# GESTIÓN DEL RIESGO EN PROYECTOS DE SOFTWARE

Rueda Sánchez,Jhon Alexander.
alexel200@gmail.com
Universidad Piloto de Colombia

*Abstract*—Combining different models, guides, standards and efficient practices such as (SSE-CMM, PMBOK, COBIT, ITIL e ISO 27001) decreases the risk of failures of software implementation or software development and adds a lot of characteristics of value like security and quality of the product, because the risk and quality factors are taken into consideration during the software implementation or software development.

*Resumen*—La combinación de los diferentes modelos, guías, estándares y buenas prácticas tales como (SSE-CMM, PMBOK, COBIT, ITIL e ISO 27001) permiten minimizar el riesgo de implementaciones o desarrollos de software fallidos y otorgan características de valor como seguridad y calidad en el producto, pues durante el desarrollo o la implementación se tienen en cuenta factores de riesgo, administración de los mismos y los factores de calidad.

## I. INTRODUCCIÓN

En un mundo tan cambiario y tan competitivo como el de la actualidad, con crecimientos tecnológicos inimaginables, factores económicos variables día a día y el rompimiento de fronteras que permiten hablar de un tierra globalizada, enmarcan el contexto empresarial en un ambiente complejo, donde solo la capacidad de innovación y de adaptación se convierten en la clave del éxito y la sostenibilidad de las empresas [1].

Uno de los factores que en la modernidad juega un papel muy importante en el ámbito empresarial, es el factor tecnológico; sí una empresa no tiene en cuenta este factor, tiene probabilidades muy altas de ser enterrada por la globalización.

La incorporación de herramientas tecnológicas en las organizaciones suponen la sobrevivencia en el mundo complejo y al mismo tiempo ayuda a agilizar y automatizar procesos, aumenta la productividad, mejora el desempeño, permite gestionar las comunicaciones, los grandes flujos de información [1] y los organiza de tal modo que la empresa minimice los errores, tome decisiones de precisas, eficaces y de este modo alcance sus objetivos propuestos.

Las herramientas tecnológicas que soportan los procesos de las organizaciones y los flujos de información producto de las actividades diarias, se traducen en programas (software), sistemas de comunicación (redes) y la parte física como computadores o servidores donde se hospeda el software [1].

Para atender los nuevos requerimientos a los que se enfrentan las empresas en el ambiente laboral. Se crean espacios de desarrollo de aplicaciones a medida [2], donde programadores atienden las necesidades de la organización por medio de una aplicación propia. Sin embargo, no es la única solución que la empresa podría encontrar, pues existen aplicaciones en el mercado, algunas de pago y otras de distribución libre conocidas como Open Source.

El software, independientemente su naturaleza (Software libre o de código cerrado), ayuda a la gestión empresarial adaptando los procesos de la empresa a sus características o incorporando nuevas funcionalidades que hacen que el software sea mucho más flexible y se pueda implementar en numerosas organizaciones y de este modo automatizar los procesos de la organización.

Entonces, la organización cuenta con la opción de crear su área de desarrollo, contratar una empresa de desarrollo, comprar o adecuar un software, pero independientemente la opción que la empresa escoja, existen riesgos inherentes a la programación o implementación de una aplicación que sin un

adecuado tratamiento, los perjuicios para la empresa pueden ser devastadores.

Con las diversas opciones que una compañía cuenta a la hora de desarrollar o implementar un software, en ocasiones se experimentan inconvenientes para tomar la decisión si se debe o no realizar el desarrollo propio, pero la decisión se debe tomar realizando un análisis exhaustivo del entorno, evaluando las ventajas y las desventajas de desarrollar o implementar un software ya existente (en caso de que exista), los problemas que se pueden tener en la fase de implementación e incluso los problemas de seguridad que hoy por hoy marcan una gran tendencia debido a las innumerables vulnerabilidades que se han descubierto a través del tiempo en los programas por deficiencias en su programación inicial.

