y cambio y Revisor Técnico.	Solicitud de cambio.
Especificación de requisitos de seguridad para el producto.	Final	Comprobar las características del producto en cuanto a la seguridad.	1. Comprobar el cumplimiento de los requisitos no funcionales de seguridad establecidos.	Asesor de SI.	Cumplimiento de los requisitos no funcionales de seguridad establecidos. Producto completamente seguro.
Modelo de despliegue. Documento de Arquitectura de Software.	Final	Despliegue.	1. Garantizar montar la aplicación en un servidor configurado de forma segura.	Asesor de SI y Gestor de Despliegue.	Producto desplegado.

Aspectos a tener en cuenta:

- Al conceptualizar el proyecto se ha de establecer quién estará al frente de la seguridad informática.
- En el procedimiento sólo se recogen las actividades que tributan en el desarrollo del software a garantizar la seguridad de los productos.
- En el campo responsable se mencionan algunos roles que en todas las metodologías no son nombrados por igual. En estos casos se sugiere asignar el homólogo que realice esta tarea.
- Aunque en todas las actividades no aparece como responsable el encargado de la seguridad informática, este tiene que velar por los pasos del procedimiento.
- El Responsable de Seguridad Informática debe autoprepararse manteniéndose actualizado con las herramientas y documentos que reporta El Proyecto Abierto de Seguridad de Aplicaciones Web (OWASP).
- El Responsable de Seguridad Informática al desarrollar estándares, políticas y guías de uso que funcionen dentro de la metodología de desarrollo ha de consultar El Ciclo de Vida de Desarrollo del Software (SDLC). Además en el modelado de amenazas y otras técnicas, para la asignación de los recursos apropiados en aquellas partes del sistema más expuestas al riesgo.
- El Responsable de Seguridad Informática ha de consultar El Manual de la Metodología Abierta de Prueba de Seguridad (OSSTMM) para establecer las diversas pruebas y métricas a utilizar, asimismo, para evaluar los detalles técnicos de los elementos que deben ser probados.
- El Responsable de Seguridad Informática una vez terminado el producto y ante de desplegarlo debe pasarlo por las pruebas que realiza LABSI para validar su seguridad.
- El Responsable de Seguridad Informática tiene que incorporar al Plan de Seguridad y Contingencias medidas que respalden el procedimiento para el desarrollo seguro.
- El Responsable de Seguridad Informática tiene que actualizar el procedimiento ante cualquier cambio sustancial, provocado por el avance o cambio en las tecnologías actuales.

3.4- Conclusiones

Con la correcta aplicación paso a paso en cada fase del procedimiento descrito, los productos pertenecientes al departamento que cuenten con las características mencionadas en capítulos

anteriores, evitarán muchos de los problemas de seguridad que enfrenta hoy día el desarrollo de software. En este sentido el procedimiento controla de una forma más organizada las actividades del proyecto que tributan a la seguridad informática y al cumplimiento de sus objetivos de manera general.

CONCLUSIONES GENERALES

Una vez finalizada la investigación se puede afirmar que se ha dado cumplimiento de manera satisfactoria al objetivo propuesto, pues:

- Se planteó un procedimiento que especifica cómo desarrollar productos en el Departamento Geoinformática, obedeciendo los objetivos de la seguridad informática.
- Se dieron a conocer las actividades y tareas de cada una de las fases propuestas, las cuales deben ser aplicadas durante el desarrollo de los productos del departamento.
- Se sometió la propuesta a una evaluación por expertos, validando satisfactoriamente la misma, concluyendo que, con un procedimiento que especifica cómo desarrollar productos en el Departamento Geoinformática, se contribuirá a que los mismos cumplan los objetivos de la seguridad informática.

RECOMENDACIONES

Se recomienda:

- Aplicar este procedimiento a los productos realizados en el Departamento Geoinformática.
- Adaptar el procedimiento a las nuevas condiciones que presenten los futuros proyectos del Departamento Geoinformática, cumpliendo con el principio de Dinamismo de la seguridad informática.
- Estimular al resto de las facultades para la aplicación del procedimiento durante el desarrollo de sus productos.

BIBLIOGRAFÍA

REFERENCIAS BIBLIOGRÁFICAS

Comerciales, Comisión de Reglamentos Técnicos. «EDI. Tecnología de la Información. Código de buenas prácticas para la gestión de la seguridad de la información.» 2007.

