UNIDAD DE GESTIÓN DE TECNOLOGÍAS

DEPARTAMENTO DE ELÉCTRICA Y ELECTRÓNICA

CARRERA DE COMPUTACIÓN

TRABAJO DE TITULACIÓN, PREVIO A LA OBTENCIÓN DEL TÍTULO DE TECNÓLOGO EN COMPUTACIÓN

TEMA: "IMPLEMENTACIÓN DE UN PROTOTIPO DE SISTEMA PARA EL CONTROL DE MANTENIMIENTO DE LOS VEHÍCULOS DE MANDO Y CONTROL EN EL COMANDO LOGÍSTICO No. 25 REINO DE QUITO"

AUTORES: CAÑAR SIZA MARCO EDUARDO
GUANOLUISA GUALLICHICO ESTEBAN TEOVALDO

DIRECTOR: ING. DIEGO ANDALUZ

LATACUNGA

2017

DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

CARRERA DE COMPUTACIÓN

CERTIFICACIÓN

Certifico que el trabajo de titulación, "IMPLEMENTACIÓN DE UN PROTOTIPO DE SISTEMA PARA EL CONTROL DE MANTENIMIENTO DE LOS VEHÍCULOS DE MANDO Y CONTROL EN EL COMANDO LOGÍSTICO No. 25 REINO DE QUITO" realizado por los señores CBOS. DE I.M. CAÑAR SIZA MARCO EDUARDO, CBOS. DE COM. GUANOLUISA GUALLICHICO ESTEBAN TEOVALDO cumple con los requisitos teóricos, científicos, técnicos, metodológicos y legales establecidos por la Universidad de Fuerzas Armadas ESPE, por lo tanto, me permito acreditarlo y autorizar a los señores CBOS. DE I.M. CAÑAR SIZA MARCO EDUARDO, CBOS. DE COM. GUANOLUISA GUALLICHICO ESTEBAN TEOVALDO para que lo sustente públicamente.

Latacunga, 06 de junio de 2017

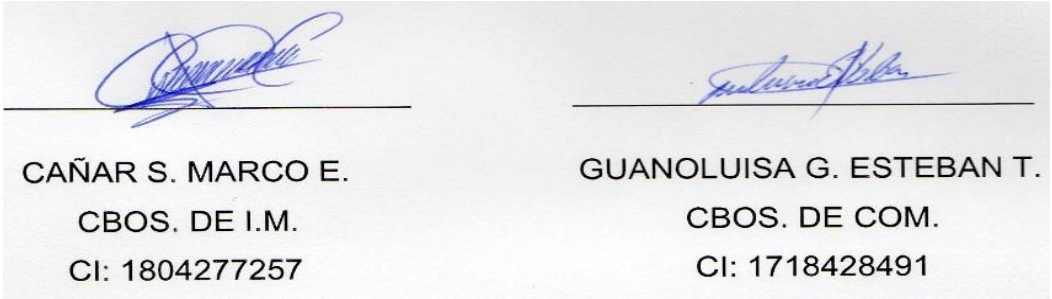
ING. DIEGO ANDALUZ
DIRECTOR DEL TRABAJO DE TITULACIÓN

DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN
CARRERA DE COMPUTACIÓN
AUTORÍA DE RESPONSABILIDAD

Yo, **CBOS. DE I.M. CAÑAR SIZA MARCO EDUARDO,** con cédula de identidad N° 1804277257, y **CBOS. DE COM. GUANOLUISA GUALLICHICO ESTEBAN TEOVALDO,** con cédula de identidad N° 1718428491, declaramos que este trabajo de titulación **"IMPLEMENTACIÓN DE UN PROTOTIPO DE SISTEMA PARA EL CONTROL DE MANTENIMIENTO DE LOS VEHÍCULOS DE MANDO Y CONTROL EN EL COMANDO LOGÍSTICO No. 25 REINO DE QUITO"** ha sido desarrollado considerando los métodos de investigación existentes, así como también se ha respetado los derechos intelectuales de terceros considerándose en las citas bibliográficas.

Consecuentemente declaro que este trabajo es de mi autoría, en virtud de ello me declaro responsable del contenido, veracidad y alcance de la investigación mencionada.

Latacunga, 06 de junio de 2017



DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

CARRERA DE COMPUTACIÓN

AUTORIZACIÓN

Yo, **CBOS. DE I.M. CAÑAR SIZA MARCO EDUARDO y CBOS. DE COM. GUANOLUISA GUALLICHICO ESTEBAN,** autorizo a la Universidad de las Fuerzas Armadas ESPE publicar en la Biblioteca Virtual de la Institución el presente trabajo de titulación **" IMPLEMENTACIÓN DE UN PROTOTIPO DE SISTEMA PARA EL CONTROL DE MANTENIMIENTO DE LOS VEHÍCULOS DE MANDO Y CONTROL EN EL COMANDO LOGÍSTICO No. 25 REINO DE QUITO"** cuyo contenido, ideas y criterios son de mi autoría y responsabilidad.

Latacunga, 06 junio del 2017

CAÑAR S. MARCO E.
CBOS. DE I.M.
CI: 1804277257

GUANOLUISA G. ESTEBAN T.
CBOS. DE COM.
CI: 1718428491

DEDICATORIA

Este trabajo fue realizado con mucho cariño y amor a todas las personas importantes en nuestra vida, que siempre estuvieron cuando más los necesitábamos, con sus consejos y apoya moral que motivaban día a día para continuar y no desmayar ante las adversidades y poder conseguir este sueño tal anhelado que hoy se hace realidad con mucha humildad esta tesis se la dedicamos a ustedes.

Esteban

Marco

AGRADECIMIENTO

A nuestra institución Fuerzas Armadas que nos dio la oportunidad y no la desperdiciamos, de prepáranos en el campo académico en la mejor Universidad del Ecuador.

A todos mis profesores por este andar por la vida e influyeron con sus lecciones y experiencias en formarnos como unas personas de bien y preparadas para los retos que pone la vida, a todos y cada uno de ellos les agradezco.

Esteban

Marco

ÍNDICE DE CONTENIDOS

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

RESUMEN

Este trabajo de titulación consiste en la implementación de un software informático con la utilización de un Sistema de Gestión de Base de Datos (Oracle) y un Lenguaje de Programación Orientado a Objetos (Java 8.0) los mismos que por estándares son manejados dentro de las Fuerzas Armadas, contiene toda la información básica y lineamientos acerca de la propuesta planteada.

Se establece el Marco Teórico, los conceptos necesarios de cada una de las herramientas que ayudaron a la realización de este proyecto, optimizando recursos y tiempo siendo capaz de contribuir de manera óptima a la resolución de la problemática antes planteada y así poder ayudar de forma directa o indirecta a las Unidades de Mantenimiento Automotriz del Ejército Ecuatoriano. Aplicando la Especificación de Requisitos del Sistema (ERS) basado en el estándar IEEE 830 en donde se obtiene de la manera más clara posible los requisitos funcionalidades y no funcionales del sistema, cumpliendo con todo el ciclo de vida del software (Analisis, Diseño, Codificación, Pruebas, Mantenimiento) con el objetivo implementar un software que tenga una interfaz amigable al usuario sea robusto y de fácil manejo.

Finalmente se puede observar las Conclusiones y Recomendaciones obtenidas durante todo el proceso del desarrollo de este trabajo de titulación.

Palabras claves:

BASE DE DATOS

ADMINISTRACIÓN

SISTEMA

OBJETO

ORIENTADO

ABSTRACT

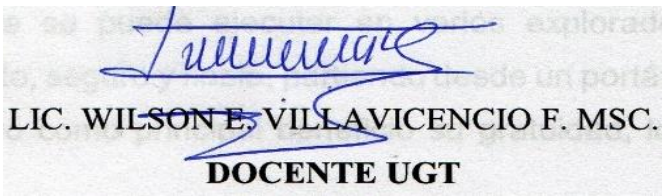
This technical written work consists of design and implementation of a software based in the use of a Database Management System (Oracle) and an Object Oriented Programming Language (Java 8.0) which via standards are controlled in the Fuerzas Armadas del Ecuador, it contains all the basic information and guidelines about the proposed proposal.

It stablishes the theoretical framework, the necessary concepts of each one of the tools that help to carry out this project, optimizing resources and time being able to give in an optimal way to solve the focused problem in order to help in a direct or indirect way to the Unidades de Mantenimiento Automortriz of the Fuerzas Armadas. Applying the System Requirements Specification (ERS) based on the IEEE 830 standard where the functionalities and non-functional requirements of the system are obtained in a clearest mode, complying with the entire software life cycle (Analysis, Design, Coding, Testing, Maintenance) with the objective of implementing a software with a friendly interface user is strong and easy to use.

Finally, we can notice the obtained conclusions and recommendations during the whole process of development of this technical written work.

Keywords:
DATABASE
MANAGEMENT
SYSTEM
OBJECT
ORIENTED

Checked by:



CAPÍTULO I

INTRODUCCIÓN

Actualmente la necesidad de los usuarios promueve al uso de nuevas tecnologías para reducir tiempo y gastos, para lo cual se estable el sistema informático de mantenimiento vehicular, con ello obtener una mejora continua que impulse a optimizar el desempeño profesional de cada uno de los miembros de la Unidad Comando Logístico N°25 Reino de Quito, en el Departamento de Mantenimiento de Vehículos de Mando y Control, que se encuentra ubicado en la provincia de Pichincha cantón Quito.

El sistema informático tiene como finalidad automatizar el proceso de gestión administrativa y el control manual que se realiza actualmente, evitando la manipulación no apropiada de la información de la unidad, la cual perjudique de manera directa o indirecta, lo que conlleva aplicar paquetes informáticos como sistemas de gestión de base de datos ORACLE, lenguaje de programación orientado a la web JAVA, y con la ayuda de metodología de desarrollo de software que permitirá el control de cada una de las fases del desarrollo de dicho sistema informático.

ORACLE es una herramienta cliente/servidor que administra las bases de datos para un mejor funcionamiento, el mismo que es reconocido a nivel mundial por su capacidad, desenvolvimiento, pero no todas las entidades o personas pueden contar con este sistema informático por su costo es elevado.

JAVA es un lenguaje de programación orientado a objetos entendible y de fácil manejo, que se puede ejecutar en varios exploradores tiene como ventajas ser rápido, seguro y fiable, partiendo desde un portátil hasta una base de datos teniendo como principal beneficio su gratuidad, lo que permite un

acceso tanto personal y como para el sector empresarial aprovechando a fondo todos beneficios.

Este proyecto de titulación se desarrolló con el propósito de implementar los conocimientos académicos adquiridos en toda la formación profesional reflejado en la Unidad de Comando N°25 Reino de Quito, que la misma de valor y uso para un mejor control de la información que en esta unidad se utiliza con respecto al ámbito vehicular.

Fomentando el profesionalismo de cada uno de sus integrantes de esta unidad, aplicando su moral y ética profesional.

1.1. Planteamiento del Problema

Hoy en día los procesos administrativos y de control del mantenimiento correctivo, preventivo y predictivo de los vehículos del Comando Logístico No. 25 Reino de Quito (COLOGE), se han llevado de forma manual, lo cual ha ocasiona la perdida de información y una distribución inadecuada del tiempo a emplearse para el mantenimiento y entrega de los vehículos, el archivo manual del historial del mantenimiento de los vehículos ha dificultado la toma de decisiones en cuanto a la adquisición de materiales, lubricantes y repuestos para el funcionamiento de las unidades, pérdida de tiempo en las gestiones administrativas, reportes de datos historiales de los vehículos y gastos de mantenimiento.