Pero para que la decisión sea conveniente, resuelva los problemas de la organización y cumpla con estándares de calidad y seguridad, se debe realizar un análisis minucioso de los aspectos que pueden causar riesgos en el desarrollo o implementación del software, acompañado de la aplicación de modelos, normas, guías de buenas prácticas y estándares internacionales como PMBOK, SSE-CMM, ISO 27001, COBIT e ITIL que garantizan el éxito del proyecto, aumenta la entrega de valor y minimiza a través del enfoque de seguridad la probabilidad de que un riesgo se materialice.

Para contextualizar un poco al lector, se realiza una breve descripción acerca de cada uno de los modelos, normas, guías de buenas prácticas y estándares internacionales, se debe destacar también, que este conjunto de elementos no se eliminan entre ellos si no que se complementan entre sí.

PMBOK: la guía PMBOK es el estándar global por excelencia para la gestión de proyectos [18].

SSE-CMM: es un modelo de referencia para la incorporación de la Ingeniería de Seguridad en las organizaciones. Dentro del alcance de SSE-CMM, se encuentra la presentación de una serie de actividades para lograr el desarrollo de productos de software confiables, y alcanzar un ciclo de vida para sistemas seguros [14].

ISO 27001: la norma ISO ha sido elaborada para suministrar requisitos para el establecimiento, implementación, mantenimiento y mejora de un sistema de gestión de la seguridad de la información, el sistema de gestión de la seguridad de la información preserva la confidencialidad, la integridad y la disponibilidad de la información, mediante la aplicación de un proceso de gestión de riesgo [16].

COBIT: los Objetivos de Control para la Información y la Tecnología relacionada (COBIT®) brindan buenas prácticas a través de un marco de trabajo de dominios y procesos, y presenta las actividades en una estructura manejable y lógica.

Las buenas prácticas de COBIT representan el consenso de los expertos. Están enfocadas fuertemente en el control y menos en la ejecución. Estas prácticas ayudarán a optimizar las inversiones habilitadas por TI, asegurarán la entrega del servicio y brindarán una medida contra la cual juzgar cuando las cosas no vayan bien [15].

ITIL: fue desarrollada al reconocer que las organizaciones dependen cada vez más de la Informática para alcanzar sus objetivos corporativos. Esta dependencia en aumento ha dado como resultado una necesidad creciente de servicios informáticos de calidad que se correspondan con los objetivos del negocio, y que satisfagan los requisitos y las expectativas del cliente. A través de los años, el énfasis pasó de estar sobre el desarrollo de las aplicaciones TI a la gestión de servicios TI.

La aplicación TI (a veces nombrada como un sistema de información) sólo contribuye a realizar los objetivos corporativos si el sistema está a disposición de los usuarios y, en caso de fallos o modificaciones necesarias, es soportado por los procesos de mantenimiento y operaciones [17].

## II. GRANDES ACONTECIMIENTOS DE DESARROLLO O IMPLEMENTACIÓN DE SOFTWARE EN LA HISTORIA

Como se mencionó en la sección anterior, el desarrollo de software o la implementación del mismo, es un proceso sensible para cualquier organización, cualquier falla en la fase de obtención de requerimientos, en la fase de desarrollo o fallas en la implementación, se traducen en fallas económicas para la organización.

A continuación se presentan algunas fallas en el desarrollo o implementación de software a través de la historia.

A finales de 1998, la sonda especial Mars Climate Orbited pasó hacer chatarra espacial después de estrellarse en marte debido a que el equipo de ingenieros, olvidaron realizar conversiones de unidades inglesas al sistema de unidades de métrica. El costo del proyecto $125 millones de dólares [3][4].

Ariane 5: la nueva era de las naves espaciales europeas, convertido en un show de fuegos artificiales tras la explosión de la nave aeroespacial 36 segundos después de su despegue [3][5].

El fallo fue causado por tratar de procesar un número de 64 bits en un espacio de 16 bits, resultando en una condición de desbordamiento que colisionó el computador principal y el computador de respaldo, ambos computadores ejecutaban exactamente el mismo software.

El costo estimado para el lanzamiento del Ariane 5 fue de 500 millones de dólares y mucho tiempo de desarrollo perdido.