López Crespo, Francisco. «MAGERIT-versión 2 Metodología de Análisis y Gestión de Riesgos de los Sistemas de Información.» 2006.

Sena, Leonardo, y Simón Mario Tenzer. «Introducción a Riesgo Informático.» 2004.

Vincent Herzog, Peter. «OSSTMM 2.1. Manual de la Metodología Abierta de Testeo de Seguridad.» 2003.

Montes de Oca, Yasser León, y Joao Lemus Martínez. «Herramienta de instalación para el Sistema de Gestión Penitenciaria Venezolano.» 2008.

OWASP, Fundación. «GUÍA DE PRUEBAS OWASP.» 2008.

Rodríguez Puente, Lic. Rafael. «Servicio de camino mínimo sobre un Sistema de Información Geográfica basado en software libre.» 2007.

Ramos, Alejandro. «SecurityByDefault.» 7 de marzo de 2010. http://www.securitybydefault.com/2010/03/metodologia-osstmm.html (último acceso: 1 de diciembre de 2010).

«WorldLingo.» 2010. http://www.worldlingo.com/ma/enwiki/es/Systems_Development_Life_Cycle (último acceso: 2 de diciembre de 2010).

Collins-Sussman, Ben, Brian W. Fitzpatrick, y C. Michael Pilato. «Control de versiones con Subversion.» 2004.

Zaninotto, François, y Fabien Potencier. *librosweb.es.* 25 de agosto de 2009. http://www.librosweb.es/symfony/capitulo6/seguridad_de_la_accion.html (último acceso: 3 de febrero de 2011).

Torrealba, Enrique. «sabd15N1.» 8 de diciembre de 2010. http://sabd15n1.wikispaces.com/PostgreSQL (último acceso: 3 de febrero de 2011).

Cuenca, Carlos Luis. «desarrolloweb.com.» 6 de junio de 2006. http://www.desarrolloweb.com/articulos/2499.php (último acceso: 7 de febrero de 2011).

Emilio. «Emilio.» 17 de febrero de 2009. http://emilio.aesinformatica.com/2009/02/17/fortificar-apache-para-aumentar-la-seguridad/ (último acceso: 7 de febrero de 2011).

Alvarez, Miguel Angel. «desarrolloweb.com.» 19 de diciembre de 2002. file:///H:/Tesis/web%20descarga/1010.php.htm (último acceso: 14 de febrero de 2011).

«PatrocinarTuFuncion.com.» 23 de mayo de 2006. file:///H:/Tesis/web%20descarga/php_seguridad_I.htm (último acceso: 14 de febrero de 2011).

del Toro González, Yoandy, y Raida Zaldívar Picasso. «Framework para desarrollar juegos Multi-jugador sobre J2ME para móviles con conexión Bluetooth.» La Habana, 2007.

Escobar Zaragoza, Mercedes. "Aplicación y Mejora del Modelo de Gestión de Riesgos MoGeRi." La Habana, 2009.

PRESSMAN. *Ingeniería de Software. Un enfoque prático.* La Habana: Felix Varela, 2005.

PMI. Guía de los Fundamentos de la Dirección de Proyectos. 2004.

Pluma Clavel, Dayana, y Yaime Oduardo Tamayo. «Aseguramiento de la Calidad en la línea de producto "Inspección de territorios".» La Habana, 2010.

Rodríguez Torres, Alexander. «Sistema de Información Geográfica de la UCI basado en tecnología OpenSource.» La Habana, 2005.

Araya Ramos, Kenyie. «desarrolloweb.com.» 16 de noviembre de 2006. http://www.desarrolloweb.com/articulos/introduccion-cross-site-scripting.html (último acceso: 1 de abril de 2011).

Manivesa, Cesar. «maestros del web.» 1997. http://www.maestrosdelweb.com/editorial/inyecsql (último acceso: 1 de abril de 2011).

BIBLIOGRAFÍA CONSULTADA

Monroy López, Daniel. «Análisis inicial de la anatomía de un ataque a un Sistema Informático.» 2009.

Hervalejo Sánchez, Alberto. «Auditorías de Seguridad Informática y la OSSTMM.» 2009.

Reyes Plano, Yandielys. «Aplicación y mejora del Modelo de Gestión de Riesgos "MoGeRi" al proyecto "Captura y Catalogación de Medias".» La Habana, 2009.

