Revista Cubana de Ciencias Informáticas
Vol. 12, No. 1, Enero-Marzo, 2018
ISSN: 2227-1899 | RNPS: 2301
Pág. 117-129
http://rcci.uci.cu

Tipo de artículo: Artículo original
Temática: Seguridad Informática
Recibido: 11/12/2017 | Aceptado: 22/01/2018

GESTIÓN DE PRUEBAS ORIENTADAS A RIESGOS EN SFTWARE CRÍTICOS CNS/ATM

MANAGEMNT OF RISK ORIENTED TEST ON CRITICAL SOFTWARE CNS/ATM

Guillermo Brito Acuña*

Empresa Cubana de Aeropuerto y Servicios Aeronáuticos (ECASA). Avenida Independencia y Final Boyeros, La Habana. guillermo.brito@aeronav.avianet.cu

***Autor para correspondencia**: guillermo.brito@aeronav.avianet.cu

Resumen

Las infraestructuras críticas y los softwares de seguridad controlan cada vez más sistemas importantes para la economía o el correcto funcionamiento de los gobiernos. El aumento de la dependencia tecnológica hace que aumenten los riesgos asociados a ellos. Para pensar en software seguro se debe identificar y evaluar los peligros potenciales que podrían afectarlo y ocasionar que falle. El presente documento integra una explicación de los métodos de clasificación de riesgos en sistemas de comunicación, navegación, vigilancia y gestión de tráfico aéreo CNS/ATM. Estos riesgos son evaluados mediante la valoración de posibles manifestaciones causadas ante la vulneración de datos específicos. Para ello se diseñó un sistema de gestión de pruebas basados en la criticidad asociada a los riesgos identificados. Se muestran también los resultados obtenidos en la aplicación de esta propuesta en software asociados con variables meteorológicas de aeródromos y sistemas de publicación de datos del servicio de información aeronáutico cubano.

Palabras clave: gestión de riesgos, infraestructuras críticas, pruebas orientadas a riesgos

Abstract

Critical infrastructure and security software control today more and more relevant systems for the economy and the proper functioning of governments. By the increase of the technological dependence, increase the risks related to them. To think on secure software requires identifying and assessing the potential hazards that might affect it and cause it to fail. This document brings together an explanation of the methods for risks classification on

Revista Cubana de Ciencias Informáticas
Vol. 12, No. 1, Enero-Marzo, 2018
ISSN: 2227-1899 | RNPS: 2301
Pág. 117-129
http://rcci.uci.cu

Communication, Navigation, Surveillance and Air Traffic Management CNS/ATM systems. These risks are evaluated through the assessment of possible appearance caused by the damage of specific data. For this purpose was designed a tests management system based on the relevance associated to the identified risks. The results obtained by the application of this proposal on software related to meteorological variables in aerodromes und data publication systems from the Cuban aeronautical information service are also shown.

Keywords: *critical infrastructure, risk management, risk oriented tests*

Introducción

Por infraestructura crítica se entiende aquellas instalaciones, redes y tecnologías, cuya interrupción o destrucción puede tener una repercusión importante en la salud, la economía o el eficaz funcionamiento de los gobiernos**.** La criticidad de estos sistemas reside cada vez más en los softwares que contienen (Eterovic & Donadello, 2015) y la función que desempeñan. Las aplicaciones que brindan servicios vitales en estas infraestructuras se conocen como "software crítico" y son considerados de alta fiabilidad y seguridad (Matei, 2015). En estos sistemas se evidencia que las amenazas informáticas no sólo comprometen el mundo digital, sino que también son un riesgo mayor, para el mundo físico (Prieto, 2014). La seguridad del software se relaciona con la calidad, es una actividad del aseguramiento que se centra en la identificación y evaluación de los peligros potenciales que podrían afectarlo y ocasionar que falle (Pressman, 2010).

