Revisión de Estado del Arte del Ciclo de Vida de Desarrollo de Software Seguro con la Metodología SCRUM

Becerra Paola, Sanjuan Marilyn
{b.paola[1], s.marilyn[2]}@unisimon.edu.co

Resumen- El desarrollo de software seguro es un asunto de alta importancia en las compañías, debido a que la mayoría de ellas dependen altamente de sus aplicaciones para su operación normal. Es por esto que se hace necesario implementar efectivamente metodologías de desarrollo seguro que sean aplicadas en cada fase del ciclo de vida: requisitos, diseño, desarrollo y pruebas. Es importante tener presente la seguridad desde las etapas más tempranas del proceso de desarrollo y no dejarla en un segundo plano. Además, se requiere investigar el estado del arte en desarrollo de aplicaciones seguras y así escoger la metodología SCRUM de cada aplicación y los requisitos de los interesados en el producto final. El objetivo de este artículo en cada etapa incrementar seguridad y herramientas existentes que se puedan implementar, añadiendo seguridad en toda la aplicación y desarrollando no sólo software de alta calidad sino también de alta seguridad.

Palabras clave- seguridad en cada una de las etapas del ciclo de vida de software, metodología Scrum.

Abstract— The development of secure software is a matter of high importance to companies because most of them highly dependent on their applications for normal operation. That is why it is necessary to effectively implement secure development methodologies that are applied at each stage of the life cycle: requirements, design, development and testing. It is important to safety from the earliest stages of the development process and not leave it in the background. In addition, it is required to investigate the state of the art in secure application development and SCRUM methodologies and choose every application and requirements of stakeholders in the final product. The aim of this article at each stage to increase security and existing tools that can be implemented by adding security throughout the application and software development not only high quality but also highly secure.

Keywords- Security software, agile methodology, software, scrum.

I. INTRODUCCION

La seguridad de los sistemas es un tema que ha despertado amplio interés desde hace mucho tiempo, porque no sólo es necesario tener aplicaciones de alta calidad sino también que tengan seguridad. Debido a que actualmente la mayor parte de ataques están enfocados a las aplicaciones y teniendo en cuenta que la mayoría de desarrolladores no tienen las habilidades y el conocimiento necesario para desarrollar código seguro [2], es necesario que las empresas apliquen metodologías y herramientas que permitan desarrollar aplicaciones seguras que cumplan con las exigencias de seguridad en este tiempo.

De acuerdo con Allen et al. [1], el Ciclo de Vida de Desarrollo de Software CDVS se puede definir como un proceso iterativo en el cual se identifican cuatro etapas principales: Requisitos, Diseño, Desarrollo y Pruebas. Existe la metodología SCRUM el cual está enfocado a la gestión de los procesos de desarrollo y hacen variadas actividades de análisis, diseño, desarrollo, implementación. Scrum suele vincularse al desarrollo de software, pero su aplicación es adecuada dentro de las especialidades más variadas

Se realizan ciclos (o iteraciones) de duración fija llamadas Sprints. Se recomienda que la duración de un Sprint sea de 2, 3 o 4 semanas, Durante el Sprint el objetivo del equipo es generar un incremento visible, utilizable, entregable.

Lo primero que se debe hacer para comenzar a implementar seguridad con la metodología SCRUM en las aplicaciones es identificar cuáles son los activos de información que se deben proteger, cómo protegerlos, cuáles son las vulnerabilidades de los elementos que interactúan con la información y como mitigarlas, al punto de reducirlas a un nivel de riesgo aceptable mediante un proceso iterativo que analice profundamente los riesgos durante todo el CVDS. También es muy importante identificar patrones de ataque en todas las fases y almacenarlos en una base de conocimiento que permita prevenir futuros ataques en otras aplicaciones [1].

La seguridad debe estar implícita desde la misma concepción del software, porque es un error dejarla para etapas posteriores del desarrollo. Se debe comenzar de los requisitos, que no deben ser simples listas de chequeos de implementación de controles, como firewalls y antivirus, sino que deben ir más enfocados a la protección de los activos críticos [1]. Para la aplicación específica que se está desarrollando se debe tener en cuenta la perspectiva del agresor. En este artículo, con el objetivo de lograr una buena aproximación al desarrollo de requisitos seguros.

II. MARCOS DE TRABAJO DE SEGURIDAD

El marco de trabajo cubre un amplio rango de intereses, considerando una gran variedad de situaciones en materia de seguridad. Cabe mencionar que la información es el activo más valioso que poseen todas las organizaciones, en donde se establecen criterios y normas para protegerse.

A continuación se detallan marcos de trabajo de la seguridad de la información, por parte de la industria y los organismos de estandarización.

