GESTION DE RIESGOS EN PROYECTO DE SOFTWARE A DESARROLLAR EN EMPRESA PRIVADA

DIRECTOR DE TRABAJO DE GRADO:

ING. FREDDY LEON REYES M.ED.

PRESENTADO POR

MARIA LILIANA ARAQUE JIMENEZ

CODIGO**:** 1301040

UNIVERSIDAD MILITAR NUEVA GRANADA

ESPECIALIZACION GERENCIA INTEGRAL DE PROYECTOS

FACULTAD DE INGENIERIA

BOGOTA D.C, NOVIEMBRE 09 DE 2015

GESTION DE RIESGOS EN PROYECTO DE SOFTWARE A DESARROLLAR EN EMPRESA PRIVADA

RISK MANAGEMENT IN SOFTWARE PROJECT TO DEVELOP IN PRIVATE ENTERPRISE

María Liliana Araque Jiménez
Ingeniera de Sistemas
Analista Calidad
Quality Vision Technologies
Bogotá D.C, Colombia
maliarji0927@hotmail.com

RESUMEN

En la actualidad las empresas desarrolladoras de software enfrentan un elemento crítico en el proceso de desarrollo de productos de software, que son los riesgos que se presentan a nivel de cada una de las fases del desarrollo, estos deben ser objeto de una gestión adecuada la cual debe ser iniciativa por la gerencia, supervisada y controlada por cada uno de los jefes de las áreas importantes involucradas en el desarrollo de software. La gestión de riesgos en proyectos de software es una actividad expresada en múltiples metodologías, pero en la práctica se aplican de forma particular dependiendo la lógica del negocio de cada organización.

En este artículo se evidencia la gestión de riesgos para el desarrollo de un software para una entidad privada basada en la guía del PMBOK, donde se hace una identificación de los riesgos más importantes en cada una de las fases del desarrollo, de igual forma se hace un análisis cualitativo, por medio del cual se haya la probabilidad de ocurrencia y el impacto de cada uno de los riesgos identificados, seguidamente se hace una clasificación por el nivel del riesgo, para posteriormente proponer un plan de respuesta a los riesgos más críticos. Esta gestión debe cumplir con un adecuado control y seguimiento, el cual permitirá llevar a cabo una identificación de nuevos riesgos o actualización de los planes de respuesta existentes, con el objetivo de prever los posibles riesgos negativos que se puedan llegar a materializar durante el desarrollo del software.

Palabras clave: Requerimiento, Riesgo, Gestión de Riesgos, PMBOK, Impacto, Probabilidad, desarrollo de software, fase.

ABSTRACT

At present, software development companies face a critical element in the development process of software products, which are the risks that arise at the level of each of the stages of development, they should be subject to appropriate management which must be initiative by manager, supervised and controlled by each of the heads of the major areas involved in software development. Risk management in software projects is an activity expressed in multiple methodologies, but in practice are applied in a particular way depending on the business logic of each organization.

In this article, risk management for the development of software for a private entity based on the PMBOK Guide, where the identification of the most important risks is done is evident in each of the phases of development, just as it is done a qualitative analysis, through which the likelihood and impact of each risk has been identified, then a classification by risk level, later to propose a plan to respond to the most critical risks. This management should comply with adequate control and monitoring, which will allow to carry out an identification of new risks or update existing mitigation plans, in order to anticipate possible negative risks that may come to realize during development software.

Keywords: Requirement, Risk, risk management, PMBOK, Impact, Probability, software development, fhase.

INTRODUCCION

La entidad financiera de caso de estudio [1], cuenta con una acreditación de más de 45 años, durante los cuales se ha caracterizado por su fidelidad, cumplimiento con sus clientes y la efectividad de sus operaciones. Inicialmente operaciones locales, y en la actualidad con su servicio de 24/7, con la ayuda de la plataforma tecnológica que permite la ejecución de operaciones a nivel electrónico, apoyado por un gran equipo de talento humano y tecnológico hacen de la entidad una de las mejores a nivel nacional.

Todas las estructuras tecnológicas que apoyan a la entidad con sus operaciones bancarias, no solo aportan una mayor efectividad y cobertura, sino que también traen consigo diferentes riesgos que pueden afectar significativamente los procesos, por este motivo se ve la necesidad de realizar una apropiada gestión de los riesgos en las fases de análisis, diseño, codificación, pruebas y entrega del software y poder proponer estrategias de mitigación de los mismos con el fin de asegurar cada una de estas etapas.

Según el PMBOK, el riesgo de un proyecto es un evento o condición incierta que, de producirse, tiene un efecto positivo o negativo en uno o más de los objetivos del proyecto, tales como el alcance, el cronograma, el costo y la calidad. Un riesgo puede tener una o más causas y, de materializarse, uno o más impactos. Una causa puede ser un requisito especificado o potencial, un supuesto, una restricción o una condición

que crea la posibilidad de consecuencias tanto negativas como positivas. Por lo tanto es de vital importancia realizar una adecuada gestión de riesgos donde se dé inicio en la primera fase del proyecto, para llevar seguimiento y controles adecuados de estos en cada una de las fases del desarrollo del software, hasta la aceptación del producto del proyecto.

Para la gestión de riesgos se ha tomado como referencia la guía del PMBOK, definiendo una metodología de gestión de riesgos la cual establece la Identificación, evaluación, plan de respuesta a riesgos, seguimiento y control, secuenciales e iterativos. Los cuales aseguraran efectividad del plan de gestión de riesgos sobre el proyecto en cuestión, por tal motivo para el proyecto de desarrollo de software se busca identificar los posibles riesgos que se pueden presentar, de igual forma categorizarlos evidenciando su probabilidad de ocurrencia e impacto en el proyecto y al final proponer unas estrategias de mitigación del riesgo en caso de llegar a materializarse. [2] Con lo anterior se asegura que el desarrollo del proyecto cumple con una adecuada planeación, en la cual se pueden prever los posibles riesgos que pueden generar alteraciones en el curso normal de este.

El desarrollo de software en la actualidad, cuenta con gran importancia en la parte financiera del país, debido al crecimiento de transacciones electrónicas, generadas por la evolución de la tecnología y los sistemas, esto genera una serie de desarrollos de software que satisfacen estas necesidades, pero se ve con preocupación que en los proyectos de desarrollo de software actualmente no se tiene la cultura de ser prevenidos en realizar una adecuada gestión de riesgos a pesar de saber que un riesgo es un gran problema donde llegue a ocurrir. La empresa específicamente para este proyecto no cuenta con una gestión de riesgos por tal motivo se evidencia una necesidad en este tema, que es de gran importancia en el desarrollo del proyecto para ayudar al equipo de trabajo de software a comprender y manejar la incertidumbre.

Teniendo en cuenta que para este proyecto se evidencia la ausencia de una gestión de riesgos, se dispuso del siguiente objetivo general llamado “Gerencia de riesgos en proyecto de software a desarrollar en empresa privada”, haciendo de esta manera que el proyecto reduzca la vulnerabilidad a eventos o condiciones que afecte el resultado del mismo.

Con los riesgos existentes hoy en día en cada uno de los proyectos desarrollados a nivel software, se evidencia la necesidad de desarrollar métodos que nos permitan identificar la probabilidad de ocurrencia e impacto de un evento atípico, facilitando el control de los mismos en el caso de que se presenten en las diferentes etapas de desarrollo. Con la utilización de una matriz de riesgos, se puede observar los impactos que estos ocasionan en el momento de materializarse, a nivel de las fases de desarrollo de software, permitiendo con los resultados arrojados tomar decisiones y ejecutar las medidas de mitigación necesarias para contrarrestar impactos negativos que estos pueden generar.

3. MATERIALES Y METODOS

La investigación desarrollada es de tipo exploratoria considerando que aunque la importancia de gestionar los riesgos en los proyectos de Software es conocida de forma general, no lo es a un nivel más particular, orientado a las fases del ciclo de vida del desarrollo de software; por tal razón las fuentes de información trabajadas son “Fuentes Secundarias” que son las búsquedas en Internet de bases virtuales, repositorio Universidad Militar Nueva Granada (UMNG), publicaciones, referencia de libros, artículos en las cuales se pueda evidenciar la gestión de riesgos en el desarrollo de proyectos informáticos detectando rasgos comunes, sugerencias y guías para su gestión los cuales puedan ser considerados como experiencia y buenas prácticas para la adaptación al proyecto de la empresa privada, “Fuente Experiencia y Conocimiento” el cual se obtiene conocimiento y experiencia de algunos procesos de las fases de desarrollo de proyecto de software, esto dado a la participación activa en diversos proyectos tanto en banca como en telecomunicaciones, apoyados en la fase de análisis, pruebas funcionales y pruebas técnicas. La metodología a utilizar para el desarrollo de este artículo se basa en la descripción de las cinco fases del desarrollo de software, en las cuales se realizara una adecuada gestión de los riesgos.