En tal virtud con la adaptación de nuevos sistemas informáticos se lograra obtener registros electrónicos del proceso de mantenimiento de los vehículos y una búsqueda eficiente de la información, reduciendo la pérdida de los registros existentes y a su vez evitar la desviación de la información que pudiera perjudicar al personal que labora en esta unidad.

1.2. Antecedentes

Comando Logístico No. 25 Reino de Quito (COLOGE), es una Unidad Militar cuyas funciones principales son: Logísticas de abastecimiento, mantenimiento y transporte, esta acantonada en la Provincia de Pichincha, Cantón Quito, en lo cual se identificó que actualmente, se lleva un control manual de las actividades relacionadas con el mantenimiento de los vehículos de Mando y Control de todo el Ejército Ecuatoriano,

El Ejército Ecuatoriano posee, el Sistema Integrado del Ejército (SIFTE) que controla de manera general módulos de Inventarios de la Institución como: (Transporte, Logística, Armamento) y administración del Personal (militar activo y civil) que labora en la misma, en este sistema mencionada, inexiste el módulo de mantenimiento de vehículos, por cuanto la implantación de este prototipo de sistema requiere de gastos económicos para su desarrollo.

Existen softwares desarrollados para este tipo de control, así se puede mencionar el trabajo de titulación desarrollado por los estudiantes Guamán. A & Ordoñez. R (2016). Diseño e Implementación del Sistema de Gestión Vehicular para el Gobierno Provincial de Loja (Tesis de grado). Universidad Técnica Particular de Loja.

1.3. Objetivo General

- Implementar un sistema informático para el control de mantenimiento de los vehículos del departamento de Mando y Control, en el Comando Logístico N°25 Reino de Quito y su incidencia en la Unidad con la utilización de herramientas y metodología de desarrollo de software.

1.4. Objetivos Específicos

- Determinar los posibles aspectos y antecedentes que se van a utilizar dentro de la investigación con el fin de obtener los resultados esperados aplicando la metodología inductiva y deductiva.
- Recopilar información existente del proceso administrativo y mantenimiento que se realiza manualmente, mediante la obtención de información detallada de las funcionalidades y requerimientos del software, para la sustentabilidad de la base teórica en el proyecto.
- Implementar el sistema informático para un mejor funcionamiento dentro del departamento y así optimizar recursos humanos y económicos que dispone dicha unidad a través de medios electrónicos.
- Proponer conclusiones y recomendaciones de acuerdo a la implementación del sistema informático en el departamento de mantenimiento de vehículos de mando y control mediante el resultado de las pruebas realizadas.

1.5. Justificación

Este software permitirá automatizar la gestión administrativa y el procesos de control de mantenimiento de vehículos que se realizan manualmente en el Departamento de Mantenimiento del Comando Logístico No. 25 Reino de Quito del Ejército Ecuatoriano, favoreciendo el desempeño del personal que labora en la sección e involucrando a la tecnología como un medio indispensable en el mejoramiento del proceso de gestión de mantenimiento, sin pérdida de tiempo e información.

Considerando todos los problemas que se presentan al llevar manualmente la gestión administrativa y el control del proceso de mantenimiento del parque automotor del Comando Logístico No. 25 Reino de Quito, el desarrollo de esta aplicación se justifica por los siguientes motivos:

- Permitirá llevar una mejor gestión administrativa minimizando el tiempo de revisión de los documentos de vehículos y conductores.
- Gestionará de forma eficiente los gastos de mantenimiento para cada vehículo obteniendo reportes diarios, semanales, mensuales y anuales de acuerdo a las necesidades del usuario, con la ayuda de datos históricos de los vehículos y la obtención de esta información permitirá que el departamento tome decisiones en forma oportuna, de esta manera evitar la pérdida de gastos económicos y tiempo en la adquisición de materiales o repuestos para el debido mantenimiento de los automotores y a la vez cumpliendo su trabajo adecuadamente más activo y eficaz.
- Mejorará el orden y control en los diferentes procesos de mantenimiento dentro del departamento, como talleres, áreas y prioridades.

El desarrollo del sistema informático para el control de mantenimiento de los vehículos de Mando y Control en el Comando Logístico No. 25 Reino de Quito, se contara directamente con el apoyo de los involucrados del COLOGE, mismos que proporcionara información necesaria para su ejecución, además dentro del mercado tecnológico se dispone de recursos de hardware y software que hizo factible su diseño, vinculando de esta forma la parte teórica y práctica de los conocimientos adquiridos a lo largo de la vida estudiantil y de esta manera satisfacer las necesidades de la institución requirente.

1.6. Alcance

El alcance del presente trabajo de investigación que se desarrolló para los vehículos del Departamento de Mando y Control de Comando Logístico N°25 del Reino de Quito, permite la gestión de los procesos para el mantenimiento de los mismos.

El cual permite obtener información oportuna, rápida y eficaz sobre el estado de los vehículos, desplegando datos verídicos y confiables, así mismo

los datos históricos que cada una de las unidades poseen, de esta manera corregir los errores que se han generado en los registros y debido a los mantenimiento tardíos o innecesarios que se encuentra estipulados en libros de acuerdo a cada una de las marcas y tipo de vehículos que pertenecen de forma directa o indirecta a esta unidad para la constatación física de lo estipulado manualmente.

Este enfoque continuo y constante impulsará a nuevas versiones abarcando a todos los vehículos del Ejército Ecuatoriano, minimizando los riesgos de perdida de información y redundancia de las mimas.

CAPÍTULO II

MARCO TEÓRICO

2.1 Ingeniería de Software

La ingeniería de software es una tecnología con varias capas, debe basarse en un compromiso organizacional con calidad, Six Sigma y otras filosofías similares alimentan la cultura de mejora continua, y es esta cultura la que lleva en última instancia al desarrollo de enfoques cada vez más eficaces de la ingeniería de software el fundamento en el que se apoya la ingeniería de software es el compromiso con la calidad.

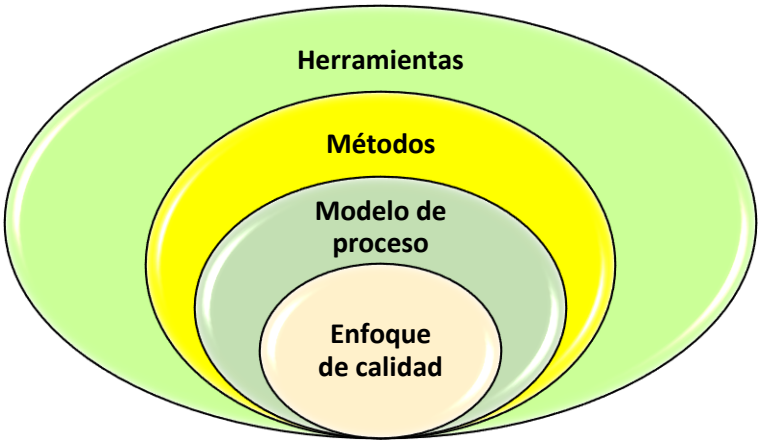


Figura 1. Capas de Ingeniería de Software

El fundamento para la ingeniería de software es la capa proceso. El proceso de ingeniería de software es el aglutinante que une las capas de la tecnología y permite el desarrollo racional y oportuno del software de cómputo. El proceso define una estructura que debe establecerse para la obtención eficaz de tecnología de ingeniería de software. El proceso de software forma la base para el control de la administración de proyectos de software, y establece el contexto en el que se aplica métodos técnicos, generando productos de trabajo (modelos, documentos, datos, reportes, formatos),

estableciendo puntos de referencia, se asegura la calidad y se administra los cambios de manera propia.

Los métodos de la ingeniería de software proporciona la experiencia técnica para elaborar software. Incluye un conjunto amplio de tareas, como comunicación, análisis de los requerimientos, modelación del diseño, construcción del programa, pruebas y apoyo. Los métodos de ingeniería de software se basan en un conjunto de principios fundamentales que gobiernan cada área de la tecnología e incluyen actividades de modelación y otras técnicas descriptivas.

Las herramientas de la ingeniería de software proporcionan un apoyo automatizado o semiautomatizado para el proceso de los métodos. Cuando se integran las herramientas de modo que la información creada por una pueda ser utilizada por otra, queda establecido un sistema llamada ingeniería de software asistido por computadora que apoya el desarrollo de software. (Roger S, 2010, págs. 11,12)

2.2. Metodología de Desarrollo de Software

Con la utilización de una metodología se busca el control y la corrección en cada etapa del desarrollo de un sistema, de esta manera permitiendo obtener a empresas desarrolladoras de software un diseño compacto del sistema y la entrega del producto de calidad en tiempo y costo estimados.

2.2.1. Metodología Agiles de Desarrollo de Software

En la actualidad las empresas desarrolladoras de software han adaptado a metodologías que les faciliten el diseño de sistemas, en el mercado de software de desarrollo se puede encontrar metodologías como (Estructurales, Orientada a objetos, Tradicionales y Agiles), siendo la más destacada y utilizada en este grupo, las Metodologías Agiles que surge de las

metodologías tradicionales para mejorar el desarrollo de sistemas, dependiendo al tipo de proyecto.

A. Características de la Metodologías agiles de Desarrollo de Software

- Un modelo de desarrollo ágil, generalmente es un proceso Incremental, (pequeños y frecuentes releases o entregas con ciclos rápidos).
- Cooperativo (Clientes y desarrolladores trabajan constantemente con una comunicación muy fina y constante).
- Sencillo (El método es fácil de aprender y modificar para el equipo, es bien documentado por medio de libros o la Web).
- Adaptativo (capaz de permitir cambios de último momento). (Metodologías Tradicionales, 2016)

B. Tipos de Metodologías Agiles de Desarrollo de Software

Entre las metodologías ágiles las destacadas hasta el momento se pueden nombrar:

- Xp (Extreme Programming)
- Scrum
- Crystal Clear
- DSDM (Dynamic Systems Developmemt Method)
- FDD (Feature Driven Development)
- ASD (Adaptive Software Development)
- XBreed
- Extreme Modeling. (Metodologías Tradicionales, 2016).

Como se puede conocer existe muchas metodologías agiles, cada una de ellas con sus propias características y atributos, misma son adoptadas dependiendo el tipo del proyecto y el entorno de desarrollo según los

requerimientos del software, siendo la más primordial para el desarrollo de un sistema en un corto plazo la metodología XP (Extreme Programming).