La sociedad se encuentra vinculada innegablemente a estas tecnologías; sin embargo, un uso tan amplio hace que existan grandes riesgos en cada una de las aplicaciones tecnológicas. Los riesgos evolucionan y aumentan con la tendencia al uso de tecnologías (Norma Martínez & Porcelli, 2015)**.** En una organización se pierden aproximadamente $6.6 millones de dólares por cada brecha de seguridad (Norton, 2012). El mayor riesgo de seguridad para una empresa no viene de competidores o espías, sino de los mismos empleados (CeBIT, 2015), que cometen errores, indiscreciones o descuidos. Análisis sobre ingeniería de software sugieren que por cada 1000 líneas de código, se cometen un promedio de 1 a 5 errores (Cid, 2014).

La clave de un software seguro es el proceso de desarrollo utilizado. En el proceso es donde se produce el producto que pueda resistir o sostenerse ante ataques no anticipados, recuperarse rápidamente y mitigar el daño causado por los ataques que no pueden ser eliminados o resistidos (Brito, 2013). Para la programación de software críticos de alta

Revista Cubana de Ciencias Informáticas
Vol. 12, No. 1, Enero-Marzo, 2018
ISSN: 2227-1899 | RNPS: 2301
Pág. 117-129
http://rcci.uci.cu

confiabilidad es imprescindible seguir una metodología en la que se defina cada paso, se utilicen una serie de herramientas de análisis, arquitecturas especiales, etc., para obtener diseños realmente confiables con unas características de coste y tiempo aceptables, y donde el esfuerzo se dirija tanto al hardware como al software **(Pérez, 2006)**, sin dejar de lado el entorno en que se desarrolla.

Materiales y Métodos

En el campo de la aeronáutica, organismos internacionales como Canso (Canso, 2014), EUROCAE (EUROCAE, 2012) y OACIS (OACI, 2010) se han pronunciado sobre estos temas; proyectando un enfoque integral de seguridad en el desarrollo de proyectos. Estos documentos recomiendan la clasificación de los elementos de la arquitectura por riesgos y una serie de medidas para la protección ante estos. Empiezan con una evaluación de la seguridad (safety) del sistema. El efecto de un fallo sobre el funcionamiento es estudiado y analizado, clasificándolo en niveles de criticidad, llamados (DAL, *Development Assurance Level*) (Fernández, 2013).

Nivel A	Catastrófico, Muchas lesiones mortales y/o gran pérdida de valores
Nivel B	Riesgo / Severo: Posibles lesiones mortales a un pequeño grupo y daños estructurales
Nivel C	Mayor: perjudica la eficiencia, molestias o posibles lesiones a los ocupantes
Nivel D	Menores: reducción de la seguridad, dentro de las capacidades del servicio
Nivel E	Sin Efecto: no afecta a la seguridad

La gestión de riesgos es el proceso más difundido para gestionar la seguridad. Permite comprender la naturaleza del riesgo y determinar el nivel del mismo. Proporciona las bases para la evaluación del riesgo y para tomar las decisiones relativas su tratamiento (ISO-NC 31000, 2015). Esto permite conocer el nivel necesario de profundidad en el análisis del elemento considerado, así como el nivel de validación y verificación (Escribano, 2013). Muchos de los defectos relacionados con la seguridad en software se pueden evitar si los desarrolladores estuvieran mejor equipados para reconocer las implicaciones de su diseño y de las posibilidades de implementación (Pressman, 2010).

Comienza con la identificación de planes y estándares de desarrollo. En forma paralela a la especificación de requisitos del sistema, nótese que hablamos de sistema y no software. Esto pertenece a la etapa de planificación y es donde se establecen los planes de certificación, desarrollo, configuración y calidad, formando parte del documento "Especificación de Software y Sistemas". Estos requisitos deben ser trazables para las pruebas de integración con el ecosistema en cuestión y las pruebas de mantenimiento. Entrando en la Especificación de Requisitos, se capturan los mismos y se determinan los estándares de diseño, requisitos y configuración, de interfaz, funcionamiento, operacionales y seguridad. Comenzado la fase de diseño a alto y bajo nivel, se determina la arquitectura, se recomienda la utilización de métodos

formales, lo cual permitirá realizar pruebas más exhaustivas. Siendo el proceso trazable desde los requisitos hasta el diseño a bajo nivel. Paralelo a estas etapas se diseñan las pruebas de software, integración y componentes y se evalúa el porciento requerido de cobertura de código, según el grado de criticidad del software en cuestión; se crea el plan de validación y el registro de resultados de las mismas. Tanto la especificación de requisitos, como el diseño de componentes deben permitir trazabilidad en las pruebas de caso de uso y las unitarias, respectivamente. Ante cualquier situación siempre es posible rediseñar o revisar los requisitos.

