UNIVERSIDAD CATÓLICA ANDRÉS BELLO
VICERECTORADO ACADÉMICO
DIRECCIÓN GENERAL DE LOS ESTUDIOS DE POSTGRADO
AREA DE CIENCIAS ADMINISTRATIVAS Y DE GESTIÓN
Postgrado en Gerencia de Proyectos

Trabajo Especial de Grado

"Elaborar el plan estratégico del proyecto de desarrollo de software que permita gestionarlas actividades de soporte técnico canalizados mediante un HelpDesk"

Presentado por
David Tineo Rangel

Para optar al título de
Especialista en Gerencia de Proyectos

Asesor
Ing. Estrella Bascarán Castanedo

ACEPTACIÓN DEL TUTOR

Por la presente hago constar que he leído el trabajo especial de grado, presentado por DAVID ALFONSO TINEO RANGEL portador de la cédula de identidad número: 13.860.979. para optar al título de Especialista en Gerencia de Proyectos, cuyo título es: **"Elaborar el plan estratégico del proyecto de desarrollo de software que permita gestionarlas actividades de soporte técnico canalizados mediante un HelpDesk"**, manifiesto que cumple con los requisitos exigidos por la Dirección General de los Estudios de Postgrado de la Universidad Católica Andrés Bello razón la cual lo considero APTO para ser evaluado por el jurado que se decida designar para tal fin.

En Caracas a los 15 días del mes de Diciembre de 2009.

Ing° Estrella Bascarán Castanedo
C.I. 05.968.206

ÍNDICE GENERAL

INTRODUCCIÓN 1

CAPÍTULO I: PROPUESTA DE LA INVESTIGACIÓN 3

1.1.- Planteamiento y Delimitación del Problema 3

1.2.- Justificación e Importancia del trabajo de investigación 7

1.3.- Objetivos 9

1.3.1.- Objetivo General 9

1.3.2.- Objetivos Específicos 9

1.4.- Factibilidad de la investigación y consideraciones éticas 10

CAPÍTULO II: MARCO CONCEPTUAL 12

2.1.- *Information Technology (IT)* 12

2.2.- Definición de *HelpDesk* 13

2.2.1.- Software utilizado para la gestión del *HelpDesk* 15

2.3.- Dirección de Proyectos 17

2.3.1.- Gestión de integración del proyecto 18

2.3.2.- Gestión del alcance del proyecto 18

2.3.3.- Gestión del tiempo del proyecto 19

2.3.4.- Gestión de los costes del proyecto 19

2.3.5.- Gestión de la calidad del proyecto 19

2.3.6.- Gestión de los recursos humanos del proyecto 20

2.3.7.- Gestión de las comunicaciones del proyecto 20

2.3.8.- Gestión de los riesgos del proyecto 21

2.3.9.- Gestión de las adquisiciones del proyecto 21

2.4.- Gerencia de proyectos del Software 23

2.4.1.- Personal 23

2.4.2.- Producto 24

2.4.3.- Proceso 24

2.4.4.- Proyecto 24

2.5.- Definición de Software 25

2.6.- Marco de trabajo genérico para el desarrollo de software 26

2.6.1.- Identificación de problemas, oportunidades y objetivos 27

2.6.2.- Determinación de los requerimientos de información 27

2.6.3.- Análisis de las necesidades del sistema 28

2.6.4.- Diseño del sistema recomendado 28

2.6.5.- Desarrollo y documentación del software 28

2.6.6.- Prueba y mantenimiento del sistema 29

2.6.7.- Implementación y evaluación del sistema 29

2.7.- Desarrollo ágil de software 29

2.8.- Programación Extrema 30

CAPÍTULO III: MARCO METODOLÓGICO 32

3.1.- Tipo de Investigación 32

3.2.- Unidad de análisis 32

3.3.- Plan de ejecución del proyecto 33

3.4.- Recolección, procesamiento y análisis de los datos 33

3.5.- Operacionalización de los objetivos 35

3.6.- Técnicas e instrumentos de recolección de datos 36

3.7.- Definición conceptual y operacional de variables 37

3.8.- Diseño de la investigación 38

CAPÍTULO IV: DISEÑO DEL PLAN 39

4.1. Determinación del alcance del proyecto 39

4.2. WBS del proyecto 42

4.3. Determinar entregables del proyecto 43

4.4. Tiempos de ejecución 44

4.5. Presupuestos de costo del proyecto 46

4.6. Calidad del proyecto 47

4.7. Metodología del trabajo a implementar 52

4.8. Tecnologías a utilizar 52

4.9. Estructura organizativa del proyecto 54

4.10. Perfil de los recursos requeridos 54

4.11. Matriz de responsabilidad del proyecto 57

4.12. Gestión de comunicaciones del proyecto 58

4.13. Identificación de los riesgos del proyecto 59

4.14. Matriz de adquisiciones del proyecto 64

CAPÍTULO V: CONCLUSIONES Y RECOMENDACIONES 65

Bibliografía 67

Otras referencias 70

ÍNDICE DE TABLAS

Operacionalización de objetivos 35

Definición de variables en el proyecto de investigación 37

Operacionalización de variables 37

Estimado de costos clase V 46

Estimado de costos por recursos 47

Estimado de costos por insumos requeridos 47

Estimado de costos por licencias de software requeridos 47

Estimado de costos por insumos requeridos 47

Estimado de costos por insumos requeridos 47

Perfil de cargo para líder de proyecto 54

Perfil de cargo para líder de proyecto 54

Perfil de cargo para consultor Sr. 55

Perfil de cargo para consultor QA 55

Perfil de cargo para DBA 55

Perfil de cargo para consultor Jr. 55

Perfil de cargo para analista testing 56

Matriz de responsabilidad del proyecto 57

Tabla de identificación de riesgos 62

Tabla de priorización de riesgos 63

Matriz de impacto en dimensión Tiempo 63

Matriz de impacto en dimensión Costo 63

Matriz de impacto en dimensión Calidad 63

Matriz de impacto en dimensión Alcance 63

Matriz de adquisiciones del proyecto 64

ÍNDICE DE FIGURAS

Niveles de atención del *HelpDesk* dentro de IT 14

Esquema de atención de incidencias 15

Plan de ejecución del proyecto 33

Árbol de problemas planteado para el proyecto 41

Árbol de objetivos planteado para el proyecto 41

WBS del proyecto 42

Diagrama de red del proyecto 45

Diagrama Gantt del proyecto 45

El modelo CRE 51

Organigrama del proyecto 54

UNIVERSIDAD CATÓLICA ANDRÉS BELLO
VICERECTORADO ACADÉMICO
DIRECCIÓN GENERAL DE LOS ESTUDIOS DE POSTGRADO
AREA DE CIENCIAS ADMINISTRATIVAS Y DE GESTIÓN
Postgrado en Gerencia de Proyectos

"Elaborar el plan estratégico del proyecto de desarrollo de software que permita gestionarlas actividades de soporte técnico canalizados mediante un HelpDesk"

Autor: Lic. David Tineo Rangel
Asesor: Ing° Estrella Bascarán Castanedo
Año: 2.009

RESUMEN

El presente trabajo tiene como finalidad diseñar un plan de ejecución que permita crear una solución estándar para la gestión diaria del proceso de HelpDesk mediante una interfaz de usuario amigable.

En Venezuela la gran mayoría de los proyectos de desarrollo de software no son analizados de forma profunda antes de dar inicio al proyecto, todos estos son llevados de forma empírica y en su totalidad poseen un grave problema de alcance y tiempo, lo cual repercute principalmente en su costo final.

El principal objetivo de esta investigación será elaborar el plan de ejecución para el desarrollo de un software especializado que permita gestionar, organizar y controlar las actividades de soporte técnico canalizados mediante un HelpDesk.

Es necesario evolucionar de un desarrollo de proyectos de software llevado al estilo "Indiana Jones", quien al encontrar alguna dificultad simplemente decía: "¡No te preocupes, pensaré en algo!" (Pressman, 2005) hacia una metodología eficiente que planifique, estime y considere los recursos tanto humanos, tecnológicos y económicos disponibles que permitan llevar a buen término cualquier proyecto de software, que aunque quizá parezca utópico en estos tipos de proyectos, terminen en el tiempo

planificado, con una excelente calidad y bajo los costos estimados además de satisfacer la necesidad del cliente

PALABRAS CLAVE

Proyecto, plan de proyecto, HelpDesk, desarrollo de software

INTRODUCCIÓN

El *HelpDesk* es un servicio prestado en muchas empresas que busca ofrecer a los usuarios internos e inclusive externos un medio de contacto que permita la notificación y posterior resolución de incidencias relacionadas con el aspecto tecnológico que afecten el diario desempeño laboral, este contacto es realizado bien sea vía telefónica o de forma electrónica mediante el uso del correo electrónico, intranets, mensajería instantánea, etc.
Las incidencias recibidas mediante el servicio de *HelpDesk* van desde fallas en equipo informático (Hardware y Periféricos) hasta aplicaciones o software especializado, todo esto en base a la labor realizada por el usuario que presente la incidencia.

Dada la gran cantidad de información que puede ser generada mediante el *HelpDesk* se hace necesario poder administrar de forma efectiva a los recursos involucrados a fin de generar diversos indicadores que permitan a los altos niveles gerenciales tomar diversas decisiones estratégicas que vayan enfocadas hacia la mejora del servicio, lo cual se traduce en mejoras de la productividad y en los niveles de satisfacción de los usuarios.
Actualmente en el mercado mundial se han desarrollado diversas aplicaciones que se especializan en la administración efectiva de las funciones del *HelpDesk*, las cuales pueden ser adquiridas con la compra de una licencia del software.

Estas aplicaciones ofrecen en sus contratos la implantación de diversos elementos operativos como centrales telefónicas, suites de software, cubos de datos para *business intelligence*, además de un contrato de mantenimiento y adiestramiento para los usuarios. Desafortunadamente estas casas de software ofrecen estas soluciones en mercados determinados en Europa y América del Norte, siendo difícil la implantación de estos en América Latina.

En el caso de Venezuela existe un vacío en este mercado en particular, ya que muchas empresas requieren la automatización de sus departamentos de *HelpDesk* y no cuentan con una herramienta apropiada, lo cual conlleva en la mayoría de los casos destinar recursos para el desarrollo interno de soluciones similares con el fin de satisfacer esta necesidad de forma temporal, descuidando así otros requerimientos internos.

Tomando en cuenta los últimos aspectos planteados se ofrece mediante esta investigación un plan estratégico para desarrollar una solución que ofrezca al mercado local una alternativa que satisfaga la necesidad de los clientes de contar con una herramienta que permita controlar de forma efectiva cada factor que interviene en el proceso de *HelpDesk,* además de un plan de acción efectiva que pueda ser adaptado para nuevos desarrollos de software de acuerdo a la necesidad planteada.

Esta investigación se distribuye de la siguiente manera:

Capítulo I – Propuesta de la investigación: Se describe la problemática actual, se expone la justificación e importancia de este trabajo y se plantean los objetivos generales y específicos que permitirán finalizar esta investigación de forma satisfactoria.

Capítulo II – Marco Conceptual: se definen aquellos conceptos necesarios para la comprensión de aquellos elementos que intervienen de forma directa en la investigación.

Capítulo III – Marco metodológico: Se plantea el tipo de investigación, el plan de ejecución del proyecto, se analizan los datos necesarios para la evaluación de forma cuantitativa de la problemática y se establece el diseño de la investigación.

Capítulo IV – Diseño del Plan: Para este capítulo se combinan y aplican los planteamientos definidos en las áreas del conocimiento del *Project Management Institute* con las diversas metodologías para la ingeniería de software, con lo cual se obtiene como resultado un plan estratégico que permitirá el desarrollo del software definido para esta investigación.

Capítulo V – Conclusiones y Recomendaciones: Describe las conclusiones generales de la investigación así como una serie de recomendaciones para futuras investigaciones de esta naturaleza.

Capítulo I.
PROPUESTA DE LA INVESTIGACION

Con el avance tecnológico de las últimas décadas, el diario desempeño de una organización requiere de la existencia de sistemas especializados que permitan el procesamiento y continua reutilización de la información generada por las tareas diarias de su ciclo de vida, y que además son vitales para determinar su permanencia en el mercado. Para velar por el correcto desempeño de estos sistemas es necesaria la existencia de un ente encargado de la administración de la plataforma informática de la organización, es conocido mayormente como *Information Technologies* o IT como será mencionada a partir de este instante. Desde el punto de vista de estructura organizativa puede estar bajo la figura de Gerencia, VicePresidencia o Departamento.

Gracias a la evolución de las telecomunicaciones, el surgimiento de la Internet, el desarrollo de la tecnología móvil y la consecuente evolución de los equipos computacionales y la popularización de estándares para la gerencia de IT como la *Information Technology Infrastructure Library* ó ITIL®[1] según sus siglas en inglés, la figura de IT logró evolucionar como una parte vital dentro del ciclo de vida de la organización.

1.1.- PLANTEAMIENTO Y DELIMITACION DEL PROBLEMA

Dentro de la estructura interna de IT tenemos al Departamento de Soporte Técnico, cuya función principal es ofrecer un completo menú de soluciones a los usuarios para cualquier incidencia a nivel informático, desde errores de hardware originados por uso rutinario o desconocimiento del usuario hasta mejoras del rendimiento del computador y protección de virus informáticos (Alex, 2008), además de instalación de software que facilite el diario desempeño de las labores cotidianas del usuario.

[1] ITIL® son mejores prácticas para IT, desarrolladas a finales de 1980 por la Agencia Central Británica para la Informática y las Telecomunicaciones (CCTA por sus siglas en inglés). Entre los beneficios de ITIL® tenemos: reducción de costos, mejora de los servicios de IT, mejora de la satisfacción del cliente e incremento de la productividad, habilidades y experiencia de los profesionales de IT; el 26 de junio de 2007 es presentada la tercera versión de este código enfocado a un nivel estratégico, ITIL® V3. (Xynco, 2009).

A medida de que estas funciones se volvieron más exigentes y las incidencias más complejas se evoluciona de un soporte técnico en sitio hacia un nuevo tipo de soporte remoto, con lo cual los casos eran atendidos vía telefónica.

Con base a la aplicación de los estándares de calidad planteados por la metodología ITIL®, con lo cual se obtiene una nueva definición que va mucho más allá de lo que se ha planteado, este concepto se denomina *HelpDesk*, cuya función principal sería: ofrecer, a partir de las instalaciones del cliente, un punto único de contacto ó SPOC (*Single Point Of Contact*) para los usuarios internos del cliente.

El *HelpDesk* es un área especializada en soporte técnico, por teléfono o vía acceso remoto, a equipos de informática y telecomunicaciones con la finalidad de resolver todos los incidentes y problemas que los clientes puedan tener (Itautec, 2009).

El proceso de *HelpDesk* es llevado a cabo mediante a los siguientes componentes (Instituto Mexicano de Teleservicios, 2009):

- Una *suite* de software que simplifica las actividades diarias de gestión de incidencias, entre los cuales tenemos *sistemas de administración de llamadas* los cuales permiten a los usuarios reportar sus inquietudes, se mencionan igualmente *sistemas de administración del conocimiento*, *software de monitoreo* y *herramientas de control de inventarios*.

- Un *Call Center* compuesto de un *staff* de técnicos que cuentan con el conocimiento y la destreza necesaria para la atención de todos los requerimientos y necesidades que pueda tener un usuario durante sus actividades diarias, mediante la oportuna aplicación de estándares ITIL® que gracias a su enfoque sistemático y profesional permite la mejora continua de los servicios de IT mediante la aplicación de mejores prácticas.

Sin embargo, todo esto trae una gran inquietud a los líderes que conforman la unidad de IT, ¿cómo medir de forma real la eficiencia de nuestros técnicos cumpliendo con la

expectativa de los usuarios y si se está logrando resolver estas incidencias en un tiempo que no afecte las operaciones diarias de nuestros clientes? Para hacer cuantificable y medible este nivel de atención y solución de casos existen en el mercado algunas herramientas que permiten canalizar las incidencias presentadas.

Tenemos a Resolve IT! desarrollado por Troika, este software posee como principal característica ofrecer una solución integral de *HelpDesk* que se adapte a las necesidades de cada empresa. Permite resolver cualquier tipo de problema de forma ordenada, rápida y eficiente, y además ayudar a organizar y controlar los activos de las organizaciones logrando una mayor productividad corporativa con la consecuente reducción de costos de soporte.

El analista de *HelpDesk* permite a los usuarios internos (empleados) y externos (clientes) ser los beneficiarios de la aplicación al conseguir una eficaz solución real de los problemas a tiempo, llevando registros, reasignaciones y seguimientos del usuario reportado hasta la satisfactoria solución del problema. (Troika, 2009).