2.2.2. Metodología Agile de Desarrollo de Software XP (Extreme Programming)

Es una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores, y propiciando un buen clima de trabajo. XP se basa en realimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios. XP se define como especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes, y donde existe un alto riesgo técnico. (Letelier, 2003)

A. Ciclo de Vida de la Metodología XP (Extreme Programmig)

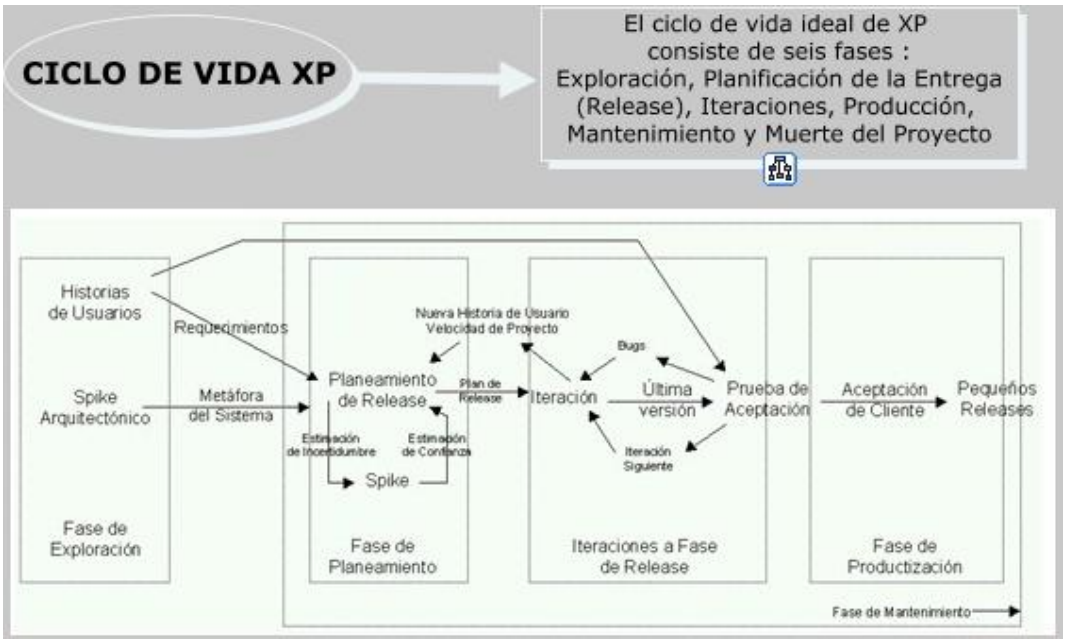


Figura 2. Ciclo de Vida de la Metodología XP (Extreme Programming)
Fuente: (http://ingsoftware072301.es/metodologia-xp-2012877,2000)

- **Fase de Exploración.-** En esta fase, los clientes plantean a grandes rasgos las historias de usuario que son de interés para la primera entrega del producto. Al mismo tiempo el equipo de desarrollo se familiariza con las herramientas, tecnologías y prácticas que se utilizarán en el proyecto.
- **Historias de Usuarios.-** Sustituyen a los documentos de especificación funcional, y a los “casos de uso”. Estas “historias” son escritas por el cliente, en su propio lenguaje, como descripciones cortas de lo que el sistema debe realizar. La diferencia más importante entre estas historias y los tradicionales documentos de especificación funcional se encuentra en el nivel de detalle requerido. Las historias de usuario deben tener el detalle mínimo como para que los programadores puedan realizar una estimación poco riesgosa del tiempo que llevará su desarrollo. Cuando llegue el momento de la implementación, los desarrolladores dialogarán directamente con el cliente para obtener todos los detalles necesarios. Las historias de usuarios deben poder ser programadas en un tiempo entre una y tres semanas. Si la estimación es superior a tres semanas, debe ser dividida en dos o más historias. Si es menos de una semana, se debe combinar con otra historia. (Joskowicz , 2008)
- **Fase de Planeamiento.-** Los programadores consideran el esfuerzo que requiere cada historia y a partir de allí se define el cronograma para realizar las entregas.
- **Fase de Producción.-** Requiere prueba y comprobación extra del funcionamiento del sistema antes de que éste pueda liberarse al cliente.
- **Fase de Mantenimiento.-** Requiere de un mayor esfuerzo para satisfacer las tareas del cliente. Así la velocidad del desarrollo puede desacelerar después de que el sistema se encuentre en producción. La fase de mantenimiento puede requerir la incorporación de nueva gente y cambiar la estructura del equipo.

- **Fase de Muerte.-** Es cuando el cliente no tiene más historias para ser incluidas en el sistema. Esto requiere que se satisfagan las necesidades del cliente en otros aspectos como rendimiento y confiabilidad del sistema; se genera la documentación final del sistema y no se realizan más cambios en la arquitectura. (Enríquez Fuel & Ayala Rosero , 2011)

B. Roles XP

Los roles de acuerdo con la propuesta original de Beck son:

- **Programador.** El programador escribe las pruebas unitarias y produce el código del sistema.
- **Cliente.** Escribe las historias de usuario y las pruebas funcionales para validar su implementación. Además, asigna la prioridad a las historias de usuario y decide cuáles se implementan en cada iteración centrándose en aportar mayor valor al negocio.
- **Encargado de Pruebas (Tester).** Ayuda al cliente a escribir las pruebas funcionales. Ejecuta las pruebas regularmente, difunde los resultados en el equipo y es responsable de las herramientas de soporte para pruebas.
- **Encargado de Seguimiento (Tracker).** Proporciona realimentación al equipo. Verifica el grado de acierto entre las estimaciones realizadas y el tiempo real dedicado, para mejorar futuras estimaciones. Realiza el seguimiento del progreso de cada iteración.
- **Entrenador (Coach).** Es responsable del proceso global. Debe proveer guías al equipo de forma que se apliquen las prácticas XP y se siga el proceso correctamente.
- **Consultor.** Es un miembro externo del equipo con un conocimiento específico en algún tema necesario para el proyecto, en el que puedan surgir problemas.

- **Gestor (Big boss).** Es el vínculo entre clientes y programadores, ayuda a que el equipo trabaje efectivamente creando las condiciones adecuadas. Su labor esencial es de coordinación.

C. Proceso XP

El ciclo de desarrollo consiste (a grandes rasgos) en los siguientes pasos:

1. El cliente define el valor de negocio a implementar.
2. El programador estima el esfuerzo necesario para su implementación.
3. El cliente selecciona qué construir, de acuerdo con sus prioridades y las restricciones de tiempo.
4. El programador construye ese valor de negocio.
5. Vuelve al paso (1).

En todas las iteraciones de este ciclo tanto el cliente como el programador aprenden. No se debe presionar al programador a realizar más trabajo que el estimado, se perderá calidad en el software o no se cumplirán los plazos. De la misma forma el cliente tiene la obligación de manejar el ámbito de entrega del producto, para asegurarse que el sistema tenga el mayor valor de negocio posible con cada iteración. (Torres Letelier & Sánchez López, 2003)

2.3. Programación Orientada a Objetos POO

- **Concepto.-** La programación orientada a objetos establece un equilibrio entre la importancia de los procesos y los datos, mostrando un enfoque más cercano al pensamiento del ser humano. Se introduce un aspecto novedoso respecto al anterior paradigma: la herencia, facilitando el crecimiento y la mantenibilidad.
- **Objeto.-** Es la abstracción de una entidad real como por ejemplo vehículos, animales, personas, etc, un objeto se compones

básicamente de dos elementos de atributos (variables) y métodos (función).

2.3.1. Características de la Programación Orientada a Objetos:

- **La Abstracción.-** Cada vez que se pronuncia una palabra, realmente lo que hace es asociar ese sonido (o ese conjunto de garabatos al escribir) con una serie de cosas. Decir que un ave es tal cosa, que una silla es tal otra, etc.

Cuando se aplica la POO, lo primero que realiza es cumplir con una vieja máxima de guerra: Divide y Vencerás. Es decir, lo que hace es seccionar nuestro código en grupos de código más pequeño que, al unirlos, hacen el trabajo. Un buen ejemplo de abstracción es el cuerpo humano, aunque el cuerpo es una unidad, está dividido por sistemas (el sistema respiratorio, el sistema linfático, cardiovascular). Estos sistemas, a su vez están compuestos por otros más pequeños: los órganos, y así sucesivamente. La abstracción permite dividir el programa en distintos objetos que se agrupan para formar cosas más complejas.

Básicamente es la capacidad de separar los elementos (al menos mentalmente) para poder verlos de forma singular. Como cuando se describe el cuerpo humano por ejemplo cabeza, brazo(s), pierna(s), etc.

También conocida como ocultamiento. Cuando se resuelve ver la televisión no interesa del modo como éste funciona, o lo que hace para cambiar de canal o aumentar el volumen. A menos que seas experto en electrónica o técnico en televisores, te pasará lo mismo: no lo sabes y no te importa; sólo sabes que al presionar un botón ocurre la magia.

- **La Encapsulación.-** Encargada de mantener ocultos los procesos internos que necesita para hacer lo que sea que haga, dándole al

programador acceso sólo a lo que necesita. Esto da dos ventajas iniciales: Lo que hace el usuario puede ser controlado internamente (incluso sus errores), evitando que todo colapse por una intervención indeseada (tú no quieres que tu mamá, que no tiene ni idea de electrónica, abra tu televisor y empiece a jugar con los circuitos para cambiar los canales manualmente ¿verdad?). La segunda ventaja es que, al hacer que la mayor parte del código esté oculto, se puede hacer cambios y/o mejoras sin que eso afecte el modo como los usuarios van a utilizar tu código. Sólo tienes que mantener igual la forma de acceder a él (en el caso del control de la tele, que los botones sigan siendo los mismos y que el botón de “apagado” no cambie el volumen). Por cierto, estas puertas de acceso que das a los usuarios son lo que se conoce como interfaz.

- **La Herencia.-** Es la capacidad que tiene una clase de derivar las propiedades y métodos de otra, un ejemplo: una gallina es un ave; esto quiere decir que las gallinas tienen características comunes con otras aves (pico, plumas, etc.), es decir que la gallina hereda las características comunes de todas las aves. Pero además, resulta que un ave es un animal, lo que significa que también comparte características comunes al caballo, el perro, el hombre (somos animales) y cualquier otra cosa que pueda ser clasificada como animal. La herencia permite, entre otras cosas, evitar tener que escribir el mismo código una y otra vez, puesto que al definir que una categoría (que en programación llamaremos clase) pertenece a otra, automáticamente atribuye las características generales de la primera, sin tener que definirlas de nuevo. (Frick, 2007)

- **La Modularidad.-** Es descomponer un sistema en un conjunto de partes. Aparecen dos conceptos muy importantes: acoplamiento y cohesión.

- El acoplamiento entre dos módulos mide el nivel de asociación entre ellos; nos interesa buscar módulos poco acoplados.
- La cohesión de un módulo mide el grado de conectividad entre los elementos que los forman; interesa buscar una cohesión alta.
- La Jerarquía es un proceso de estructuración de varios elementos por niveles. La programación orientada a objetos implementa estos cuatro conceptos con los siguientes elementos: clases y objetos, atributos y estado, métodos y mensajes, herencia y polimorfismo.

- **El Polimorfismo.-** Capacidad de que un mismo mensaje funcione con diferentes objetos. Es aquél en el que el código no incluye ningún tipo de especificación sobre el tipo concreto de objetos sobre el que se trabaja. El método opera sobre un conjunto de posibles objetos compatibles. (Bernal Bermúdez , 2012, págs. 7,8)

2.4. Lenguaje Orientado a Objetos

Mediante la evolución de la tecnología actualmente existen gran cantidad de lenguajes orientados que entre ellos los más destacados en el mercado son:

Python, VB.NET, Delphi, Eiffel, Java, PHP, PowerBuilder, Python, Ruby y Smalltalk.

Sobre saliendo entre la lista Java, lenguaje de alto nivel que cuenta con características como: (lenguaje orientado a objetos, sencillo, Independiente de plataforma, brinda un gran nivel de seguridad, capacidad multihilo, gran rendimiento, creación de aplicaciones distribuidas), todos estas especialidades crean un ambiente amigable para el desarrollador.