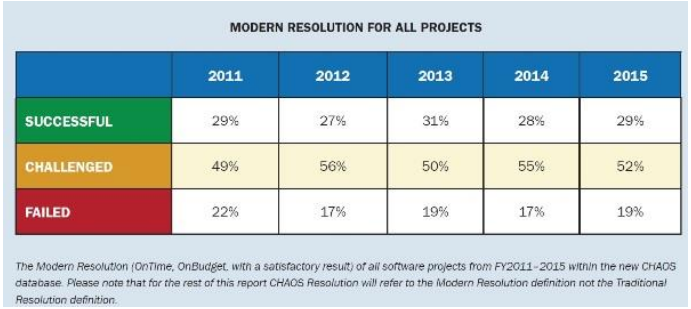
EDS Child Support System: en el 2004, se introdujo un sistema informático de gran complejidad en el Reino Unido, específicamente en la CSA (Child Support Agency). Al mismo tiempo, el Departamento de Trabajo y Pensiones (DWP), decidió reestructurar la agencia completa. Los dos pedazos de software fueron completamente incompatibles. El sistema aunque pudo pagar a 1.9 millones de personas, dejó de pagar a otras 700,000. Teniendo $7 billones en pagos no recolectados para el soporte de niños. Un atraso de 239.000 casos, 36.000 nuevos casos “parados” en el sistema y costó más de un billón de dólares a los contribuyentes del Reino Unido [6].

En internet existen más ejemplos de desastres causados por el desarrollo e implementación de software donde incluso, grandes compañías no han podido escapar, ejemplo Hewlett Packard, Nike y Avon. Empresas que buscaban mejorar los procesos internos por medio de la implementación de ERP’s (Enterprise Resource Planning – Planificación de Recursos Empresariales), terminaron gastando millones de dólares por no probar el sistema antes de ponerlo en producción como el caso de Nike, pobre planificación y pruebas inadecuadas en el caso HP y no capacitar adecuadamente al personal caso de HP y AVON [7][8][9].

En los siguientes capítulos, se presentarán medidas que permiten minimizar los riesgos de fallas en el desarrollo e implementación del software.

## III. PLANEACIÓN DEL DESARROLLO O IMPLEMENTACIÓN DE SOFTWARE

De acuerdo con las estadísticas [10] en promedio, tan solo el 30% de los proyectos de software terminan según lo planeado.



MODERN RESOLUTION FOR ALL PROJECTS

| | 2011 | 2012 | 2013 | 2014 | 2015 |
|---|---|---|---|---|---|
| SUCCESSFUL | 29% | 27% | 31% | 28% | 29% |
| CHALLENGED | 49% | 56% | 50% | 55% | 52% |
| FAILED | 22% | 17% | 19% | 17% | 19% |

*The Modern Resolution (OnTime, OnBudget, with a satisfactory result) of all software projects from FY2011-2015 within the new CHAOS database. Please note that for the rest of this report CHAOS Resolution will refer to the Modern Resolution definition not the Traditional Resolution definition.*

***Figura 1. Resumen de los resultados de proyectos de los últimos cinco años, con la nueva definición de éxito. Tomado de [10].***

Las fallas o demoras en el Proyecto, generalmente están asociadas a falencias en la planificación del proyecto, falta de compromiso de las partes interesadas, insuficiencia en la gestión de los

riesgos del proyecto [11], entre otros muchos factores.

Al conocer la facilidad con la que factores externos e internos de un proyecto de desarrollo o implementación de software se ve afectado, claramente se necesita de modelos, guías, buenas prácticas y estándares internacionales como PMBOK, SCE-CMM, ITIL, ISO 27001, COBIT, que sin ser necesario aplicar todas en conjunto, se puede tomar las mejores características o las características relacionadas con el desarrollo de software para realizar una buena mezcla que garanticen el éxito, la seguridad y la calidad del producto.

Si la organización no cuenta con el sistema de gestión de la seguridad de la información (SGSI), se debería establecer todo el control A.14.2, definido en la norma NTC-ISO IEC 27001, especialmente la política de desarrollo seguro (control A.14.2.1), que busca establecer y aplicar reglas para el desarrollo de software y de sistemas, a los desarrollos dentro de la organización.