También tenemos al software denominado *ServiceDesk Plus* presentado por la marca AdventNet. El mismo se ofrece en tres ediciones: *Standard*, *Professional* y *Enterprise.* En la versión *Standard,* la cual es la más básica de las versiones, tenemos los siguientes módulos: *HelpDesk* (manejo de requerimientos), *Self-Service Portal* (permite consultar la base de datos del *Knowledge Base* y rastrear las incidencias abiertas), *Knowledge Base* (habilita el uso del *Knowledge Base* de la aplicación para los técnicos), *SLA Management* (administra los SLA y permite la creación de SLA personalizados) y *HelpDesk Reports* (motor de reportes para esta herramienta que permite la búsqueda mediante parámetros determinados) (AdventNet inc, 2009).

Con base en las referencias anteriores, se plantea para esta investigación diseñar un plan de ejecución que sirva como guía para la construcción de un software con las características básicas que posee una solución estándar para la gestión diaria del proceso de *HelpDesk* como: administración de usuarios, administración de SLA, gestión de incidencias, gestión de un *Knowledge Base*, reportes, control de inventarios mediante una interfaz de usuario amigable.

Sin embargo, la diferencia radica en que la mayoría de las soluciones existentes poseen una licencia muy costosa además de que las casas de software que la desarrollan están fuera de Venezuela y no poseen filiales en el país y el soporte técnico es un costo adicional. A esto se suma el hecho de que estas herramientas requieren una central telefónica, la cual, por diversos motivos, no está disponible en algunas empresas, excepto aquellas cuya razón de ser no sea otra sino la atención telefónica. Por eso se propone una aplicación "hecha en casa", ofreciendo las siguientes bondades:

- Al ser desarrollada bajo la filosofía de ambiente web garantizaría el acceso de los usuarios que existan dentro de la organización mediante la intranet de la empresa.

- Será configurable a las características de infraestructura de IT que la empresa disponga aprovechando al máximo los recursos a nivel de servidores de aplicaciones y de base de datos sin platear la adquisición de nuevos servidores o componentes de software adicionales.

- Dispondrá de un motor de reportes que permita a cada usuario obtener reportes personalizados, mediante una interfaz que sea de fácil comprensión al usuario lo cual sería de gran utilidad ya que obtendría una guía que le permita saber dónde está posicionado en cuanto a la opinión del cliente, conocer si están cumpliendo con los tiempos estimados para cada actividad e inclusive conocer si existe alguna carencia de recursos.

Cabe señalar que para la construcción de una solución de tales características, se plantea la siguiente inquietud **¿Cuáles deberían ser los componentes de un plan para el desarrollo de un software especializado que permita gestionar las actividades de soporte técnico canalizados mediante un *HelpDesk*?.**

La respuesta a esta interrogante permitirá vislumbrar el camino adecuado para obtener un plan de ejecución que dé como resultado una guía que defina las actividades a seguir, el personal que hará el trabajo y los mecanismos para valorar los riesgos, controlar los cambios y evaluar la calidad (Pressman, 2005) del software planteado, en este caso la

construcción de un software experto, el cual se define como la automatización del conocimiento de un experto para solucionar un problema específico en una organización, siendo sus principales componentes la base de conocimientos, un motor de inferencia que conecta al usuario con el sistema mediante determinadas transacciones y la interfaz del usuario (Kendall y Kendall, 2005); la aplicación de este concepto a este caso de estudio permitirá definir una solución que incremente la labor productiva del personal de soporte técnico y por consiguiente de los usuarios mediante un entorno amigable y fácil de manejar y que emita además de forma procesada la información manejada por los diversos usuarios que interactúen con la misma.

1.2.- JUSTIFICACIÓN E IMPORTANCIA DEL TRABAJO DE NVESTIGACIÓN

El plantear un plan de ejecución para el desarrollo de un software, es una actividad indispensable, ya que la construcción de software de computadora es una empresa compleja que puede involucrar a muchas personas en un período de tiempo determinado. En este proceso se encuentran involucradas diferentes actividades que determinarán si el producto final cumple con las expectativas planteadas en un inicio, entre estas actividades resaltan la planificación del tiempo y de los recursos además de la supervisión y control del personal, ya que estas actividades permitirán la evolución del producto desde su concepto preliminar hasta su implementación operativa, todo esto mediante una metodología que nos permita la comprensión de las cuatro P en la gerencia de proyectos de software: *personal, producto, proceso y proyecto*. El *personal* debe estar organizado para realizar el trabajo de software con efectividad. La comunicación con el cliente y otros involucrados debe ocurrir de modo que sean comprensibles el ámbito y los requisitos del *producto*. Se debe seleccionar un *proceso* adecuado para el *personal* y el *producto*, y por último el *proyecto* se debe planificar para estimar el esfuerzo y el tiempo para cumplir las labores del trabajo (Pressman, 2005).

Al plantear el desarrollo de un software tenemos una serie de pasos, que vienen determinados por una metodología de desarrollo de software denominado programación extrema o PE que debemos cubrir, los cuales son los siguientes: *planeación*, *codificación*, *diseño* y *pruebas* (*Extreme Programming*, 2009).

Pressman (2005) define las fases de la programación extrema de la siguiente forma:

En la *planeación* se definen las actividades y los tiempos de desarrollo de cada una de estas actividades, con esta actividad se estima abarcar la fase de *inicio* del ciclo de vida del proyecto de acuerdo a lo planteado por el PMI (*Project Management Institute*, 2004).

En la fase de *diseño* se determina una guía de implementación para una historia escrita, esta fase es equivalente a la etapa de *planificación*.

Con la *codificación* se procede al desarrollo de cada una de las actividades definidas previamente en la fase de planeación, con esta fase se cubre la etapa de *ejecución*.

En *pruebas* se validan cada uno de los entregables realizados durante el proceso de codificación para su futura implementación, lo cual nos permite realizar el *cierre* del proyecto.

Es muy importante señalar que el fundamento de la programación extrema es la rápida respuesta a nuevas necesidades que puedan surgir durante el ciclo de vida de desarrollo del software, ya que el cliente puede agregar historias, cambiar el valor de la historia existente, dividir historias o eliminarlas. Entonces el equipo de desarrollo considera de nuevo los lanzamientos restantes y modifica sus planes de acuerdo a ello, de allí que las fases de *seguimiento* y *control*, se estén realizando de forma continua e iterativa hasta la culminación del producto.

1.3.- Objetivos

1.3.1.- Objetivo General

Elaborar el plan estratégico del proyecto de desarrollo de software que permita gestionarlas actividades de soporte técnico canalizados mediante un *HelpDesk.*

1.3.2.- Objetivos específicos

- Determinar el alcance del proyecto en firme a través de la aplicación de las técnicas S.M.A.R.T. Mapa de problemas y objetivos.

- Elaborar el WBS del proyecto.

- Conceptualizar los entregables del proyecto mediante la descripción breve de las actividades del proyecto.

- Estimar los tiempos de ejecución para los tiempos más probable, optimista y pesimista mediante la consulta a expertos de la empresa.

- Elaborar el presupuesto de costos del proyecto con un nivel de estimado clase 5.

- Determinar el nivel de calidad del producto deseado, de acuerdo a las exigencias del mercado, además de cumplir con los estándares exigidos para este tipo de Software.

- Determinar la estructura organizativa del proyecto.

- Describir el perfil de los RRHH requeridos para proyecto.

- Elaborar la matriz de responsabilidad del proyecto.

- Diseñar las pautas para plan de gestión de las comunicaciones del proyecto.

- Identificar cualitativamente los riesgos del proyecto.

- Describir la matriz de las adquisiciones de proyecto y su plan de acción.

1.4.- Factibilidad de la investigación y consideraciones éticas

Este proyecto de plan de ejecución desde el punto de vista técnico y económico es totalmente factible ya que se cuentan con los recursos necesarios para llevar a cabo esta investigación, igualmente se cuenta con total acceso a la información ya que al ser un proceso (*HelpDesk*) estándar para la gran mayoría de las organizaciones, la información se encuentra en muchos repositorios de información, como revistas electrónicas, artículos especializados, e-zines, etc.

El tiempo planteado para el desarrollo de esta investigación es de 12 semanas.

Durante esta investigación se utilizarán diversas fuentes de información: Libros de reconocidos autores en el área de informática como: Roger Pressman, Kenneth Kendall y Julie Kendall, Joseph Schmuller, además publicaciones electrónicas provenientes de diversas organizaciones e instituciones como la Universidad de Málaga, el instituto de ingenieros eléctricos y electrónicos (IEEE), la organización internacional para estandarización (ISO), el Consejo Superior de Administración Electrónica del Gobierno de España, además de diversas consultoras en el área de Gerencia de Proyectos de Software, entre otros.

Igualmente se hace referencia al Project Management Book of Knowlege (PMBOK®) version 2004 ya que esta investigación comenzó antes del mes de julio del 2009, cabe señalar que algunas instituciones realizan las certificaciones para PMP® en base a dos fechas de corte: versión 2004 del PMBOK® para exámenes presentados antes del 30 de junio del 2009 y versión 2008 del PMBOK® para los exámenes presentados a partir del 1ro. De julio del 2009 (P&M Consulting and Training Specialists, 2009).

Desde el punto de vista ético, dado que el producto final permitirá realizar un software comercial destinado a cubrir una necesidad para el mercado local e internacional, cabe

señalar que se respetan los derechos de autor de cada fuente de información consultada a lo largo de la investigación ya que ésta fue obtenida de fuentes libres para el acceso al público.

Los resultados de este trabajo estarán disponibles como material técnico y de apoyo para aquellas personas que requieran una guía para futuras investigaciones de esta naturaleza.

Capítulo II.
MARCO CONCEPTUAL

Motivado a que este trabajo de investigación se enfoca principalmente en obtener como resultado final un plan de ejecución para el desarrollo de un software especializado para la medición y gestión de los procesos que se llevan a cabo en un departamento de *HelpDesk*, se requiere previamente definir una serie de conceptos, que permitirán la comprensión de los diferentes elementos que se encuentran involucrados en el proceso de *HelpDesk*, su asociación al proceso de planificación de proyectos y sus particularidades para el caso de desarrollo de software, los cuales son necesarios para la consecución de esta investigación

2.1.- *Information Technology* (IT)

Se puede definir IT como el estudio, diseño, desarrollo, implementación, soporte o administración de sistemas de información, particularmente software o hardware de computador a fin de convertir, almacenar, proteger, procesar, transmitir y resguardar de forma segura información. (*Information Technology Association of America* (ITAA), c.p. Hill, 2008).

Como puede apreciarse el concepto es muy amplio en comparación al paradigma que se mantenía hace 10 años atrás en la cual se sostenía que la labor de IT era algo similar a la de un taller de reparación de autos; aquellas personas tenían el talento, la lógica, y hacían el trabajo, pero solamente se les consultaba por cualquier cosa que no fuera reparaciones (Panner, 2008). Hoy en día IT abarca muchos aspectos de computación y tecnología, así como también muchas otras áreas relacionadas tales como: manejo de data, redes, ingeniería del hardware, diseño de software, diseño de bases de datos y administración de sistemas.

La historia de IT data de años atrás, cuando su administración era llevada por un simple operador de equipos de computación quien respaldaba determinada información en cintas magnéticas las cuales eran luego almacenadas en espacios destinados para este fin. Luego, con el pasar del tiempo y a fin de mejorar las funciones asociadas con el campo de

tecnología, se comenzaron a implementar computadores, servidores, funciones de administración de bases de datos y sistemas criptográficos. (Hill, 2008).

2.2.- Definición de *HelpDesk*

HelpDesk es un equipo de personas especializadas que mediante diversos medios de comunicación como el teléfono, la internet, el correo electrónico, son centro de acopio y de resolución de aquellas inquietudes expuestas por diversos usuarios que se encuentran en la organización, dichas inquietudes pueden ir de las más sencillas, hasta aquellas que son capaces de paralizar de forma parcial las actividades cotidianas de un área clave en específico (Instituto Mexicano de Teleservicios, 2009). Revisar con lo escribiste en el capítulo I suena repetido

Inicialmente el proceso de *HelpDesk* vendría dado por las siguientes fases (Instituto Mexicano de Teleservicios, 2009):

- **Recepción de reportes de eventos**: En esta fase se reciben las incidencias por parte de los usuarios bajo los siguientes criterios (Soporte Remoto de México, 2009):

    - Al existir una interrupción de la operación normal del trabajo.

    - Cuando se requiriera soporte sobre el hardware y/o software instalado.

    - Cuando se requiera la instalación de nuevos productos de hardware, software o periféricos.

    - Al requerir consulta y/o asesoramiento respecto al funcionamiento y/o utilización de los recursos informáticos disponibles. Luego se determinará si es necesario asignar el caso a un técnico que se apersone al puesto de trabajo del usuario o si por el contrario el caso puede ser resuelto de forma remota por el mismo operador de *HelpDesk.*

- **Registros de eventos**: Dichos eventos captados en el proceso anterior, deben ser registrados de forma detallada a fin de que el técnico asignado tenga pleno conocimiento de la incidencia en cuestión.

- **Elevación y notificación de eventos**: En los casos que sea necesario se deben escalar los casos a los niveles superiores pertinentes: administración de redes, desarrollo, proveedores externos, etc. (ver figura 1)

- **Registro de soluciones**: Una vez resueltos los casos los mismos deben ser documentados en el *Knowledge Base* a fin de que en caso de presentarse nuevamente un caso con las mismas características, solamente deba consultarse esta fuente de información para así resolver de forma más eficiente la incidencia.

- **Autoservicio**: En los casos que sea necesario, es buena práctica adiestrar a los usuarios en aquellas solicitudes que no requieran la apertura de un caso en el *HelpDesk*, como por ejemplo el reemplazo de un consumible.

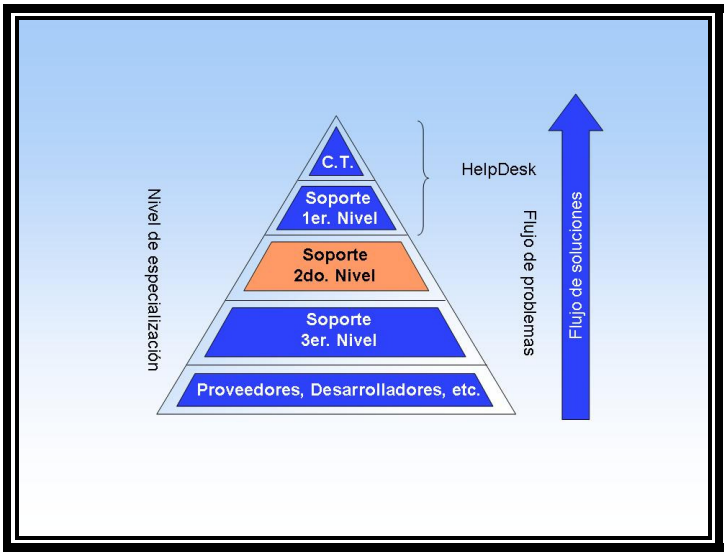

Figura 1. Niveles de Atención del *HelpDesk* dentro de IT
(Fuente: Instituto Mexicano de Teleservicios, 2009)

También tenemos que el avance de la tecnología ha logrado que la actividad de soporte técnico sea realizada de forma más eficiente gracias a la existencia de software

especializado que permite que el técnico realice algunas de sus tareas de una manera más fácil y rápida con el fin de incrementar el nivel de atención y solución de incidencias presentadas en un momento dado (ver figura 2).

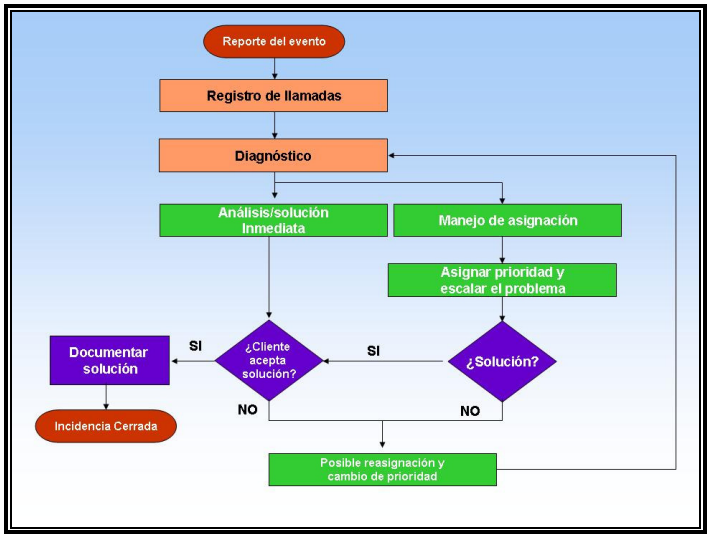

Figura 2. Esquema de atención de incidencias
(Fuente: Instituto Mexicano de Teleservicios, 2009)

2.2.1.- Software utilizado para la gestión del HelpDesk

Entre los diferentes tipos de software con que cuenta el personal de *HelpDesk* tenemos (Instituto Mexicano de Teleservicios, 2.009):

- **Sistemas de administración de llamadas**: Que permiten mediante un menú auditivo y con el uso del teclado digital del teléfono el marcado de ciertas opciones de acuerdo a la necesidad del usuario.