En la fase de codificación no existe un documento indicado por la norma, se recomienda la utilización de un registro de (errores) bugs, los cuales permitirá al equipo desarrollador identificar vulnerabilidades en cuanto a la estrategia de programación. En esta etapa el diseño de pruebas tiene un carácter informal, para aumentar la efectividad de las pruebas, se recomienda un sistema de comprobación en pares, ya que los proyectos software que la incorporan, tienden a estar mejor diseñados, a contener menos errores, y por tanto, a ser más robustos (Vector, 2016). Esta etapa debe mantener un detallado control de cambios, un registro de incidencias y una línea base donde se detalla la adición de componentes. Además de mantener el registro de verificación se obtiene como resultado código fuente y propuesta de ejecutable. La fase de verificación será explicada detalladamente mediante el procedimiento de Gestión de Pruebas de Software y Sistemas en el acápite

Determinación de Niveles de Criticidad

La determinación de los niveles de criticidad de los software y sistemas se realiza en la etapa de planificación del producto, luego de realizada la captura de requisitos. Para ello es necesaria la presencia de un equipo multidisciplinario, donde estén representados todos los sectores. Habitualmente estos equipos están compuestos por representantes y especialistas de las direcciones operacionales donde se involucra el producto, especialistas en las diferentes ramas de comunicaciones asociadas, dígase: seguridad, redes, equipos especiales u otros y presencia del equipo de desarrollo.

Cada experto evalúa dentro de su campo las variables más críticas para el dominio en cuestión. Teniendo en consideración el cúmulo de variables críticas se someten al criterio de expertos, sobre el peor escenario posible donde las fallas se ocurran de manera simultánea, sin implicar redundancias u otras medidas que se estén proyectando.

A partir del escenario trazado se clasifica según nivel de riesgo. Es importante notar que si las variables de riesgo pertenecen a otro programa o módulo a desarrollar se puede clasificar el módulo de forma independiente a las demás

tareas. Cada nivel de riesgo impone una serie de condiciones de ejecución durante el ciclo de vida. Lo importante es gestionar la mayor cantidad de riesgos posibles. Los niveles de riesgo utilizados son:

Nivel 1: Riesgo Ninguno. Cuando la perdida de los programas o datos generados por ellos no tiene ninguna repercusión para el dominio en cuestión. Estos sistemas no procesan información con variables críticas.

Nivel 2: Riesgo Menor. Cuando la pérdida de la integridad o disponibilidad de la información o los programas reducen los márgenes de seguridad de la infraestructura, las variables críticas y la transmisión de la información no dependen de esta aplicación, por lo que la afectación puede ser solucionada sin afectaciones perceptibles para el dominio.

Nivel 3: Riesgo Mayor. Cuando la pérdida de una variable el sistema o sus partes, puede causar daños al dominio o a los que se benefician con estos servicios. Al punto de sufrir altos niveles de estrés o lesiones.

Nivel 4: Riesgo Severo: Cuando la perdida de la aplicación y los datos puede producir daños en el dominio o a los que se benefician con estos servicios, Estos daños pueden incluir grandiosos daños materiales, de imagen e incluso muerte a un número limitado de personas.

Nivel 5: Riesgo Catastrófico: Cuando la afectación de la integridad o disponibilidad del sistema pone en riego de catástrofe a los implicados con el dominio. Riesgo de pérdidas de muchas vidas humanas y la destrucción total o parcial de los prestadores y receptores del servicio.