- Estándares ISO/IEC 27000 [4]:

Es un conjunto de 6 estándares internacionales desarrollados por ISO (International Organization for Standardization) e IEC (International Electrotechnical Commission), que proporcionan un marco de gestión de la seguridad de la información utilizable por cualquier tipo de organización, pública o privada, grande o pequeña [5]. También se encuentra el ISO 27001 [6], donde se especifica los requisitos necesarios para establecer, implantar, mantener y mejorar un Sistema de Gestión de Seguridad de la Información (SGSI), además la norma ISO 27002, define un código de buenas prácticas para la gestión de la seguridad por las organizaciones.

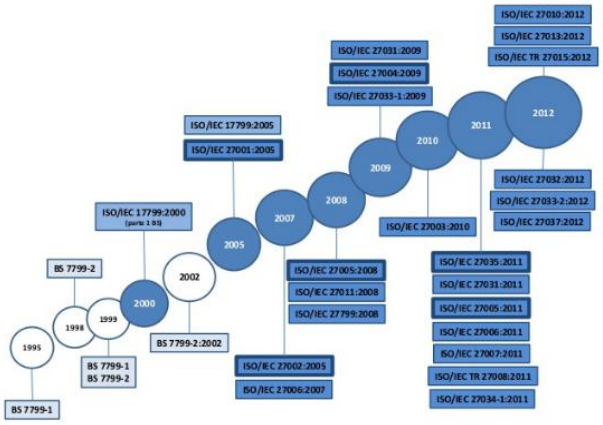

Fig. 1 Familia ISO 27000 [7]

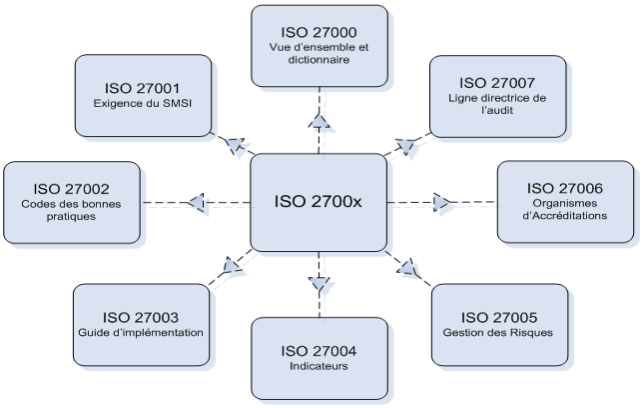

Fig. 2 ISO 2700 [8]

- ISO/IEC 15408:2009 Common Criteria Framework (CCF):

Estándar que propone un marco de trabajo común para la realización de las pruebas y evaluación de la seguridad [9].

- ISO/IEC 21827:2008 SSE-CMM:

(Systems Security Engineering Capability Maturity Model) en el cual se define un modelo de proceso de madurez para mejorar y evaluar la capacidad de la ingeniería de la seguridad de una organización [10].

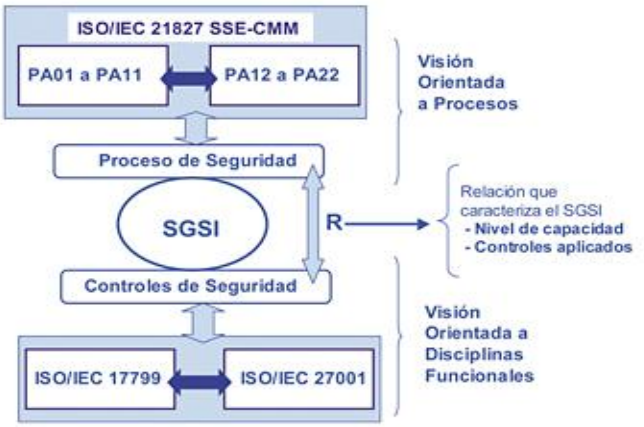

Fig. 3 ISO/IEC 21827 [11]

- IEEE P1074-2005:

Estándar internacional que trata la Gestión del Software y sus ciclos de vida. Centra su aproximación en aplicar una apropiada priorización de la seguridad en los proyectos de software así como durante la construcción de los controles de seguridad en los productos [12].

- COBIT 5.0:

(Control Objectives for Information and related technology), es el único marco de negocio para el gobierno y la gestión de TI de la empresa. Esta versión evolutiva incorpora las últimas ideas en técnicas de gestión, y proporciona principios globalmente aceptados, prácticas, herramientas analíticas y modelos para ayudar a aumentar la confianza y el valor de los sistemas de información [13].

Fig.42 COBIT 5 [14]

III. METODOLOGIA SCRUM

Scrum: define un marco para gestión de proyectos, el desarrollo de software se realiza mediante iteraciones, denominadas sprints, con una duración de 30 días. El resultado de cada sprint es un incremento ejecutable que se muestra al cliente. La segunda característica importante son las reuniones a lo largo proyecto, entre ellas destaca la reunión diaria de 15 Minutos del equipo de desarrollo para coordinación e integración.