Posteriormente y ya conociendo el problema o el requerimiento realizado por parte de la organización, es imprescindible formalizar el inicio del proyecto y el alcance del mismo. Este evento se puede llevar a cabo mediante un acta de inicio como se propone en la guía PMBOK o mediante el dominio de planeación y organización descrito en COBIT. De esta forma se da inicio al proyecto, asegurando el compromiso de las partes interesadas para posteriormente pasar a la fase de requerimientos para contextualizar las necesidades del requerimiento, entre otras cosas.

## IV. FACE DE REQUERIMIENTOS

La fase de requerimientos es una etapa sensible en la ingeniería de software que comprende el levantamiento, análisis, especificación y validación de los requerimientos del proyecto. Cualquier deficiencia en la ejecución de las tareas anteriormente mencionadas, afectaría críticamente el proyecto en la etapa de desarrollo como la etapa de implementación, incluso, sería una fuente errónea de toma de decisiones a la hora de seleccionar un software ya desarrollado.

Es la etapa que permite reconocer la situación del entorno y los problemas del mismo que el software va a resolver. Es la etapa precisa donde se evalúa si se necesita un desarrollo o si se debe implementar un software ya existente, los pro’s y las contra’s del camino a tomar y además, permite la identificación actores del sistema que comprende los clientes y usuarios del mismo para posteriormente definir la interacción con los miembros del equipo desarrollador.

Entonces, existen diferentes metodologías y técnicas que ayudan al equipo desarrollador a entender las necesidades del cliente, realizar el análisis y levantamiento de requerimientos.

El éxito o el fracaso del proyecto, depende del acierto en la selección de la metodología o de las técnicas usadas en la fase de requerimientos, tales como las entrevistas, conformación de grupos focales (expertos discutiendo sobre temas específicos, ayudando a determinar requerimientos), observación, modelado (uso de diagramas de flujo, casos de uso, diagramas de estado, UML, entre otros) y prototipos, entre otros.

Es propicio en esta fase realizar la identificación, evaluación y tratamiento de los riesgos del proyecto. En el modelo SSE-CMM aplican las áreas de proceso PA03 – Valoración de riesgos de seguridad y PA10 – Especificar necesidades de seguridad.

En la norma ISO 27001, aplica el control A14.2.5 – Principio de construcción de los sistemas seguros; Las organizaciones deben establecer y proteger adecuadamente los ambientes de desarrollo seguros para las actividades de desarrollo e integración de sistemas que comprendan todo el ciclo de vida de desarrollo de sistemas.

COBIT, también posee controles que pueden ser aplicados en esta fase; PO6 - Comunicar las aspiraciones y directrices de la Administración. PO8 – Administrar la Calidad (incorporación de estándares de calidad en etapas tempranas de desarrollo) y PO9. Valoración y administración de riesgos.

En el PMBOK, las actividades que están estrechamente relacionadas con esta fase son: recolectar requerimientos, definir el alcance, planificar la calidad, identificar los riesgos, análisis cualitativo de riesgos y el plan de respuesta a los riesgos.

## V. ESPECIFICACIÓN, DISEÑO Y ARQUITECTURA

Acorde con la Ingeniería de software, el desarrollo de software consta de una fase de especificación y otra fase de diseño y arquitectura:

- Especificación: es la tarea de describir el funcionamiento del software, la interacción con el usuario y con otros sistemas de una forma rigurosa.
- Diseño y Arquitectura: se refiere a determinar cómo funcionará el software de forma general sin entrar en detalles. Consisten en incorporar consideraciones de la implementación tecnológica, como el hardware, la red, etc. Se definen los casos de uso para cubrir las funciones que realizará el sistema, y se transformarán las entidades definidas en el análisis de requisitos en clases de diseño, obteniendo un modelo cercano a la programación orientada a objetos [12].

El PMBOK a través de las actividades de planificación de las comunicaciones y elaboraciones de EDT (Donde se planifican las entregas del proyecto), así mismo COBIT a través del control PO6 – Comunicar las aspiraciones y directrices de la Administración, aplican para estas dos fases de la ingeniería de software que para un mejor entendimiento del lector se presenta en este artículo como una sola sección, pues con los mismos controles se busca fortalecer la comunicación entre las partes interesadas y la aprobación de los avances del proyecto.