El lenguaje Java fue creado por Sun Microsystems, Aparece en el año 1995 y creó en su origen para que fuese un lenguaje multiplataforma. Para ello se compila en un código intermedio: bytecode y necesita de una máquina virtual que lo ejecute. Normalmente, no utiliza código nativo, es decir, no se puede ejecutar directamente por el procesador. Se disponen de varias plataformas Java para el desarrollo.

Para el desarrollo y compilación de aplicaciones Java, utilizo: Standard Edition (Java SE) o Java Development Kit (JDK) de Sun:

Se trabaja mediante comandos en consola. Incluye, entre otras, las siguientes utilidades:

- El compilador: javac.exe. Un ejemplo: javac Fich.java.
- El intérprete: java.exe. Ej. java Fich.
- Un compresor: jar.exe. Ej. jar –cvf fich.jar Uno.class Dos.class.
- El generador de documentación: javadoc.exe. Ej. javadoc *.java.
- Un analizador de clases: javap.exe. Ej. javap Fich

Para la ejecución de aplicaciones Java (Delivery Platforms) es el Java Runtime Environment (JRE). Se instala automáticamente con Java SE.

Java se ejecuta en la mayoría de sistemas operativos como: Windows 8, Windows 7, Vista, Windows XP, Macintosh OS X, Windows 10.

Entonces JAVA es un lenguaje de programación y una plataforma informática que utiliza un compilador, para traducir del código fuente al código ejecutable, teniendo característica de ser un lenguaje interpretado es decir el bytecode pasa a por lenguajes interpretados como son: Perl, Gambas, Php, Python, Java.

2.4.1. JRE

Es un conjunto de utilidades que permite la ejecución de programas java, En su forma más complicada, el entorno en tiempo de ejecución de Java está conformado por una Máquina Virtual de Java o JVM, un conjunto de Java y otros componentes innecesarios para que una aplicación escrita en lenguaje c++ pueda ser ejecutada. El JRE actúa como un intermediario entre el sistema y Java.

La JVM es el programa que interpreta el código Java mientras que las librerías de clases estándar son las que implementan el API de Java. Ambas JVM y API deben ser consistentes entre sí, de ahí que sean distribuidas de modo conjunto.

Un usuario sólo necesita el JRE para ejecutar las aplicaciones desarrolladas en lenguaje Java, mientras que para desarrollar nuevas aplicaciones en dicho lenguaje es necesario un entorno de desarrollo, denominado JDK, que además del JRE (mínimo imprescindible) incluye, entre otros, un compilador para Java. (Latorre, 2010)

2.4.2. Java Development Kit (JDK)

Se puede definir como un conjunto de herramientas, utilidades, documentación y ejemplos para desarrollar aplicaciones Java. JDK consta de una serie de aplicaciones y componentes, para realizar cada una de las tareas de las que es capaz de encargarse.

Todos los programas permiten la inclusión de una serie de opciones sobre su ejecución antes del primer argumento. Estas opciones se indican precedidas de un menos (-) Todas las opciones que los ejecutables del JDK presentan se muestran llamando al programa sin parámetros o con las opciones -? o -help: (Olivares, 2010)

2.5. Entorno de Desarrollo Java

Los entornos de desarrollo Java son aplicaciones que permiten al programador implementar las abstracciones del mundo real en un aplicación concreta mediante la introducción de secuencias de código con sus estructuras de programación.

Actualmente existen dos entornos de desarrollo Java de amplio uso en el sector profesional. Este entorno de desarrollo Java que son NetBeans, Eclipse, para niños y adolescentes (Scratch, Alice, Greenfoot, BlueJ) para móviles (AppInventor, AndroidStudio) estos son los más utilizados en el mundo.

2.5.1. NetBeans

Se trata de unos de los mejores entornos de desarrollo Java. Es libre y su uso principal es el desarrollo de aplicaciones Java, aunque también permite el desarrollo de aplicaciones en otros lenguajes de programación tales php, html.

Se trata de un producto gratuito y libre sin restricciones de uso. Además ofrece la posibilidad de ampliación, existen programadores que desarrollan nuevos módulos que se pueden añadir a dicho entorno.

Figura 3. Logo de Netbeans
Fuente: (https://netbeans.org/, 2016)

2.5.2. Eclipse

Se trata de una aplicación de programación multiplataforma que permite el desarrollo de aplicaciones para Android que también utilizan el lenguaje

Java para su implementación, existen otros entornos de desarrollo Java (JBuilder, JCreator) pero dado que se trata de software comercial tienen una menor difusión que los aquí tratados. (Alonso, 2014)

Figura 4. Logo de Eclipse
Fuente: (https://eclipse.org/, 2016)

2.6. ¿Qué es un Sistema Web?

Los “sistemas Web” o también conocido como “aplicaciones Web” son aquellos que están creados e instalados no sobre una plataforma o sistemas operativos (Windows, Linux). Sino que se alojan en un servidor en Internet o sobre una intranet (red local). Su aspecto es muy similar a páginas Web que se observa normalmente, pero en realidad los ‘sistemas Web’ tienen funcionalidades muy potentes que brindan respuestas a casos particulares.

Los sistemas Web se pueden utilizar en cualquier navegador Web (chrome, firefox, Internet Explorer) sin importar el sistema operativo. Para utilizar las aplicaciones Web no es necesario instalarlas en cada computadora, los usuarios se conectan a un servidor donde se aloja el sistema.

Las aplicaciones Web trabajan con bases de datos que permiten procesar y mostrar información de forma dinámica para el usuario.

Los sistemas desarrollados en plataformas Web, tienen marcadas diferencias con otros tipos de sistemas, lo que lo hacen muy beneficioso tanto para las empresas que lo utilizan, como para los usuarios que operan en el sistema.

- Un ejemplo claro de un sistema es un panel de administración, con él se podrán modificar y actualizar diferentes contenidos dentro de la página sin necesidad de llamar a tu proveedor para que lo haga.
- Este tipo de diferencias se ven reflejada en los costos, en la rapidez de obtención de la información, en la optimización de las tareas por parte de los usuarios y en alcanzar una gestión estable. (Gonzales, fraktalweb, 2015)

2.7. Hosting o Servidor WEB

El alojamiento web (en inglés web hosting) es el servicio que provee a los usuarios de Internet un sistema para poder almacenar información, imágenes, vídeo, o cualquier contenido accesible vía web. Es una analogía de "hospedaje o alojamiento en hoteles o habitaciones" donde uno ocupa un lugar específico, en este caso la analogía alojamiento web o alojamiento de páginas web, se refiere al lugar que ocupa una página web, sitio web, sistema, correo electrónico, archivos etc. en internet o más específicamente en un servidor que por lo general hospeda varias aplicaciones o páginas web.

Las compañías que proporcionan espacio de un servidor a sus clientes se suelen denominar con el término en inglés web host ó Hosting.

Se puede definir como "un lugar para tu página web o correos electrónicos", aunque esta definición simplifica de manera conceptual el hecho de que el alojamiento web es en realidad espacio en Internet para prácticamente cualquier tipo de información, sea archivos, sistemas, correos electrónicos, videos etc. (Aprigliano, 1999)

2.8. Motor de Bases de Datos

2.8.1. Introducción

Los Motores de Bases de Datos (MBD) son programas específicos, dedicados a servir de intermediarios entre las bases de datos y las aplicaciones que las utilizan como función principal, cada uno de estos cumple una tarea específica, que van desde crear la Base de Datos hasta administrar el uso y acceso a esta.

Un MBD está compuesto principalmente por tres lenguajes, siendo estos DDL (Lenguaje de definición de datos), DML (Lenguaje de manipulación de datos), y finalmente el SQL (Lenguaje de consulta).

Teniendo en cuenta la gran cantidad de información que se almacena en una Base de Datos y que cada vez son más las aplicaciones y personas que requieren de su uso, los Motores de Bases de Datos, nacen como alternativa para optimizar el procedimiento de acceso, consulta y extracción o inyección de información de las Bases de Datos, creando así un entorno más sencillo, agradable y eficaz a la hora de Utilizarlas.

Un Motor de Bases de Datos debe cumplir con los siguientes parámetros:

- Abstracción de la información.
- Independencia.
- Redundancia mínima.
- Consistencia
- Seguridad.
- Integridad.
- Respaldo y recuperación.
- Control de la concurrencia.
- Tiempo corto de respuesta (Fortinet, 2016)

2.8.2. Definición de Bases de datos Oracle

Una Base de Datos de Oracle es una colección de datos tratados como una unidad. El propósito de una base de datos es para almacenar y recuperar información relacionada. Un servidor de base de datos es la clave para resolver los problemas de gestión de la información. En general, un servidor gestiona fiable una gran cantidad de datos en un entorno multiusuario para que muchos usuarios pueden acceder simultáneamente a los mismos datos. Todo esto se realiza al tiempo que ofrece un alto rendimiento. Un servidor de base de datos también impide el acceso no autorizado y ofrece soluciones eficientes para la recuperación de errores. (Oracle, 2007)

2.9. Conexiones ODBC

Conectividad de base de datos abierta (Open Database Connectivity, ODBC) es una interfaz de programación de aplicaciones (API) de estándar abierto para acceder a una base de datos. Mediante el uso de sentencias de ODBC en un programa, usted puede acceder a los archivos de diferentes bases de datos, incluyendo Access, dBase, DB2, Excel y Text. Además del software de ODBC, se necesita un módulo o controlador independiente para acceder a cada base de datos. El principal promotor y proveedor del soporte de programación ODBC es Microsoft.

ODBC se basa y está estrechamente alineado con el estándar Interfaz de Nivel de Llamada en Lenguaje de Consultas Estructuradas (Structured Query Language Call-Level Interface) de The Open Group. Permite a los programas utilizar peticiones SQL que tendrán acceso a bases de datos sin necesidad de conocer las interfaces propietarias hacia las bases de datos. ODBC controla la solicitud SQL y la convierte en una solicitud que el sistema de base de datos individual entiende. (Rouse, 2015).

2.10. Especificación De Requisitos De Software (ERS)

2.10.1. Introducción

El análisis de requisitos es una de las tareas más importantes en el ciclo de vida del desarrollo de software, puesto que en ella se determinan los planos de la nueva aplicación.

En cualquier proyecto software los requisitos son las necesidades del producto que se debe desarrollar. Por ello, en la fase de análisis de requisitos se deben identificar claramente estas necesidades y documentarlas. Como resultado de esta fase se debe producir un documento de especificación de requisitos en el que se describa lo que el futuro sistema debe hacer. Por tanto, no se trata simplemente de una actividad de análisis, sino también de síntesis.

El análisis de requisitos se puede definir como el proceso del estudio de las necesidades de los usuarios para llegar a una definición de los requisitos del sistema, hardware o software, así como el proceso de estudio y refinamiento de dichos requisitos, definición proporcionada por el IEEE [Piattini, 1996]. Asimismo, se define requisito como una condición o capacidad que necesita el usuario para resolver un problema o conseguir un objetivo determinado. Esta definición se extiende y se aplica a las condiciones que debe cumplir o poseer un sistema o uno de sus componentes para satisfacer un contrato, una norma o una especificación.

En la determinación de los requisitos no sólo deben actuar los analistas, es muy importante la participación de los propios usuarios, porque son éstos los que mejor conocen el sistema que se va a automatizar. Analista y cliente se deben poner de acuerdo en las necesidades del nuevo sistema, el cliente no suele entender el proceso de diseño y desarrollo del software como para redactar una especificación de requisitos software (ERS) y los analistas no

suelen entender completamente el problema del cliente, debido a que no dominan su área de trabajo.