3.1 DESCRIPCION DE LAS FASES DEL DESARROLLO DE SOFTWARE

3.1.1 FASE DE ANÁLISIS

Esta fase es la primera y es la de mayor importancia, ya que es la base para continuar con las demás fases del desarrollo de software. El objetivo de esta fase es la descripción del propósito del sistema, donde el cliente, ingeniero de requerimientos y usuarios identifican el problema y posteriormente definen a través de lenguaje natural un sistema que da solución al problema sin generar ambigüedades. En esta fase después del estudio del arte realizado, se definieron los procesos de captura de requerimientos, análisis y negociación de requerimientos, especificación de requerimientos, validación de requerimientos y documentación de requerimientos. [3,4]

Figura 1. Procesos de la Fase de Análisis

3.1.1.1 Captura de Requerimientos: Es el proceso por medio del cual se obtienen o capturan los requerimientos funcionales y no funcionales donde los primeros hace referencia a las interacciones entre el usuario y el sistema, los segundos tratan sobre las restricciones o atributos del sistema como tiempo de respuesta y precisión. Se da inicio con la extracción de datos al cliente y usuarios finales del sistema haciendo uso

de técnicas eficientes y seguras tales como introspección, entrevistas de cuestionarios, entrevistas de final abierto y discusiones, etc.

3.1.1.2 Análisis y Negociación de Requerimientos: En el análisis de requerimientos se descubre problemas en la captura de requerimientos, se evalúa la necesidad de todos los requisitos, se analiza su consistencia y completitud, se asegura la viabilidad en cuanto a la parte técnica, de costes, y planificación. Seguidamente se pasa a la negociación de requerimientos donde se discute las inconsistencias, limitaciones, carencias encontradas y finalmente llegar a un mutuo acuerdo que satisfaga a todos los usuarios para así poder realizar la priorización de requerimientos.

3.1.1.3 Especificación de Requerimientos: Es el proceso por medio del cual se describe a manera de borrador los requerimientos ya que está sujeto a modificaciones.

3.1.1.4 Validación de Requerimientos: La validación la realiza un equipo integral que puede estar conformado por ingeniero de requerimientos, desarrollador, cliente y usuarios con el fin de detectar y corregir errores, ambigüedades u omisiones para garantizar la consistencia y precisión de los requerimientos. También se evalúa que los requerimientos acordados son los que realmente definen el sistema que el cliente necesita y quiere ver como producto final.

3.1.1.5 Documentación de Requerimientos: Se realiza la documentación oficial de cada uno de los requerimientos acordados, de forma clara, detallada y verificable.

3.1.2 FASE DE DISEÑO

El diseño de un sistema de información produce los elementos que establecen cómo el sistema cumplirá los requerimientos identificados durante la fase de análisis. A esta fase se le conoce también con el nombre de Diseño Lógico.
El primer paso en el diseño de sistemas es identificar los informes y las salidas que el sistema producirá; a continuación los datos específicos de cada uno de éstos se señalan, incluyendo su visualización exacta sobre el papel, la pantalla de despliegue o cualquier otro medio.

El diseño también describe los datos calculados y/o almacenados en la base de datos. Los datos y los procedimientos de cálculo se describen con detalle. Se seleccionan las estructuras de los archivos y los dispositivos de almacenamiento, como son discos o cintas magnéticas o papel. Los procedimientos deben mostrar cómo se van a procesar los datos y cuáles van a ser las salidas. Los documentos que contiene las especificaciones del diseño se pueden representar por medio de diagramas, tablas y símbolos especiales. El último paso del diseño es pasar el documento de diseño al grupo de desarrollo para que se inicie la fase de codificación del software. Por lo tanto después de realizar el análisis respectivo de la fase se propone los procesos de Diseño de datos, Diseño Arquitectónico, Diseño Procedimental, Diseño de Interfaz de usuario, Documento de Diseño Preliminar, Documento de Diseño Detallado como se puede observar en la siguiente figura. [7].

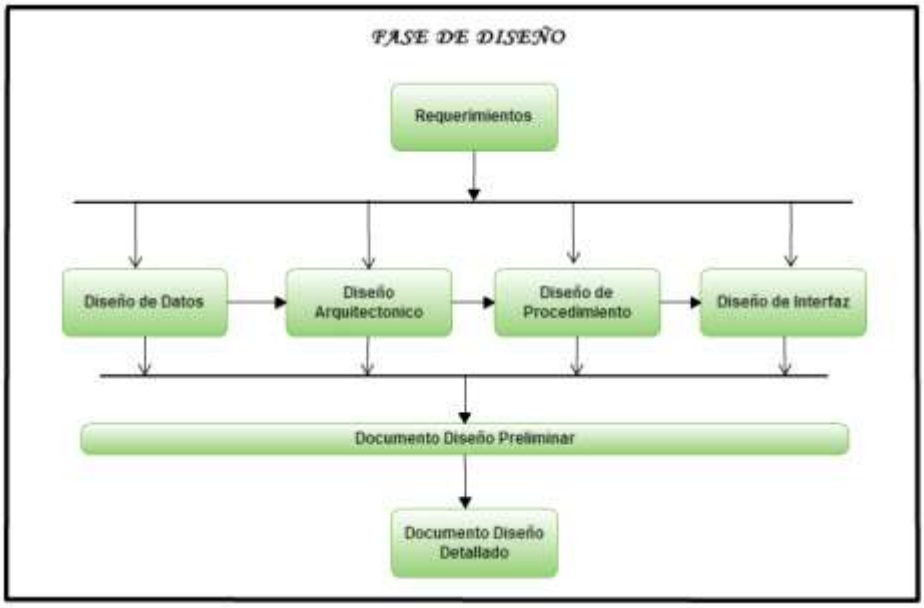

Figura 2. Procesos de la Fase de Diseño.

3.1.2.1 Diseño de Datos: Se toma el documento de requerimientos y se definen aspectos, como los tipos de datos que se requieren para lograr el funcionamiento correcto de toda la aplicación, al igual, estos datos pueden ser registros, archivos o bases de datos. De igual forma se tiene que definir aquellas personas o entidades que van a estar involucradas, al igual que las relaciones que se pueden presentar entre estos dependiendo el tipo de procesos u operaciones que deba cumplir la aplicación. Es de suma importancia definir las reglas que se deben seguir esto para establecer los criterios de cómo se deben realizar los procesos para que ejecuten estrictamente lo que se desea.

El diseño de datos consiste en definir completamente los procesos y las características de los datos de la aplicación, este puede ser un proceso donde se perfecciona desde lo elemental como los datos que requiere la aplicación hasta los procesos en los cuales intervienen, con un buen diseño de datos se puede garantizar, que el acceso a los datos de la aplicación se llevara de manera ágil, y se podrán mantener, al igual que puede aceptar mejoras a los datos que se requieran realizar más adelante. En este proceso se identifican los datos, la definición de tipo de estos datos, y los mecanismos de almacenamiento adecuados, también se definen reglas de empresa y mecanismos de exigencia los cuales garanticen la integridad de los datos [8].

3.1.2.2 Diseño Arquitectónico: En este se define la relación entre los principales elementos estructurales del software. Patrones de diseño que se puedan presentar, y que ayuden a cumplir los requisitos que se han definido para el sistema. Y las diferentes restricciones que puedan afectar la aplicación de los patrones de diseño arquitectónico. Es el proceso que identifica todos los subsistemas existentes y establece un marco para el control y la comunicación entre todos estos subsistemas, se identifican los principales componentes del sistema y las comunicaciones que puedan existir entre estos. Existen ventajas con el diseño arquitectónico que pueden

facilitar este proceso, la comunicación con los usuarios, esto facilita la discusión entre diferentes usuarios involucrados, otra es el análisis del sistema, lo que permite tener control sobre los requerimientos críticos del sistema, rendimiento, fiabilidad, y mantenibilidad y por último la reutilización a gran escala, esto dado a que la arquitectura del sistema es a menudo la misma para sistemas con requerimientos similares, y esto permite que se pueda llevar a cabo una reutilización a gran escala, a veces es posible desarrollar arquitecturas que se pueden utilizar en varios sistemas [8].

3.1.2.3 Diseño Procedimental: Este transforma los elementos estructurales en una descripción procedimental del software. Es la especificación mediante herramientas de modelado (Star UML, Vsparading) tales como algoritmos, diagramas de flujo, del funcionamiento interno de cada uno de los módulos que compone el sistema, definidos en los subproceso Arquitectónico y las relaciones u operaciones que deben realizar dentro de la aplicación final, esto permite al desarrollador tener una visión más detallada del procedimiento de desarrollo que se debe realizar, estos diagramas y algoritmos definidos se convertirán en parte funcional de la aplicación ya que contienen los componentes primarios y secundarios, las relaciones y operaciones que se deben realizar entre ellos para cumplir cada uno de los requerimientos del cliente [9].

3.1.2.4 Diseño de Interfaz: En esta se define cada una de las interfaces que se van a utilizar para permitir la interacción de la aplicación como tal, existen diversos aspectos que se deben tener en el diseño de interfaz, aspectos como navegación, amigabilidad, colores, ayuda, control de acceso, textos, entre otros. Teniendo en cuenta los anteriores aspectos podemos garantizar que el diseño de interfaz va a cumplir con la necesidad para la cual es creado, ya que este finalmente va ser el utilizado por el cliente como mecanismo para interactuar y operar los diferentes componentes que vienen inmersos en la aplicación final [10].