Atendiendo a lo expuesto anteriormente, queda claro la importancia que recubre el tratamiento de los proyectos que se pretendan aplicar en infraestructuras críticas. Atendiendo a ello, a la compatibilización con los requerimientos de las normas internacionales e indicaciones de la ISO 25000, se estableció la siguiente estrategia de gestión de pruebas. En ella se establecen un grupo de pruebas, validaciones y verificaciones dependientes del nivel de criticidad determinado para el producto. Es importante antes de comenzar a explicar el procedimiento, establecer que estas pruebas son acumulativas, por lo que, existe la obligación de realizar todas las pruebas establecidas para los niveles de criticidad inferiores. Para su mejor comprensión se organiza en la tabla 1.

Tabla 1. Organización de las Pruebas de Calidad en dependencia del nivel de criticidad del Sistema.

Calidad / Riesgo	**Ninguno**	**Menor**	**Mayor**	**Severo**	**Catastrófico**
PPQA	Revisión del formato procedimentado	Detección de Requisitos Saturados	Revisión de calidad documental externa		
	Detección de Requisitos	Detección de Requisitos muy	La revisión es realizada por pares		

	Duplicados	específicos			
	Detección de Requisitos sin límites claros	El test manager participa en el diseño	Revisión de la Línea Base		
	Requisitos con Problemas de Ortografía				
	Detección de Requisitos Ambiguos				
	Requisitos con Problemas de Redacción				
	Límites del Producto Establecidos				
Seguridad					
Confidencialidad.	Penetración en múltiples superficies de ataques	Información de la mensajería			
Integridad	Encriptación	Vulnerabilidad de productos en los que se construye	integridad de datos I/O de sistemas externos		
	Análisis de Valores Límites	Validación de Datos en interfaces automáticas	Sin funciones inseguras		
No repudio	Pruebas de Acceso	Configuración de permisos rw en ficheros y DB	Trazabilidad de acciones de modulo	Trazabilidad de acciones de objeto	
		Sincronismo en el tiempo			
Responsabilidad	Escalamiento e Inyecciones	Trazabilidad con los requisitos			
Fiabilidad					
Madurez	Integración		Requiere cobertura de código completo	Cobertura del código a nivel de decisión	Cobertura del código a nivel de condición
Disponibilidad	Unitarias	Garantía de redundancia simple	Garantías con redundancias múltiples		
Tolerancia a fallos	Accesos desconectados	Desconexión de módulos	Desconexión múltiples	Desconexión con estrés	Desconexiones aleatorias con estrés y datos incompletos
Capacidad de recuperación	Emisión de alertas	Funcionamiento luego de desconexión	Análisis de Código para tratamiento de fallo		

Autenticidad		Trazabilidad de acciones de usuario			
Adecuación Funcional					
Completitud funcional	Verificación del cumplimiento de los requisitos		Revisión de los requisitos formalmente a bajo nivel		
Corrección funcional	Permite adecuada gestión de sus datos	Verificación de los resultados esperados	Requiere integración continua		
Pertinencia funcional	Listar funcionalidades adicionadas	Cumplimiento con las consideraciones de Organismos	Cumplir con estilo de programación internacional		
Eficiencia de desempeño					
Utilización de recursos	Rendimiento por aplicación o servicio	Estrés			
Capacidad	Carga por aplicación o servicio	Volumen por aplicación o servicio			
Compatibilidad					
Coexistencia	Instalación de otros software como el antivirus	Probar que no se afecta otros sistemas por este	Funcionalidad e instalación con otro software crítico		
Interoperabilidad		Integración a Sistemas	Comportamiento errático de datos externos		
Usabilidad					
Capacidad para reconocer su adecuación	Pruebas de aceptación piloto				
Estética de la interfaz de usuario	Lista de verificación de usabilidad				
Capacidad de aprendizaje.					
Capacidad para ser usado	Compruebe limites de los requerimientos	Compruebe posibles picos de botella			
Protección contra errores de usuario	Validación de Datos en Interfaz de Usuario	Gestión de la configuración			
Capacidad para ser instalado	Probar múltiples entornos respetando la arquitectura				
Accesibilidad.	Solo en páginas web corporativas				