- **Sistemas de administración del conocimiento**: Comúnmente conocidos como *Knowledge Base*, los cuales permiten la documentación de ciertos casos que tengan una frecuencia determinada a fin de facilitar al técnico o al usuario dependiendo de la incidencia, la búsqueda de una solución apropiada entre los casos que hayan sido documentados previamente o en caso contrario, introducir nueva información para futuras eventualidades.

- **Herramientas de monitoreo y soporte remoto**: Estas permiten que el técnico tenga acceso de forma remota a los computadores que se encuentren dentro de la red informática de la empresa o inclusive fuera de la misma, siempre que tengan la permisología administrativa correspondiente, facilitando así la labor cotidiana de atención de incidencias ya que el técnico no debe estarse desplazando entre diferentes lugares para la solución de una incidencia menor.

- **Herramientas de control de inventarios**: la cual permite conocer la cantidad de insumos (periféricos, hardware y licencias de software) existente en el inventario general de IT, conocer bajo qué condiciones están y además mantener un censo de los equipos de computación y periféricos (impresoras, unidades de disco externos, *headsets,* etc.) que se encuentran funcionando dentro de la organización y los diversos programas que tengan instalados.

Para el control de las incidencias o casos reportados, se crean en el proceso de *HelpDesk* los *Service Level Agreement* ó SLA, estos agrupan a los servicios en ciertos niveles de soporte, que engloban características como: naturaleza de la incidencia (falla de hardware, software, falla de comunicación, etc.), grupo de IT que corresponde (administración de redes, desarrollo, proveedor externo) y tiempo de solución de la incidencia (German, 2005). Dichos SLA deben ser continuamente validados para detectar cuáles deben ser adicionados o desincorporados por cambios en el negocio, hardware o software.

Como toda organización IT debe también determinar ciertas métricas que permitan garantizar la continuidad operativa de la empresa y promover la mejora continua del departamento como tal, estas métricas aplicadas al *HelpDesk* permiten conocer el nivel de rendimiento de atención para el usuario y para la organización. Como ejemplo de algunas de estas métricas tenemos (Miller, 2007):

- **Tiempo de espera en cola**: Es el tiempo que espera el usuario en la línea hasta ser atendido por el operador.

- **Tiempo promedio de conversación**: Es el tiempo invertido en la conversación con el usuario.

- **SLA**: Es el tiempo promedio para la solución de una incidencia determinado por el SLA.

- **Solución a la primera llamada**: Promedio de aquellas incidencias resueltas en la primera llamada en las cuales el usuario no continuó reportándolas repetidamente.

Una vez definido lo que sería el proceso de *HelpDesk*, además de los elementos que lo componen, se procede a conceptualizar las fases de la dirección de proyectos que nos permitirá definir un plan que permita la automatización de los elementos mencionados anteriormente.

2.3.- Dirección de Proyectos

Es la aplicación de conocimientos, habilidades, herramientas y técnicas a las actividades de un proyecto para satisfacer los requisitos del proyecto, mediante la aplicación e integración de los procesos de dirección de proyectos (PMI, 2004).

La dirección del proyecto incluye los siguientes aspectos (PMI, 2004):

- Identificar los requisitos.
- Establecer los objetivos claros y posibles de realizar.
- Equilibrar las demandas concurrentes de calidad, alcance, tiempo y costes.
- Adaptar las especificaciones, los planes y el enfoque a las diversas inquietudes y expectativas de los diferentes interesados.

Para lograr la consecución de estos objetivos el PMI organizó los procesos de la dirección de proyectos en nueve áreas de conocimientos (*Project Management Institute*, 2004):

2.3.1.- Gestión de la integración del proyecto

Donde se detallan los procesos y actividades que forman parte de los diversos elementos de la dirección del proyecto, los cuales se resumen en los siguientes:

- Desarrollar el acta de constitución del proyecto.
- Desarrollar el enunciado preliminar del alcance del proyecto.
- Desarrollar el plan de gestión del proyecto.
- Dirigir y gestionar la ejecución del proyecto.
- Supervisar y controlar el trabajo del proyecto.
- Control integrado de cambios.
- Cerrar proyecto.

2.3.2.- Gestión del alcance del proyecto

Describe aquellos procesos necesarios para asegurar que el proyecto incluya todo lo requerido para completar el mismo de forma satisfactoria, y se compone de las siguientes fases:

- Planificación del alcance.
- Definición del alcance.
- Crear EDT.
- Verificación del alcance

- Control del alcance.

2.3.3.- Gestión del tiempo del proyecto

Va enfocado hacia la puntualidad en la conclusión del proyecto y se compone de:

- Definición de las actividades.
- Establecimiento de la secuencia de las actividades.
- Estimación de recursos de las actividades.
- Estimación de la duración de las actividades.
- Desarrollo del cronograma.
- Control del cronograma.

2.3.4.- Gestión de los costes del proyecto

Describe los procesos de planificación, estimación, presupuesto y control de costes, para así completar el proyecto dentro del presupuesto aprobado y viene dado por:

- Estimación de costes.
- Preparación del presupuesto de costes.
- Control de costes.

2.3.5.- Gestión de la calidad del proyecto

Permite asegurar que el proyecto cumpla con los objetivos por los cuales el proyecto ha sido emprendido. Sus elementos son los siguientes:

- Planificación de la calidad.

- Realizar aseguramiento de la calidad.

- Realizar control de la calidad.

2.3.6.- Gestión de los recursos humanos del proyecto

Va orientado hacia los procesos que organizan y dirigen el equipo del proyecto. Se compone de los siguientes procesos:

- Planificación de los recursos humanos.

- Adquirir el equipo de proyecto.

- Desarrollar el equipo del proyecto.

- Gestionar el equipo del proyecto.

2.3.7.- Gestión de las comunicaciones del proyecto

Describe los procesos relacionados con la generación, recogida, distribución, almacenamiento y destino final de la información del proyecto en tiempo y forma. Viene dado por las siguientes actividades:

- Planificación de las comunicaciones.

- Distribución de la información.

- Informar el rendimiento.

- Gestionar a los interesados.

2.3.8.- Gestión de los riesgos del proyecto

La gestión de los riesgos en el proyecto permitirá conocer aquellas eventualidades que pueden afectar de forma positiva o negativa el desempeño de las actividades durante desarrollo del proyecto.

- Planificación de la gestión de riesgos.
- Identificación de riesgos.
- Análisis cualitativos de riesgos.
- Análisis cuantitativo de riesgos.
- Planificación de la respuesta a los riesgos.
- Seguimiento y control de riesgos.

2.3.9.- Gestión de las adquisiciones del proyecto

Son aquellos procesos para adquirir o comprar productos, servicios o resultados, así como para contratar procesos de dirección:

- Planificar las compras y adquisiciones.
- Planificar la contratación.
- Solicitar respuestas de vendedores.

- Selección de vendedores.

- Administración de contrato.

- Cierre del contrato.

Para el ciclo de vida de un proyecto para el desarrollo de software la organización *Extreme Programming* plantea las siguientes fases: *planeación*, *codificación*, *diseño* y *pruebas*, las cuales se describen de la siguiente manera (*Extreme Programming*, 2009):

En la *planeación* se cuenta con una documentación denominada historias del usuario, donde se describen las características y las funciones requeridas para el software, estas funciones se colocan en una carta índice, el cliente les asigna una prioridad y por último el equipo de desarrolladores les asigna un peso equivalente a semanas de desarrollo; una vez definidas las historias del usuario las mismas pueden ser ordenadas de las siguientes maneras: por implementación inmediata, por tiempo (valor más alto) o por su riesgo.

Con esta carta índice se procede a determinar la velocidad del proyecto, con lo cual pueden estimarse las fechas de entrega y el programa para lanzamientos subsecuentes y saber además si existió un compromiso excesivo en las historias determinadas previamente.

La fase de *diseño* nos permitirá obtener como resultado una guía de implementación de acuerdo a lo acordado durante la elaboración de la historia del usuario para una identificación y la organización de las clases orientadas al objeto destinados al incremento del software, lo cual será útil para el desarrollo de prototipos operacionales, a fin de implementarlos y evaluarlos, con el fin de reducir el riesgo al momento de realizar la verdadera implementación.

La *codificación* se realiza mediante una serie de pruebas de unidad donde se "ejercitan" cada una de las historias que vayan a implementarse, estas pruebas vienen determinadas por los resultados obtenidos durante el *diseño*; la actividad del desarrollo (*codificación*) se

realiza en pareja de programadores, ya que esto proporciona un mecanismo de resolución de problemas en tiempo real (dos cabezas piensan mejor que una), luego el trabajo resultante se integra con el trabajo de otros.

La fase de *pruebas* es realizada de manera continua y puede ser utilizado como indicador del avance del proyecto además de emisor de emergencias en caso de que exista alguna dificultad en la codificación de alguna historia.

2.4.- Gerencia de Proyectos de Software

De acuerdo a lo planteado en el capítulo anterior, tenemos que una gestión eficaz de gerencia de proyectos del software se enfoca sobre las cuatro P: *personal*, *producto*, *proceso* y *proyecto*, cabe señalar que el orden no es arbitrario, el proceso de ingeniería de software aplicado a la gerencia de proyectos es una empresa intensamente humana, un gestor que fracasa en alentar la comunicación amplia con los participantes se arriesga a construir una solución para un problema equivocado (Pressman, 2005).

2.4.1.- Personal

Tenemos que el factor humano es tan importante que el *Software Engineering Institute* ha desarrollado un modelo de madurez de la capacidad de gestión de personal o MMCGP (Pressman, 2005), esto es con el fin de "aumentar la rapidez con la cual las organizaciones de software acometen las aplicaciones cada vez más complejas al ayudar a atraer, aumentar, motivar, desplegar y retener el talento necesario para mejorar su capacidad de desarrollo de software" (Curtis, 1994, c.p. Pressman, 2005); dicho modelo de madurez abarca las siguientes áreas claves para el personal de software: reclutamiento, selección, gestión del desempeño, entrenamiento, retribución, desarrollo de la carrera, diseño de la organización y trabajo, y desarrollo de la cultura en equipo.

2.4.2.- Producto

Antes de la planificación del proyecto deberían establecerse los objetivos y el ámbito del producto, considerar soluciones alternativas e identificar las restricciones técnicas y de gestión, esto es con el fin de definir estimaciones razonables y precisas del costo, una valoración efectiva del riesgo, una distribución realista de las tareas y un cronograma de trabajo manejable.

La definición del alcance del proyecto debería ser acordado entre el desarrollador del software y el cliente, es decir, se determina el ámbito del producto o los datos primarios, sus funciones y su comportamiento de forma cuantitativa.

2.4.3.- Proceso

Este es el marco de trabajo desde el cual se puede establecer un plan detallado para el desarrollo del software, este plan viene definido por un conjunto de actividades compuestos por elementos tales como: tareas, hitos, productos de trabajo y puntos de control de calidad.

2.4.4.- Proyecto

Los proyectos de software se realizan de manera planificada y controlada ya que de esta forma se puede determinar su complejidad.

Para evitar el fracaso de un proyecto, se debe tomar en consideración una serie de señales y advertencias comunes, comprender los factores de éxito críticos y desarrollar un enfoque de sentido común para planificar, supervisar y controlar el proyecto.

2.5.- Definición de Software

Pressman (2005) define a software como el producto que los ingenieros de software construyen y después mantienen a largo plazo e incluye los programas que se ejecutan dentro de una computadora sin importar su arquitectura física.

Su importancia radica en que afecta de forma muy cercana todos los aspectos de nuestras vidas y se ha vuelto omnipresente en el comercio, la cultura o las actividades cotidianas.

En la actualidad el software tiene un papel de producto y de vehículo mediante el cual se entrega un producto. Como producto es un medio transformador de información, es decir, realiza la producción, el manejo, la adquisición, la modificación, el despliegue o la transmisión de la información sin importar el lugar donde resida el software, bien sea en un celular o dentro de una computadora central.

A diferencia de otras cosas que hace el ser humano, el software es un elemento lógico con determinadas características que lo permiten diferenciarse de otros componentes del computador como el hardware:

Las características principales del software son las siguientes:

- *El software se desarrolla o se construye; no se manufactura en el sentido clásico*, es decir, a pesar de que en la manufactura del hardware y en el desarrollo del software se busca la alta calidad mediante un buen diseño, y ambas actividades dependen de personas, los enfoques son diferentes, ya que principalmente los costos del software se centran en la ingeniería y por lo tanto los proyectos de software no se pueden manejar como proyectos de manufactura.

- *El software no se "desgasta"*, mientras que el hardware puede mostrar un alto nivel de "fallas" al inicio de su vida (por defectos de diseño o arquitectura); este nivel tiende a bajar por la aplicación oportuna de correcciones y tiende a estabilizarse por algún período de tiempo, sin embargo la tasa de fallas se eleva nuevamente conforme los

agentes ambientales como la acumulación de polvo, la vibración, el desgaste, el abuso y las temperaturas extremas hacen mella en el componente (hardware).

El software es inmune a estos males ambientales que desgastan el hardware, sin embargo el software puede presentar tasas altas de fallas motivado a defectos en la programación sin descubrir pero éstos pueden ser corregidos y estabilizar el nivel de fallas que pueda presentar el programa.

Sin embargo, a pesar de que el software no se desgasta, si puede sufrir deterioro, ya que a lo largo de su ciclo de vida, el software tiende a sufrir cambios, conforme estos son aplicados, se presenta la oportunidad de introducir errores, lo que ocasiona que durante su vida el software tenga algunos "picos" en su curva de fallas, lo que trae como consecuencia que el nivel mínimo de fallas se eleve y el software comience a deteriorarse por los continuos cambios.

- *A pesar de que la industria tiene una tendencia hacia la construcción por componentes*, la mayoría del software aún se construye a la medida. En el mundo del hardware, la reutilización de componentes es una parte natural del proceso de ingeniería al utilizar por ejemplo tornillos y circuitos integrados, lo cual permite al ingeniero concentrarse en los elementos que en realidad sí son innovadores en el diseño.

  En el caso del software un componente se debe diseñar e implementar de modo tal que permita utilizarlo en muchos programas diferentes, encapsulando tanto a los datos como el proceso que aplica a esto. Sin embargo esta es una práctica que a diferencia del desarrollo de hardware, apenas se ha comenzado a extender.

2.6.- Marco de trabajo genérico para el desarrollo de software

A continuación se lista una serie de pasos que de forma genérica marcan las etapas del ciclo de vida de desarrollo de software (Kendall y Kendall, 2005)

2.6.1.- Identificación de problemas, oportunidades y objetivos

Esta fase es crítica para el éxito del resto del proyecto, ya que una buena identificación del problema ahorrará muchas horas de trabajo al analista, ya que se evitaría desperdiciar esfuerzo trabajando en un problema que no era el que se debía resolver. En esta actividad (identificación del problema) el analista en conjunto con otros miembros de la organización, con base en la observación objetiva del negocio, precisará con determinación cuáles son los problemas; las oportunidades son aquellas situaciones que se consideran susceptibles de mejorar utilizando sistemas de información computarizados. Por último, se tiene la identificación de los objetivos, para esto se debe identificar plenamente lo que el negocio trata de conseguir, para así determinar si alguna de las aplicaciones de los sistemas de información o software puede contribuir a que el negocio alcance sus objetivos.

Como producto final de esta fase tenemos un informe de viabilidad que incluye una definición del problema y un resumen de los objetivos; una vez obtenida esta información, la administración decidirá la continuación o no del proyecto en base a lo presupuestado.

2.6.2.- Determinación de los requerimientos de información

Para esta fase el analista se vale de instrumentos tales como: las entrevistas, los muestreos, la investigación de datos impresos y la aplicación de cuestionarios. Todo esto con la finalidad de comprender la información que es requerida por los usuarios para llevar a cabo sus actividades, además de confirmar las ideas respecto a la organización y sus objetivos que se obtuvieron previamente de la identificación del problema.

En esta etapa es fundamental conocer a detalle las funciones del sistema actual en base al quién (involucrados), el qué (la actividad del negocio), el dónde (el entorno donde se realizan las actividades), el cuándo (el momento oportuno) y el cómo (la manera como se realizan los procedimientos actuales).

2.6.3.- Análisis de las necesidades del sistema

En esta fase, mediante el uso de herramientas especializadas como diagramas de flujos para graficar las entradas de una forma estructurada, se determinaran las entradas y salidas de las funciones del negocio para así desarrollar un diccionario de datos que enliste todos los datos utilizados en el sistema además de sus respectivas especificaciones.