3.1.2.5 Documento de Diseño Preliminar: Se centra en la transformación de los requerimientos de la fase de análisis, en datos y la arquitectura propia del desarrollo de software.

3.1.2.6 Documentos de Diseño Detallado: Se ocupa del refinamiento y de la representación arquitectónica que lleva a una estructura de datos refinada y a las representaciones a través de diagramas de flujo del software.

3.1.3 FASE DE CODIFICACIÓN

Después de terminar la fase de diseño, se inicia la fase de codificación lo que en el lenguaje de ingeniería se conoce como la programación, se toman los algoritmos y se llevan a un lenguaje de programación determinado, este lenguaje es escogido dependiendo su utilidad y la necesidad de la aplicación si es adaptable o no a lo requerido por el programador. Después de realizar un análisis a la base teórica de esta fase se procede a proponer los subprocesos de codificación, depuración del código, pruebas de unidad, pruebas de componentes, pruebas de integración, documento técnico como se observa en la siguiente figura.

Figura 3. Procesos de la Fase de Codificación

3.1.3.1 Codificación: Este proceso lo compone los subprocesos de *"Código Fuente"* el primer paso en la codificación y consta de líneas de texto, que son instrucciones que se deben seguir para ejecutar el programa, *"Código Objeto"* es el resultado de llevar a cabo la compilación del código fuente, *"Código Ejecutable"* es el que nos permite saber que la compilación tuvo éxito, y verifica que en el programa no existan errores de manejo, de esta forma si no se encuentran errores se dice se dice que el programa funciona correctamente, ya que está libre de errores de variables, signo y otros, *"Código Compilador"* es el programa informático que permite traducir el código fuente de un programa en lenguaje de alto nivel (Lenguaje cercano a cómo piensa el humano), a un lenguaje de bajo nivel (Lenguaje de máquina), de esta forma el compilador reporta los posibles errores encontrados en el código fuente, *"Código Máquina"* es el que proviene de la tarea de compilación efectuada directamente sobre el código fuente, con el que se obtiene posteriormente el (Bytecode), que es la integración de distintos archivos que forman parte de ejecutables para que el ordenador pueda hacer uso del código programado anteriormente [11]..

3.1.3.2 Depuración Del Código: Es el proceso de identificar la raíz de un error y corregirlo, la depuración representa el 50% del tiempo de desarrollo de un programa, la mayoría de errores y fallas son mecanográficas, las cuales pueden ser detectadas con facilidad, con la ayuda de la observación del código o con ayuda del depurador.

3.1.3.3 Pruebas De Unidad: Estas también llamadas pruebas de caja blanca, o pruebas modulares ya que estas nos permiten determinar si un módulo del programa está listo y correctamente terminado, para estas pruebas se debe tener en cuenta el tamaño de los módulos y preferiblemente separarlos acuerdo a su funcionalidad. El objetivo de las pruebas unitarias es el aislamiento de partes de código y demostrar que estas no contienen errores. Los beneficios de estas pruebas son Simplificación de la integración, Refactorización del código, Documentación, y Diseño.

3.1.3.4 Pruebas De Componentes: En este se llevan a cabo pruebas a componentes de manera individual, con este se logra encontrar defectos internos dentro de este. Por medio de un caso de prueba se debe probar la funcionalidad de ese componente de esta manera se confirma la buena ejecución del componente, de igual forma con estas pruebas se determina la resistencia que este pone a entradas de datos inválidos.

3.1.3.5 Pruebas De Integración: También llamadas pruebas de interfaz comprueba la integración entre los diferentes componentes después de la integración de todos los módulos o componentes que integran el sistema.

3.1.3.6 Documento Técnico: En el documento se describe de forma general la ejecución de cada proceso que se ha codificado, de igual forma se debe describir la funcionalidad y operatividad de cada componente. También se especifica los paquetes que intervienen si se debe realizar algún despliegue, especificación e interacción de la base de datos, especificar como se deben ingresar cada uno de los datos para la ejecución de los componentes o de la aplicación, especificar y demostrar la correcta utilización de cada una de las interfaces utilizadas, y la correcta navegación y ejecución de cada uno de los procesos que se ejecutan dentro de la aplicación por parte del usuario [11].

3.1.4. FASE DE PRUEBAS

En esta fase se realiza una verificación dinámica del comportamiento de la codificación del software contra el comportamiento esperado según los requerimientos definidos en el documento de la fase de análisis, haciendo uso de un conjunto de casos de pruebas, seleccionados de manera adecuada según los diferentes escenarios que se pueden presentar por cada validación puntual del software que se desee probar.

Las pruebas están enfocadas principalmente en la evaluación o valoración de la Calidad de los productos software, permitiendo Encontrar y documentar defectos existentes, Sugerir mejoras según la calidad percibida, Validar que los requisitos estén implementados correcta y completamente [5,6]. A partir del estudio realizado de la fase se dedujo los subprocesos, los cuales contribuyen a la eficiente ejecución de pruebas de software, para posteriormente certificar el desarrollo del software, los cuales se observan en la siguiente figura.

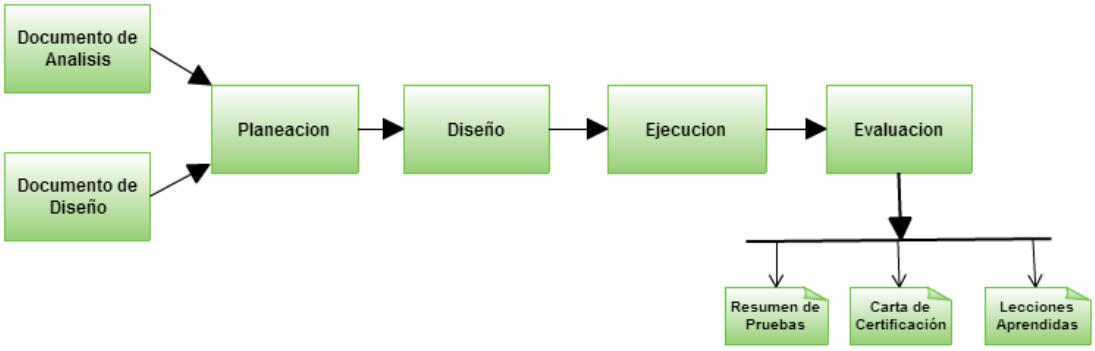

Figura 4. Procesos de la Fase de Pruebas

3.1.4.1 Planeación: Se define como una preparación y planificación en la cual se contextualiza al grupo de pruebas sobre la aplicación o software a probar, se establece el entorno de pruebas, se disponen de los procedimientos y los criterios a aplicar en las pruebas, también se selecciona la estrategia de pruebas adecuada. La planificación

de pruebas es de suma importancia en la fase de pruebas ya que es la base para los subsiguientes procesos de la misma fase, por tanto se debe tener claro los Tipos Pruebas, Numero de Ciclos de pruebas, Recursos Físicos, Recursos humanos, Calendario de pruebas, Restricciones/Limitaciones, Dependencias.

3.1.4.2 Diseño: Identificar y describir los casos de prueba y escenarios identificados en los requerimientos recibidos del cliente, los cuales serán ejecutados posteriormente en la fase de ejecución. Además, permite identificar el conjunto de datos que servirán para ejecutar las pruebas, crear nuevos casos de prueba necesarios que sean detectados en el transcurso del proyecto y controlar la cobertura de los requerimientos dados.

3.1.4.3 Ejecución: Realizar la ejecución de todos los casos de prueba que fueron diseñados en la fase de Diseño y lograr los objetivos propuestos en la fase de Planeación, mediante la adecuada gestión de incidencias encontradas, el mantenimiento de la trazabilidad de los casos de prueba ejecutados y el seguimiento y control del estado de las pruebas.

3.1.4.4 Evaluación: Determinar si los criterios de completitud definidos para un requerimiento han terminado a satisfacción y valorar los resultados generados. Se pretende igualmente establecer la calidad del producto desarrollado y del proceso mismo. Para ello se prepara un informe final con un balance del proyecto, proporcionando una valoración del cumplimiento de las actividades planeadas y de la calidad en general del proyecto y del producto obtenido. Generando como entregables el resumen de pruebas, carta de certificación y el documento de lecciones aprendidas.

3.1.5 FASE DE ENTREGA

En esta fase se realizan las diferentes actividades para poner a disposición de los usuarios la aplicación desarrollada, se hace la preparación de la infraestructura necesaria para configurar el entorno, la instalación de los componentes, la activación de los procedimientos manuales y automáticos asociados, y cuando sea necesario la migración o carga inicial de datos. Adicional a esto se debe cumplir con la capacitación a los usuarios finales, en esta se debe garantizar la correcta puesta en marcha, y la corrección de los errores que se lleguen presentando al momento de la integración con el ambiente del cliente. Donde finalmente se debe obtener la firma del acta de entrega por parte del cliente. [12] En esta fase se entrega la última versión del proyecto probada y aprobada; con el análisis realizado a la base documental se procese a proponer los subprocesos más importantes como se plasman en la siguiente figura.