Ortiz, Kadir Hector. «eumed.net.» 2009. http://www.eumed.net/libros/2009c/583/Requerimientos%20no%20Funcionales.htm (último acceso: 1 de abril de 2011).

Gutierrez, Manu. «Las principales 5 vulnerabilidades web.» 11 de octubre de 2007. http://www.desarrolloweb.com/articulos/principales-vulnerabilidades-web.html (último acceso: 6 de abril de 2011).

San Miguel Carrasco, Rafael. «borrmart,s.a.» 2005. http://www.borrmart.es/articulo_redseguridad.php?id=463&numero=17#.# (último acceso: 6 de abril de 2011).

Comerciales, Comisión de Reglamentos Técnicos. «EDI. Tecnología de la Información. Código de buenas prácticas para la gestión de la seguridad de la información.» 2007.

López Crespo, Francisco. «MAGERIT-versión 2 Metodología de Análisis y Gestión de Riesgos de los Sistemas de Información.» 2006.

Sena, Leonardo, y Simón Mario Tenzer. «Introducción a Riesgo Informático.» 2004.

Vincent Herzog, Peter. «OSSTMM 2.1. Manual de la Metodología Abierta de Testeo de Seguridad.» 2003.

Montes de Oca, Yasser León, y Joao Lemus Martínez. «Herramienta de instalación para el Sistema de Gestión Penitenciaria Venezolano.» 2008.

OWASP, Fundación. «GUÍA DE PRUEBAS OWASP.» 2008.

Rodríguez Puente, Lic. Rafael. «Servicio de camino mínimo sobre un Sistema de Información Geográfica basado en software libre.» 2007.

Ramos, Alejandro. «SecurityByDefault.» 7 de marzo de 2010. http://www.securitybydefault.com/2010/03/metodologia-osstmm.html (último acceso: 1 de diciembre de 2010).

«WorldLingo.» 2010. http://www.worldlingo.com/ma/enwiki/es/Systems_Development_Life_Cycle (último acceso: 2 de diciembre de 2010).

Collins-Sussman, Ben, Brian W. Fitzpatrick, y C. Michael Pilato. «Control de versiones con Subversion.» 2004.

Zaninotto, François, y Fabien Potencier. *librosweb.es.* 25 de agosto de 2009. http://www.librosweb.es/symfony/capitulo6/seguridad_de_la_accion.html (último acceso: 3 de febrero de 2011).

Torrealba, Enrique. «sabd15N1.» 8 de diciembre de 2010. http://sabd15n1.wikispaces.com/PostgreSQL (último acceso: 3 de febrero de 2011).

Bibliografía

Cuenca, Carlos Luis. «desarrolloweb.com.» 6 de junio de 2006. http://www.desarrolloweb.com/articulos/2499.php (último acceso: 7 de febrero de 2011).

Emilio. «Emilio.» 17 de febrero de 2009. http://emilio.aesinformatica.com/2009/02/17/fortificar-apache-para-aumentar-la-seguridad/ (último acceso: 7 de febrero de 2011).

Alvarez, Miguel Angel. «desarrolloweb.com.» 19 de diciembre de 2002. file:///H:/Tesis/web%20descarga/1010.php.htm (último acceso: 14 de febrero de 2011).

«PatrocinarTuFuncion.com.» 23 de mayo de 2006. file:///H:/Tesis/web%20descarga/php_seguridad_I.htm (último acceso: 14 de febrero de 2011).

del Toro González, Yoandy, y Raida Zaldívar Picasso. «Framework para desarrollar juegos Multi-jugador sobre J2ME para móviles con conexión Bluetooth.» La Habana, 2007.

Escobar Zaragoza, Mercedes. "Aplicación y Mejora del Modelo de Gestión de Riesgos MoGeRi." La Habana, 2009.

PRESSMAN. *Ingeniería de Software. Un enfoque prático.* La Habana: Felix Varela, 2005.

PMI. Guía de los Fundamentos de la Dirección de Proyectos. 2004.

Pluma Clavel, Dayana, y Yaime Oduardo Tamayo. «Aseguramiento de la Calidad en la línea de producto "Inspección de territorios".» La Habana, 2010.

Rodríguez Torres, Alexander. «Sistema de Información Geográfica de la UCI basado en tecnología OpenSource.» La Habana, 2005.