Así pues, el documento de especificación de requisitos debe ser legible por el cliente, con lo que se evita el malentendido de determinadas situaciones, el cliente participa activamente en la extracción de dichos requisitos. (Piattini Velthuis, 1996)

2.10.2. Objetivos de la ERS.

Los principales objetivos que se identifican en la especificación de requisitos software son:

1. Ayudar a los clientes a describir claramente lo que se desea obtener mediante un determinado software: El cliente debe participar activamente en la especificación de requisitos, éste tiene una visión mucho más detallada de los procesos que se llevan a cabo. Asimismo, el cliente se siente partícipe del propio desarrollo.
2. Ayudar a los desarrolladores a entender qué quiere exactamente el cliente: En muchas ocasiones el cliente no sabe exactamente qué es lo que quiere. La ERS permite al cliente definir todos los requisitos que desea y al mismo tiempo los desarrolladores tienen una base fija en la que trabajar. Si no se realiza una buena especificación de requisitos, el costo de desarrollo puede tener un aumento considerable, se deben hacer cambios durante la creación de la aplicación.
3. Servir de base para desarrollos de estándares de ERS particulares para cada organización: Cada entidad puede desarrollar sus propios estándares para definir sus necesidades.

Una buena especificación de requisitos software ofrece una serie de ventajas entre las que destacan el contrato entre cliente y desarrolladores (como ya se ha indicado con anterioridad), la reducción del esfuerzo en el

desarrollo, una buena base para la estimación decostes y planificación, un punto de referencia para procesos de verificación y validación, y una base para la identificación de posibles mejoras en los procesos analizados.

La ERS es una descripción que debe decir ciertas cosas y al mismo tiempo debe decirlas de una determinada manera. En este documento se presentará una de las formas que viene especificada por el estándar IEEE 830.

Una ERS forma parte de la documentación asociada al software que se está desarrollando, por tanto debe definir correctamente todos los requerimientos, pero no más de los necesarios. Esta documentación no debería describir ningún detalle de diseño, modo de implementación o gestión del proyecto, los requisitos se deben describir de forma que el usuario pueda entenderlos. Al mismo tiempo, se da una mayor flexibilidad a los desarrolladores para la implementación. (Monferrer Agut, 2001)

2.10.3. Características.

Las características deseables para una buena ERS en el IEEE son las siguientes:

- **Corrección.-** La ERS es correcta si y sólo si todo requisito que figura en ella refleja alguna necesidad real. La corrección de la ERS implica que el sistema implementado será el sistema deseado.
- **Ambigüedad.-** Cada característica del producto final debe ser descrita utilizando un término único y, en caso de que se utilicen términos similares en distintos contextos, se deben indicar claramente las diferencias entre ellos. Incluso se puede incluir un glosario en el que indicar cada significado específicamente.

Los analistas deben poner un cuidado especial a la hora de especificar los requisitos. El hecho de utilizar el lenguaje natural para hacer la ERS

comprensible a los usuarios supone un riesgo muy elevado, porque el lenguaje natural puede llegar a ser muy ambiguo.

En términos generales, el lenguaje natural es de los más ambiguos. Por el contrario existen los lenguajes formales que no son ambiguos, pero son más difíciles de aprender y menos comprensibles para el que no los conoce.

- **Completitud.-** Incluye todos los requisitos significativos del software (relacionados con la funcionalidad, ejecución, diseño, atributos de calidad o interfaces externas).

Existe una definición de respuestas a todas las posibles entradas, tanto válidas como inválidas, en todas las posibles situaciones.

Aparecen etiquetadas todas las figuras, tablas, diagramas, así como definidos todos los términos y unidades de medida empleados.

- **Verificabilidad.-** Un requisito es verificable si existe algún proceso no excesivamente costoso por el cual una persona o una máquina puedan verificar que el software satisface dicho requerimiento.

- **Consistencia.-** Una ERS es consistente si y sólo si ningún conjunto de requisitos descritos en ella son contradictorios o entran en conflicto. Se pueden dar tres casos:
    - Requisitos que describen el mismo objeto real utilizando distintos términos.
    - Las características especificadas de objetos reales. Un requisito establece que todas las luces son verdes y otro que son azules.
    - Conflicto lógico o temporal entre dos acciones determinadas. Se llega a un punto en el que dos acciones serían perfectamente válidas.

- **Clasificación.-** No todos los requisitos son igual de importantes. Los requisitos pueden clasificarse por diversos criterios:

    - Importancia: Pueden ser esenciales, condicionales u opcionales.
    - Estabilidad: Cambios que pueden afectar al requisito.

Lo ideal es el establecimiento de prioridades, de modo que la implementación de un requisito de menor prioridad no emplee excesivos recursos.

- **Modificabilidad.-** Una ERS es modificable si cualquier cambio puede realizarse de manera fácil, completa y consistente. Para ello, es deseable tener una organización coherente y fácil de usar en la que aparezca el índice o una tabla de contenidos fácilmente accesible.
- **Explorabilidad (Traceability).-** Una ERS es explorable si el origen de cada requerimiento es claro tanto hacia atrás (origen que puede ser un documento, una persona.) como hacia delante (componentes del sistema que realizan dicho requisito).
  Utilizable Durante Las Tareas De Mantenimiento Y Uso.

En la ERS también se deben tener en cuenta las necesidades de mantenimiento. El personal que no ha intervenido directamente en el desarrollo debe ser capaz de encargarse de su mantenimiento. Así, dicha ERS actúa a modo de plano de la aplicación, permitiendo incluso modificaciones que no requieran un cambio en el diseño. (Piattini Velthuis, 1996).

2.10.4. Lenguaje Unificado de Modelado (UML)

UML es ante todo un lenguaje. Un lenguaje proporciona un vocabulario y unas reglas para permitir una comunicación. En este caso, este lenguaje se centra en la representación gráfica de un sistema.

Este lenguaje indica cómo crear y leer los modelos, pero no dice cómo crearlos. Esto último es el objetivo de las metodologías de desarrollo.

Los objetivos de UML son muchos, pero se pueden sintetizar sus funciones:

- Visualizar: UML permite expresar de una forma gráfica un sistema de forma que otro lo puede entender.
- Especificar: UML permite especificar cuáles son las características de un sistema antes de su construcción.
- Construir: A partir de los modelos especifica.
- Diagrama de colaboración. dos se pueden construir los sistemas diseñados.
- Diagrama de estados.
- Documentar: Los propios elementos gráficos sirven como documentación del sistema desarrollado que pueden servir para su futura revisión.

A. Diagramas UML

Un diagrama es la representación gráfica de un conjunto de elementos con sus relaciones. En concreto, un diagrama ofrece una vista del sistema a modelar. Para poder representar correctamente un sistema, UML ofrece una amplia variedad de diagramas para visualizar el sistema desde varias perspectivas. UML incluye los siguientes diagramas:

- Diagrama de casos de uso.
- Diagrama de clases.
- Diagrama de objetos.
- Diagrama de secuencia.
- Diagrama de colaboración.
- Diagrama de estados.

- Diagrama de actividades.
- Diagrama de componentes.
- Diagrama de despliegue.

Los diagramas más interesantes (y los más usados) son los de casos de uso, clases y secuencia, por lo que nos centraremos en éstos. Pare ello, se utilizará ejemplos de un sistema de venta de entradas de cine por Internet.

Diagrama de Casos de Usos.- Representa gráficamente los casos de uso que tiene un sistema. Se define un caso de uso como cada interacción supuesta con el sistema a desarrollar, donde se representan los requisitos funcionales. Es decir, se está diciendo lo que tiene que hacer un sistema y cómo. En la figura 5 se muestra un ejemplo de casos de uso, donde se muestran tres actores (los clientes, los taquilleros y los jefes de taquilla) y las operaciones que pueden realizar (sus roles).

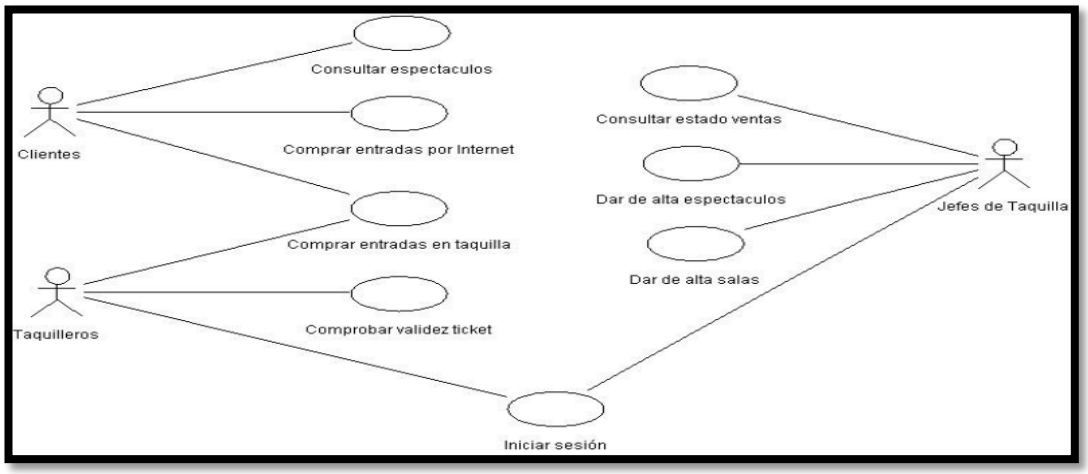


Figura 5. Diagrama de Caso de Uso
Fuente: (Hernández Orallo, Enrique, 2010)

Diagrama de Clases.- Muestra un conjunto de clases, interfaces y sus relaciones. Éste es el diagrama más común a la hora de describir el diseño de los sistemas orientados a objetos. En la figura 6 se muestran las clases globales, sus atributos y las relaciones de una posible solución al problema de la venta de entradas.