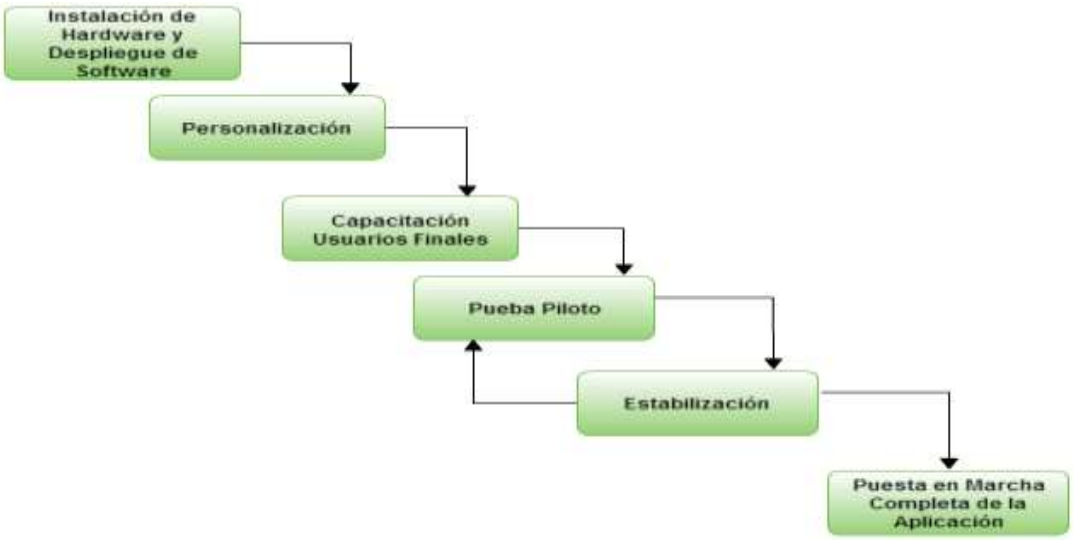

Figura 5. Procesos de la Fase de Implementación

3.1.5.1 Instalación de Hardware y Despliegue de Software: En esta se procede a realizar la instalación de todos los elementos tecnológicos que permitirán instalar y ejecutar la aplicación de proyecto, los cuales deben cumplir requisitos mínimos que se determinaron en la fase de pruebas del proyecto, con esto se garantiza que se posee la estructura de hardware sobre la cual va a estar bien sementado el software que se va a requerir para garantizar la funcionalidad del producto de software.

3.1.5.2 Personalización: En esta se debe adaptar la aplicación a los requerimientos del usuario final, en esta se ajustan los parámetros establecidos en el software a las particularidades que necesita el cliente.

3.1.5.3 Capacitación a Usuarios Finales: En esta fase se realiza un plan de capacitación a usuarios finales, donde se debe impartir tanto conocimiento de la información que controlara la aplicación, como la forma de operación de la misma, con esto se cumple con un requisito indispensable. Se imparte capacitación a los administradores de la aplicación, como a los usuarios finales, el objetivo es no crear dificultades ni resultados erróneos con la ejecución de la aplicación.

3.1.5.4 Prueba Piloto: Es aquella que se realiza en un ambiente vivo del cliente en donde se observa su funcionamiento y los errores que se van evidenciando se van corrigiendo sobre la marcha de esta prueba esta es indispensable ya que es la antesala a la entrega final del proyecto al cliente.

3.1.5.5 Estabilización: En este proceso se van integrando cada una de las funcionalidades contenidas en el proyecto, donde se realizan unas pruebas llamadas “pruebas beta”, con estas al ejecutarlas se va observando su operación y se van corrigiendo los errores, esto se realiza hasta que se hayan incorporado todas las funcionalidades, con esto lo que se busca es tener un producto estable operacionalmente.

3.1.5.6 Puesta en Marcha de la Aplicación Completa: En esta se tiene la aplicación en funcionamiento directamente en el ambiente del cliente, en esta finalmente es donde se obtiene la aceptación del cliente y se firma el acta de entrega del producto de software [13].

3.2 GESTIÓN DE RIESGOS EN LAS FASES DEL DESARROLLO DE SOFTWARE EN LA EMPRESA PRIVADA

Los riesgos según el PMBOOK se define como la probabilidad de que ocurra un evento ya sea positivo o negativo, en caso de ser positivo se considera como una oportunidad y si es negativo representa una amenaza, que puede afectar la ejecución del proyecto ya sea en el alcance, tiempo, costo, y calidad del producto o servicio.

Los Tipos de Riesgos son ***"Riesgos Conocidos"*** que hacen referencia a los que han sido identificados y analizados, lo cual permite planificar el plan de respuesta a estos y se les debe asignar una reserva para contingencias, los *"Riesgos Desconocidos"* hacen referencia a los riesgos que se desconocen y se les puede asignar una reserva de gestión. También es importante tener en cuenta que existe una *"Probabilidad de Ocurrencia*" que hace referencia a la medida para estimar la posibilidad de que ocurra un incidente o evento.

3.2.1 Relación del Riesgo con las fases del desarrollo de software

Los riesgos se reflejan desde el inicio del proyecto, por esta razón se debe realizar la gestión en el grupo de procesos de planificación del proyecto. De igual forma esta gestión debe ser aplicada a cada una de las fases del desarrollo de software, lo cual permitirá garantizar la disminución de riesgos presentes en estas, evitando sobre costos y demoras en el proyecto. Lo anterior se representa en la siguiente figura.

Figura 6. Relación del Riesgo con las fases del desarrollo de Software

3.2.2 Gestión De Riesgos

La gestión de riesgos puede definirse como el proceso sistemático de identificación, análisis, respuesta a los riesgos y control y seguimiento, esto aplicado a cada una de las fases del desarrollo de software en la empresa privada de caso de estudio [1]. Su objetivo principal es minimizar la probabilidad y las consecuencias de los eventos negativos.

Para la gestión de riesgos del proyecto de desarrollo de software de la empresa privada se va utilizar la siguiente metodología con los procesos de identificación, análisis cualitativo, plan de respuesta y control y seguimiento de los riesgos, la cual se estableció basada en la guía del PMBOK.

Figura 7. Gestión de Riesgos.

3.2.2.1 Identificación

Se tomó como base la documentación del proyecto, y la información sobre proyectos similares en los que se ha tenido participación, al igual que las fuentes de información secundaria. Se realizó un listado de posibles riesgos que se pueden materializar en cada una de las fases del desarrollo de software, se realizó el análisis de la causa raíz asociada a cada uno de los riesgos identificados por cada una de las fases.

3.2.2.2 Análisis Cualitativo

En este proceso se toma como entrada la lista de riesgos identificados y analizados en cada una de las fases. Se utiliza la tabla de probabilidad con cinco niveles, esta nos ubica el riesgo en un nivel de ocurrencia, los cuales son raro, improbable, posible, probable, casi seguro, como se muestra en la siguiente figura.

Tabla 1. Niveles de Probabilidad

NIVEL	DESCRIPTOR	DESCRIPCION
1	Raro	El evento puede ocurrir sólo en circunstancias excepcionales.
2	Improbable	El evento puede ocurrir en algún momento.
3	Posible	El evento podría ocurir en algún momento.
4	Probable	El evento probablemente ocurrirá en la mayoría de las circunstancias.
5	Casi seguro	Se espera que en evento ocurra en la mayoría de las circunstancias.

Fuente. Basado en la guía del PMBOK

Para medir el nivel de impacto causado por los riesgos en el momento de llegarse a materializar, los cuales pueden afectar directamente los objetivos del proyecto, como alcance, tiempo, costo y calidad. Para esta establecer esta medida se utiliza la tabla de impactos donde se especifican cinco niveles insignificante, menor, moderado, mayor, catastrófico. Estos se muestran en la siguiente figura.

Tabla 2. Niveles de Impacto.

NIVEL	DESCRIPTOR	DESCRIPCION
1	Insignificante	Si el hecho llegara a presentarse, tendría consecuencias o efectos mínimos.
2	Menor	Si el hecho llegara a presentarse, tendría bajo impacto.
3	Moderado	Si el hecho llegara a presentarse, tendría mediano impacto.
4	Mayor	Si el hecho llegara a presentarse, tendría alto impacto.
5	Catastrófico	Si el hecho llegara a presentarse, tendría desastrosas consecuencias.

Fuente. Basado en la guía del PMBOK

Para calificar cada uno de los riesgos identificados se realiza la matriz de probabilidad e impacto, en esta se refleja la multiplicación del valor numérico del nivel de probabilidad de ocurrencia del riesgo por el nivel numérico del impacto del riesgo sobre los objetivos, este resultado es el que determina el nivel del riesgo.

Tabla 3. Probabilidad e Impacto.