También se analizan las decisiones estructuradas que se hayan tomado, es decir, aquellas en las cuales se pueden determinar las condiciones, las alternativas de condición, las acciones y las reglas de acción; para esta fase se obtiene como producto final una propuesta de sistemas que sintetiza sus hallazgos, proporciona un análisis de costo/beneficio de las alternativas y ofrece recomendaciones de lo que se debe hacer.

2.6.4.- Diseño del sistema recomendado

En la fase de diseño se utiliza la información recopilada en los pasos anteriores para realizar un diseño lógico del sistema de información, se diseñan además aquellos procedimientos precisos para la captura de datos que aseguran que lo ingresado al sistema sea lo correcto.

El producto final obtenido de esta fase son aquellas interfaces de usuario que forman parte del diseño lógico del sistema, los archivos o bases de datos que sirvan de almacén primario de la información recolectada de las interfaces de usuario y aquellos controles y procedimientos de respaldo que protejan al sistema y a los datos, además de aquellos paquetes de especificaciones de programa para los desarrolladores.

2.6.5.- Desarrollo y documentación del software

En esta fase se aplicarán técnicas estructuradas para el diseño y la documentación del código generado en el desarrollo de software (diagramas de estructura, diagramas de Nassi-Schneiderman y pseudocódigo).

Además, en esta fase se desarrolla la documentación de usuario efectiva para el software, como manuales de procedimiento, ayuda en línea y sitios web de preguntas y respuestas (*Frequently Asked Questions o* FAQ's).

2.6.6.- Prueba y mantenimiento del sistema

Aquí se realiza la correcta depuración del sistema antes de su puesta en producción; es menos costoso encontrar de forma preventiva los errores y corregirlos antes que el sistema sea entregado a los usuarios; además, en esta fase se define el mantenimiento preventivo del software y la periodicidad de la ejecución del mismo.

2.6.7.- Implementación y evaluación del sistema

En esta fase se capacita al usuario respecto al manejo de la herramienta, además se planifica el pase de la información previa generada por el sistema o proceso anterior hacia los repositorios de datos nuevos (en caso de aplicar).

2.7.- Desarrollo ágil de software

En el entorno de ingeniería de software la agilidad se define como la velocidad de reacción y respuesta de una forma apropiada a los cambios, es decir, cambios en el software que se va a construir, cambios entre los miembros del equipo, cambios debido a nuevas tecnologías, cambios de todo tipo que puede incidir en el producto que se construye o en el proyecto que crea el producto. Un equipo ágil reconoce que el software lo desarrollan individuos que trabajan en equipo y que las aptitudes de esa gente, y su capacidad para colaborar, son esenciales para el éxito del proyecto (Jacobson, 2002, c.p. Pressman, 2005).

A continuación se listan los principios para quienes deseen alcanzar la agilidad (alianza ágil, 2001, c.p. Pressman, 2005):

- Nuestra mayor prioridad es satisfacer al cliente mediante la entrega temprana y continua de software valioso.

- Bienvenidos los requisitos cambiantes, incluso en las fases tardías del desarrollo. La estructura de los procesos ágiles cambia para la ventaja competitiva del cliente

- Entregar con frecuencia software en funcionamiento, desde un par de semanas hasta un par de meses, con una preferencia por la escala de tiempo más corta.

- La gente de negocios y los desarrolladores deben trabajar juntos a diario a lo largo del proyecto.

- Construir proyectos alrededor de individuos motivados. Darles el ambiente y el soporte que necesitan, y confiar en ellos para obtener el trabajo realizado.

- El método más eficiente y efectivo de transmitir información hacia y dentro de un equipo de desarrollo es la conversación cara a cara.

- El software en funcionamiento es la medida primaria del progreso.

- Los procesos ágiles promueven el desarrollo sustentable. Los patrocinadores, desarrolladores y usuarios deben ser capaces de mantener un paso constante de manera indefinida.

- La atención continua a la excelencia técnica y al buen diseño mejora la agilidad.

- La simplicidad – el arte de maximizar la cantidad de trabajo no realizado – es esencial.

- Las mejores arquitecturas, los mejores requisitos y los mejores diseños emergen de equipos autoorganizados.

- A intervalos regulares el equipo refleja la forma en que se puede volver más efectivo; entonces su comportamiento se ajusta y se adecua en concordancia.

2.8.- Programación extrema

Dentro del movimiento ágil tenemos la existencia de diferentes modelos de proceso, todas enmarcadas dentro del manifiesto *para el desarrollo de software* la cual plantea que al haber descubierto mejores formas de desarrollar software y al ayudar a otros a hacerlo, se ha llegado a valorar (Pressman, 2005):

- A los individuos y sus interacciones sobre los procesos y las herramientas.
- Al software en funcionamiento sobre la documentación extensa.
- A la colaboración del cliente sobre la negociación del contrato.
- A la respuesta al cambio sobre el seguimiento de un plan.

Dicho manifiesto expone además que de lo planteado se valoran más los términos de la izquierda que los aspectos de la derecha sin desconocer que éstos tienen valor.

Para esta investigación se hará énfasis en la metodología denominada Programación extrema o PE, la cual es una disciplina dirigida al desarrollo de software, con un enfoque dirigido a la programación orientada a objetos; esta metodología está diseñada para la entrega oportuna de software cuando es requerida por el cliente, además de impulsar al desarrollador a responder de forma efectiva a cualquier cambio en el software por parte del cliente aún inclusive durante el ciclo de vida del desarrollo de software; la PE contempla además cuatro actividades del marco de trabajo las cuales son *planeación*, *codificación*, *diseño* y *pruebas* (*Extreme Programming*, 2009).

Para los efectos de esta investigación se plantea un plan de trabajo basado en esta disciplina, esto es motivado a que existen factores tales como el tiempo por ejemplo, el cuál influye notablemente en el desarrollo de software, tomando en consideración que esta actividad no puede extenderse por años, y este modelo plantea el manejo efectivo de este factor en particular además de otros como el personal, recursos, etc.

Capítulo III.
MARCO METODOLÓGICO

Luego de haber planteado el problema y haber definido aquellos conceptos fundamentales que permitirán conocer mejor el objeto de la investigación, es necesario definir los diversos métodos y técnicas que permitirán obtener de forma concisa aquellos datos que sustenten la solución al problema propuesto en este trabajo de investigación.

3.1.- Tipo de investigación

Para este trabajo el tipo de investigación será *aplicada*, ya que busca la aplicación concreta y efectiva de las conclusiones teóricas al plano de la vida real (Sabino, 1992), enmarcándose además dentro de la clasificación *proyecto factible* ya que se propone el desarrollo de un producto para cubrir una necesidad específica.

Cabe señalar además que esta investigación es del tipo de estudio diagnóstico, incorporada por algunos autores dentro de la investigación descriptiva, sin embargo la investigación diagnóstico, busca captar, reconocer y evaluar sobre el terreno, los componentes y las relaciones que se estableces en una situación determinada, con el propósito de lograr su verdadera comprensión y avanzar hacia su resolución (Balestrini, 2002).

3.2.- Unidad de análisis

La unidad de análisis son aquellos sujetos u objetos a ser estudiados y medidos con el fin de conocer sus características y determinar el impacto de determinada situación en ellos, estos objetos van en función de la delimitación del problema y de los objetivos de la investigación (Balestrini, 2002).

Para esta investigación se determinó como unidad de análisis un proyecto de desarrollo de software que automatice la gestión diaria del proceso de *HelpDesk*, dicha unidad de análisis será sometida a un estudio, mediante el cual se determinará si cada uno de los

subproductos obtenidos cumplen con los requisitos exigidos por el PMI® para la gestión de proyectos.

3.3.- Plan de ejecución del proyecto

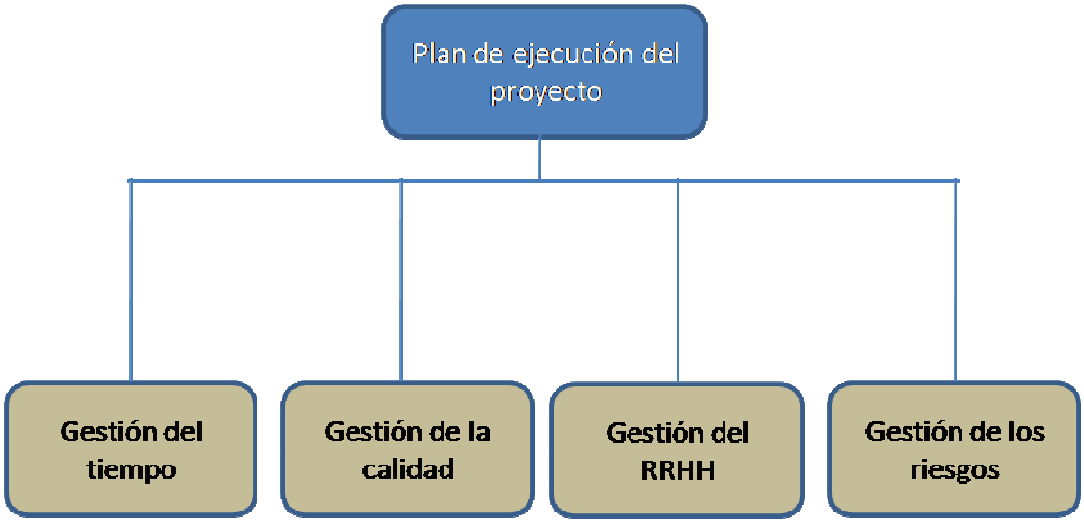

Figura 3. Plan de ejecución del proyecto

3.4.- Recolección, procesamiento y análisis de los datos

En esta investigación la gestión de las nueve áreas del conocimiento será realizada enmarcada dentro de la metodología de *extreme programming*, de la siguiente manera: en la fase de *planeación* se realiza la *gestión del alcance del proyecto*, ya que los *stakeholders* del proyecto definen la necesidad planteada y realizan continuas revisiones hasta obtener un documento de especificaciones definitivo.

Mediante la *gestión del tiempo*, el equipo de desarrolladores determina el tiempo de dedicación a una actividad determinada, generando así un cronograma de actividades dónde las mismas serán ejecutadas de acuerdo al nivel de complejidad que estas presenten. Con la *gestión de los costos*, se determinan los requerimientos a nivel tecnológico que requieran un coste adicional.

El gerente del proyecto mediante la *gestión de los RRHH*, determina los recursos que participarán en el proyecto y sus roles en el desarrollo del software, adicionalmente se realizará en conjunto con los integrantes del equipo de trabajo la *gestión de los riesgos*, con el fin de evaluar cualquier eventualidad que pueda afectar los tiempos de desarrollo y por consiguiente el cronograma de actividades.

En la fase de *diseño*, se obtiene la estructura preliminar del proyecto culminado, es decir, se obtendrá la estructura de la aplicación, como se distribuirán los módulos de la misma y la estructura general de la base de datos. Para la definición de estos elementos se hace necesaria una efectiva *gestión de la comunicación* entre cada uno de los miembros del equipo de trabajo con el fin de indicar observaciones oportunas que eviten contratiempos durante la fase de *codificación*, adicionalmente se realizan pruebas en frío con los resultados obtenidos de la evaluación de los riesgos a fin de determinar en que grado se podrían materializar, su impacto en el proyecto además de la forma de controlarlos y manejarlos, y por último se aplican estándares de programación y planes de pruebas unitarios lo cual permite aplicar controles de calidad iniciales (*gestión de la calidad*).

Durante la fase de *codificación* ya que se realiza la materialización del producto obtenido en el *diseño*, mediante software especializado se "escribe" en un lenguaje propietario la interfaz del usuario, las clases que comunicarán cada acción del usuario con el repositorio de datos y otros elementos que intervengan en la interacción del usuario, se realizan pruebas unitarias de cada interfaz en conjunto con el usuario para que éste indique si cumple con su expectativa, además de conocer si el módulo es lo suficientemente intuitivo para el usuario.

La fase de *pruebas* se realiza con la finalidad de asegurar que el producto desarrollado cumpla con la calidad determinada durante el diseño de la interfaz, además se valida la funcionalidad, la forma en que cada módulo interactúe entre sí para culminar con la implementación final del producto, previa aprobación del cliente.

3.5.- Operacionalización de objetivos

Objetivos Específicos	Variable(s)	Definición	Indicador(es)	Técnica / Instrumentos
Determinar el alcance del proyecto en firme a través de la aplicación de las técnicas S.M.A.R.T. Mapa de problemas y objetivos.	1.- Causas de la problemática 2.- Consecuencias de la problemática 3.- Definición de la problemática 4.- Medio de solución del problema 5.- Objetivo de la solución 6.- Finalidad de la solución	Variables que permitirán determinar la causa de la problemática, las soluciones a cada una de estas circunstancias y el resultado esperado luego de la aplicación de dichas soluciones	1.- Definición 2.- Precisión 3.- Alcance 4.- Posibilidad 5.- Tiempo	1.- Técnica S.M.A.R.T. 2.- Árbol de Problemas 3.- Árbol de Objetivos
Elaborar el WBS del proyecto	1.- Alcance del Proyecto 2.- Productos Entregables 3.- Plan Gestión Alcance	Se estructuran las actividades en productos entregables más pequeños	1.- Estrategias 2.- Métodos 3.- Técnicas	1.- Plantillas de la estructura de desglose de trabajo 2.- Descomposición
Conceptualizar los entregables del proyecto mediante la descripción breve de las actividades del proyecto	1.- Software 2.- Documentación 3.- Procedimientos 4.- Manuales de usuario	Determinar los entregables del proyecto o productos finales	1.- Estrategias 2.- Métodos 3.- Técnicas	1.- Diagramas de casos de uso 2.- Diagramas de clase 3.- Diagramas de secuencia 4.- Diagramas de entidad - relación
Estimar los tiempos de ejecución para los tiempos mas probable, optimista y pesimista mediante la consulta a expertos de la empresa	1.- Tiempo 2.- Actividad	Definir los tiempos de ejecución de cada actividad	1.- Métodos 2.- Técnicas	1.- Diagrama de Red
Elaborar el presupuesto de costos del proyecto con un nivel de estimado clase 5.	1.- Tiempo 2.- Actividad 3.- Recursos.	Definir el costo de cada actividad	1.- Unidades Tiempo. 2.- Cantidad Horas. 3.- Costo Unitario	1.- Juicio Experto. 2.- Antecedentes otros proyectos similares.
Determinar el nivel de calidad del producto deseado, de acuerdo a las exigencias del mercado, además de cumplir con los estándares exigidos para este tipo de Software.	1.- Propiedades del producto/entregable 2.- Características del producto/entregable 3.- Requerimiento cliente	Determina el nivel de fiabilidad del producto y el grado de satisfacción del cliente	1.- Funcionalidad 2.- Confiabilidad 3.- Usabilidad 4.- Eficiencia 5.- Soporte 6.- Portabilidad	1.- ISO – 9126 2.- COTS-Based Requirements Engineering (CRE)
Determinar la estructura organizativa del proyecto	1.- Recursos	Determinar los Roles de cada miembro del equipo del proyecto	1.- Métodos 2.- Técnicas	1.- Organigrama
Describir el perfil de los RRHH requeridos para proyecto.	1.- Cargos	Determinar perfil profesional de cada cargo	1.- Métodos 2.- Técnicas	1.- Juicio Experto.
Elaborar la matriz de responsabilidad del proyecto	1.- Tiempo 2.- Actividad 3.- Recursos.	Determinar roles y responsabilidades para cada actividad definida en el WBS del proyecto.	1.- Métodos 2.- Técnicas	1.- Juicio Experto.
Diseñar las pautas para plan de gestión de las comunicaciones del proyecto.	1.- Tiempo 2.- Actividad 3.- Recursos.	Definir la forma de comunicación que predominará durante el ciclo de vida del proyecto	1.- Métodos 2.- Técnicas	1.- Plan de comunicaciones
Identificar cualitativamente los riesgos del proyecto.	1.- Riesgos del proyecto 2.- Riesgos técnicos 3.- Riesgos de mercado	Realizar la administración de riesgos para así lograr los resultados implícitos en el plan del proyecto	1.- Riesgos 2.- Probabilidad de impacto 3.- Costo del riesgo	1.- Lista de Chequeo 2.- Juicio experto 3.- Tabla de Riesgos
Describir la matriz de las adquisiciones de proyecto y su plan de acción.	1.- Costo 2.- Mantenimiento 3.- Garantía 4.- Tiempo de Entrega	Realizar la selección de la adquisición de los insumos entre diversos proveedores	1.- Costo	1.- Propuestas y Ofertas 2.- Criterios de selección

Tabla 1. Operacionalización de Objetivos.

3.6.- Técnicas e instrumentos de recolección de datos

Los instrumentos y técnicas de recolección de datos, son aquellos de los cuales se vale el investigador para acercarse a los fenómenos y extraer información de ellos, esto con el fin de alcanzar la necesaria correspondencia entre teoría y hechos, para así satisfacer aquellas interrogantes iniciales planteadas en la investigación (Sabino, 2002).