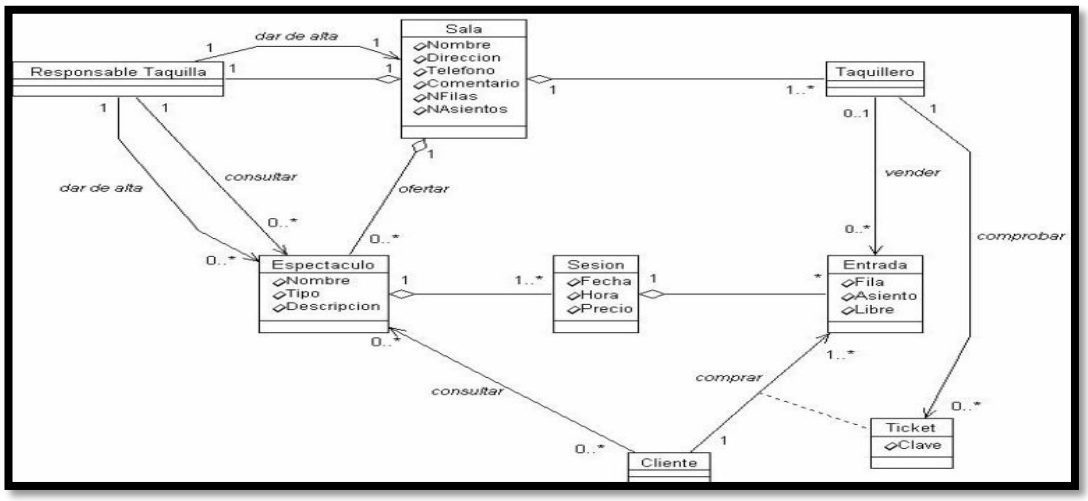


Figura 6. Diagrama de Clases

Fuente: (Hernández Orallo, Enrique, 2010)

Diagrama de Secuencia.- Muestra la interacción de los objetos que componen un sistema de forma temporal. Siguiendo el ejemplo de venta de entradas, la figura 7 muestra la interacción de crear una nueva sala para un espectáculo. (Hernández Orallo, 2010)

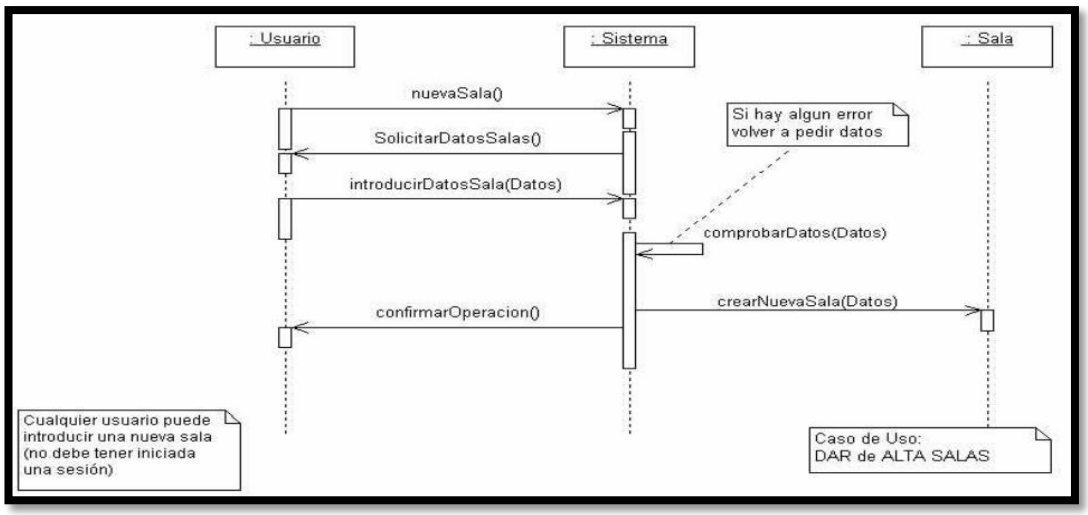


Figura 7. Diagrama de Secuencia

Fuente: (Hernández Orallo, Enrique, 2010)

2.11. Estándar para Modelar Base de Datos

Tablas.- Los nombres de estos objetos en su primera letra deben especificar el sistema que se está automatizando (“F”, “E”, “L”), de ser necesario la segunda letra identificará el módulo dentro del sistema, los dos o tres siguientes caracteres contendrán las siglas del nombre de la tabla (hasta aquí deben completarse 4 caracteres) seguido de un guión bajo (underscore) y luego un texto que identifique a la tabla. El nombre completo de la tabla no debe sobrepasar los 15 caracteres.

En la tabla Plan Anual de Cursos EPLC_PLANCURSO, la letra E representa el sistema de Educación, PLC son las siglas de la tabla y PLANCURSO es un texto que identifica a la tabla.

En la tabla de Misiones CIMI_MISION, la letra C representa el sistema de Comando de Operaciones Terrestres, I representa el Módulo de Defensa Interna, MI son las siglas de la tabla y MISION es el texto que identifica a la tabla.

Campos.- Serán nombrados basándose en las 3 siglas del nombre de la tabla, seguido de un underscore y un texto que identifique a la columna. El nombre completo de la columna no debe sobrepasar los 15 caracteres. El campo secuencia de la tabla de plan de curso sería nombrado del siguiente modo PLC_SECUENCIA.

Tablas Temporales.**-** Los nombres de estos objetos en su primera letra deben especificar el sistema que se está automatizando (“F”, “E”, “L”), luego los caracteres “TMP” que significan que la tabla es temporal, seguido de un guión bajo (underscore) y luego un texto que identifique a la tabla. El nombre completo de la tabla no debe sobrepasar los 15 caracteres. En la tabla temporal depreciación FTMP_DEPRECIACION, la letra F representa la dirección de Finanzas, TMP significa que es una tabla temporal y DEPRECIACION es el texto que identifica a la tabla.

Las tablas temporales que requieran crearse deben ser temporales propias de Oracle. No usar tablas permanentes para procesos temporales.

Vistas.- En lo posible no se utilizarán vistas, pero en el caso de requerirse, su nombre empezará con el prefijo “V_”, seguido de un texto que identifique a la tabla o a la consulta. El nombre completo de la vista no debe sobrepasar los 15 caracteres. Una vista de inventarios de logística se debería nombrar de la siguiente manera V_INVENTARIO.

Sinónimos.- Los sinónimos que se creen para los objetos de la base de datos tendrán el mismo nombre de los objetos originales. Solo se utilizará sinónimos para los objetos de los que no sean propietarios. Estos se utilizarán para las interfaces de las diferentes aplicaciones del SIFTE y en los esquemas auxiliares que se creen para acceso de los usuarios finales y para la administración de seguridades de cada una de las aplicaciones. El sinónimo para la tabla del Personal Militar PMEM_MILIT que debe ser creada en todos los esquemas que necesiten su acceso debe ser PMEM_MILIT.

Funciones.- Se nombraran colocando el prefijo “F_” seguido de un texto que identifique a la función. El nombre completo de la función no debe sobrepasar los 15 caracteres. Una función que realice la suma de existencias se llamaría F_SUMA_EXIST.

Procedimientos Almacenados.- Tendrá como prefijo “SP_” seguido de un texto que identifique al procedimiento. El nombre completo del procedimiento no debe sobrepasar los 15 caracteres. Un procedimiento almacenado que realice el proceso de amortización se llamaría SP_AMORTIZA.

Paquetes.- Tendrá como prefijo “PACK_” seguido de un texto que identifique al paquete. El nombre completo del paquete no debe sobrepasar los 15 caracteres. Ejemplo: PACK_MATER_GUERRA.

Triggers.- Se debe colocar el prefijo “TR_” seguido de un texto que identifique al trigger. El nombre completo del trigger no debe sobrepasar los 15 caracteres. Ejemplo: TR_SUMA_SALDO.

Vistas Materializadas.- Se crearán con el prefijo “MV_” seguido de un texto que identifique a la vista materializada. El nombre completo de la vista materializada no debe sobrepasar los 15 caracteres. Ejemplo: MV_HISTOR_ROL.

Secuencias.- Tendrán como prefijo “SEQ_” seguido de un texto que identifique a la secuencia. El nombre completo de la secuencia no debe sobrepasar los 15 caracteres. Ejemplo: SEQ_INVENTARIO.

Claves Foráneas (Relaciones).- Se deben nombrar colocando el prefijo “FK_”, a continuación los 8 caracteres iniciales de la tabla hija, seguido de un underscore y los 8 caracteres iniciales de la tabla padre. El nombre completo de la clave foránea no debe sobrepasar los 20 caracteres. Por ejemplo si se tiene la tabla de Participantes en una Misión CIPA_PARTICIPANTE y la tabla de Misiones CIMI_MISION la relación se nombrará de la siguiente manera FK_CIPA_PAR_CIMI_MIS.

Claves Primaria.- Se deben nombrar colocando el prefijo “PK_” seguido del nombre de la tabla. El nombre completo de la clave primaria no debe sobrepasar los 15 caracteres. Ejemplo: la tabla “PMEM_MILIT” tiene una clave primaria que se llama PK_ PMEM_MILIT.

Checks.- Se deben nombrar colocando el prefijo “CHK_” seguido de los 4 primeros caracteres del nombre de la tabla, luego un undescore seguido de

los ocho primeros caracteres del nombre del campo que tiene la restricción. El nombre total del CHECK no debe sobrepasar los 20 caracteres. Ejemplo: la tabla “PMEM_MILIT” tiene un check que se llama CHK_PMEM_MEM_VIV.

Indices.- Se deben nombrar colocando el prefijo “INDX_” seguido de los cuatro primeros caracteres del nombre de la tabla y a continuación los campos por los que se está indexando en forma abreviada. El nombre total del índice no debe sobrepasar los 25 caracteres Ejemplo: la tabla “SHRL_HISRO” tiene un índice que se llama INDX_SHRL_A_M_IES_TIE_CED.

Nota: Los índices que se generan para las PK (claves primarias) y FK (claves foráneas) mantendrán el siguiente estándar:

Los índices que se generan para las PK deben nombrarse de la siguiente manera: “PK_” seguido del nombre de la tabla. El nombre completo del índice de la clave primaria no debe sobrepasar los 15 caracteres. Ejemplo: El índice generado para la clave primaria de la tabla PMEM_MILIT se llama PK_ PMEM_MILIT.

Los índices que se generan para las FK deben nombrarse de la siguiente manera: los 8 primeros caracteres de la tabla hija, seguido de un underscore, luego los 8 caracteres iniciales de la tabla padre, seguido de un underscore y las letras “FK”. El nombre completo del índice de la clave foránea no debe sobrepasar los 20 caracteres. Por ejemplo si se tiene la tabla de Participantes en una Misión CIPA_PARTICIPANTE y la tabla de Misiones CIMI_MISION la clave foránea será FK_CIPA_PAR_CIMI_MIS y el índice se llamará de la siguiente manera: CIPA_PAR_CIMI_MIS_FK

Polices.- Se deben nombrar colocando el prefijo “POLI_” seguido del nombre de la tabla sobre la cual se aplica la política. El nombre completo del pólice no debe sobrepasar los 15 caracteres. Ejemplo: la tabla “PMEM_MILIT” tiene un police que se llama POLI_PMEM_MILIT.

Directorios.- Se deben nombrar colocando tres letras que identifique el tipo de datos que almacenará el directorio “IMG_” seguido un texto que identifique al directorio. El nombre completo del directorio no debe sobrepasar los 15 caracteres. Ejemplo: el directorio que contiene las fotos se llama “IMG_FOTOS”. (Ejército)

2.12. Tipos de Mantenimiento

Los tipos de mantenimiento son: preventivo, correctivo y restaurativo. Constituyen acciones de mantenimiento que permiten conservar y preservar un artículo o equipo en condiciones normales de empleo.

La ejecución de los tipos de mantenimiento correctivo y restaurativo se da en función de la capacidad técnica del personal de mantenimiento y del equipamiento de que dispongan las unidades logísticas.

2.12.1. Mantenimiento Preventivo

Es el conjunto de acciones programadas y repetitivas que permitan verificar y mantener un estado o condición de funcionamiento dado a fin de evitar su desgaste prematuro.

El mantenimiento preventivo se efectúa para reducir la probabilidad de falla de los vehículos; se realiza cada cierto intervalo de tiempo, dependiendo de la marca de cada uno de ellos y a su kilometraje recorrido, o cuando se alcanza una condición, cuyos parámetros han sido fijados previamente.

A. Mantenimiento Preventivo Indicativo

Las tareas que se realizan son:

1. Observación e inspección visual.

2. Inspección básica.
3. Medición básica.
4. Prueba básica.
5. Verificación.

B. Mantenimiento Preventivo Conservativo

Agrupan las siguientes tareas de mantenimiento:

1. Limpieza.
2. Lavado.
3. Ajuste.
4. Inspección detallada.
5. Desmontaje.
6. Montaje.
7. Lubricación.
8. Engrase.
9. Relleno - cambio.
10. Reemplazo de elemento antes de la falla.
11. Mantenimiento correctivo básico.