PROBABILIDAD	IMPACTO				
	Insignificante (1)	Menor (2)	Moderado (3)	Mayor (4)	Catastrófico (5)
Raro (1)					
Improbable (2)					
Posible (3)					
Probable (4)					
Casi seguro (5)					

Fuente. Basado en la guía del PMBOK

Con base en el resultado de la matriz de probabilidad e impacto, se propone una escala numérica la cual permite clasificar el riesgo según su nivel, estos pueden ser muy bajo, bajo, medio, alto, y muy alto.

Tabla 4. Nivel de Riesgo.

NIVEL DE RIESGO	PROBABILIDAD X IMPACTO
Muy Alto	>80
Alto	51- 80
Medio	31-50
Bajo	11 - 30
Muy Bajo	<10

Fuente. Basado en la guía del PMBOK

3.2.2.3 Plan De Respuesta

Se procede a priorizar los riesgos según su nivel de riesgo Muy alto y alto para proponer el plan de contingencia por cada uno de los riesgos priorizados, haciendo uso de las estrategias para amenazas las cuales son evitar, transferir, mitigar y aceptar.

3.2.2.4 Control Y Seguimiento

Para cada uno de los riesgos que se les da respuesta, así mismo se designa a un responsable el cual debe supervisar la correcta ejecución del plan de contingencia de riesgos, al igual este debe evaluar si existe la necesidad de replantear los riesgos existentes o si se debe identificar nuevos riesgos, también se debe realizar una revisión periódica (cada dos meses) para validar el estado y/o afectación sobre cada una de las fases del desarrollo de software.

4. RESULTADOS Y DISCUSIONES

Para cada una de las fases de desarrollo de software se realizó un análisis detallado de cómo funciona cada fase para posteriormente basado en la experiencia y el apoyo de fuentes de información secundaria se procediera a listar los riesgos más

significativos y que en la realidad se pueden dar, teniendo en cuenta diversos factores tales como la lógica del negocio, el compromiso por parte del cliente, los requerimientos, la calidad al realizar el diseño, codificación y pruebas, conocimiento y experiencia por parte del personal, manejo de herramientas y metodologías entre otros, lo cual dio pauta para proponer el listado de riesgos para cada fase y posteriormente aplicar paso a paso la respectiva gestión de riesgos propuesta y explicada en el numeral 3.2.2, lo anterior genera como resultado la matriz de evaluación de riesgos por cada una de la fases del desarrollo de software, con un listado de 43 riesgos los cuales se identificaron por cada una de las fases del desarrollo de software, donde los niveles de riesgos que se arrojaron fueron Muy Alto, Alto, Medio y Bajo teniendo mayor relevancia los tres primeros respectivamente.

Para la fase de análisis se identificó siete riesgos con nivel de riesgo Muy alto uno, Alto uno y Medio cinco, en la fase de diseño se identificó seis riesgos con nivel de riesgo Muy alto uno, Alto dos y Medio tres, en la fase de codificación se identificaron once riesgos con nivel de riesgo Muy alto uno, Alto tres, Medio seis y bajo uno, en la fase de pruebas se identificaron diez riesgos con nivel de riesgo Muy Alto dos, Alto dos, Medio cinco y bajo uno, en la fase de entrega del producto se identificó nueve riesgos con nivel de riesgo Muy Alto uno, Medio seis y bajo dos, ver tabla 5.

Después de tener la matriz de evaluación de riesgos, se priorizo trece riesgos en total por todas las fases con nivel Muy alto y alto para realizarles el respectivo plan de respuesta. Para la fase de Análisis se seleccionaron los riesgos con nivel Muy alto R-001 y Alto R-002, en la fase de diseño se seleccionó los riesgos con nivel Muy alto R-011, Alto R-012 y R-013, en la fase de Codificación se seleccionó los riesgos con nivel Muy Alto R-020, Alto R-017, R-018 y R-019, en la fase de pruebas se seleccionó los riesgos con nivel Muy Alto R-025, R-026, Alto R-027 y R-028, en la fase de Entrega de producto se seleccionó el riesgo R-019 con nivel Muy alto.

Tener claro que en la realidad, se debe realizar plan de respuesta a todos los riesgos identificados y evaluados sin importar su nivel de riesgo, en este trabajo por efectos académicos solo se realizara el plan de respuesta a los riesgos anteriormente mencionados. Por cada uno de los riesgos se realizó un análisis para deducir cuál de las estrategias de plan de respuesta como son evitar, transferir, mitigar y aceptar para riesgos tipo amenaza propuestas por la guía del PMBOK les aplicaba, donde como resultado dio para cada uno de los riesgos el tipo de plan de respuesta es Mitigar ya que esta permite implementar acciones tempranas para reducir la probabilidad de ocurrencia de un riesgo y/o su impacto sobre el proyecto. Para proponer las estrategias de mitigación por cada riesgo se hizo con base a los conocimientos y experiencia adquirida al participar en la mayoría de las fases del desarrollo de software y de igual manera con el apoyo del estudio realizado por cada una de las fases, ver tabla 6.

Tabla 5. MATRIZ DE EVALUACION DE RIESGOS POR CADA UNA DE LA FASES DEL DESARROLLO DE SOFTWARE

ANALISIS

CÓDIGO DEL RIESGO	DESCRIPCIÓN DEL RIESGO	FASE AFECTADA	CAUSA RAÍZ	ENTREGABLES AFECTADOS	ESTIMACIÓN PROBABILIDAD	OBJETIVO AFECTADO	ESTIMACIÓN IMPACTO	PROBABILIDAD X IMPACTO	NIVEL DE RIESGO
R-001	Requerimientos incompletos o ambiguos.	Analisis	Los Requerimientos no se definieron de manera clara y completa.	Documento de requerimientos.	5	Alcance	4	20	Muy Alto
						Tiempo	5	25	
						Costo	5	25	
						Calidad	4	20	
						Total Probabilidad x Impacto		90	
R-002	Falta de acompañamiento de los usuarios en el levantamiento de requerimientos.	Analisis	Usuarios que no colaboran o no se compreten con la definición de los requerimientos	Documento de requerimientos.	4	Alcance	5	20	Alto
						Tiempo	4	16	
						Costo	5	20	
						Calidad	4	16	
						Total Probabilidad x Impacto		72	
R-003	Retrasos en la especificacion de requerimientos.	Analisis	La reuniones con el cliente para el levantamiento de requerimientos se posponen. Las especificaciones de las interfaces esenciales no estan a tiempo.	Documento de requerimientos.	3	Alcance	3	9	Medio
						Tiempo	4	12	
						Costo	4	12	
						Calidad	3	9	
						Total Probabilidad x Impacto		42	
R-004	Incorporación continua de nuevos requerimientos.	Analisis	El cliente no tiene claridad de lo que desea. Necesidad nueva por por parte del mercado, del gobierno o del negocio.	Documento de requerimientos.	3	Alcance	4	12	Medio
						Tiempo	4	12	
						Costo	4	12	
						Calidad	3	9	
						Total Probabilidad x Impacto		45	
R-005	Modificacion continua de requerimientos.	Analisis	Actualizacion necesaria debido a una deficiente definicion de requerimientos incialmente.	Documento de requerimientos.	4	Alcance	3	12	Medio
						Tiempo	3	12	
						Costo	3	12	
						Calidad	3	12	
						Total Probabilidad x Impacto		48	
R-006	Modificaciones incorrectas de las especificaciones	Analisis	Actualizacion incorecta de los requerimientos devido a la ausencia de un estudio detallado previo.	Documento de requerimientos.	3	Alcance	4	12	Medio
						Tiempo	4	12	
						Costo	4	12	
						Calidad	4	12	
						Total Probabilidad x Impacto		48	
R-007	Entendimiento errado de los requerimientos	Analisis	El ingeniero de requerimientos entiende y documenta de manera equivocada las necesidades expuestas por el cliente.	Documento de requerimientos.	3	Alcance	4	12	Medio
						Tiempo	4	12	
						Costo	4	12	
						Calidad	4	12	
						Total Probabilidad x Impacto		48	

Fuente. Plantilla adaptada con base a la Fuente: http://dharmacon.net/herramientas/gestion-proyectos-formatos/