Araya Ramos, Kenyie. «desarrolloweb.com.» 16 de noviembre de 2006. http://www.desarrolloweb.com/articulos/introduccion-cross-site-scripting.html (último acceso: 1 de abril de 2011).

Manivesa, Cesar. «maestros del web.» 1997. http://www.maestrosdelweb.com/editorial/inyecsql (último acceso: 1 de abril de 2011).

Código de prácticas para la administración de la seguridad de la información. Norma ISO-IEC 17799.

RESOLUCION No. 127/2007. Anexo: REGLAMENTO DE SEGURIDAD PARA LAS TECNOLOGÍAS DE LA INFORMACIÓN.

GLOSARIO DE TÉRMINOS

Framework: Estructura de soporte definida, en la cual otro proyecto de software puede ser organizado y desarrollado (del Toro González y Zaldívar Picasso 2007).

Gestión: Comprende todas las actividades de una organización que implican el establecimiento de metas u objetivos, así como la evaluación de su desempeño y cumplimiento (Escobar Zaragoza 2009).

OpenSource: Término en inglés con el que se conoce al software distribuido y desarrollado libremente (Montes de Oca y Lemus Martínez 2008) .

OpenGIS: Creado en 1994 y cuyo fin es la definición de estándares abiertos e interoperables dentro de los Sistemas de Información Geográfica (SIG). Persigue acuerdos entre las diferentes empresas del sector que posibiliten la interoperación de sus sistemas de Geoprocesamiento y facilitar el intercambio de la información geográfica en beneficio de los usuarios (Rodríguez Puente 2007).

Proceso: Conjunto de tareas, actividades o acciones interrelacionadas entre sí que, a partir de una o varias entradas de información, materiales o de salidas de otros procesos, dan lugar a una o varias salidas también de materiales (productos) o información con un valor añadido (PRESSMAN 2005).

Proyecto: Esfuerzo temporal que se lleva a cabo para crear un producto, servicio o resultado único (PMI 2004).

Producto: Es cualquier bien, servicio o idea capaz de motivar y satisfacer a un comprador (Escobar Zaragoza 2009).

Procedimiento: Es la acción de proceder o el método de ejecutar algunas cosas. Se trata de una serie común de pasos definidos, que permiten realizar un trabajo de forma correcta (Pluma Clavel y Oduardo Tamayo 2010).

UML: Lenguaje Unificado de Modelado. Es el lenguaje de modelado de sistema de software más conocido en la actualidad (Rodríguez Torres 2005).

ANEXOS

Anexo1: Entrevista realizada a los líderes del proyecto de seguridad informática de la facultad2

1. ¿Cuentan con alguna solución que respalde la seguridad de las aplicaciones durante el desarrollo?
2. Si la respuesta anterior es afirmativa ¿En qué fases de desarrollo del producto la aplican?
3. ¿Creen que de esa forma mitigarían muchos de los riesgos que afectan el desarrollo de los productos?
4. ¿Lograrán que una vez finalizada la solución esta se encuentre libre de vulnerabilidades?

Anexo2: Entrevista realizada al Departamento Geoinformática

1. ¿Qué proyectos se encuentran inscritos en el Departamento Geoinformática?
2. ¿Cuál es el organigrama que se establece?
3. ¿Cuáles son las principales actividades que tienen definidas?
4. ¿Cuentan con el documento "**Gestión de Riesgo"**?
5. ¿Por quién/quiénes fue realizado?
6. ¿Son controladas las actividades que enmarcan este documento? ¿Por quién?
7. El documento de "**Arquitectura de Software"** ¿Lo tienen elaborado?
8. ¿Por quién/quiénes fue desarrollado?
9. ¿Quiénes son los máximos responsables de controlar las actividades que enmarcan este documento?
10. Al iniciar el proyecto ¿Ustedes tienen en cuenta la seguridad informática en el desarrollo de los productos?
11. ¿Cuentan con el documento **"Plan de Seguridad Informática y Contingencia"**?

    Si la respuesta es positiva
12. ¿Por quién/quiénes fue realizado?
13. ¿Quién es el responsable de controlar las actividades que enmarcan este documento?

    Si la respuesta es negativa
14. ¿Cuáles son los requisitos no funcionales de seguridad que tienen en cuenta?
15. Una vez inicializado el proyecto ¿Quién/quiénes pueden acceder a la información necesaria?