## VI. PROGRAMACIÓN

Es la fase donde se traduce el diseño en código fuente escrito en un lenguaje de programación.

Sea cual sea la metodología seleccionada para el desarrollo de software (SCRUM, XP, CASCADA. Etc...), es propicio contar con entornos de desarrollo, pruebas y de producción, tal y como lo específica el control A12.1.4, de la norma ISO 27001 - Separación de los ambientes de desarrollo, pruebas, y operación, con el siguiente argumento: Se deben separar los ambientes de desarrollo, prueba y operación, para reducir los riesgos de acceso o cambios no autorizados al ambiente de operación.

Así mismo la norma ISO en el control A.14.2.6 - Ambiente de desarrollo seguro, establece que las organizaciones deben establecer y proteger adecuadamente los ambientes de desarrollo seguros para las actividades de desarrollo e integración de sistemas que comprendan todo el ciclo de vida de desarrollo de sistemas [16].

Adicionalmente, al finalizar una iteración en la metodología seleccionada, sería óptimo realizar un nuevo análisis y administración de riesgos para el posterior ciclo de iteración, con el fin de tomar lecciones aprendidas resultantes de la iteración anterior o iteraciones anteriores y minimizar riesgos o problemas en la nueva iteración.

En esta sección, vale la pena aclarar que si el software está siendo desarrollado por terceros, se deben establecer controles que permitan verificar y controlar las actividades realizadas por estos terceros con el fin de minimizar riesgos, el control está definido en la norma ISO 27001 en el control A.14.2.7 – Desarrollo Contratado externamente. COBIT, también contempla esta opción a través del control DS2 –Administración de servicios prestados por terceros.

## VII. PRUEBAS

Consiste en comprobar que el software realice correctamente las tareas indicadas en la especificación del problema. Una técnica de prueba es probar por separado cada módulo del software y luego probarlo de forma integral, para así llegar al objetivo. Se considera una buena práctica que las pruebas sean efectuadas por alguien distinto al desarrollador que la programó [12].

En esta fase es necesario comprobar que el software, módulo o funcionalidad cumpla con los requisitos exigidos en la fase de requerimientos. Si

no se realiza desarrollo sino implementación de un software ya desarrollado, se debe verificar que este cumpla con todos los requisitos exigidos y las pruebas deben ser extremadamente rigurosas.

Además de realizar todas las comprobaciones de seguridad y cumplimiento de la calidad. Estas pruebas deben ser rigurosas con el fin de aprobar o no el software liberado.

Para validar la seguridad, se puede recurrir al modelo SSE-CMM, en el área de proceso PA11. Verificación y Validación de la Seguridad. En COBIT, con el control DS10. Administración de problemas, se puede enfocar hacia: "los problemas de fallas de seguridad de software son parte del conocimiento histórico que todos los desarrolladores de software de la organización deberían tener a su disposición, de manera tal, de evitar la repetición de errores cometidos, en este aspecto, se debe aprovechar este proceso COBIT" [14].

Los controles A.14.2.8 y A.14.2.9 de la Norma ISO 27001, hace referencia a Pruebas de seguridad de sistemas y Prueba de aceptación de sistemas respectivamente. Estos controles aplican para esta fase, pues el primer control detalla que durante el desarrollo se debe realizar pruebas de funcionalidad de la seguridad y el segundo control, indica que para los sistemas de información nuevos, actualizaciones y nuevas versiones, se deben establecer programas de prueba para aceptación y criterios de aceptación relacionados.

En cuanto a las pruebas de calidad, se puede realizar siguiendo la guía PMBOK, se realiza el control de calidad del entregable definido en la actividad planeación de la calidad, es decir, se valida lo que se planeó contra el entregable. Así mismo el control PO8. Administrar la Calidad – de COBIT puede ser enfocado al control de la calidad del producto.

## VIII. DOCUMENTACIÓN

Realización del manual de usuario, y posiblemente un manual técnico con el propósito de mantenimiento futuro y ampliaciones al sistema. Las tareas de esta etapa se inician ya en la primera fase pero sólo finalizan una vez terminadas las pruebas.