Para la recolección de datos se empleará la Observación participante, la cual implica que el investigador deba integrarse al grupo, para lo cual el investigador debe primeramente integrarse al grupo a la par de ir recolectando los datos para la investigación, lo cual en este caso se aplica ya que se cuenta con la experiencia tanto en el desarrollo de software de forma activa como participación directa en el área de Helpdesk.

3.7.- Definición conceptual y operacional de variables

Proyecto	Desarrollo de un software que permita gestionar las actividades de soporte técnico canalizados mediante un HelpDesk
Objetivo General	Elaborar el plan estratégico del proyecto de desarrollo de software que permita gestionar las actividades de soporte técnico canalizados mediante un HelpDesk
Objetivos Específicos	1.-Determinar el alcance del proyecto en firme a través de la aplicación de las técnicas S.M.A.R.T. Mapa de problemas y objetivos. 2.- Elaborar el WBS del proyecto. 3.-Conceptualizar los entregables del proyecto mediante la descripción breve de las actividades del proyecto. 4.-Estimar los tiempos de ejecución para los tiempos mas probable, optimista y pesimista mediante la consulta a expertos de la empresa. 5.- Elaborar el presupuesto de costos del proyecto con un nivel de estimado clase 5. 6.-Determinar el nivel de calidad del producto deseado, de acuerdo a las exigencias del mercado, además de cumplir con los estándares exigidos para este tipo de Software. 7.- Determinar la estructura organizativa del proyecto. 8.- Describir el perfil de los RRHH requeridos para proyecto. 9.- Elaborar la matriz de responsabilidad del proyecto. 10.- Diseñar las pautas para plan de gestión de las comunicaciones del proyecto. 11.- Identificar cualitativamente los riesgos del proyecto. 12.- Describir la matriz de las adquisiciones de proyecto y su plan de acción.
Variables	1.- Plan estratégico 2.- Desarrollo de software

Tabla 2. Definición de variables en el Proyecto de Investigación.

Variables	Dimensión	Indicador	Técnicas / Herramientas e instrumentos de medición	Resultado Esperado
Plan Estratégico	1.- Alcance del Proyecto. 2.- Estructura del Proyecto. 3.- Entregables del Proyecto. 4.- Tiempo de Ejecución. 5.- Presupuesto de costos. 6.- Nivel de Calidad. 7.- Estructura organizativa.	1.- Objetivos. 2.- Estándares de calidad. 3.- Costos. 4.- Tiempo. 5.- Recursos.	1.- S.M.A.R.T. 2.- Diagrama Gantt. 3.- Organigrama. 4.- Estimado costos.	Estrategia de ejecución del proyecto
Desarrollo de Software	1.- Planeación. 2.- Diseño. 3.- Codificación. 4.- Pruebas	1.- Base de Datos. 2.- Pantallas del sistema. 3.- Clases. 4.- Objetos.	1.- Diagrama E-R. 2.- Desarrollo Ágil. 3.- Extreme Programming.	Producto final y entregables

Tabla 3. Operacionalización de Variables.

3.8.- Diseño de la investigación

El diseño de la investigación se define como el plan global de investigación que integra de un modo coherente y adecuado técnicas de recolección de datos a utilizar, análisis previstos y objetivos, dando así respuestas a aquellas interrogantes surgidas durante la investigación (Martín, 1986, c.p. Balestrini, 2002).

Motivado a que esta investigación es del tipo diagnóstico, el diseño será de campo ya que al buscar la interacción entre los objetivos y la realidad de la situación, se observan y recolectan los datos directamente de la realidad, en su situación natural a fin de determinar correlaciones entre cada uno de ellos (Balestrini, 2002).

Capítulo IV
Diseño del Plan

En este capítulo se determinarán aquellos pasos y estrategias que serán necesarias para llevar a cabo la elaboración del Software que permita la administración de un HelpDesk.

Con la finalidad de lograr la consecución del objetivo general de esta investigación se procederá a desarrollar los siguientes pasos:

4.1.- Determinación del alcance del proyecto:

El PMI define a la gestión del alcance del proyecto de la siguiente manera "La Gestión del alcance del proyecto incluye todos los procesos necesarios para asegurarse que el proyecto incluya todo el trabajo requerido, y sólo el trabajo requerido, para completar el proyecto satisfactoriamente" (PMI, 2004)

Para la determinación del alcance del proyecto se disponen diversas técnicas como la técnica S.M.A.R.T la cual permite marcar principios que ayudan a clarificar y estipular adecuadamente los objetivos de determinado proyecto (López, 2008).

S.M.A.R.T viene dado por los siguientes principios:

Un objetivo debe venir definido por las siguientes variables:
S – Debe ser específico.
M – Debe ser mensurable.
A – Debe ser alcanzable.
R – Debe ser realista.
T – Debe ser culminado en un tiempo determinado.

Los resultados obtenidos luego de aplicar la técnica S.M.A.R.T. serán plasmados en un árbol de problemas y un árbol de objetivos.

En base a lo expuesto anteriormente tenemos como principal objetivo del proyecto: Presentar una solución tecnológica del tipo software que permita organizar y controlar las actividades de soporte técnico canalizados mediante un HelpDesk.

Este objetivo cumple con lo definido por S.M.A.R.T. ya que es:

Especifico: Tiene como finalidad presentar un software para cubrir una determinada necesidad en un mercado específico

Mensurable: Este proyecto será desarrollado en un período de tiempo determinado, dentro de este periodo de tiempo se establecerán sub actividades que igualmente disponen de un inicio y un fin en el tiempo, además de otros indicadores que permitirán conocer los avances y resultados alcanzados.

Alcanzable: Es alcanzable ya que se disponen de las herramientas, los recursos y cualquier otro medio necesario que permita la consecución exitosa y satisfactoria del proyecto planteado

Realista: Es realista ya que el desarrollo de este proyecto se basa bajo estándares conocidos y aplicados en muchas situaciones de la vida real.

Tiempo: El proyecto dispone de un periodo de tiempo especifico que permitirá su consecución

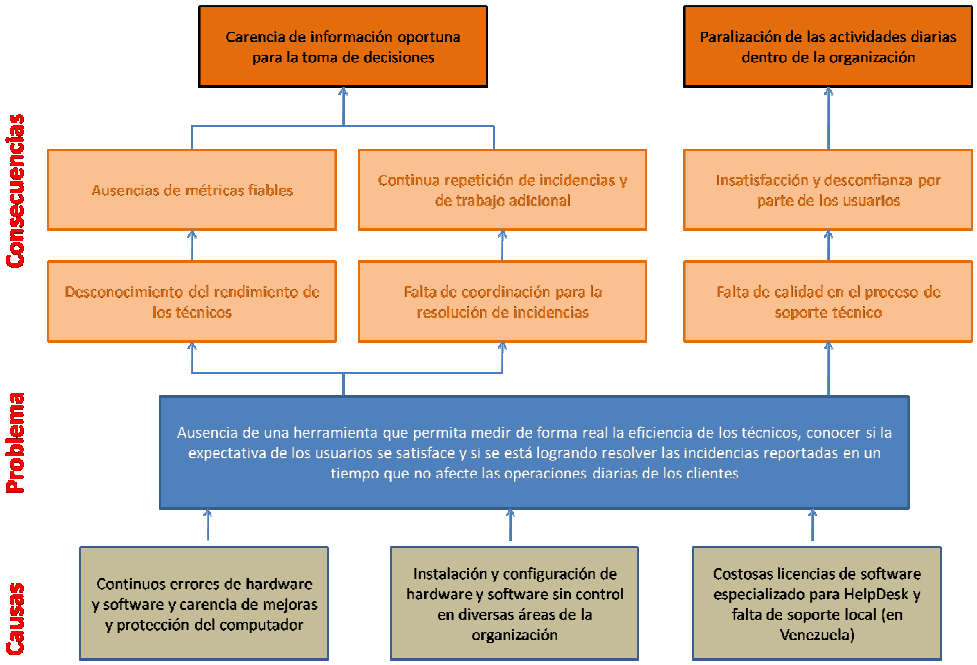

Figura 4. Árbol de problemas planteado para el proyecto

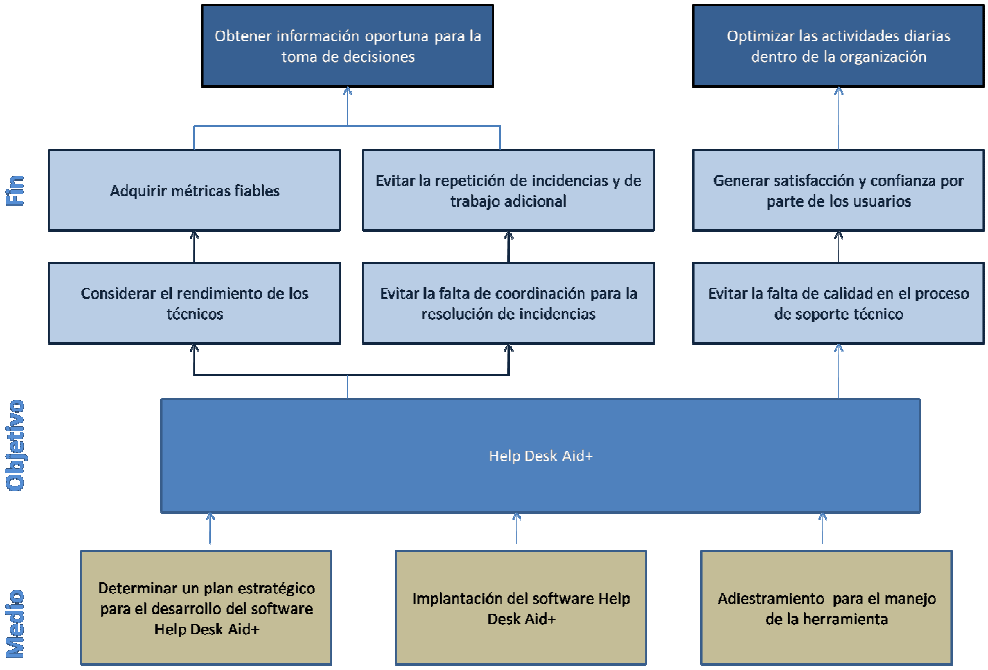

Figura 5. Árbol de objetivos planteado para el proyecto

Luego de determinar los problemas y los objetivos, tenemos como alcance del proyecto la creación de un software que permita:

- Obtener información oportuna para la toma de decisiones:
  - Obtener métricas fiables
  - Evitar la repetición de incidencias y de trabajo adicional
  - Considerar el rendimiento de los técnicos y especialistas
  - Evitar la falta de una coordinación oportuna para la resolución de incidencias
- Optimizar las actividades diarias dentro de la organización:
  - Generar satisfacción y confianza por parte de los usuarios
  - Evitar la falta de calidad en el proceso de soporte técnico

4.2.- WBS del Proyecto

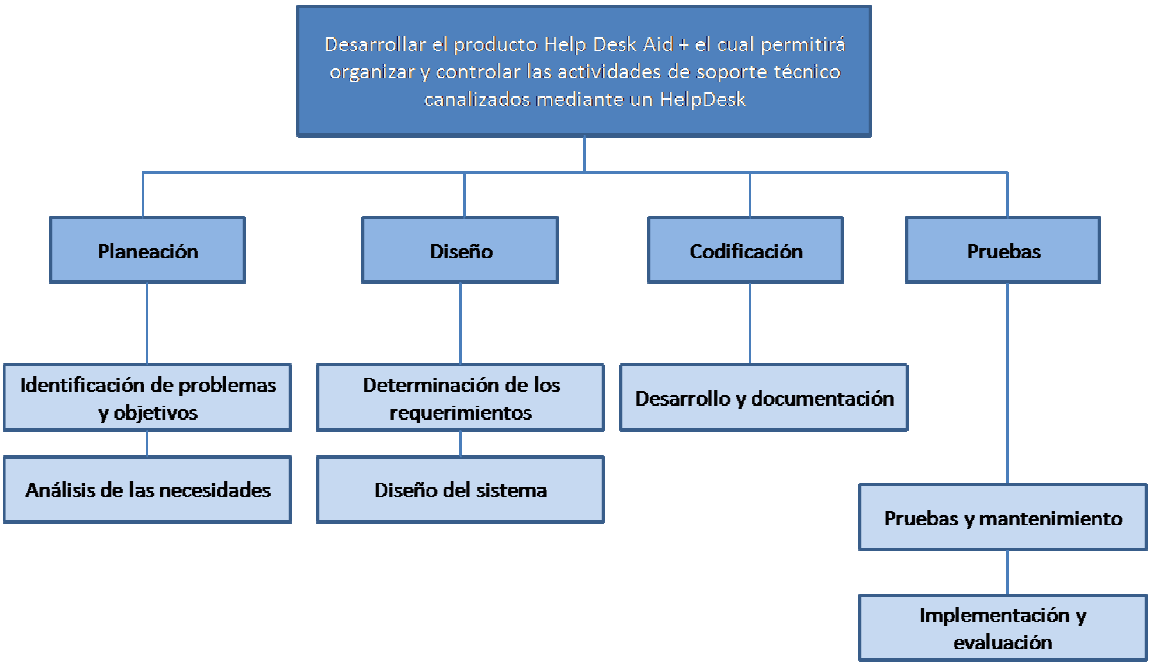

Figura 6. WBS. Del Proyecto

4.3.- Determinar entregables del proyecto

Como entregables del proyecto o productos finales se determinan: El software, documentación, procedimientos, manuales de usuario o sesiones de capacitación de acuerdo a los términos especificados en un contrato (Kendall y Kendall, 2005)
Durante la actividad de *planeación* se obtendrá como producto final el alcance del proyecto mediante la creación de un árbol de objetivos el cual resulta de un análisis de la necesidad plasmada en un árbol de problemas.
En la fase de *diseño* se obtendrán diversos documentos que permitirán al equipo de desarrollo producir el software requerido, entre estos documentos tenemos:

- *Diagramas de casos de uso:* Este es un poderoso concepto que ayuda a un analista a comprender la forma en que un sistema deberá comportarse, permitiéndole así obtener los requerimientos desde el punto de vista del usuario. Esta representación viene dada de la siguiente manera: Existe un actor que inicia el caso de uso y otro que puede ser el que inició el caso de uso, que recibirá algo de valor de él. Un elipse representa a un caso de uso y una figura agregada a un actor, el actor que inicia se encuentra a la izquierda y el que recibe a la derecha, una línea asociativa une a los actores con el caso de uso, mostrando así los confines entre el sistema y el mundo exterior (Schmuller, 2000).

- *Diagramas de clases:* Este diagrama ilustra las entidades en un sistema o dominio y la forma en que cada entidad se relaciona entre sí, las entidades en este caso vienen representadas por un rectángulo (Schmuller, 2000).

- *Diagramas de secuencia:* Representa la forma en que interaccionan los objetos entre sí al paso del tiempo. Los objetos se colocan en la parte superior, y el tiempo procede desde la parte superior hacia la parte inferior del diagrama. Las flechas denotan mensajes o algún tipo de flujo que van de objeto a objeto (Schmuller, 2000).

- *Diagramas de entidad – relación:* Esta técnica permite modelar los datos que administra una organización, proporcionando un modelo independiente de cualquier almacenamiento de datos y métodos de acceso, viene conformado por entidad (objeto

que representa el contenedor de algún dato o un conjunto de información determinado) y una relación (La asociación entre dos o más entidades) (IUTEP, s.f.)

Para la actividad de *Codificación* se obtendrá como producto final el software, el cual se deriva de la generación del código, bien sea manual o automatizado, y la realización de pruebas preliminares que permitan detectar algún error en el código, además de validar si el producto que se está codificando está cumpliendo con lo exigido durante la fase de planeación (Pressman, 2005).

En la actividad de *pruebas* se realizan determinadas pruebas de rendimiento y desgaste del software a fin de realizar una evaluación que permitirá conocer si el producto cumple con los estándares y la calidad necesaria para garantizar la completa funcionalidad y comercialización del mismo (Pressman, 2005), todo esto con el fin de recibir como entregable el software ya certificado.