16. ¿Tienen establecido algún software para el control de acceso a la información?
17. ¿Quién da los permisos?
18. ¿Cada qué tiempo salvan la información? ¿Diariamente?, ¿Semanalmente?, ¿Mensualmente?, ¿Anualmente?
19. ¿Quién revisa este procedimiento?
20. ¿Tienen en cuenta la experiencia de líderes y desarrolladores?
21. ¿Se capacita al personal?
22. ¿Creen que sea suficiente con los cursos actuales?
23. ¿Durante el desarrollo y terminación del producto hay algún chequeo o control de la seguridad alcanzada o el cumplimento de los Requerimientos No Funcionales (RNF) de Seguridad definidos al inicio?

    Si la respuesta es el equipo de calidad
24. ¿Cómo desarrollan esta tarea?

Anexo3: Evaluación de expertos

1. ¿Conoce usted las distintas amenazas y vulnerabilidades a las que se enfrentan actualmente los productos realizados en el Departamento Geoinformática?

   Sí___ No___ ¿Por qué?
2. ¿Con este procedimiento cree usted que serán mitigadas muchas de las amenazas y vulnerabilidades existentes en los productos realizados en el Departamento Geoinformática?

   Sí___ No___ ¿Por qué?
3. Al aplicar el procedimiento ¿Cree que serían eliminados muchos de los riesgos presentes en los proyectos?

   Sí___ No___ ¿Por qué?
4. ¿Con este procedimiento cree Ud. que los productos realizados cumplirán con la confidencialidad requerida?

   Sí___ No___ ¿Por qué?
5. ¿Con este procedimiento cree Ud. que los productos realizados cumplirán con la integridad requerida?

   Sí___ No___ ¿Por qué?

6. ¿Con este procedimiento cree Ud. que los productos realizados cumplirán con la disponibilidad requerida?
   Sí___ No___ ¿Por qué?
7. En una escala del 1 al 5 confiera un valor a la propuesta según el criterio siguiente:
   ___ Satisfacción de los clientes.
   ___ Cumplimiento de los requerimientos.
   ___ Contribución a bajo nivel de incidentes informáticos.
   ___ Disminución de riesgos.
   ___ Control y organización de las actividades dentro del proyecto que tributan a la seguridad.

Como las respuestas a las preguntas podía ser de afirmación o negación se le otorgó en cada caso valor 1 a la respuesta (Sí) y valor 0 a la respuesta (No).