La documentación es una fase muy importante dentro del desarrollo y por ende, debe ser entregado a satisfacción del cliente o usuario y debe cumplir con normas y estándares de calidad.

PMBOK a través de la elaboración de EDT puede controlar la calidad de la documentación y la entrega a satisfacción del cliente. También, se puede controlar a través de COBIT, mediante los controles PO6 y PO8, donde se define el compromiso con la calidad y se administra la calidad respectivamente.

## IX. MANTENIMIENTO

Mantener y mejorar el software para solventar errores descubiertos y tratar con nuevos requisitos. El mantenimiento puede ser de cuatro tipos: perfectivo (mejorar la calidad interna de los sistemas), evolutivo (incorporaciones, modificaciones y eliminaciones necesarias en un producto software para cubrir la expansión o cambio en las necesidades del usuario), adaptativo (modificaciones que afectan a los entornos en los que el sistema opera, por ejemplo, cambios de configuración del hardware, software de base, gestores de base de datos, comunicaciones) y correctivo (corrección de errores).

Esta es una fase propensa para la aplicación de ITIL ya que la librería soporte al servicio se preocupa de todos los aspectos que garanticen la continuidad, disponibilidad y calidad del servicio prestado al usuario [17].

Este enfoque puedo ser dirigido a los casos o incidentes que aparecen pos implantación del sistema, casos referentes a capacitación, soporte y mantenimiento del sistema implantado ya sea que se halla desarrollado o se halla implementado.

Otra librería importante es provisión de servicios, que busca Influyen de forma directa con los objetivos de seguridad, cuando se definen dentro de la organización y se asegura la demanda que requiere el negocio en cuanto a los activos tecnológicos que tiene y requiere para el desempeño

de sus actividades. Esto es un aporte dado que los productos software son un activo tecnológico [14].

## X. CONCLUSIONES

El desarrollo e implementación de software es una actividad crítica en las organizaciones del mundo actual, deficiencias en la ejecución del proyecto o un mal diseño del software, podrían generar pérdidas incalculables para las organizaciones y en algunos casos para la humanidad.

De acuerdo con las estadísticas, aproximadamente el 30% de los proyectos de software son exitosos y la mayoría de los mismos se terminan a destiempo y con sobrecostos.

Para minimizar los riesgos de que el desarrollo o la implementación del software fracasen, existen diferentes modelos, guías, estándares y buenas prácticas que se podrían llegar a aplicar en conjunto, elevando la probabilidad de finalizar el desarrollo o la implementación de software de manera exitosa y otorgando características de valor como la seguridad y la calidad del producto.

La seguridad siendo un factor crítico en la sociedad actual, debido al aprovechamiento de vulnerabilidades y obtención no autorizada de información, es un factor clave en el desarrollo de software pues la información que ingresa y es procesada a través de la aplicación puede ser de vital importancia para la organización, razón por la cual, la aplicación de la norma ISO 27001, COBIT, SSE-CMM, ITIL y PMBOK, es pertinente pues el enfoque de seguridad, si no es el core, es tenida en cuenta, además de que combinadas pueden ser un gran complemento para los factores de seguridad y calidad del software.

## REFERENCIA


[1] Chullitohost.com. “Importancia del Software en las Empresas”. Internet: http://softwarearequipa.blogspot.com.co/2010/05/importancia-del-software-en-las.html, 29 de mayo de 2010 [04 de octubre de 2015].

[2] IT GROUP.” Software A La Medida”. Internet: http://www.itgroup.com.co/desarrollo-de-software/productosyservicios.html#software, 12 de Junio de 2012 [04 de octubre de 2015].

[3] Nick Harley. “10 of the most costly software errors in history”. Internet: https://raygun.io/blog/2014/05/10-costly-software-errors-history/, 29 de mayo de 2014 [04 de noviembre de 2015].

[4] Javier Valenzuela (2 OCT 1999). “La "Mars Climate" se estrelló en Marte porque la NASA no tradujo kilómetros a millas”. El Pais. [On-line]. Disponible: http://elpais.com/diario/1999/10/02/sociedad/938815207_850215.html, [04 de noviembre de 2015].