DISEÑO

R-008	Incorrecta definicion y estructuracion de los datos establecidos.	Diseño	Pobre definicion de tipos de datos e integridad, y poco entendimiento sobre la relacion o dependencia de los mismos	Documento de Diseño Detallado.	3	**Alcance**	**3**	**9**	**Medio**
						Tiempo	**3**	**9**	
						Costo	**3**	**9**	
						Calidad	**3**	**9**	
						Total Probabilidad x Impacto		**36**	
R-009	Diseño de interfaces incompleto	Diseño	Desconocimiento de todas las interfases que pueden afectar la solución	Documento de Diseño Detallado.	3	**Alcance**	**4**	**12**	**Medio**
						Tiempo	**4**	**12**	
						Costo	**4**	**12**	
						Calidad	**3**	**9**	
						Total Probabilidad x Impacto		**45**	
R-010	Subestimacion del tamaño de la aplicación.	Diseño	Al realizar el diseño se subestima el software con respecto a las necesidades del cliente.	Documento de Diseño Detallado.	3	**Alcance**	**4**	**12**	**Medio**
						Tiempo	**4**	**12**	
						Costo	**5**	**15**	
						Calidad	**3**	**9**	
						Total Probabilidad x Impacto		**48**	
R-011	Falta de Especificación de la arquitectura logica	Diseño	No se Define adecuadamente las interconexiónes y recursos logicos entre módulos del sistema de manera apropiada para su diseño detallado y administración.	Documento de Diseño Detallado	5	**Alcance**	**4**	**20**	**Muy Alto**
						Tiempo	**5**	**25**	
						Costo	**5**	**25**	
						Calidad	**4**	**20**	
						Total Probabilidad x Impacto		**90**	
R-012	Falta de Especificación de la arquitectura fisica	Diseño	No se define correctamentete el conjunto de dispositivos fisicos que se va utilizar para que la arquitectura logica funcione correctamente.	Documento de Diseño Detallado	3	**Alcance**	**4**	**12**	**Alto**
						Tiempo	**4**	**12**	
						Costo	**5**	**15**	
						Calidad	**4**	**12**	
						Total Probabilidad x Impacto		**51**	
R-013	Desconocimiento de la logica de negocio	Diseño	Mala interpretacion y/o interpretacion superficial de los requisitos para hacer el diseño detallado del sistema	Documento de Diseño Detallado	3	**Alcance**	**5**	**15**	**Alto**
						Tiempo	**4**	**12**	
						Costo	**5**	**15**	
						Calidad	**4**	**12**	
						Total Probabilidad x Impacto		**54**	

CODIFICACION

R-014	Bajo rendimiento de la herramienta CASE	Codificacion	Las herramientas CASE que se utilizan como apoyo no tienen el rendimiento y las funcionalidades esperadas	Implementacion del software.	3	**Alcance**	**3**	**9**	**Medio**
						Tiempo	**4**	**12**	
						Costo	**3**	**9**	
						Calidad	**4**	**12**	
						Total Probabilidad x Impacto		**42**	
R-015	Manejo inadecuado en liberacion de versiones	Codificacion	Despliegue incompleto de version de la aplicación, El no despliegue de la ultima la version de la aplicación , Despliegue de version con direccionamiento equivocada a bases de datos.	Implementacion del software.	3	**Alcance**	**3**	**9**	**Medio**
						Tiempo	**4**	**12**	
						Costo	**4**	**12**	
						Calidad	**3**	**9**	
						Total Probabilidad x Impacto		**42**	

R-016	Falta de documentacion en codigo fuente	Codificacion	Limitacion del tiempo. Aplicación de malas practicas de desarrollo y ausencia de revisiones	Implementacion del software.	3	**Alcance**	**2**	**6**	**Bajo**
						Tiempo	**3**	**9**	
						Costo	**2**	**6**	
						Calidad	**3**	**9**	
						Total Probabilidad x Impacto		**30**	
R-017	Modificacion cronograma actividades	Codificacion	Actividades no contempladas. Adicion de nuevas actividades. Complejidad del desarrollo de actividades no estimadas. Retrasos en la ejecucion de actividades por improvistos indirectos.	Implementacion del software.	4	**Alcance**	**3**	**12**	**Alto**
						Tiempo	**4**	**16**	
						Costo	**4**	**16**	
						Calidad	**3**	**12**	
						Total Probabilidad x Impacto		**56**	
R-018	No disponibilidad de hardware y/o software.	Codificacion	El hardware y/o software esencial no es entregado a tiempo.	Implementacion del software.	4	**Alcance**	**5**	**20**	**Alto**
						Tiempo	**5**	**20**	
						Costo	**4**	**16**	
						Calidad	**4**	**16**	
						Total Probabilidad x Impacto		**72**	
R-019	El Software es complejo de implementar	Codificacion	El desarrollo de la aplicacion tiene un nivel alto de complejidad. El modelado del sistema realizado en la fase de diseño no fue tan clara y especifica.	Implementacion del software.	4	**Alcance**	**4**	**16**	**Alto**
						Tiempo	**4**	**16**	
						Costo	**4**	**16**	
						Calidad	**4**	**16**	
						Total Probabilidad x Impacto		**64**	
R-020	Compleja la integración de módulos del software	Codificacion	Al codificar y comenzar la integración se hace evidente que la especificación está incompleta o contiene requisitos contradictorios o hay falencias en el diseño del software.	Implementacion del software.	5	**Alcance**	**4**	**20**	**Muy Alto**
						Tiempo	**5**	**25**	
						Costo	**5**	**25**	
						Calidad	**4**	**20**	
						Total Probabilidad x Impacto		**90**	
R-021	Retiro de personal con conocimiento y experiencia	Codificacion	Al ser las unicas personas que maneja ciertos temas especificos y/o complejos al irsen generan retraso en el curso de las tareas.	Implementacion del software.	4	**Alcance**	**3**	**12**	**Medio**
						Tiempo	**3**	**12**	
						Costo	**3**	**12**	
						Calidad	**3**	**12**	
						Total Probabilidad x Impacto		**48**	
R-022	No hay buena comunicación y/o sinergia en el equipo.	Codificacion	La comunicación entre el personal del area de desarrollo no es el mas optimo y asi mismo su sinergia no es la mas eficaz para el cumplimiento de objetivos en comun.	Implementacion del software.	3	**Alcance**	**4**	**12**	**Medio**
						Tiempo	**4**	**12**	
						Costo	**4**	**12**	
						Calidad	**4**	**12**	
						Total Probabilidad x Impacto		**48**	
R-023	Falta de conocimiento y Experiencia sobre las tareas asignadas y las herramientas a utilizar.	Codificacion	El personal no es idoneo o no tiene la espertise necesaria para el rol asignado.	Implementacion del software.	3	**Alcance**	**4**	**12**	**Medio**
						Tiempo	**4**	**12**	
						Costo	**4**	**12**	
						Calidad	**4**	**12**	
						Total Probabilidad x Impacto		**48**	
R-024	Pérdida de backups	Codificacion	Perdida de la copia de seguridad de la version de software actual causado por virus o por remplazo de version sin sacar la copia previamente.	Implementacion del software.	3	**Alcance**	**3**	**9**	**Medio**
						Tiempo	**4**	**12**	
						Costo	**4**	**12**	
						Calidad	**3**	**9**	
						Total Probabilidad x Impacto		**42**	

PRUEBAS

R-025	Alcance de las pruebas No definido completamente.	Pruebas	No se definio desde el inico de la fase el alcance debido a que no se tenia documentacion o la que existia era muy superificial o estaba desactualizada	Aseguramiento de calidad del software.	5	Alcance	4	20	Muy Alto
						Tiempo	5	25	
						Costo	5	25	
						Calidad	5	25	
						Total Probabilidad x Impacto		95	
R-026	Documentación de requisitos insuficiente, desactualizada, contradictoria o ambigua.	Pruebas	Los casos de prueba no quedan cubiertos en su totalidad, debido a que pueden existir cambios y/o mejoras que no se encuentran actualizados a la fecha.	Aseguramiento de calidad del software.	5	Alcance	5	25	Muy Alto
						Tiempo	5	25	
						Costo	5	25	
						Calidad	5	25	
						Total Probabilidad x Impacto		100	
R-027	Realizar pruebas en ambiente desarrollo	Pruebas	Alta Inestabilidad del ambiente Funcionalidades probadas pero no certificadas al 100%.	Aseguramiento de calidad del software.	4	Alcance	4	16	Alto
						Tiempo	4	16	
						Costo	5	20	
						Calidad	5	20	
						Total Probabilidad x Impacto		72	
R-028	No se realiza completitud en las pruebas.	Pruebas	El conjunto de pruebas realizadas no son lo suficientes para garantizar la caidad del software esto sucede por omision y/o por falta de tiempo.	Aseguramiento de calidad del software.	4	Alcance	4	16	Alto
						Tiempo	4	16	
						Costo	4	16	
						Calidad	4	16	
						Total Probabilidad x Impacto		64	
R-029	No se realiza priorizacion en las ejecucion de las pruebas	Pruebas	No se le da prioridad para probar las funcionalidades más importantes y complejas del software Al final se descubre defectos bloqueantes los cuales nesecita tiempo para ser solucionados.	Aseguramiento de calidad del software.	3	Alcance	4	12	Medio
						Tiempo	4	12	
						Costo	4	12	
						Calidad	4	12	
						Total Probabilidad x Impacto		48	
R-030	Demoras excesivas en la reparación de defectos encontrados en las pruebas	Pruebas	Solucion de defectos no priorizada por parte de los desarrolladores lo cual retrasa las pruebas.	Aseguramiento de calidad del software.	3	Alcance	3	9	Medio
						Tiempo	4	12	
						Costo	4	12	
						Calidad	3	9	
						Total Probabilidad x Impacto		42	
R-031	Problemas de disponibilidad con el ambiente de pruebas.	Pruebas	Problemas con el alistamiento, adecuación y estabilización del ambiente donde se ejecutan las pruebas, afectando cronogramas y retrasando el inicio de cada ciclo.	Aseguramiento de calidad del software.	3	Alcance	4	12	Medio
						Tiempo	4	12	
						Costo	4	12	
						Calidad	4	12	
						Total Probabilidad x Impacto		48	
R-032	Retraso Testing debido a nuevos errores despues de despliegues	Pruebas	Retrabajo causado por despliegues los cuales dañan funcionalidades ya esxitosas, generando asi nuevos defectos.	Aseguramiento de calidad del software.	3	Alcance	4	12	Medio
						Tiempo	4	12	
						Costo	4	12	
						Calidad	4	12	
						Total Probabilidad x Impacto		48	
R-033	Pobre Productividad	Pruebas	Tiempos muertos en subfases iniciales de la fase de pruebas que no se pueden recuperar por entregas tardias de desarrollo.	Aseguramiento de calidad del software.	3	Alcance	2	6	Bajo
						Tiempo	3	9	
						Costo	2	6	
						Calidad	2	6	
						Total Probabilidad x Impacto		27	
R-034	No hay suficientes recursos y/o ingresan demasiado tarde	Pruebas	Ingreso tardio del personal a las roles necesitados. Personal reducido donde se nesecitan mas de los asignados.	Aseguramiento de calidad del software.	3	Alcance	4	12	Medio
						Tiempo	4	12	
						Costo	4	12	
						Calidad	4	12	
						Total Probabilidad x Impacto		48	