C. Mantenimiento Preventivo Predictivo

Se trata de un conjunto de tareas de mantenimiento, cuya aplicación requiere de laboratorios especiales y equipos de prueba. El mantenimiento predictivo comprende la serie de acciones que se toman y las técnicas que se aplican con el objeto de detectar fallas y defectos de la maquinaria o equipo, a fin de evitar que dichos daños se agraven durante la operación.

Aquí se agrupan las siguientes labores de mantenimiento:

1. Examen.

2. Análisis.
3. Evaluación.
4. Investigación.
5. Peritaje.
6. Diagnóstico.

2.12.2. Mantenimiento Correctivo

Incluye el grupo de actividades que se ejecutan después de la aparición de un daño o falla, sobre el vehículo y que permitirán restablecerlo a su estado operativo. Consiste en la realización de trabajos ejecutados por el personal de mantenimiento especializado, a fin de dar solución a mencionadas fallas, se incluyen las siguientes tareas:

1. Extracción.
2. Corrección.
3. Reparación.
4. Intercambio directo
5. Sustitución.
6. Instalación.
7. Eliminación.
8. Reemplazo de elementos después de la falla que afecta a un sistema.

Como parte del mantenimiento correctivo se efectúa también el intercambio controlado de elementos (“canivalización”). Únicamente con la autorización respectiva, las unidades de mantenimiento podrán aprovechar las piezas o repuestos en buen estado de aquellos materiales o equipos que no se encuentran operables.

Algunos trabajos de tipo de mantenimiento correctivo son:

1. Rectificación de un disco de freno.
2. Reparación del silenciador de un sistema de escape.

3. Intercambio del motor de arranque de un camión Hino.

2.12.3. Mantenimiento Restaurativo

Si a pesar del trabajo de mantenimiento preventivo y correctivo que se haya ejecutado sobre el material o equipo, persiste la falla o daño, se deben ejecutar acciones que permitan remediar definitivamente la anomalía. A este conjunto de operaciones se lo conoce como mantenimiento restaurativo.

El mantenimiento restaurativo es efectuado por personal altamente especializado y es ejecutado prioritariamente por el COLOG N° 25; En el mantenimiento restaurativo se incluyen las siguientes tareas:

1. Modificaciones, actualizaciones.
2. “Overhaul” (reparación completa).
3. Revisión general (inspección mayor).
4. Inspección y reparación, si es necesario.

2.13. Modos de Mantenimiento

Se entiende por modos de mantenimiento las limitaciones dadas por los fabricantes a los artículos, materiales o equipos para que sean sujetos a una revisión (inspección) o para definir su período de vida útil.

De acuerdo con las necesidades institucionales y con un análisis técnico a través del COLOGE, se podrán modificar las limitaciones de los artículos, materiales o equipo de que dispone la Fuerza Terrestre, siempre y cuando cumplan con los estándares de seguridad.

Los modos de mantenimiento constituyen la base fundamental para la realización del tipo de mantenimiento preventivo. Existen tres modos de mantenimiento:

1. Por tiempo límite
2. Según estado o condición
3. Con vigilancia de comportamiento.

2.13.1. Mantenimiento por Tiempo Límite

Se dice que un artículo, material o equipo es objeto de un mantenimiento con tiempo límite (TL) cuando está limitado para realizar su revisión (inspección), o para colocarlo fuera de utilización por haber cumplido su vida útil. Las limitaciones pueden estar expresadas en kilometraje. El modo de mantenimiento por tiempo límite se divide en: tiempo límite de revisión (TLR) y tiempo límite de vida (TLV).

2.13.2. Tiempo Límite de Revisión (TLR)

Significa que el artículo, material o equipo debe sufrir intervenciones en un taller especializado, cuya finalidad es darle un nuevo período de servicio.

2.13.3. Tiempo Límite de Vida (TLV)

El artículo, material o equipo debe ser retirado del servicio al alcanzar el plazo indicado y se lo debe dar de baja. Los tiempos límites de vida son valores que se deben respetar imperativamente. Las unidades logísticas llevarán un control de los tiempos límites de cada uno de los artículos, materiales o equipos de que se dispongan.

2.13.4. Mantenimiento Según Estado o Condición

Se hace cuando el fabricante del artículo, material o equipo no impone períodos para realizar una intervención técnica ni tampoco describe su período de vida útil.

La forma de realizar el mantenimiento sobre este material es simplemente cumpliendo con los programas de mantenimiento preventivo establecidos por el usuario, y que están sujetos a verificación del estado o condición para determinar la ausencia de alteraciones. Los artículos, materiales o equipos se colocarán fuera de servicio en función del resultado del mantenimiento preventivo.

2.13.5. Mantenimiento con Vigilancia de Comportamiento

Es un modo de mantenimiento que solo se debe llevar a cabo en los artículos, materiales o equipos que presentan un daño o falla que aún no afecta a su operatividad.

Las actividades de mantenimiento preventivo permitirán vigilar el comportamiento del daño o falla del artículo, material o equipo; esto ayudará a determinar si pueden continuar en condiciones de uso.

En el siguiente gráfico se presenta la programación de mantenimiento preventivo de un vehículo administrativo, donde se especifican las periodicidades en kilometraje.

Tabla 1.

Periodicidades de Inspecciones de Vehículos

TIPOS DE INSPECCIONES							
COMPLEMENTARIA							PERIÓDICA
5.000	10.000	15.000	20.000	25.000	30.000	35.000	40.000

Las inspecciones que se realizan en este ejemplo tienen una periodicidad de 5000 km., y son complementarias; al llegar a los 40000 km., se cumplirá una inspección periódica de este recorrido; en este período culmina el ciclo de mantenimiento, es decir, la próxima inspección es una complementaria de

5000 km., con lo que empieza otra vez el ciclo y así sucesivamente hasta complementar otra inspección periódica.

2.14. Lineamientos Generales

En el proceso de mantenimiento y control, se debe considerar lo siguiente:

- El mejoramiento del proceso debe estar acorde a las normativas y lineamientos; así como a las normativas de calidad vigentes.
- Las actividades a desarrollarse deberán cumplir con los procedimientos, cartas de servicio y manuales del fabricante; y optimizar al máximo los recursos asignados, a fin de evitar los reprocesos que ocasionan pérdidas de tiempo y elevan los costos de mantenimiento.
- Este proceso, se debe basar en la programación de mantenimiento vigente del Centro de Transportes perteneciente al Batallón de Mantenimiento “QUISQUIS”.
- La ejecución del mantenimiento se debe realizar en base a los manuales, procedimientos, cartas de servicios, manuales del fabricante y formatos vigentes pertenecientes al Centro de Transportes (vehículos administrativos o tácticos).
- La ejecución del mantenimiento se debe efectuarse siempre bajo una orden de trabajo
- La supervisión y control de calidad se llevara a cabo al finalizar las fase de mantenimiento y al final del procedimiento en las pruebas o ensayos, aplicando los formularios vigentes
- El intercambio controlado de elementos se realizara solo para los vehículos tácticos
- El mantenimiento preventivo, correctivo y restaurativo se realizara en base al número de kilómetros y tiempo calendario estableciendo, el TLV. TLR y potenciales para la programación de mantenimiento.

- El control y supervisión se realizara en base a los parámetros establecidos en la carpeta del supervisor.
- La ejecución del mantenimiento se realizaran considerando las normas, plan de seguridad ocupacional o industrial, reglamentos, normativas, leyes y ordenanzas vigentes de seguridad industrial, nacionales e internacionales.
- Al finalizar el mantenimiento se deberá liquidar los materiales y repuestos
- Una vez finalizado el mantenimiento se deberá clasificar y etiquetar los desechos industriales.
- Se deberá observar y rediseñar las distribuciones de la planta o taller para que éxito mayor eficiencia y eficacia en el mantenimiento y producción considerando los procesos, servicio y productos.

2.15. Lineamientos Institucionales

El proceso de mantenimiento y control debe considerar los siguientes lineamientos:

- Los lineamientos del COLOGE y COLOG se encontraran establecidos mediante directivas, instructivos, reglamentos y manuales vigentes relacionados con el mantenimiento (Obando)

CAPÍTULO III

DESARROLLO DEL SISTEMA DE WEB

3.1. Introducción

El presente capítulo se expondrá la Implementación de un Sistema informático Para El Control De Mantenimiento De Los Vehículos De Mando Y Control En El Comando Logístico No. 25 Reino De Quito, a partir de la metodología de desarrollo de software XP, con cada una de sus fases desde su análisis de hasta la aceptación de cliente.

Sabiendo que las metodologías agiles facilitan el diseño de un software de calidad con la entrega a tiempo del producto, se adoptado la metodología XP (Extreme programmig) para la Implementación de un Prototipo de Sistema para el Control de Mantenimiento de los Vehículos de Mando y Control Comando Logístico No. 25 Reino De Quito, misma que proporcionan herramientas para el desarrollo del software, basado en la realimentación continua entre cliente-equipo de desarrollo, comunicación fluida entre todos los participantes y enfrenta cambios que pudiera suscitar en el trascurso del desarrollo del sistema.

3.2. Especificación de Requisitos del Sistema (ERS)

La recolección de requisitos se basa en los estándares IEEE 830 con las necesidades del cliente llegando a constituir los requerimientos funcionales y no funcionales del sistema, de esta manera el equipo desarrollador del prototipo de sistema analiza los requisitos, plasma las funcionalidades y obtienen la base para el desarrollo del software.

3.2.1. Requerimientos Funcionales

Tabla 2.

RF-01 Autenticación de Usuario

Identificación del requerimiento:	RF01
Nombre del Requerimiento:	Autentificación de Usuario.
Características:	Los usuarios deberán identificarse o validar para acceder a cualquier parte del sistema.
Descripción del requerimiento:	Para ingresar al sistema, el cliente validará con su respectivo Usuario y contraseña por medios del Sistema Integrado Del Ejército (SIFTE), el sistema desplegará características dependiendo al perfil que de cada usuario.
Prioridad del requerimiento: Alta	

Tabla 3.

RF-02 Gestionar Vehículos

Identificación del requerimiento:	RF02
Nombre del Requerimiento:	Gestionar Vehículos
Características:	Permite gestionar información de los vehículos de mando y control.
Descripción del requerimiento:	Mediante la asignación de permisos del SIFTE el administrador del prototipo de sistema registrar la información referente a los vehículos.
Prioridad del requerimiento: Alta	

Tabla 4.

RF-03 Gestión de Matriz Recepción de Vehículos

Identificación del requerimiento:	RF03
Nombre del Requerimiento:	Gestión de Matriz recepción de Vehículos
Características:	Permite gestionar información referente a la recepción de vehículos
Descripción del requerimiento:	Permite al usuario una vez accedido al sistema, el ingreso o la selección de la información referente a los vehículos, personal encargado del vehículo, tiempo que tardará el mantenimiento, fecha/hora y el tipo de mantenimiento que se va a realizar al vehículo.
Prioridad del requerimiento: Alta	

Tabla 5.

RF-04 Gestión Orden de Trabajo

Identificación del requerimiento:		RF04
Nombre del Requerimiento:		Gestión Orden de Trabajo
Características:		Permite gestionar información orden de trabajo
Descripción del requerimiento:		El Sistema permitirá al usuario generar de forma automática el formulario Orden de trabajo disminuyendo el tiempo empleado de llenado del mismo.
Prioridad del requerimiento:		Alta

Tabla 6.