[5] Álvaro Ibañez. “El error de software que convirtió un lanzamiento espacial en carísimos fuegos artificiales”. Internet: http://www.rtve.es/noticias/20140604/error-software-convirtio-lanzamiento-espacial-carisimos-fuegos-artificiales/948262.shtml, 04 de junio de 2014 [04 de noviembre de 2015].

[6] Wikipedia. “List of failed and overbudget custom software projects”. Internet: https://en.wikipedia.org/wiki/List_of_failed_and_overbudget_custom_software_projects, 12 de noviembre de 2015 [14 de noviembre de 2015]

[7] Melissa Monahan. “Top Six ERP Implementation Failures”. Internet: http://blog.360cloudsolutions.com/Top-Six-ERP-Implementation-Failures, 30 de mayo de 2013 [04 de noviembre de 2015].

[8] Ben Kepes. “UPDATED - Avon's Failed SAP Implementation A Perfect Example Of The Enterprise IT Revolution”. Interent: http://www.forbes.com/sites/benkepes/2013/12/17/avons-failed-sap-implementation-a-perfect-example-of-enterprise-it-revolution/, 17 de diciembre de 2013 [04 de noviembre de 2015].

[9] Thomas Wailgum. “10 Famous ERP Disasters, Dustups and Disappointments”. Internet: http://www.cio.com/article/2429865/enterprise-resource-planning/10-famous-erp-disasters--dustups-and-disappointments.html, 24 de marzo de 2009 [04 de noviembre de 2015].

[10] Shane Hastie, Stéphane Wojewoda. “Standish Group 2015 Chaos Report - Q&A with Jennifer Lynch”. Internet: http://www.infoq.com/articles/standish-chaos-2015, 04 de octubre de 2015 [15 de noviembre de 2015].

[11] LaboratorioTI, “Los 36 Errores Clásicos de los Proyectos de Desarrollo de Software”. Internet: http://www.laboratorioti.com/2015/06/16/los-36-errores-clasicos-de-los-proyectos-de-desarrollo-de-software/, 16 de junio de 2015 [15 de Noviembre de 2015].

[12] Inteco, “Ingeniería del Software: Metodologías Y Ciclos De Vida” 1ra edición, Marzo de 2009.

[13] Nicolás Aristizábal Mejía y Miguel Eduardo Torres Moreno , “Técnicas de Levantamiento de Requerimientos con Innovación”. Presentado en el Cuarto Congreso Colombiano de Computación 4CCC. Sociedad Colombiana de Computación S(CO)2, Universidad Autónoma de Bucaramanga, 2009.

[14] Edmundo Tovar, José Carrillo, Vianca Vega y Gloria Gasca, “Desarrollo De Productos De Software Seguros En Sintonía Con Los Modelos SSE-CMM, COBIT e ITIL”. PM-AEMES, vol. 3, N° 2, Septiembre de 2006.

[15] IT Governance Institute. COBIT 4.1, 2007.
[16] ICONTEC. “NTC-ISO-IEC 27001 - Tecnología de la Información. Técnicas de Seguridad. Sistemas de Gestión de la Seguridad de la Información. Requisitos“. 11 de diciembre de 2013.
[17] OSIATIS. “ITIL®-Gestión de Servicios TI - Soporte al Servicio”.Internet: http://itil.osiatis.es/Curso_ITIL/Gestion_Servicios_TI/fundamentos_de_la_gestion_TI/que_es_ITIL/soporte_al_servicio.php, [21 de noviembre de 2015].
[18] PMI. “PMBOK® Guide”. Internet: http://www.pmi.org/pmbok-guide-and-standards/pmbok-guide.aspx, [22 de noviembre de 2015].
[19] OSIATIS. “ITIL®-Gestión de Servicios TI - ¿Qué es ITIL®?”.Internet: http://itil.osiatis.es/Curso_ITIL/Gestion_Servicios_TI/fundamentos_de_la_gestion_TI/que_es_ITIL/que_es_ITIL.php, [22 de noviembre de 2015].