ENTREGA DEL PRODUCTO

R-035	Capacitacion superficial a usuarios finales	Entrega del Producto.	Por limitacion o subestimacion del tiempo se realiza una capacitacion incompleta sobre el uso de la aplicacion.	Puesta en produccion del software.	3	**Alcance**	**4**	**12**	**Medio**
						Tiempo	**4**	**12**	
						Costo	**4**	**12**	
						Calidad	**4**	**12**	
						Total Probabilidad x Impacto		**48**	
R-036	La aplicacion no procesa transacciones por segundo como se esperaba.	Entrega del Producto.	La capacidad a nivel de hardware no es acorde al numero de transacciones solicitadas. La codificacion del procedimiento de la transaccion es poco eficiente en el tiempo de respuesta.	Puesta en produccion del software.	3	**Alcance**	**4**	**12**	**Medio**
						Tiempo	**4**	**12**	
						Costo	**4**	**12**	
						Calidad	**4**	**12**	
						Total Probabilidad x Impacto		**48**	
R-037	Fallas del hardware limitan la funcionalidad del software	Entrega del Producto.	Inestabilidad de la red y/o Internet. Caida o afectacion por virus de los servidores.	Puesta en produccion del software.	3	**Alcance**	**4**	**12**	**Medio**
						Tiempo	**4**	**12**	
						Costo	**4**	**12**	
						Calidad	**4**	**12**	
						Total Probabilidad x Impacto		**48**	
R-038	Arquitectura inadecuada por parte del cliente	Entrega del Producto.	Especificacion superficial de los requisitos basicos para la arquitectura hardware y software. Modificacion de la arquitectura con respecto a la definida inicialmente.	Puesta en produccion del software.	3	**Alcance**	**4**	**12**	**Medio**
						Tiempo	**4**	**12**	
						Costo	**4**	**12**	
						Calidad	**4**	**12**	
						Total Probabilidad x Impacto		**48**	
R-039	Documentacion sobre el uso de la aplicación.	Entrega del Producto.	Generación pobre de documentos necesarios para la instalacion y uso efectivo de la aplicación.	Puesta en produccion del software.	3	**Alcance**	**2**	**6**	**Bajo**
						Tiempo	**3**	**9**	
						Costo	**2**	**6**	
						Calidad	**3**	**9**	
						Total Probabilidad x Impacto		**30**	
R-040	Vulnerabilidades del software presentadas en produccion.	Entrega del Producto.	Omision de validaciones en la fase de pruebas. Ambiente produccion como es un ambiente real se pueden presentar defectos que no se presentaron en el ambiente semireal de pruebas.	Puesta en produccion del software.	3	**Alcance**	**3**	**9**	**Medio**
						Tiempo	**3**	**9**	
						Costo	**5**	**15**	
						Calidad	**5**	**15**	
						Total Probabilidad x Impacto		**48**	
R-041	Resistencia del personal para cambiar las prácticas del pasado	Entrega del Producto.	El personal que va utilizar el nuevo software presenta miedo al cambio debido a la costumbre de utilizar el anterior software.	Puesta en produccion del software.	3	**Alcance**	**2**	**6**	**Bajo**
						Tiempo	**3**	**9**	
						Costo	**2**	**6**	
						Calidad	**2**	**6**	
						Total Probabilidad x Impacto		**27**	
R-042	Software contiene numerosos errores cuando se entrega al cliente.	Entrega del Producto.	El cliente por el afan de salir a produccion toma el riesgo de salir con defectos existentes en la aplicación que aun no se han solucionado por parte del equipo de desarrollo.	Puesta en produccion del software.	3	**Alcance**	**3**	**9**	**Medio**
						Tiempo	**3**	**9**	
						Costo	**5**	**15**	
						Calidad	**5**	**15**	
						Total Probabilidad x Impacto		**48**	
R-043	Presentacion de defectos en ambiente produccion.	Entrega del Producto.	Hallazgo de defectos que no se detectaron previamente o que no se presentaron en el ambiente de pruebas.	Puesta en produccion del software.	5	**Alcance**	**4**	**20**	**Muy Alto**
						Tiempo	**4**	**20**	
						Costo	**5**	**25**	
						Calidad	**5**	**25**	
						Total Probabilidad x Impacto		**90**	

Tabla 6. MATRIZ DE MITIGACION DE RIESGOS POR CADA UNA DE LAS FASES DEL DESAROLLO DE SOFTWARE

GRUPO DE PROCESOS	AREA CONOCIMIENTO	ACTIVIDAD
Planificación	Gestión de los Riesgos	Plan de mitigación a los Riesgos con nivel Muy Alto y Alto en las fases de desarrollo de software en una empresa privada.

NOMBRE DEL PROYECTO	SIGLAS O ABREVIATURA DEL PROYECTO
GESTION DE RIESGOS EN PROYECTO DE SOFTWARE A DESARROLLAR EN EMPRESA PRIVADA	HW: Hardware; SW: Software.

[1] Evitar, Mitigar, Transferir, Aceptar.

CÓDIGO DEL RIESGO	AMENAZA / OPORTUNIDAD	DESCRIPCIÓN DEL RIESGO	FASE	NIVEL DE RIESGO	TIPO DE RESPUESTA[1]	RESPONSABLE	PLAN DE MITIGACION
R-001	Amenaza	Requerimientos incompletos o ambiguos.	Análisis	Muy Alto	Mitigar	Líder Funcional	*Usuarios e ing. requerimientos capacitarlos sobre la lógica del Negocio. *Usuarios tener claro lo que desean. *Listado de preguntas sobre los temas poco claros en reuniones previas. *Incorporar los nuevos requerimientos o los cambios necesarios de forma clara y completa para que se cumpla con la funcionalidad solicitada.
R-002	Amenaza	Falta de acompañamiento de los usuarios en el levantamiento de requerimientos.	Análisis	Alto	Mitigar	Líder Funcional	*Reuniones periódicas con el cliente. *Participación del usuario en la definición de requerimientos. *Compromiso y responsabilidad por parte de los usuarios.
R-011	Amenaza	Falta de Especificación de la arquitectura lógica	Diseño	Muy Alto	Mitigar	Líder de Diseño de sistemas	*Definir la arquitectura lógica correcta y más eficiente con base a la especificación de requerimientos. *Revisión y apoyo por parte del líder de desarrollo. *Realizar el diseño de la arquitectura lógica teniendo en cuenta la arquitectura que posee el cliente. *Utilizar modelos, vistas y diagramas para el diseño de la arquitectura lógica.
R-012	Amenaza	Falta de Especificación de la arquitectura física	Diseño	Alto	Mitigar	Líder de Diseño de sistemas	*Tener la última versión aprobada del documento de especificación de requerimientos. *Definir la arquitectura física correcta y más eficiente con base a la especificación de requerimientos.