Revista Cubana de Ciencias Informáticas
Vol. 12, No. 1, Enero-Marzo, 2018
ISSN: 2227-1899 | RNPS: 2301
Pág. 117-129
http://rcci.uci.cu

Mantenibilidad					
Modularidad	Implementación de modificación				
Reusabilidad	Implementación de modificación				
Analizabilidad	Facilidad de implementación de los casos de prueba				
Capacidad de ser Probado	Facilidad de implementación de los casos de prueba				
Modularidad	facilidad ante modificación				
Portabilidad					
Facilidad de Instalación	Instalación en otra PC con el mismo Sistema. Operativo				
capacidad de ser remplazado	Posibilidad de cambiarse por el mismo				
Adaptabilidad	Facilidades de Configuración y Multiplataforma				

Generalidades del Procedimiento de Gestión de Pruebas de Software y Sistemas

El propósito de este, es planificar, ejecutar y documentar pruebas de calidad del sistema a realizar y su ejecución. Contempla las pruebas unitarias que son responsabilidad del equipo de desarrollo, pero no las pruebas de aceptación emitidas por el cliente. Luego de finalizar, el programa se encuentra completamente ensamblado, y se han encontrado y corregido los errores entre los módulos, métodos, clases y objetos. Incluye la etapa de las pruebas de validación, de requerimientos, de integración, rendimiento, stress, usabilidad, confiabilidad, seguridad, madurez y calidad de los documentos. Los requisitos deben ser escritos según el siguiente patrón:

"Debe" indica que el requerimiento es de obligatorio cumplimiento. Estos casos deberán ser objeto de validación del funcionamiento del sistema en las pruebas de aceptación.

"Debería" indica que el requerimiento es recomendable, pero pueden estar o no contenidos en las pruebas de aceptación.

"Podría" indica que el requerimiento se considera opcional.

Las pruebas se deben planificar antes de escribir el código fuente, se controlarán los procedimientos, datos y que el proceso de creación del software sea repetible. Se documentarán siguiendo el estándar establecido para documentar las pruebas y pudiendo tener los siguientes resultados:

Éxito: El resultado de la prueba es conforme al resultado esperado.

Aceptable: El resultado de la prueba indica que el sistema difiere de la especificación, pero es aceptable, no son necesarios cambios en la aplicación, pero requiriendo un cambio en la Especificación Funcional o los Requisitos.

Tolerable: El resultado de la prueba es incorrecto, la aplicación trabaja y podría ser aceptada, pero la falla deberá rectificarse en el tiempo acordado.

Intolerable: El resultado de la prueba es incorrecto, y la falla debe ser corregida antes de concluir la fase de prueba.

Error: El resultado de la prueba observado es correcto, pero los resultados esperados de acuerdo a los scripts de prueba son incorrectos.

Pendiente: Pruebas que realizarán en otro protocolo o iteración.

El resultado de estas pruebas, será documentado en el PSV y formará parte del expediente de proyecto. El control de este procedimiento será independiente a los procedimientos de control de seguridad informática. Se recomienda para el diseño de las pruebas la utilización de historias de usuario, las cuales pueden validar múltiples requisitos para cada caso de prueba, aumentando la eficacia de las mismas.

Nivel 1: Sin riesgos para la actividad

Para este caso se exige la revisión de los requisitos y la realización de pruebas. Se deben realizar los documentos PSAA, SSS y PSV. La revisión PPQA incluye la revisión del formato de los documentos, detección de problemas en la captura de requisitos o límites del producto. En cuanto a seguridad se detectan posibles superficies de ataque las cuales se intentan penetrar. Se valora la integridad mediante análisis de valores límites y calidad de la encriptación. Se realizan pruebas de acceso incluyendo ataques de fuerza bruta. Se intentan escalamientos y ataques con inyecciones. Se le realizan pruebas de integración, de tolerancia a fallos como desconexiones y emisión de alertas, con estas se mide fiabilidad. La adecuación funcional es medida a través de sus sub-características con pruebas que valoran el cumplimiento y pertinencia de los requisitos y la adecuada gestión de los datos mediante el mismo. Las pruebas de desempeño se realizan midiendo la carga y el rendimiento de la aplicación. Se realizan pruebas de aceptación pilotos mediante historias de usuario y se llena la Lista de Verificación de Usabilidad**.** Se valida la entrada de los datos y su capacidad de instalarse y ejecutarse siempre que se respete la arquitectura. En el caso de páginas corporativas se

valora la accesibilidad. Además se da un estimado de la facilidad de analizar, modificar y probar los datos atendiendo a facilidad de realizar e implementar pruebas desde diferentes ambientes. Las características como portabilidad son valoradas solo en este nivel de criticidad.