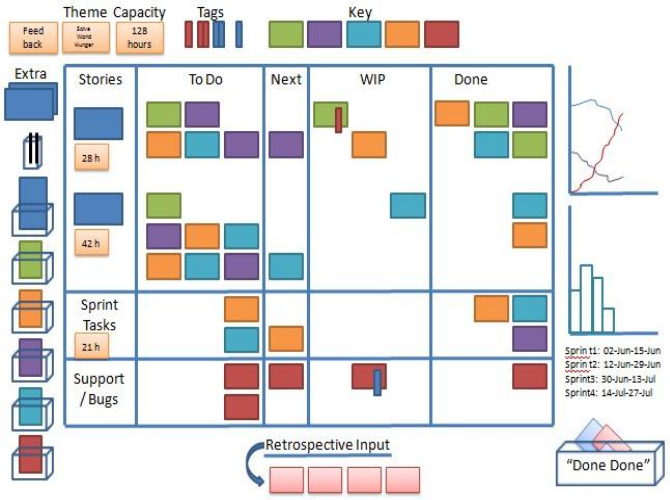

Figura 1. Planificación Metodología Scrum.

En scrum se llevan a cabo una planificación de iteraciones el primer día se divide en dos partes, la selección de requisitos que debe durar 4 horas máximo en el el equipo le pregunta al cliente las dudas que tiene y se selecciona los requisitos mas prioritarios que se compromete a completar en la iteración de manera que puedan ser entregados por solicitud del cliente, ya la otra parte es la planificación de la iteración en esta el equipo elabora las listas de tareas de la iteración necesarias para desarrollar los requisitos a que se a comprometido a cumplir, es característico en la metodología scrum que cada miembro del equipo inspeccione al resto del grupo además en las reuniones cada miembro del grupo responde tres preguntas como ¿Qué he hecho desde la última reunión ?¿qué voy a hacer a partir de ahora? y ¿Qué impedimentos voy a tener?. Durante la iteración el facilitador se encarga de que el equipo pueda cumplir con sus compromisos y de que no caiga la productividad.

El último de la iteración se realiza la reunión de revisión de la iteración que igual que al principio tiene 2 partes demostración y retrospectiva, en la demostración el equipo presenta al cliente los requisitos completados en la iteración en forma de incremento de producto preparado para ser entregado con el mínimo esfuerzo, en retrospectiva el equipo analiza como ha sido su manera de trabajar y cuáles son los problemas que pueden surgir en el futuro que le impida seguir avanzando adecuadamente, podemos decir entonces que las principales ventajas de utilizar scrum son resultados anticipados, mitigación sistemática de los riesgos , flexibilidad y adaptación y gestión sistemática del retorno de inversión.

ANALISIS DE RIESGO EN EL DESARROLLO DE SOFTWARE SEGURO

Después de haber estudiado varias metodologías y estándares, relacionados con la seguridad en el desarrollo del software, se evidencia que el análisis de riesgos, con la identificación de activos del sistema es considerado tarea clave en el proceso de establecimiento de la seguridad y paso necesario para su gestión [15].

A continuación se describen algunas de las metodologías específicas del Análisis de Riesgos, las cuales han sido adoptadas como estándares en sus países respectivos;

- CRAMM (CCTA Risk Analysis and Management method) [16], es el método de análisis y Gestión de Riesgos del CTTA (Carmarthenshire College of Technology and Art), adoptado por el Gobierno del Reino Unido, el cual es conforme al estándar ISO 27001.
- MAGERIT (Método de Análisis y Gestión del Riesgo del Ministerio de Administraciones Públicas [17]), es el método de Análisis y Gestión del riesgo definido por la Administración Pública Española, actualizada a la versión 2.0 y conforme al estándar ISO 15408, Common Criteria.
- OCTAVE (Operationally Critical Threat, Asset and Vulnerability Evaluation), [18, 19]). Método de análisis de riesgos desarrollado por el Carnegie Mellon – Software Engineering Institute- aceptado en USA.

IV. CICLO DE VIDA DEL SOFTWARE

Es un marco de referencia que contiene los procesos, las actividades y las tareas involucradas en el desarrollo, la explotación y el mantenimiento de un producto de software, abarcando la vida del sistema desde la definición de los requisitos hasta la finalización de su uso [19]. El ciclo de Vida según:

Fábregas [20]:

1. Requerimientos
2. Análisis / Diseño
3. Construcción
4. Pruebas
5. Producción/Mantenimiento

Pressman [21]:

1. Análisis
2. Diseño
3. Codificación
4. Prueba
5. Mantenimiento

Senn [22]:

1. Investigación Preliminar
2. Determ. De Requerimientos
3. Desarrollo del Software
4. Prueba del Sistema
5. Implantación y Evaluación

Fig. 4 Ciclo de Vida del Software [23]

Ventajas:

• Permite una construcción rápida del sistema.
• Es útil para sistemas de un tamaño muy reducido, que no requiera más de dos o tres programadores y que no requiera un mantenimiento posterior.
• No pierde tiempo en las etapas de planificación, control de calidad y documentación.
• Cualquiera, sin preparación técnica, lo puede utilizar.