4.4.- Tiempos de ejecución:

Para este proyecto se definirán los tiempos de ejecución de cada actividad mediante la creación de un diagrama de red que ilustre cada actividad de acuerdo a la dependencia de cada una, luego se empleará la metodología AON o actividad en el nodo, en donde cada actividad está representada por un único cuadro donde se muestra el numero de la actividad y el plazo de duración (Lledó y Rivarola, 2007)

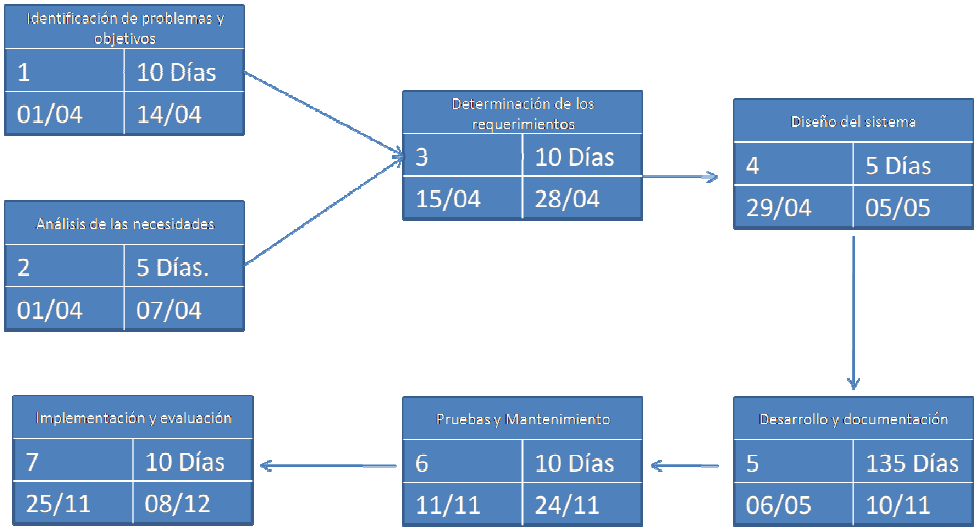

Figura 7. Diagrama de Red del Proyecto

Este Diagrama de Red, queda plasmado de la siguiente manera en el diagrama Gantt del proyecto.

Tracking Gantt

	Task Name	Duration	Start	Finish	Predecessors	Resource Names
1	**Help Desk Aid +**	**189 days**	**Thu 01/04/10**	**Tue 21/12/10**		
2	**Planeación**	**15 days**	**Thu 01/04/10**	**Wed 21/04/10**		
3	Identificación de problemas y objetivos	10 days	Thu 01/04/10	Wed 14/04/10		Líder Proyecto
4	Análisis de las necesidades	5 days	Thu 15/04/10	Wed 21/04/10	3	Líder Proyecto
5	**Informe de planeación**	0 days	Wed 21/04/10	Wed 21/04/10	4	
6	**Diseño**	**15 days**	**Thu 22/04/10**	**Wed 12/05/10**		
7	Determinación de los requerimientos	10 days	Thu 22/04/10	Wed 05/05/10	2	Líder Proyecto
8	**Documento de Requisitos Funcionales**	0 days	Wed 05/05/10	Wed 05/05/10	7	
9	**Diseño del sistema**	**5 days**	**Thu 06/05/10**	**Wed 12/05/10**		
10	Creación Diagrama Casos - Usos	1 day	Thu 06/05/10	Thu 06/05/10	7	Líder Proyecto / Consultor Sr.
11	Creación Diagrama Clases	1 day	Fri 07/05/10	Fri 07/05/10	7,10	Líder Proyecto / Consultor Sr.
12	Creación Diagrama Secuencia	1 day	Mon 10/05/10	Mon 10/05/10	7,11	Líder Proyecto / Consultor Sr.
13	Desarrollo Diagrama Entidad-Relación	1 day	Tue 11/05/10	Tue 11/05/10	7,12	DBA / Consultor Sr.
14	Diseño Pantallas Usuario	1 day	Wed 12/05/10	Wed 12/05/10	7,13	Líder Proyecto / Consultor Sr.
15	**Documentos de Diseño del Sistema**	0 days	Wed 12/05/10	Wed 12/05/10	9	
16	**Codificación**	**140 days**	**Wed 12/05/10**	**Tue 23/11/10**		
17	**Desarrollo y Documentación**	**140 days**	**Wed 12/05/10**	**Tue 23/11/10**		
18	Normalización Base Datos	5 days	Wed 12/05/10	Tue 18/05/10	13	DBA
19	Desarrollo Base Datos	10 days	Wed 19/05/10	Tue 01/06/10	18	DBA
20	Desarrollo Clases	45 days	Wed 02/06/10	Tue 03/08/10	10,11,12,19	Consultor Sr. / Consultor(es) Jr.
21	Desarrollo Pantallas Usuario	45 days	Wed 04/08/10	Tue 05/10/10	20,14	Consultor Sr. / Consultor(es) Jr.
22	Pruebas Unitarias	30 days	Wed 06/10/10	Tue 16/11/10	21	Consultor QA / Analista Testing
23	Documentación	5 days	Wed 17/11/10	Tue 23/11/10	22	Consultor QA / Analista Testing
24	**Entrega Sistema Preliminar**	0 days	Tue 23/11/10	Tue 23/11/10	17	
25	**Pruebas**	**20 days**	**Wed 24/11/10**	**Tue 21/12/10**		
26	Pruebas y mantenimiento	10 days	Wed 24/11/10	Tue 07/12/10	16	Analista Testing / Consultor(es) Jr.
27	Implementación y evaluación	10 days	Wed 08/12/10	Tue 21/12/10	26	Líder Proyecto / Consultor Sr.
28	**Entrega del Sistema Final**	0 days	Tue 21/12/10	Tue 21/12/10	26,27	

Figura 8. Diagrama Gantt del Proyecto.

4.5.- Presupuestos de costo del proyecto:

Por la naturaleza del proyecto, la cual es el desarrollo de un software, se realizó el estimado de costos clase V en base a horas hombre, lo cual generó los costos por cada actividad determinada en el WBS del proyecto.

El estimado de costo clase V resultante quedó distribuido de la siguiente manera

Estimación de costos					
Paquete WBS	**Paquete de Trabajo**	**Unidades**	**Cantidad Horas**	**Costo Unitario**	**Total**
1.- Planeación					
	1.1.- Identificación de Problemas y objetivos	h/h	80	32,98	2.638,40
	1.2.- Análisis de las necesidades	h/h	40	32,98	1.319,20
2.- Diseño					
	2.1.- Determinación de los requerimientos	h/h	80	32,98	2.638,40
	2.2.- Diseño del sistema				
	2.2.1.- Creación Diagrama Casos - Usos	h/h	8	54,45	435,60
	2.2.2.- Creación Diagrama Clases	h/h	8	54,45	435,60
	2.2.3.- Creación Diagrama Secuencia	h/h	8	54,45	435,60
	2.2.4.- Desarrollo Diagrama Entidad-Relación	h/h	8	54,45	435,60
	2.2.5.- Diseño Pantallas Usuario	h/h	8	54,45	435,60
3.- Codificación					
	3.1.- Desarrollo y Documentación				
	3.1.1.- Normalización Base Datos	h/h	40	21,47	858,80
	3.1.2.- Desarrollo Base Datos	h/h	80	21,47	1.717,60
	3.1.3.- Desarrollo Clases	h/h	360	54,45	19.602,00
	3.1.4.- Desarrollo Pantallas Usuario	h/h	360	54,45	19.602,00
	3.1.5.- Pruebas Unitarias	h/h	240	32,46	7.790,40
	3.1.6.- Documentación	h/h	40	32,46	1.298,40
4.- Pruebas					
	4.1.- Pruebas y mantenimiento	h/h	80	27,48	2.198,40
	4.2.- Implementación y evaluación	h/h	80	54,45	4.356,00
Totales			**1.520,00**	**669,88**	**66.197,60**

Tabla 4. Estimado de costos clase V.

Igualmente se estimaron los costos de los recursos participantes y los insumos necesarios para el desarrollo del proyecto

Cargo	Salario	Cantidad de recursos	Total
Líder Proyecto	5.805,00	1	**5.805,00**
Consultor Sr.	3.780,00	1	**3.780,00**
Consultor QA	3.780,00	1	**3.780,00**
DBA	3.780,00	1	**3.780,00**
Consultor Jr.	2.902,50	2	**5.805,00**
Analista Testing	1.935,00	1	**1.935,00**
Totales	**21.982,50**	**7**	**24.885,00**

Tabla 5. Estimado de costos por recursos.

Equipo	Precio Unitario	Cantidad	Total
Computador Computador INTEL Dual Core E5200 + DD 250GB + MEM 2GB	1.950,00	6	**11.700,00**
Computador INTEL Quad Q8200 + DD 500GB + MEM 4GB	2.960,00	1	**2.960,00**
Portatil HP COMPAQ CQ40-621LA	3.700,00	2	**7.400,00**
Totales	**8.610,00**	**9**	**22.060,00**

Tabla 6. Estimado de costos por insumos requeridos.

Licencia	Precio Unitario	Cantidad	Total
SQL Server Developer Edition	100,00	1	**100,00**
Visual Studio Microsoft Visual Studio 2008 Professional	1.332,97	1	**1.332,97**
Microsoft Office Professional 2007 FULL VERSION	752,00	1	**752,00**
Totales	**2.184,97**	**3**	**2.184,97**

Tabla 7. Estimado de costos por licencias de software requeridos.

4.6.- Calidad del producto:

La calidad del producto viene dada por la totalidad de las propiedades y características de un producto o servicio, que conformen la aptitud de satisfacer los requerimientos explícitos del cliente (Lledó y Rivarola, 2007).

Motivado a que este proyecto va enfocado hacia el desarrollo de un software basado en componentes, tenemos que la calidad viene determinada por el producto y por el proceso.

Para medir la *calidad en el producto* final desarrollado mediante DBSC se emplearan las métricas definidas en la norma ISO – 9126, la cual publicada en 1992 bajo el nombre de "*Information Technology – Software product evaluation: Quality characteristics and guidelines for their use*" establece que cualquier componente de la calidad del software puede ser descrito en términos de Funcionalidad, Confiabilidad, Usabilidad, Eficiencia, Soporte , Portabilidad (Abud, 2004).

Cada uno de estas características vienen definidas de la siguiente manera (International Organization for Standarization ISO, c.p. Abud, 2004):

Funcionalidad: Este grupo conjuga una serie de atributos que permiten calificar si el producto final maneja de forma adecuada el conjunto de funciones que satisfagan las necesidades para las cuales fue diseñado. Dentro de este grupo tenemos las siguientes métricas:

- Adecuación: Evalúa si el software cuenta con un conjunto de funciones apropiadas para efectuar las tareas que fueron especificadas en su definición.
- Exactitud: Permite evaluar si el software presenta los resultados acordes a las necesidades para las cuales fue diseñado.
- Interoperabilidad: Evalúa la habilidad del software para interactuar con otros sistemas previamente especificados.
- Conformidad: Valida si el software se adhiere a estándares, convenciones o regulaciones en leyes y/o prescripciones similares.
- Seguridad: Se refiere a la habilidad de prevenir el acceso no autorizado, accidental o premeditado, a los programas y datos.

Confiabilidad: Viene definido por aquellos atributos que permiten mantener el nivel de ejecución del software bajo condiciones normales, durante un período de tiempo determinado, sus subcaracterísticas son las siguientes:

- Nivel de madurez: Mide la frecuencia de fallas por errores en el software.
- Tolerancia a fallas: Es la habilidad de mantener un nivel especifico de funcionamiento en caso de fallas en el software o infracciones de la interfaz.
- Recuperación: Es la capacidad de restablecer el nivel de operación y recobrar los datos que hayan sido afectados directamente por una falla, así como el tiempo y el esfuerzo necesario para lograrlo

Usabilidad: Son aquellos atributos que permiten evaluar el esfuerzo necesario que deberá invertir el usuario para utilizar el sistema:

- Comprensibilidad: Es el esfuerzo requerido por el usuario para reconocer la estructura lógica del sistema y los conceptos relativos a la aplicación del software.
- Facilidad de aprender: Es aquél dirigido al esfuerzo que el usuario debe hacer para aprender a usar la aplicación.
- Operabilidad: Son los conceptos que evalúan la operación y control del sistema.

Eficiencia: Evalúa la relación entre el nivel de funcionamiento del software y la capacidad de los recursos empleados:

- Comportamiento con respecto al tiempo: Son aquellos relacionados al tiempo de respuesta y de procesamiento de los datos.
- Comportamiento de acuerdo a recursos: Son aquellos relacionados a la cantidad de recursos empleados y la duración de su uso en la realización de sus funciones

Soporte: Son los atributos que miden el esfuerzo necesario para realizar modificaciones al software, ya sea por corrección de errores o por incremento de la funcionalidad, y viene determinado por las siguientes características:

- Capacidad de análisis: Relativo al esfuerzo necesario para diagnosticar las deficiencias o causas de fallas, o para identificar las partes que deberán ser modificadas.
- Capacidad de modificación: Mide el esfuerzo necesario para modificar aspectos del software, remover fallas o adaptar el software para que funcione en un ambiente diferente.

Estabilidad: Evalúa los riesgos de efectos inesperados debido a modificaciones realizadas al software.

- Facilidad de prueba: Es aquél esfuerzo necesario para validar el software una vez modificado.

Portabilidad: Es la capacidad del software para ser transferido de un ambiente a otro y considera:

- Adaptabilidad: Referido a adaptar el software a diferentes ambientes sin necesidad de aplicar modificaciones.
- Facilidad de instalación: Es el esfuerzo necesario para instalar el software en un ambiente necesario.
- Conformidad: Evalúa si el software se adhiere a estándares o convenciones relativas a portabilidad.
- Capacidad de reemplazo: Se refiere a la oportunidad y el esfuerzo empleado en sustituir el software por otro producto con condiciones similares.

Luego de determinar la metodología a emplear para medir la calidad en el producto final, se empleara el modelo COTS[2] -*Based Requirements Engineering* (CRE) para la medición de *calidad del proceso* para el desarrollo del software definido en el proyecto.
CRE se desarrolla para facilitar el proceso de selección y evaluación basado en los requisitos, prestando especial atención al análisis de los requisitos extra-funcionales. La selección de los componentes se realiza por rechazo, es decir, si alguno de los componentes candidatos no cumple con alguno de los requisitos del cliente, estos van siendo rechazados y retirados de la lista de candidatos, a medida que esta lista de componentes va decreciendo se hace necesario aumentar el número y detalle de los requisitos. Generando como resultado final un proceso iterativo mediante el cual el proceso de adquisición de los requisitos permite la selección de productos a utilizar, generando información para la obtención de nuevos requisitos de calidad. CRE va

[2] COTS o *Commercial-Off-The-Shell* son componentes que presentan las siguientes características (Bass, 1999, C.P. Bertoa, Troya y Valecillo, s.f.):

1.- Son vendidos o licenciados al público en general.

2.- Los mantiene y actualiza el propio vendedor, quien conserva los derechos de propiedad intelectual

3.- Están disponibles en forma de múltiples copias y su código no puede ser modificado por el usuario.

orientado a objetivos, y se compone de cuatro fases iterativas (Bertoa, Troya y Vallecillo, s.f.):

- *Identificación*: En esta fase se definen una serie de objetivos, basados en un análisis detallado de los factores que influyen en el proceso de desarrollo.

- *Descripción*: En esta fase los criterios de evaluación se definen con mayor detalle, tomando mayor importancia los requisitos extra-funcionales como discriminadores entre productos similares. Aunque se reconoce la dificultad de obtener, expresar, cuantificar y probar este tipo de requisitos (Cysneiros y Leite, 1999, c.p. Bertoa, Troya y Vallecillo, s.f.)

- *Evaluación*: Esta fase usa determinadas técnicas para el análisis de los datos de evaluación, basándose en el coste estimado versus el análisis de beneficios para cada alternativa COTS.

- *Aceptación*: Concierne a la negociación de los contratos legales con los suministradores de componentes para el producto final.

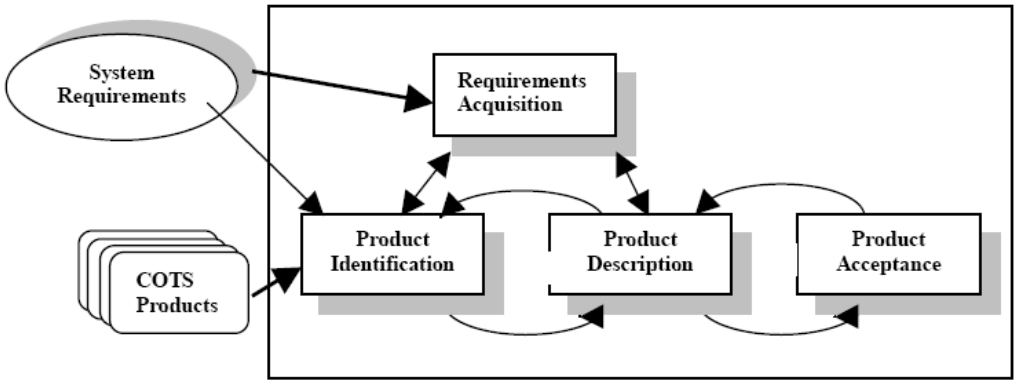

Figura 9. El Modelo CRE (Fuente: Universidad Nacional de Pernambuco, 2000)

4.7.- Metodología de trabajo a implementar:

Como metodología de trabajo se empleará el desarrollo ágil de software, específicamente el modelo Programación Extrema, el cual plantea cuatro actividades básicas: Planeación, Diseño, Codificación y Pruebas, ya descritas anteriormente.