Fuente. Plantilla adaptada con base a la Fuente: http://dharmacon.net/herramientas/gestion-proyectos-formatos/

CÓDIGO DEL RIESGO	AMENAZA / OPORTUNIDAD	DESCRIPCIÓN DEL RIESGO	FASE	NIVEL DE RIESGO	TIPO DE RESPUESTA[1]	RESPONSABLE	PLAN DE MITIGACION
							*Revisión y apoyo por parte del líder de desarrollo e infraestructura. *Realizar el diseño de la arquitectura física teniendo en cuenta la arquitectura que posee el cliente. *Utilizar modelos, vistas y diagramas para el diseño de la arquitectura física.
R-013	Amenaza	Desconocimiento de la lógica de negocio	Diseño	Alto	Mitigar	Líder de Diseño de sistemas	*Capacitación sobre la lógica del negocio a los encargados del diseño. *Facilitación de documentación sobre la lógica del negocio de la empresa. *Reuniones para la aclaración de dudas sobre temas puntuales. *Asignar personal proactivo y con experiencia.
R-020	Amenaza	Compleja la integración de módulos del software	Codificación	Muy Alto	Mitigar	Líder de Desarrollo	*Tamaño del módulo relativamente pequeño, para minimizar el impacto al hacer un cambio, corrupción de error o un rediseño. *Independencia modular, permite de manera más fácil y flexible trabajar con ellos. Al desarrollar un nuevo modulo no es necesario conocer detalles internos de otros módulos *Realizar programación en equipo y desarrollar módulos paralelamente. *Realizar pruebas de componentes.
R-017	Amenaza	Modificación cronograma actividades	Codificación	Alto	Mitigar	Líder de Desarrollo	*Recibir a tiempo los documentos de diseño. *Recibir contextualización y apoyo por parte del equipo de diseño. *Tener claro el alcance del desarrollo de la aplicación. *Realizar una buena planeación de recursos, tareas y tiempos para evitar posibles desfases. * Incluir en la planeación un tiempo racional por si ocurren imprevistos indirectos.
R-018	Amenaza	No disponibilidad de hardware y/o software.	Codificación	Alto	Mitigar	Líder de Desarrollo	*Definir en el plan de desarrollo de software los requisitos tanto de equipos físicos como herramientas software necesario para la codificación de la aplicación. *Gestionar por parte del equipo de desarrollo con anterioridad los recursos HW y SW para poder ejecutarse las tareas planeadas. *Tener plan alterno por si llegan a fallar algunos de los recursos software y/o software.
R-019	Amenaza	El Software es complejo de implementar	Codificación	Alto	Mitigar	Líder de Desarrollo	*Apoyo al equipo de desarrollo por parte de un experto en el tema. *Utilizar un modelo de desarrollo de software de acuerdo al tamaño de la aplicación, tiempos, documentación, etc. *Apoyo entre los integrantes del grupo. *Realizar reutilización de software. *Realizar pruebas unitarias.

CÓDIGO DEL RIESGO	AMENAZA / OPORTUNIDAD	DESCRIPCIÓN DEL RIESGO	FASE	NIVEL DE RIESGO	TIPO DE RESPUESTA[1]	RESPONSABLE	PLAN DE MITIGACION
R-025	Amenaza	Alcance de las pruebas No definido completamente.	Pruebas	Muy Alto	Mitigar	Líder de Pruebas	*Reuniones de contextualización con el desarrollador y funcional. * Aclarar dudas y recibir apoyo por parte del funcional. *Aprobación del plan de pruebas por parte de los funcionales.
R-026	Amenaza	Documentación de requisitos insuficiente, desactualizada, contradictoria o ambigua.	Pruebas	Muy Alto	Mitigar	Líder de Pruebas	*Los funcionales deben entregar la última versión de la documentación la cual debe estar actualizada hasta la fecha. *Funcionales deben informar y explicar los cambios que se den. *Funcionales deben entregar las nuevas versiones de los documentos.
R-027	Amenaza	Realizar pruebas en ambiente desarrollo	Pruebas	Alto	Mitigar	Líder de Pruebas	*Realizar las pruebas en un ambiente aislado del de desarrollo.
R-028	Amenaza	No se realiza completitud en las pruebas.	Pruebas	Alto	Mitigar	Líder de Pruebas	*Usar técnicas de pruebas y buenas prácticas para cubrimiento total de las pruebas. *Asignar Analistas de calidad con conocimiento y experiencia sobre el tema. *Realizar una óptima planeación de ejecución de actividades. *Realizar pruebas de usuario. *Los desarrolladores no deben ser parte del grupo de pruebas.
R-028	Amenaza	Presentación de defectos en ambiente producción.	Entrega del Producto.	Muy Alto	Mitigar	Líder de Estabilización	* Realizar pruebas en ambiente preproducción. * Realizar pruebas piloto en ambiente producción. *No pasar a producción con defectos ya que estos pueden generar nuevos defectos.

[1] Evitar, Mitigar, Transferir, Aceptar.

CONCLUSIONES

- ✓ Es importante entender de forma específica y clara cada una de las fases del desarrollo de software, ya que de estas se deben identificar los riesgos más importantes que las afectan.

- ✓ La identificación de los riesgos es de suma importancia ya que permitirá concentrarse en los riesgos que se pueden llegar a materializar en cada fase del desarrollo del software.

- ✓ Al definir el nivel de la probabilidad de ocurrencia y el nivel de impacto de los riesgos, permitirá hallar el nivel de riesgo y así focalizarse con mayor grado de atención a los que den niveles críticos.

- ✓ Se debe realizar una adecuada determinación del nivel del riesgo por cada uno de los riesgos para proponer el plan de respuesta que más se ajuste al riesgo.

- ✓ Es de vital importancia la propuesta de planes de respuesta de riesgos que garanticen un adecuado control y seguimiento sobre estos.

- ✓ La matriz de riesgos permite orientar al encargado de riesgos para tomar y aplicar el mejor plan de mitigación sobre el riesgo en cuestión.

- ✓ Es necesario contar con personal que maneje el tema de tratamiento, control y seguimiento de los riesgos presentados en las diferentes fases del desarrollo del proyecto de software.

- ✓ Se deben tener planes de respuesta claros, actualizados y efectivos, que permitan el control y un adecuado manejo de los riesgos para reducir la probabilidad y/o impacto de ocurrencia.

- ✓ Se debe tener implementado el sistema gestor de riesgos para tener un control y seguimiento sobre los riesgos encontrados analizados y mitigados en cada una de las fases del desarrollo de software.

- ✓ Actualmente no hay libros específicamente sobre la gestión de riesgos en las fases de desarrollo de software.

REFRENCIAS BIBLIOGRAFICAS

[1] Por temas de confidencialidad se reserva el nombre de la compañía objeto del análisis y se hará referencia a una Entidad Financiera.

[2] Project Management Institute, I. (2013). GUÍA DE LOS FUNDAMENTOS PARA LA DIRECCIÓN DE PROYECTOS (Guía del PMBOK). Estados Unidos de América.: Quinta Edición.

[3] Roger S. Pressman. Ingeniería del Software. Mc Graw Hill. 155-173p.

[4] María del Carmen Gómez Fuentes. (2011). Análisis De Requerimientos. México D.F. Universidad Autónoma Metropolitana. 3-11p.

[5] Ian Sommerville. (2005). Ingeniería de Software. Madrid (España). Pearson Educación, S.A.

[6] International Software Quality Institute. (2005). ISTQB Certified Tester Foundation Level. Germany. CHOUCAIR.

[7] Alejandro Peña Ayala. (2006). Ingeniería de Software. México. Instituto Politécnico Nacional. 67-80p

[8] Roger S. Pressman. Ingenieria del Software. Mc Graw Hill 201-207p

[9] Gustavo Marcelo Torossi. (2005). Diseño de Sistemas En: http://exa.unne.edu.ar/informatica/anasistem2/public_html/apuntes/de1.pdf (12/10/2015)

[10] Willian J. Giraldo, Cesar A. Collazos, Faber D. Giraldo (2009). Desarrollo Basado en Modelos de la Interfaz de Usuario de Sistemas. En: http://www.bdigital.unal.edu.co/23464/1/20375-68787-1-PB.pdf (12/10/2015)

[11] José M, Drake, Patricia López. (2009). Ingeniería de Programación. En: http://www.ctr.unican.es/asignaturas/Ingenieria_Software_4_F/Doc/M7_09_VerificacionValidacion-2011.pdf. (11/10/2015)

[12] Institución Universitaria de Envigado. (2008). Gerencia de Proyectos. En: http://www.iue.edu.co/documents/emp/aspectosGenProyecto.pdf. (10/10/2015)

[13] Barrios Judith. (2010). Conceptualización del Proceso de Implementación de Software. En: http://erevistas.saber.ula.ve/index.php/cienciaeingenieria/article/viewFile/1147/1102. (12/10/2015)

[14] Software Testing Help (2015). How to Manage Risks During Test Planning Phase – Risk Based Testing. http://www.softwaretestinghelp.com/risk-management-during-test-planning-risk-based-testing/

[15] Sebastián Uchitel (2007). Ingeniería de Requerimientos. http://www-2.dc.uba.ar/materias/isoft1/teoricas_2009_1/01-Introduccion_IR_BN.pdf

[16] Maniasi S. D. Identificación de Riesgos de Proyectos de Software en Base a Taxonomías. Tesis de Magíster En Ingeniería de Software (2005). Universidad de Postgrado ITBA. Pag 460.

[17] Del Rio J. L. Sistema De Asistencia A La Gestión De Riesgos En Proyectos Software De Sistemas Industriales De Automatización Y Control. Tesis de Magíster En Ingeniería de Software (2006). Universidad de Postgrado ITBA. Pag 361.

[18] Ledo N, Palacios D. Sistema Experto para la Identificación de Riesgos en el Desarrollo de Software (2007). Universidad de Buenos Aires, Facultad de Ingeniería. Pag 156.