Desventajas:
• Carece de cualquier control y Gestión del Proceso
• No dispone de las fases necesarias en todo proyecto de software: especificaciones, diseño...
• Se dificulta la corrección de errores y el mantenimiento, al carecer de una documentación del proceso adecuada.
• No proporciona medios de evaluación ni de prevención de riesgos.
• Resulta peligroso para proyectos grandes.

V. CONCLUSIONES

Se presentó un marco de trabajo para el desarrollo de software con la metodología SCRUM para, identificación de riesgos y vulnerabilidades, desde las fases tempranas, las cuales permiten analizar los riesgos y las vulnerabilidades, que se puedan llegar a presentar, sin que afecte el producto final, permitiendo obtener información importante, para la mitigación del riesgo. hay que tener en cuenta que toda metodología debe ser adaptada al contexto del proyecto, La metodología SCRUM ofrecen una solución casi a medida para una gran cantidad de proyectos que tienen estas características. Una de las cualidades más destacables en una metodología ágil es su sencillez, tanto en su aprendizaje como en su aplicación, reduciéndose así los costos de implantación en un equipo de desarrollo, podemos decir que es la solución más eficaz para desarrollar productos en corto tiempo, aunque existen varios libros sobre metodologías agiles mi grupo de trabajo y yo llegamos a la conclusión que Scrum es la más sencilla de aplicar cuando queremos entregar un producto en cortos plazos

REFERENCIAS

[1] J. H. Allen et al. “Software Security Engineering: A guide for Project Managers”. Addison-Wesley. 2008.
[2] M. Brown & A. Paller. “Secure software development:
[3] Why the development world awoke to the challenge”. Information Security Technical Report, Vol. 13, No. 1, pp. 40-43, 2008.
[4] ISO/IEC27000. ISO/IEC 27000 Information technology http://www.27000.org/. 2009 [Cited 2011].
[5] Agustín López Neira, Javier Spohr http://www.iso27000.es/index.html
[6] ISO27001, ISO/IEC 27001 Information technology –Security techniques – Information security management systems - Requirements. 2005.
[7] http://image.slidesharecdn.com/iso27000evolucinenero 2013pb-130131071629-phpapp02/95/slide-1-638.jpg?cb=1359638399
[8] http://4.bp.blogspot.com/_t2DdWClfIY/Uzs5gGE3vCI/AAAAAAAAA YE/RRhvYldZTyY/s72c/iso2700x_familly.png
[9] Joaquín Lasheras Velasco “Marco de Trabajo deRepresentación y Reusos de Requisitos de Seguridad”.
[10] Kotonya, G. and I. Sommerville, RequirementsEngineering. Processes and Techniques. 1998, JohnWiley & Sons.
[11] http://www.revistasic.com/revista75/imagenes75/articulo B.jpg
[12] IEEE, IEEE P1704 Standard for Developing Software Project Life Cycle Processes. 1997.
[13] http://www.isaca.org/cobit/ pages/ default.aspx
[14] COBIT5 http://www.itsmportal.com/system/files/vijf.png
[15] Magerit. Methodology for Information Systems Risk Analysis and Management. V 2.0,http://www.csi.map.es/csi/pg5m20.htm. 2006 [cited 2011].
[16] Cramm. United Kingdom Central Computer and Telecommunication Agency. CCTA Risk Analysis and Management Method: User Manual, ver. 5.2. HMSO. http://www.cramm.com/. 2005 [cited 2011].
[17] Magerit. Methodology for Information Systems Risk Analysis and Management. V 2.0, http://www.csi.map.es/csi/pg5m20.htm. 2006 [cited 2011].
[18] Octave. Operationally Critical Threat, Asset and Vulnerability Evaluation (OCTAVE). V 2.0, Carnegie Mellon - Software Engineering Institute,http://www.cert.org/octave/. 1999 [cited 2011].
[19] C. Alberts. and A. Dorofee, Managing information
[20] security risks: The OCTAVE (SM) approach. 2002, Boston, Addison Wesley
[21] Llorens, Fábregas, Roger,Pressman http://www.taringa.net/posts/ciencia-educacion/12689892/Ingenieria-del-Software---Roger-Pressman-6ta-Edicion.html
[22] James A. Senn, http://www.comusoft.co /ciclos-de-vida-proceso-de-desarrollo-del-software