4.8.- Tecnologías a utilizar:

Para el desarrollo de esta solución se plantea el uso de la programación orientada a objetos, la cual busca desarrollar software que refleje particularmente un esquema del mundo, mediante la construcción de objetos o clases (Schmuller, 2000) las cuales poseen diversos atributos, propiedades y acciones bajo las cuales se desenvolverá el software una vez codificado.

Dentro de la programación orientada a objetos tenemos diversas extensiones, entre las cuales destaca el Desarrollo de Software Basado en Componentes (DSBC). Ésta es una disciplina que se apoya en la combinación adecuada de componentes desarrollados por terceros, es decir, construir una aplicación (software) se convierte en la búsqueda y ensamblaje de "piezas" prefabricadas, desarrolladas en su mayoría por terceras casas y los cuales no pueden ser modificados, con la finalidad de reducir el tiempo de desarrollo, mejorar la fiabilidad del producto final mediante el uso de componentes ya probados y ser más competitivo en costos (Bertoa, Troya y Vallecillo, s.f.).

Esta metodología permite emplear plataformas de objetos distribuidos para la codificación del software como la tecnología *.NET Framework* de Microsoft, la cual viene definida como un componente integral de Windows que permite la creación y la ejecución de la siguiente generación de aplicaciones y servicios Web XML. El diseño del .NET Framework está enfocado a cumplir los siguientes objetivos (Microsoft, 2007):

- Proporcionar un entorno coherente de programación orientada a objetos, en el que el código de los objetos se pueda almacenar y ejecutar de forma local, ejecutar de forma local pero distribuida en Internet o ejecutar de forma remota.

- Proporcionar un entorno de ejecución de código que reduzca lo máximo posible la implementación de software y los conflictos de versiones.
- Ofrecer un entorno de ejecución de código que fomente la ejecución segura del mismo, incluso del creado por terceras personas desconocidas o que no son de plena confianza.
- Proporcionar un entorno de ejecución de código que elimine los problemas de rendimiento de los entornos en los que se utilizan secuencias de comandos o intérpretes de comandos.
- Ofrecer al programador una experiencia coherente entre tipos de aplicaciones muy diferentes, como las basadas en Windows o en el Web.
- Basar toda la comunicación en estándares del sector para asegurar que el código de *.NET Framework* se puede integrar con otros tipos de código.

Como manejador de base de datos se plantea emplear a *Microsoft SQL Server 2008*, el cual posee numerosas ventajas sobre otros manejadores de base de datos como las siguientes (Microsoft, 2008):

- Reducción de vulnerabilidades de datos
- Consolidación de los sistemas de datos y reducción de los requerimientos de energía
- Maximización de los recursos de hardware con gestión de recursos
- Ahorro de costos en inversión de hardware
- Monitoreo de sistemas centralizado
- Reducción de costos administrativos con mantenimiento automatizado
- Mejora de análisis del negocio

4.9.- Estructura organizativa del Proyecto:

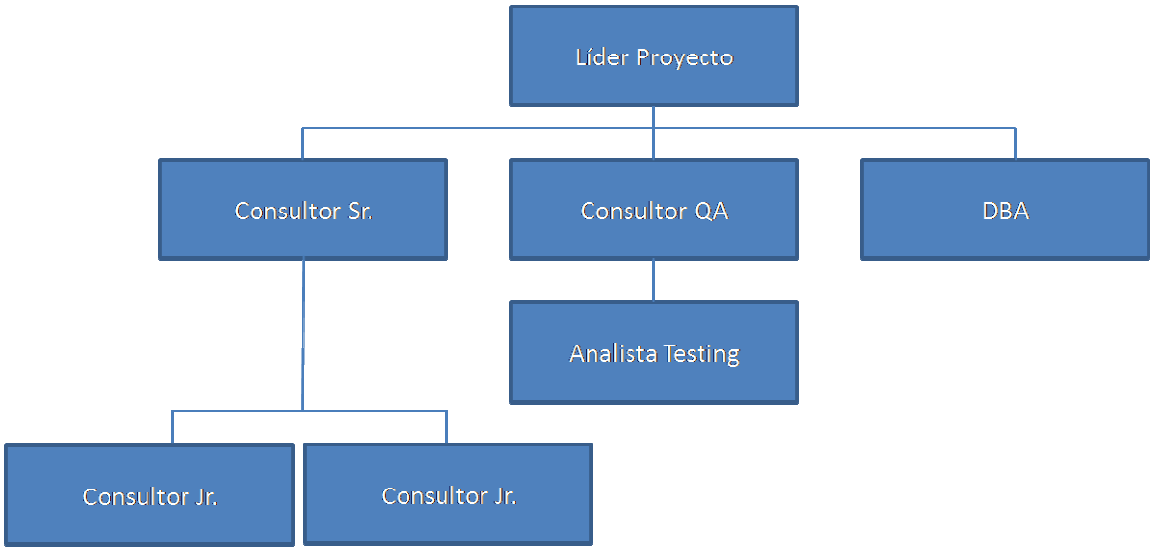

Figura 10. Organigrama del proyecto

4.10.- Perfil de los recursos requeridos:

Para este proyecto se definieron los siguientes perfiles para los cargos requeridos

Líder de Proyecto
Funciones y Responsabilidades
Conducir y establecer los diferentes estándares necesarios para el desarrollo de software a fin de garantizar la satisfacción del usuario final. Buscar mejoras de desempeño. Evaluar el desarrollo, implantación, administración, operación de los proyectos, y garantizar la calidad de los mismos, ademas de otras actividades inherentes a la naturaleza de sus funciones
Nivel Académico
Especialización
Requisitos del Cargo
Manejo de bases de datos , programación, estándares de calidad ISO - 9126, seguridad informática, licenciamientos de Software. Habilidades: proactivo, con capacidad de liderazgo, manejo y trabajo en equipo, habilidad numérica, capaz de laborar bajo presión, organizado, comunicativo.
Experiencia
Al menos 5 años en cargos similares

Tabla 8. Perfil de Cargo para Líder de Proyecto

Consultor Sr.
Funciones y Responsabilidades
Análisis de información con el fin de definir los requerimientos para el posterior desarrollo de la aplicación. Análisis, diseño y elaboración de los documentos técnicos a fin de ajustar y/o desarrollar nuevas funcionalidades de aplicación. Diseño de arquitectura de la solución de acuerdo al contexto de la aplicación.
Nivel Académico
Universitario graduado con certificaciones Microsoft (MCSD)
Requisitos del Cargo
Manejo de bases de datos . Habilidades: proactivo, con capacidad de liderazgo, manejo y trabajo en equipo, capaz de laborar bajo presión, organizado, comunicativo.
Experiencia
Al menos 5 años en cargos similares

Tabla 9. Perfil de Cargo para Consultor Sr.

Consultor QA.
Funciones y Responsabilidades
Garantizar el correcto diseño y elaboración de los documentos técnicos necesarios. Conocimientos en estándares de desarrollo tales como ISO 9126, COTS y CRE
Nivel Académico
Universitario graduado con certificaciones Microsoft
Requisitos del Cargo
Conocimientos básicos de normalización para bases de datos , conocimientos básicos en programación. Habilidades: proactivo, con capacidad de liderazgo, manejo y trabajo en equipo, capaz de laborar bajo presión, organizado, comunicativo.
Experiencia
Al menos 5 años en cargos similares

Tabla 10. Perfil de Cargo para Consultor QA.

DBA
Funciones y Responsabilidades
Análisis de información con el fin de definir los requerimientos necesarios para el posterior desarrollo de las bases de datos de la aplicación. Análisis, diseño y elaboración de triggers, stored procedures, paquetes de integration services. Diseño de arquitectura de la base de datos de acuerdo al contexto de la aplicación.
Nivel Académico
Universitario graduado con certificaciones en manejadores de Bases de Datos SQL Server
Requisitos del Cargo
Conocimientos básicos en programación, Conocimientos de Cubos OLAP y Business Inteligence, Habilidades: proactivo, con capacidad de liderazgo, manejo y trabajo en equipo, capaz de laborar bajo presión, organizado, comunicativo.
Experiencia
Al menos 5 años en cargos similares

Tabla 11. Perfil de Cargo para DBA

Consultor Jr.
Funciones y Responsabilidades
Análisis de información para el posterior desarrollo de la aplicación.
Nivel Académico
Universitario en últimos años de estudio o T.S.U.
Requisitos del Cargo
Conocimientos de programación bajo .Net Framework y/o Lenguaje orientado a Objetos. Habilidades: proactivo y capacidad de trabajo en equipo, capaz de laborar bajo presión, organizado, comunicativo.
Experiencia
Al menos 2 años de experiencia

Tabla 12. Perfil de Cargo para Consultor Jr.

Analista Testing
Funciones y Responsabilidades
Análisis de información para el posterior desarrollo de la aplicación.
Nivel Académico
Universitario en últimos años de estudio o T.S.U.
Requisitos del Cargo
Conocimientos básicos de programación bajo .Net Framework y/o Lenguaje orientado a Objetos. Habilidades: proactivo y capacidad de trabajo en equipo, capaz de laborar bajo presión, organizado, comunicativo.
Experiencia
Al menos 1 año de experiencia

Tabla 13. Perfil de Cargo para Analista Testing

4.11.- Matriz de responsabilidad del proyecto:

Actividad	Rol	Responsabilidad	Observaciones
1.- Planeación			En esta actividad se define de forma previa el requerimiento, identificando problemas, objetivos y necesidades a satisfacer
1.1.- Identificación de problemas y objetivos	Líder Proyecto	Identificar necesidades y objetivos	
1.2.- Análisis de las necesidades	Líder Proyecto	Analizar las necesides y definir plan de acción	
2.- Diseño			Para esta actividad se realiza un preliminar de la solución a fin de disponer de una guía que permita guiar hacia la consecución satisfactoria de la necesidad planteada, ademas de definir la documentación estándar para el proyecto
2.1.- Determinación de los requerimientos	Líder Proyecto / Consultor Sr.	Determinar los requerimientos	
2.2.- Diseño del sistema			
2.2.1.- Creación Diagrama Casos - Usos	Líder Proyecto / Consultor Sr.	Realizar diagrama caso de uso	
2.2.2.- Creación Diagrama Clases	Líder Proyecto / Consultor Sr.	Realizar diagrama caso de clases	
2.2.3.- Creación Diagrama Secuencia	Líder Proyecto / Consultor Sr.	Realizar diagrama caso de secuencia	
2.2.4.- Desarrollo Diagrama Entidad-Relación	DBA / Consultor Sr.	Realizar diagrama caso de Entidad - Relación	
2.2.5.- Diseño Pantallas Usuario	Líder Proyecto / Consultor Sr.	Realizar diseño de pantallas de usuario	
3.- Codificación			Esta fase es la construcción de la solución mediante el uso de las herramientas idóneas, además de la documentación técnica de la solución
3.1.- Desarrollo y Documentación			
3.1.1.- Normalización Base Datos	DBA	Definir estructura y arquitectura de Base de Datos	
3.1.2.- Desarrollo Base Datos	DBA	Elaborar e implementar Base de Datos y demás funcionalidades (Trigger, Stored Procedures, otros)	
3.1.3.- Desarrollo Clases	Consultor Sr. / Consultor(es) Jr.	Elaborar las clases necesarias para interacción con bases de datos	
3.1.4.- Desarrollo Pantallas Usuario	Consultor Sr. / Consultor(es) Jr.	Desarrollar las pantallas de interacción de la aplicación con el usuario final	
3.1.5.- Pruebas Unitarias	Consultor QA / Analista Testing	Realizar pruebas a cada funcionalidad desarrollada a fin de garantizar el cumplimiento de estándares	
3.1.6.- Documentación	Consultor QA / Analista Testing	Realizar la documentación técnica necesaria a fin de futuros desarrollos de mantenimiento	
4.- Pruebas			En esta fase se culmina el requerimiento, se realizan los últimos ajustes y pruebas antes de entregar la solución definitiva
4.1.- Pruebas y mantenimiento	Analista Testing / Consultor(es) Jr.	Realizar pruebas de desgaste en diversos escenarios a fin de comprobar el comportamiento de los diversos módulos de la aplicación y realizar ajustes necesarios	
4.2.- Implementación y evaluación	Líder Proyecto / Consultor Sr.	Implantar en ambiente de producción el producto final de manera conjunta con el cliente	

Tabla 13. Matriz de responsabilidad del proyecto.

4.12.- Gestión de comunicaciones del Proyecto:

El proceso de las comunicaciones en el proyecto es aquél que incluye los procesos necesarios para asegurar que el proyecto genere, recolecte, distribuya, almacene y disponga de información para así lograr un vínculo crítico entre las personas, las ideas y la información necesaria para el éxito del proyecto (Lledó y Rivarola, 2007).

De acuerdo a lo planteado anteriormente, se define un plan de comunicaciones para el proyecto de estudio de la siguiente manera:

- Estructura de Archivos: Se determinaron una serie de documentos, mediante los cuales se recolectarán y almacenarán los distintos tipos de información, tales como: Informes, Documento de requisitos, Documentos de control de cambios, Manual técnico, Diagrama Casos – Usos, Diagrama Clases, Diagrama Secuencia, Diagrama Entidad-Relación, Documentos de diseño para pantallas de usuarios, manual de usuario

- Estructura de distribución: Para la distribución de la información se empleará el correo electrónico, mensajería instantánea y en aquellos casos necesarios correspondencia física.

- Descripción de la información: El formato a utilizar para la creación de los documentos será estándar de acuerdo al empleado por diversas empresas, es decir, se especificará el numero de versión, creadores, titulo del documento, alcance y otra información que sea de relevancia para el documento

- Agenda de producción: Se agendarán reuniones periódicas de avances con el fin de determinar el avance de determinadas actividades y detectar posibles incidencias que afecten el desarrollo del proyecto

- Métodos de acceso a la información: Se determinará un espacio en un servidor de la organización en donde se almacenarán de acuerdo a ciertos criterios los documentos realizados y sus versiones.

- Actualización del plan de comunicaciones: Se realizarán actualizaciones al plan de comunicaciones de forma periódica de acuerdo al nivel de avance del proyecto o la actividad.

4.13.- Identificación de los riesgos del Proyecto:

Debido a que todos los proyectos y negocios implícitamente conllevan algún tipo de riesgo, se hace necesario realizar la administración de estos para así lograr los resultados implícitos en el plan del proyecto (Lledó y Rivarola, 2007).

En el caso de desarrollo de proyectos de software se han realizado diversas categorizaciones de riesgos (Pressman, 2005):

- *Riesgos del proyecto*: los cuales amenazan el plan del proyecto, trayendo como consecuencia que de concretarse estos riesgos afecten la calendarización del proyecto y los costos aumenten.
- *Riesgos técnicos*: amenazan la calidad y actualidad del software, provocando que la implementación se vuelva difícil o imposible.
- *Riesgos de negocios*: los cuales amenazan la viabilidad del software que se construirá. Los riesgos de negocios más comunes son:
- *Riesgo de mercado:* es decir un producto o sistema excelente que nadie quiere.
- *Riesgo de estrategia*: un producto que no encaja con la estrategia comercial global de la compañía.
- *Riesgo de ventas:* la construcción de un producto que la fuerza de ventas no sabe cómo vender.

- *Riesgo administrativo:* la pérdida de apoyo de los altos ejecutivos motivado a un cambio en el enfoque o en el personal.
- *Riesgo presupuestal:* originada por la pérdida presupuestaria o del personal asignado.

Para lograr una gestión de riesgos efectiva en el desarrollo de software, se plantean siete principios efectivos (Software Engineering Institute SEI, 2002, c.p. Pressman, 2005):

Mantenimiento de una perspectiva global: ver los riesgos del software dentro del contexto del sistema en el que está el componente y el problema de negocios que se intenta resolver

Tener una visión previsora: Pensar en los riesgos que pudiesen surgir en el futuro

Alentar la comunicación abierta: Si alguien establece un riesgo potencial, este no debe ser descartado, si un riesgo se propone de manera informal, debe ser considerado. Todos los participantes y usuarios deben ser alentados a sugerir riesgos en cualquier momento

Integración: en el proceso de desarrollo de software debe estar integrada una consideración de los riesgos

Enfatizar un proceso continuo: esto implica atención por parte del equipo a lo largo del proceso de desarrollo del software, modificar los riesgos identificados y agregar unos nuevos a medida que se logre un mejor criterio.

Desarrollo de una visión conjunta del producto: esto permitirá una mejor identificación y evaluación de riesgos gracias a que todos los participantes comparten la misma visión del software

Alentar el trabajo en equipo: esto implica combinar los talentos, habilidades y conocimientos de los participantes cuando se lleven a cabo actividades de gestión de riesgos.