Nivel 2: Riesgo Menor

Para este nivel de riesgo además de lo anterior debe documentarse el diseño de alto nivel, se requiere redundancia simple a nivel lógico. El entorno de prueba será muy similar al de explotación, se especificará el ciclo de vida a utilizar para el desarrollo del sistema. Se realizará un análisis de riesgo inicial que aportará nuevos requerimientos y posibles condiciones de fallo. El responsable de las pruebas participa en la reunión de los grupos de desarrollo y prepara las pruebas con anterioridad. Se analizan y corrigen vulnerabilidades conocidas de los productos de desarrollo. Se controlan los permisos de lectura - escritura en ficheros y bases de datos. Debe permitir sincronismo de tiempo y crear trazas. Soportará desconexiones de módulos y mantendrá su funcionamiento. Permitirá gestión de la configuración y se deberá revisar el cumplimiento de las normas internacionales que le apliquen. Se le realizarán pruebas de volumen y estrés y se comprobará su compatibilidad con otros sistemas y que no afecta su puesta en marcha a ninguno.

Nivel 3: Riesgo Mayor

Para este nivel, se exige redundancia física. Los documentos deben reflejar el diseño a bajo nivel y trazabilidad hasta el código. Las pruebas deben ser documentadas por pares con miembros externos al grupo productor del sistema. Se documenta la línea base, se verifica la integridad de los datos que entran y salen del sistema, así como que la programación no contenga funciones vulnerables en el análisis del código fuente para tratar los fallos. Requiere cobertura completa del código, revisión formal de los requisitos a bajo nivel, adoptar el estándar de programación internacional y utilizar integración continua. Se probará su interoperabilidad con otro sistema crítico y el comportamiento ante inyecciones de datos erráticos externos.

Nivel 4: Riesgo Severo

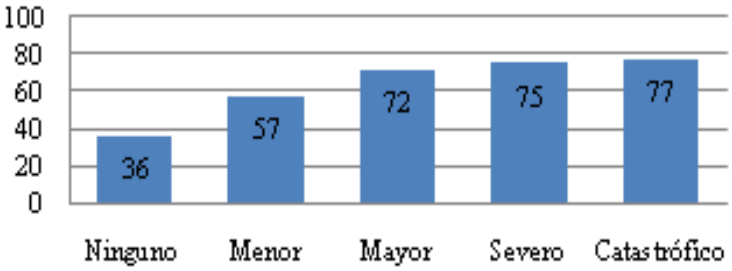
Se documentan las pruebas para niveles de criticidad anteriores, con la particularidad que para este nivel de riesgo las pruebas deben ser documentadas por pares externos a la entidad productora del sistema. La trazabilidad llega a niveles de objetos, la cobertura de código se hace a nivel de decisión. La tolerancia a fallos se valida en un ambiente de stress mediante desconexión. Se exige redundancia múltiple.

Nivel 5: Riesgo Catastrófico

Revista Cubana de Ciencias Informáticas
Vol. 12, No. 1, Enero-Marzo, 2018
ISSN: 2227-1899 | RNPS: 2301
Pág. 117-129
http://rcci.uci.cu