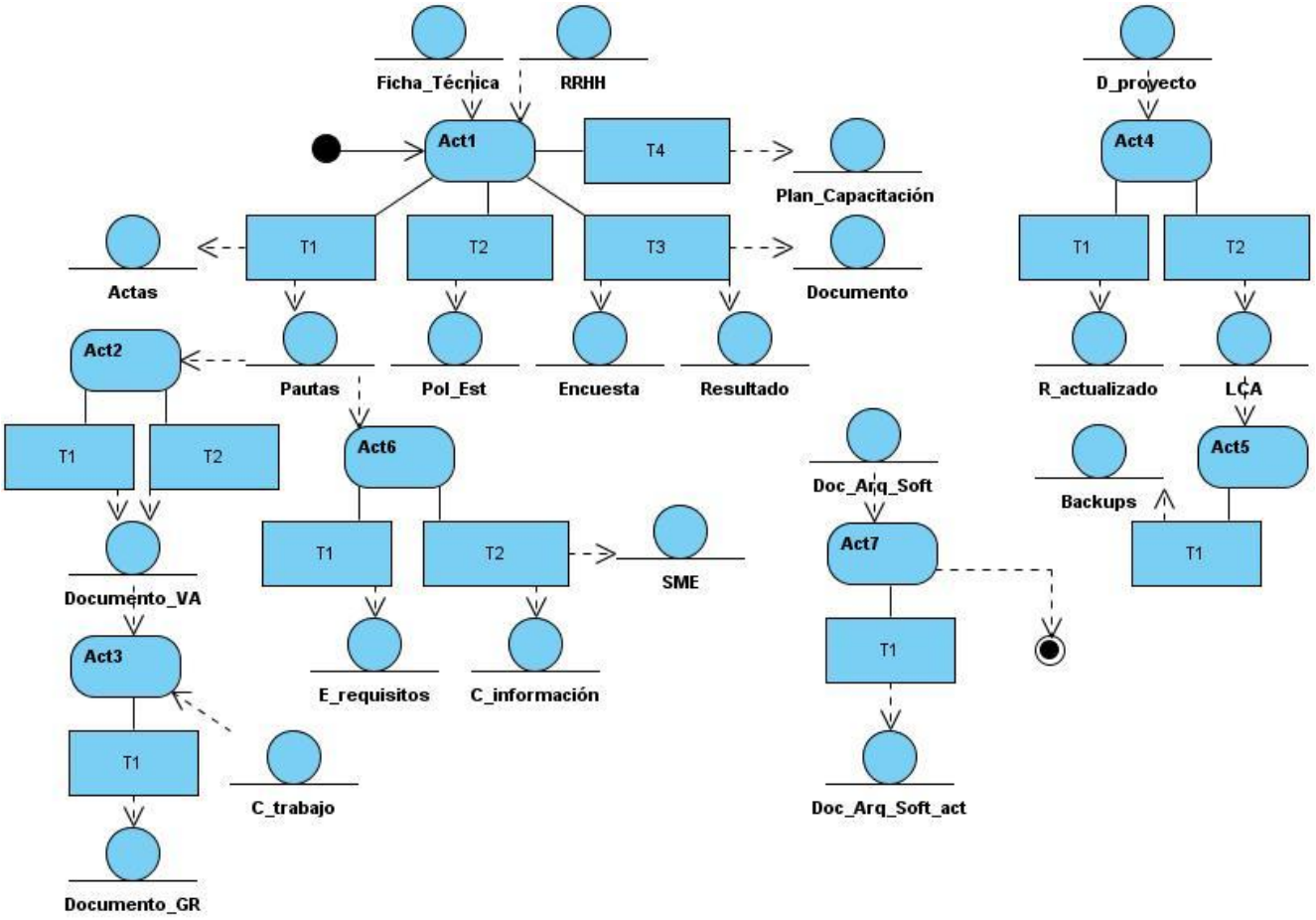
Los resultados obtenidos para cada pregunta formulada aparecen en la siguiente tabla:

Preguntas	**Expertos**							**%**
	1	**2**	**3**	**4**	**5**	**6**	**7**	
1	x		x	x			x	57.1
2	x	x	x	x	x	x	x	100
3	x	x	x	x	x	x	x	100
4	x	x	x	x	x		x	85.7
5	x	x	x	x	x	x	x	100
6	x	x	x	x		x	x	85.7
7	3	4	4	4	5	3	4	77.1
	2	4	5	5	5	4	4	82.9
	3	4	4	4	4	3	4	74.3
	3	3	4	4	5	3	5	77.1
	4	4	4	4	4	5	4	82.9

A partir de los resultados obtenidos se puede concluir que:

1. La proporción de personas que aseguran conocer las distintas amenazas y vulnerabilidades a las que se enfrentan actualmente los productos realizados en el Departamento Geoinformática es de un 57.1%, lo que sugiere pensar que se debe seguir trabajando en este aspecto.
2. El 100% de los miembros del panel de expertos reconocen que con la aplicación del procedimiento serán mitigadas muchas de las amenazas y vulnerabilidades existentes en los productos realizados en el Departamento Geoinformática.
3. El 100% de los encuestados reconoce que al aplicar el procedimiento serían eliminados muchos de los riesgos presentes en los proyectos.
4. El porcentaje de personas que afirman que con este procedimiento los productos realizados cumplirán con la confidencialidad requerida es de un 85.7%.
5. El 100% de las personas encuestadas afirman que con este procedimiento los productos realizados cumplirán con la integridad requerida.
6. La proporción de personas que aseguran que con este procedimiento los productos realizados cumplirán con la disponibilidad requerida es de un 85.7%.
7. El porcentaje del panel de expertos que asegura que la propuesta proporcionará contribución a bajo nivel de incidentes informáticos es del 74.3%. El 77.1% reconoce que la propuesta dará satisfacción al cliente y disminuirá los riesgos. El 82.9% afirma que se dará cumplimiento a los requerimientos y habrá un control y organización de las actividades dentro del proyecto que tributan a la seguridad.

Anexo4: Relación entre las tareas de la fase inicial del procedimiento



Anexo5: Relación entre las actividades de cada una de las fases del procedimiento