Con base a lo anterior tenemos que para este proyecto se implementara la siguiente estrategia para administrar efectivamente los riesgos que pudieran presentarse durante el desarrollo del proyecto:

Identificar los riesgos: Para la identificación del riesgo se creará un checklist basado en las siguientes subcategorías genéricas (Pressman, 2005):

- Tamaño del producto: riesgos asociados con el tamaño global del software
- Impacto en el mercado: Riesgos asociados con las restricciones impuestas en el mercado
- Características del cliente: Riesgos asociados con la sofisticación del cliente y la habilidad del desarrollador para comunicarse con él de forma oportuna
- Definición del proceso: Riesgos asociados con el grado en el que se ha definido el proceso del software y en el que le da seguimiento la organización o quien lo desarrolla
- Entorno del desarrollo: Riesgos asociados con la disponibilidad y calidad de las herramientas que se utilizarán en la construcción del software
- Tecnología a construir: Riesgos asociados con la complejidad del sistema a construir y la "novedad" de la tecnología implícita en el sistema.
- Tamaño y experiencia de la plantilla de personal: Riesgos asociados con la experiencia global técnica y en el proyecto de los desarrolladores y profesionales encargados de realizar el proyecto.

Medir probabilidad de impacto: para determinar el impacto de los riesgos determinados en el paso anterior se realizará una tabla de riesgos, en dicha tabla se determinaran mediante columnas los riesgos, su afectación en tiempo, costo, calidad y alcance, además de su impacto en el proyecto, para este último se determinará una escala definida de la siguiente manera:

Muy bajo = 0,10

Bajo = 0,30

Moderado = 0,50

Alto = 0,70

Muy Alto = 0,90

La ponderación del impacto será determinada mediante la aplicación de juicio experto basado en anteriores experiencias y proyectos de desarrollo de software anteriores, luego de completada la información en las columnas anteriores, cada ítem se ordenarán según la probabilidad de ocurrencia e impacto, los riesgos de alta probabilidad e impacto se filtran hacia lo alto de la tabla y los de baja probabilidad irán hacia el fondo de la tabla, obteniéndose una priorización de primer orden.

Identificación de Riesgos						
No.	**Riesgo**	**Impacto**	**Tiempo**	**Costo**	**Calidad**	**Alcance**
1	Cambio de requerimiento	Alto	x	x		x
2	Falta de Licencias	Alto	x	x	x	
3	Bajo rendimiento participantes	Alto	x		x	
4	Ausencia de algún analista	Moderado	x			
5	Ausencia de PC	Alto	x		x	
6	Bajo conocimiento de los analistas	Moderado	x		x	
7	Virus en los PC	Alto	x	x		
8	Virus en el servidor	Muy Alto	x	x		
9	Pérdida de información en Base de Datos	Muy Alto	x	x		
10	Corte del servicio de Internet	Moderado	x			
11	Falta de mercado para el producto	Muy Alto	x	x		

Tabla 14. Tabla de identificación de riesgos.

El resultado de esta tabla será la priorización de los riesgos detectados en el proyecto, es decir, se ordenarán los ítems de acuerdo al impacto y su probabilidad de ocurrencia, por ejemplo, un evento de alto impacto pero con baja probabilidad de ocurrencia no debería absorber una cantidad significativa de tiempo en comparación con otro que tenga alto impacto y una probabilidad de ocurrencia entre alta y media.

Identificación de Riesgos						
No.	**Riesgo**	**Impacto**	**Tiempo**	**Costo**	**Calidad**	**Alcance**
8	Virus en el servidor	Muy Alto	x	x		
9	Pérdida de información en Base de Datos	Muy Alto	x	x		
11	Falta de mercado para el producto	Muy Alto	x	x		
1	Cambio de requerimiento	Alto	x	x		x
2	Falta de Licencias	Alto	x	x	x	
3	Bajo rendimiento participantes	Alto	x		x	
5	Ausencia de PC	Alto	x		x	
7	Virus en los PC	Alto	x	x		
6	Bajo conocimiento de los analistas	Moderado	x		x	
4	Ausencia de algún analista	Moderado	x			
10	Corte del servicio de Internet	Moderado	x			

Tabla 15. Tabla de priorización de riesgos.

Finalmente para determinar la priorización en el tratamiento de los riesgos, los mismos serán colocados en una matriz de probabilidad de impacto.

Probabilidad						
Muy alta			4,6,10	1,2,3,5,7	8,9,11	
Alta						
Moderada						
Baja						
Muy Baja						
	Muy bajo	**Bajo**	**Moderado**	**Alto**	**Muy Alto**	**Impacto**

Tabla 16. Matriz de impacto en dimensión Tiempo.

Probabilidad						
Muy alta						
Alta						
Moderada				1,2,7	8,9,11	
Baja						
Muy Baja						
	Muy bajo	**Bajo**	**Moderado**	**Alto**	**Muy Alto**	**Impacto**

Tabla 17. Matriz de impacto en dimensión Costo.

Probabilidad						
Muy alta						
Alta		2,3,5				
Moderada		6				
Baja						
Muy Baja						
	Muy bajo	**Bajo**	**Moderado**	**Alto**	**Muy Alto**	**Impacto**

Tabla 18. Matriz de impacto en dimensión Calidad.

Alcance **Probabilidad**						
Muy alta						
Alta	1					
Moderada						
Baja						
Muy Baja						
	Muy bajo	**Bajo**	**Moderado**	**Alto**	**Muy Alto**	**Impacto**

Tabla 19. Matriz de impacto en dimensión Alcance.

Para la cuantificación de los riesgos determinados se aplicará la fórmula de exposición al riesgo global (Hall, 1998, c.p. Pressman, 2005) la cual se determina mediante la siguiente fórmula:

ER = P x C

P = Probabilidad de ocurrencia

C = Costo al proyecto en caso de ocurrir

4.14.- Matriz de adquisiciones del Proyecto:

Para determinar el proveedor más conveniente para la adquisición de los insumos de hardware necesarios se evaluarán tres proveedores reconocidos en el área de venta de equipos de computación, se tomarán criterios principales como: Costo, Mantenimiento, Garantía y Tiempo de Entrega a los cuales se le dará una ponderación del 1 al 10 siendo 1 el más bajo y 10 el más alto.

		VJA		Mundo PC		M.A. Computers	
Criterio	**Ponderación**	**Calificación**	**Puntaje**	**Calificación**	**Puntaje**	**Calificación**	**Puntaje**
Costo	50%	5	2,5	4	2	3	1,5
Mantenimiento	25%	6	1,5	4	1	2	0,5
Garantía	15%	5	2,5	3	0,45	3	0,45
Tiempo de entrega	10%	4	0,4	5	0,5	4	0,4
Totales	**100%**		**6,9**		**3,95**		**2,85**

Tabla 20. Matriz de adquisiciones del proyecto.

El resultado de la evaluación de cada uno de los criterios, de forma preliminar indica que el proveedor más conveniente es VJA Compute, el cuál obtuvo el puntaje más alto entre el resto de los proveedores, Para el caso de las licencias se estudiará la posibilidad de que las mismas sean proporcionadas por Microsoft Inc. Mediante algún tipo de acuerdo, o por el contrario las mismas serán adquiridas mediante algún proveedor en U.S.A.

Capítulo V
Conclusiones y Recomendaciones

Luego de culminar el presente trabajo de investigación se llegaron a las siguientes conclusiones:

- El proyecto de desarrollo de software lejos de lo que se pueda pensar es uno de los más difíciles de controlar en cuanto tiempo y recursos, más si los recursos pertenecen a diferentes áreas como base de datos o redes, prácticamente cada actividad que se plantea en un Project chart entran dentro del denominado camino crítico ya que casi siempre son dependientes las unas con las otras, de allí la importancia de administrar los recursos que integren el proyecto, de hacerlos sentir que forman parte de un sistema que interactúa entre sí y se retroalimenta de forma continua y que sin la colaboración o el fallo de uno de estos elementos, se afecta de forma irremediable el proyecto y al final el producto.

- Una de las causas por la cual muchos proyectos de desarrollo de software fallan o son proyectos de "nunca acabar" se debe principalmente a que la mayoría no son analizados de forma profunda antes de dar inicio al mismo. Todos son llevados de forma empírica y en su totalidad poseen un grave problema de alcance y tiempo, lo cual repercute principalmente en su costo final.

- El tiempo, la credibilidad y la confianza frente al cliente son elementos difíciles de recuperar.

Igualmente se cree de forma errónea que al disponer de una excelente plantilla de profesionales con certificaciones y años de experiencia se garantiza el buen desenvolvimiento de un proyecto para toparse en algún momento del avance del mismo que se está desarrollando algo que simplemente no es lo que el cliente requería, o el cliente cambio de parecer respecto al planteamiento inicial, trayendo consigo graves consecuencias respecto al desenvolvimiento del proyecto. Para evitar que continúen ocurriendo situaciones como esta se plantean las siguientes recomendaciones:

- Se debe tener especial cuidado en los tiempos de entrega, especialmente para el desarrollo de componentes de software (codificación).
- Se deben tomar en cuenta cualquier tipo de riesgo que pueda existir, es decir, tomar en consideración la pérdida de información en un servidor, la existencia de algún virus que pueda propagarse por la red, ya que estos eventos fácilmente pueden atrasar un proyecto en días.
- Siempre se debe contar con la asesoría de un recurso con experiencia por cada área que se determine (Base de Datos, Calidad y Desarrollo para este caso) este recurso debe ser el apoyo de aquellos que no cuenten con la suficiente destreza en este tipo de proyecto.
- No por contar con una amplia cantidad de certificaciones en herramientas de desarrollo o de base de datos signifique que el recurso es el indicado para ser asignado en determinada actividad, para proyectos de alta presión la experiencia es mejor consejera.
- Es altamente recomendable dedicar el tiempo necesario para revisar el requerimiento planteado y hacer las preguntas puntuales, ya que una duda en mitad de una actividad puede afectar el sentido del proyecto.
- Evitar en reuniones informales adelantar sin realizar un estudio adecuado del problema elementos tan delicados como tiempo y costo al cliente, ya que esta fecha y/o costo es la que el cliente siempre tendrá en consideración.
- Siempre se debe contar con el cliente o stakeholder del proyecto al alcance de la mano y siempre debe ser consultado ante cualquier eventualidad o notificar al cliente de los avances del proyecto
- Siempre se debe dejar constancia de cualquier acuerdo que afecte los tiempos de una actividad, así sea un post-it, esto no debe tomarse como desconfianza, ya que cualquier decisión puede afectar el curso de proyecto.
- Contar siempre con un repositorio centralizado de datos, es decir, que cada uno de los integrantes trabaje con sus propias fuentes de código, y que cada cierto tiempo éste sea respaldado en un servidor para tal función.
- Mantener un hilo de comunicación constante entre todos los integrantes del equipo de proyecto.

Bibliografía

Abud, M. (2004). Calidad en la Industria del Software. La Norma ISO-9126. Recuperado en Noviembre 02, 2009, de http://www.revistaupiicsa.20m.com/Emilia/RevEneAbr04/Antonieta1.pdf

AdventNet Inc. (2009). Descripción del Producto ServiceDesk Plus. Recuperado en Junio 01, 2009, de http://www.adventnet.co.nz/products/service-desk/index.html

Alex, N. (2008, abril 3). What Computer Support Experts Do? Recuperado en Junio 01, 2009, de http://ezinearticles.com/?What-Computer-Support-Experts-Do?&id=1088601

Balestrini, M. (2002). Como se elabora el proyecto de Investigación (6ª ed). Caracas: BL Consultores Asociados Servicio Editorial.

Bertoa, M. Troya, J. Y Valecillo A. Aspectos de Calidad en el Desarrollo de Software Basado en Componentes. Recuperado en Noviembre 01, 2009, de http://www.lcc.uma.es/~av/Publicaciones/02/CalidadDSBC.pdf

Extreme Programming (2009). Extreme Programming: a gentle introduction. Recuperado en Junio 24, 2009, de http://www.extremeprogramming.org/

German, H. (2005, febrero 22). The dark side of HelpDesk SLAs. Recuperado en Junio 01, 2009, de http://ezinearticles.com/?The-Dark-Side-of-Help-Desk-SLAs&id=15795

Gobierno de España - Consejo Superior de Administración Electrónica (2009) Métrica. Versión 3, Metodología de Planificación, Desarrollo y Mantenimiento de sistemas de información. Recuperado en Noviembre 02, 2009 de http://www.csi.map.es/csi/metrica3/index.html

Hill, S. (2008, abril 14). Definition of Information technology. Recuperado en Mayo 31, 2009, de http://ezinearticles.com/?Definition-of-Information-Technology&id=1109986

Instituto Mexicano de Teleservicios (2009). El Proceso de Mesa de Ayuda (Primera Parte). Recuperado en Mayo 20 2009, de http://www.imt.com.mx/recontact/07/proceso.php

ITAUTEC (2.009). Help Desk. Recuperado en Mayo 20, 2009, de http://www.itautec.com.br/iPortal/sp-SP/97aacba6-0e82-4295-9ee5-bf859d6f5796.htm

Kendall, K. Y Kendall, J. (2005). Análisis y diseño de sistemas (6ª ed). México: Pearson educación.

Lledó, P. Y Rivarola G. Gestión de Proyectos (1ª ed). Buenos Aires: Prentice Hall - Pearson Educación.

Microsoft (2009). Información general y conceptual sobre .NET Framework. Recuperado en Noviembre 01, 2009, de http://msdn.microsoft.com/es-ve/library/zw4w595w.aspx

Microsoft (2009). Microsoft SQL Server 2008. Recuperado en Noviembre 01, 2009, de http://www.microsoft.com/latam/sqlserver/ahorro.aspx.

Miller, S. (2007, junio 23). Helpdesk perfomance metrics. Recuperado en Junio 01, 2009, de http://ezinearticles.com/?Helpdesk-Performance-Metrics&id=618616

P&M Consulting and Training Specialists (2009). Virtual Class Index. Recuperado en Noviembre 01, 2009, de http://www.pmconsulting.com.mx/VirtualClass/index.html

Panner, M. (2008). Becoming a better Project Manager: Five steps to becoming a strategic project manager. Recuperado en Abril 03, 2009, de http://solutions.internet.com/3851_betterpm

Pressman, R. (2005). Ingeniería del software: un enfoque práctico (6ª ed). México: Mc. Graw Hill.

Project Management Institute (2004). Fundamentos de la dirección de proyectos (Guía del PMBOK) (3ª ed). E.E.U.U. Project Management Institute. Inc.

Sabino, C. (1992). El proceso de investigación. Caracas: Panapo.

Schmuller, J (2000). Aprendiendo UML en 24 horas, México: Pearson Educación.

Santalla, Z. (2008). Guía para la elaboración formal de reportes de investigación. Caracas: Publicaciones U.C.A.B.

Soporte Remoto de México (2009). Diferencia entre operación de un Help Desk y administración de un Service Desk. Recuperado en Mayo 21, 2009, de http://www.soporteremoto.com.mx/help_desk/articulo01.html

Troika Inc. (2009). Descripción del Producto Resolve IT!, recueperado en Mayo 19, 2.009, de http://www.resolveit.com/spanish/helpdesk.htm

Universidad Católica Andrés Bello (2009). Instructivo Integrado para Trabajos Especiales de Grado. Caracas.

Universidade Nacional de Pernambuco (2000). Requirements Engineering for COTS Selection. Recuperado en Noviembre 01, 2009, de http://wer.inf.puc-rio.br/WERpapers/artigos/artigos_WER00/alves.pdf

Xynco (2009). Beneficios de ITIL®. Recuperado en Mayo 21 2009 de http://www.xynco.com/servicios/consultorias/itil.html

Otras Referencias

Blanco, A. (2008), Formulación y Evaluación de Proyectos (7ª ed). Caracas: Editorial Texto.

Cárdenas, P. (2007). Desarrollo de un sistema de información para el registro de la declaración formal de los accidentes de trabajo reportados ante el INPSASEL desde la perspectiva de la gerencia de Proyectos. U.C.A.B.

Palacios, L. (2007), Gerencia de Proyectos un enfoque latino (4ª ed). Caracas: Publicaciones U.C.A.B.

Paradiso, A. (2006). Desarrollo del Proyecto de "Diseño de un sistema en entorno Web basado en estándares de Calidad y Técnicos , que acople en un solo producto varias soluciones de software creadas por la empresa Stratos Virtual, para satisfacer los requerimientos específicos de un cliente. U.C.A.B.

Rivero, A. (2007). Propuesta de gestión de un sistema de información para el registro y control de los procesos referentes a imprentas autorizadas por el SENIAT a nivel nacional. U.C.A.B.

Vazquez, M. Y Carrera, L. (2007). Técnicas del trabajo de Investigación. Caracas: Editorial Panapo