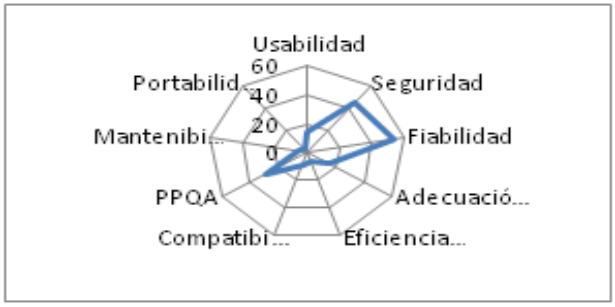
Se realizan las pruebas contenidas en todos los niveles de riesgos anteriores. Como caso puntual para este nivel de criticidad se realiza la cobertura del código de la variable crítica a nivel de condición, esta cobertura debe tener traza desde los requisitos y el diseño de bajo nivel como muestra de madurez del software. La tolerancia a fallos, será probada en un ambiente de estrés de todas las redundancias con desconexiones múltiples aleatorias.

Resultados



Viendo la propuesta de implementación de pruebas atendiendo a la gestión de riesgos es fácil identificar que el aumento de la exhaustividad aumenta según lo hace la criticidad del software.

Cuando lo miramos valorando las pruebas mínimas según las características de calidad del software se notas una planificación en factores que se consideran importantes potenciar según el criterio de los especialistas. Estos como muestra el grafico, son la Seguridad, la Fiabilidad. Estos garantizan la robustez del programa, su capacidad para resistir ataques y fallos. Luego se potencian los controles de PPQA, que aunque no es una característica de los sistemas establecida en las ISO 25000, se consideró, producto que garantizar la calidad del proceso y el producto, garantiza a su vez, la calidad de los requisitos, del diseño y su trazabilidad hasta el nivel de código y pruebas. Luego se encuentra la Adecuación Funcional y Usabilidad, que en general garantizan una utilización eficiente del sistema en cuestión, estas características en sistemas críticos garantizan respuesta oportuna y a tiempo de la información necesaria.



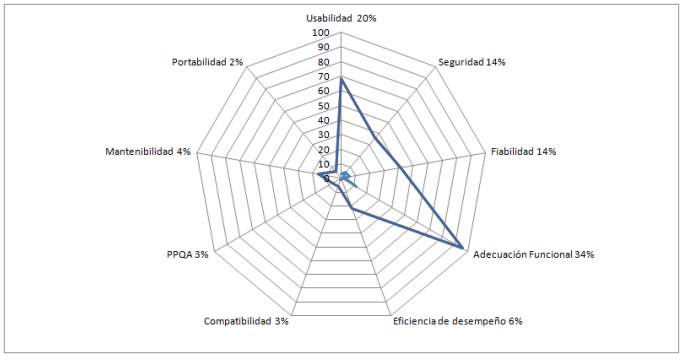
Esta propuesta fue utilizada en 2 sistemas útiles para las infraestructuras de Comunicación, Navegación, Vigilancia y Gestión del Tráfico Aéreo. El primero el sitio web AIS-Cuba, que brinda el Servicio de Información Aeronáutica cubana en tiempo real. Está considerado de Criticidad 1 (Sin riesgo para la aeronavegación), porque aunque la ausencia de los datos que visualiza puede llegar a tener criticidad 3 (Mayor) en el caso de los datos de fenómenos climatológicos o NOTAM, esta aplicación no los genera o gestiona. Estos datos son obtenidos mediante la red AFTN, se relacionan con el software mediante una interface de respaldo de la información que no pertenece al dominio del sitio web. Es un sistema CNS/ATM producto que constituye una herramienta de apoyo a la aeronavegación. Consiste

Revista Cubana de Ciencias Informáticas
Vol. 12, No. 1, Enero-Marzo, 2018
ISSN: 2227-1899 | RNPS: 2301
Pág. 117-129
http://rcci.uci.cu

en publicar la información de datos climatológicos, incluyendo METAR, SPECI y mapas climáticos. Publica también el AIP Electrónico de Cuba, lo cual constituye un requerimiento de la OACI sobre los requerimientos y obstáculos del espacio aéreo. Permite diferenciar los NOTAM para los diferentes aeródromos y filtrar los más riesgosos. Además, publica el posicionamiento de los aeródromos e información de contacto, misión y visión de la Dirección de Aeronavegación (UNAGO).

El segundo es el Sistema Automatizado de Observación Meteorológica de Aeródromo (AEROMET 2). Compuesto por la integración de 2 módulos informáticos y 1 sistema de equipos para la medición y transmisión de datos climatológicos. Es utilizado en el control de los aeródromos para informar las variables climatológicas en los diferentes aeropuertos del país. Los sistemas informáticos en general están categorizados como nivel 2 (Menor) y el módulo de medición y transmisión de datos es considerado nivel 3 (Mayor). El sistema contiene 2 datos de criticidad mayor, el QNH (presión hiperbatica por altura) dato meteorológico utilizado para estimar el ángulo de entrada de la aeronave, los controladores aéreos están facultados para cerrar un aeródromo en tanto este dato falle. El segundo dato crítico es viento este dato es importante para establecer condiciones de despegue y aterrizaje de las aeronaves, para prevenir sean desplazadas por el viento. La automatización de estos sistemas representa un requerimiento de la OACI y tiene integración con otros softwares críticos.

Como primer dato es importante esclarecer que ambos proyectos fueron exitosos. Las etapas de pruebas en concordancia con el modelo W utilizado para el desarrollo de los mismos, se realizó en 4 iteraciones. Para lo cual se organizaron en correspondencia con la propuesta un total de 362 pruebas las cuales se distribuyeron de la manera que representa la gráfica. Donde se evidencia que los factores más probados son Adecuación Funcional, Usabilidad, Seguridad y Fiabilidad, en correspondencia con las características de estas infraestructuras. Seguido por pruebas de eficiencia que aumentan principalmente en relación al nivel de las variables críticas.



Los sistemas en cuestión se comportaron de la siguiente forma de las 362 pruebas diseñadas en la primera iteración se ejecutaron 362, de ellas, fueron exitosas 298, tolerables 5 y aceptables 7. Ninguna fue clasificada como

inaceptable, 18 fueron error y 17 fueron pospuestas. Es importante destacar que estos números reflejan la totalidad de las iteraciones y que al finalizar las mismas, se contaban con 360 éxitos y 2 pospuestas para la fase de liberación, por restricciones de la maqueta.

Referencias

CeBIT. (2015, Marzo 18). *DeutscheWelle*. Retrieved Enero 06, 2016, from Auge de la seguridad informática: http://www.dw.com/es/cebit-2015-auge-de-la-seguridad-informática

Cid, D. (2014, Noviembre 24). *Protección Contra Vulnerabilidades de Software Desconocida.* Retrieved Febrero 13, 2016, from SucuriEspañol: https:/SucuriEspanol.com/

Eterovic, J., & Donadello, D. (2015). Identificación de los niveles de Integridad de la Seguridad en el desarrollo de software crítico para sistemas ferroviarios. *Universidad Nacional de La Matanza* .

ISO-NC 31000. (2015). *Gestión de Riesgo, Principios y Directrices.* La Habana, Cuba: Oficina Nacional de Normalización (NC) Calle E No. 261 El Vedado.

Matei, A. (2015, Marzo 17). *Guía para el desarrollo de software seguro.* Retrieved 06 29, 2016, from Archivo Diital UPM: http://oa.upm.es/34770/

Norma Martínez , A., & Porcelli, A. (2015). IMPLICANCIAS DE LAS TECNOLOGÍAS INFORMÁTICAS EN EL AMBIENTE Y NUEVAS TENDENCIAS EN EL DESARROLLO DE LA INFORMÁTICA VERDE COMO APORTE AL DESARROLLO SUSTENTABLE. *Actualidad Jurídica Ambiental* (50).

Norton. (2012). *Norton Cybercrime Report*. Retrieved Enero 12, 2016, from http://us.norton.com/cybercrimereport

Pressman, R. S. (2010). *Ingeniería de Software un enfoque práctico.séptima edición* (Vol. S.A. DE C.V). McGRAW-HILL INTERAMERICANA EDITORES.

Prieto, M. D. (2014, Octubre 13). *Detección proactiva de amenazas en infraestructuras críticas.* Retrieved Octubre 23, 2015, from III Congreso Iberoamericano de Ciberseguridad Industrial: blogthinkbig.com