RF-05 Validación Matriz de Lubricantes o Repuestos

Identificación del requerimiento:		RF05
Nombre del Requerimiento:		Validación Matriz Lubricantes o Repuestos
Características:		Permite validar lubricantes o repuestos para el mantenimiento de vehículos
Descripción del requerimiento:		El Sistema permitirá al usuario la selección o ingreso de lubricantes o repuestos.
Prioridad del requerimiento:		Alta

Tabla 7.

RF-06 Gestión Libro de Vida del Vehículo

Identificación del requerimiento:		RF06
Nombre del Requerimiento:		Gestión Matriz Libro de Vida del Vehículos
Características:		Permite gestionar la matriz libro de vida de vehículos
Descripción del requerimiento:		El sistema de permitirá generar la matriz libro de vida de vehículo extrayendo información de la tabla orden de trabajo.
Prioridad del requerimiento:		**Alta**

Tabla 8.

RF-07 Consulta Información

Identificación del requerimiento:		RF07
Nombre del Requerimiento:		Consultar Información.
Características:		El sistema ofrecerá al usuario información general acerca de: matriz (recepción de vehículos, orden de trabajo, solicitud de lubricantes o repuestos, acta de entrega, libro de vida de vehículos),
Descripción del requerimiento:		Muestra información general sobre matriz (recepción de vehículos, orden de trabajo, solicitud de lubricantes o repuestos, acta de entrega, libro de vida de vehículos).
Prioridad del requerimiento:		Alta

Tabla 9.

RF-08 Gestionar Reporte

Identificación del requerimiento:	RF008
Nombre del Requerimiento:	Gestionar Reportes.
Características:	El sistema permitirá generar reportes.
Descripción del requerimiento:	Permite al administrador imprimir reportes de los eventos realizados: recepción de vehículos, orden de trabajo, solicitud de lubricantes o repuestos, acta de entrega, libro de vida de vehículos
Prioridad del requerimiento:	Alta

3.2.2. Requerimientos No Funcionales.

Tabla 10.

RNF-01 Interfaz del Sistema

Identificación del requerimiento:	RNF01
Nombre del Requerimiento:	Interfaz del Sistema.
Características:	El sistema poseerá una interfaz amigable para usuario facilitando
Descripción del requerimiento:	El sistema contendrá de botones y menús que facilite al usuario el manejo eficiente del sistema
Prioridad del requerimiento:	Alta

Tabla 11.

RNF-02 Mantenimiento

Identificación del requerimiento:	RNF02
Nombre del Requerimiento:	Mantenimiento.
Características:	El sistema deberá de tener manual de usuario para facilitar los mantenimientos y movimiento dentro del sistema que serán realizados por los usuarios.
Descripción del requerimiento:	El sistema debe disponer de una documentación fácilmente actualizable que permita realizar operaciones de mantenimiento del mismo.
Prioridad del requerimiento:	Alta

Tabla 12.

RNF-03 Seguridad en Información

Identificación del requerimiento:	RNF03
Nombre del Requerimiento:	Seguridad en Información
Características:	El sistema garantizara a los usuarios una seguridad en cuanto a la información que se administra dentro del sistema.
Descripción del requerimiento:	Garantizar el sistema la seguridad de la información que se manejan como, documentos, archivos y contraseñas.
Prioridad del requerimiento:	Alta

3.3. Requisitos Comunes de las Interfaces

3.3.1. Interfaces de Usuario

La interfaz del usuario está compuesta por un conjunto de ventanas con botones amigables, listas y campos de textos referentes al control de mantenimiento de vehículos.

3.3.2. Interfaces de Hardware

Para la implementación del Software es necesario disponer de equipos de cómputo en perfecto estado con las siguientes características:

- Adaptadores de red.
- Procesador Intel Core i3 2.13GHz o superior.
- Memoria Ram 4.00 GB
- Disco duro 500 GB
- Mouse.
- Teclado.

3.3.3. Interfaces de Software

- Sistema Operativo: Windows 7 o superior.
- Explorador: Mozilla o Chrome.

3.3.4. Interfaces de Comunicación

Los servidores, clientes y aplicaciones se comunicarán entre sí, mediante protocolos de comunicación, a través de cables enlazados a sus respectivos puertos de conexión o conexiones inalámbricas con el fin de permitir el intercambio de información.

3.4. Requisitos de Rendimiento

Garantizar el proceso de las consultas, ingresos, modificación, reportes no afecten a desempeño de la base de datos y a la manipulación de interfaces de sistema.

- **Seguridad.-** Garantizar la confiabilidad y el desempeño del sistema informático respecto al manejo de información y datos, tales como archivos y documentos de mantenimiento de los vehículos o datos y contraseñas de los usuarios, toda esta información almacenada podrán ser consultados y actualizados permanente y simultáneamente, sin producir algún error o desbordamiento de memoria afectando al tiempo de respuesta del software.

- **Fiabilidad.-**El sistema presentara una interfaz amigable para usuario facilitando el manejo sencillo para el mismo, ajustando a las características de la web.

- **Disponibilidad.-** La disponibilidad del sistema debe ser continua para el servicio de los usuarios de 7 días por 24 horas, garantizando un adecuado funcionamiento durante la utilización del software.

- **Mantenibilidad.-** El sistema deberá tener el manual de usuario para facilitar los mantenimientos y movimiento dentro del sistema, que

facilite al personal con poca experiencia en el uso de aplicaciones informáticas.

- **Portabilidad.-**El sistema será viable implantar bajo plataforma de Windows, Linux, Mac.

3.5. Diseño de Interfaces

Se mostrara el diseño de las interfaces que posee el prototipo de sistema para la interacción con el usuario.

- **Interfaz ingreso al Sistema**

El usuario ingresará al sistema con su respectivo usuario y password

Figura 8. Interfaz Ingreso al Sistema

- **Interfaz de Inicio.-** Pantalla de inicio muestra los diferentes menús que posee el prototipo de sistema de mantenimiento de vehículos.

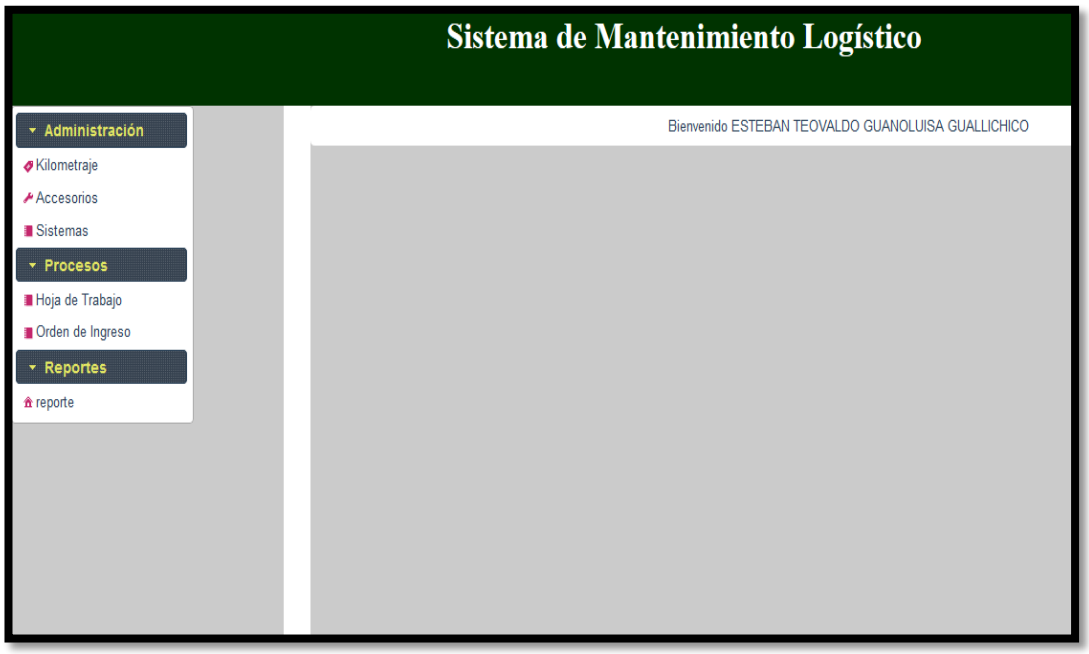


Figura 9. Interfaz de Inicio

- **Interfaz Orden de Recepción de Vehículo.-** Permite al usuario el ingreso o la selección de la información referente al vehículo, personal encargado del vehículo, mecánico encargado del mantenimiento, tiempo que tardara el mantenimiento, fecha/hora , numero de recepción y tipo de mantenimiento a realizar el mismo.

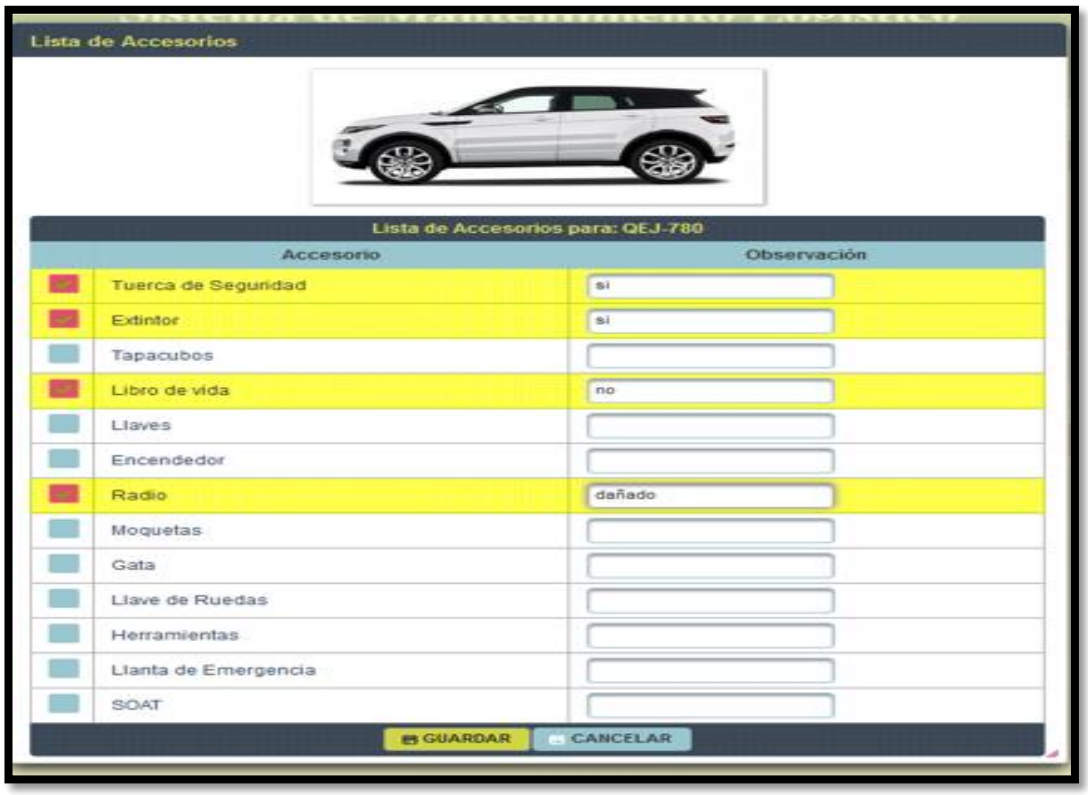


Figura 10. Interfaz Recepción de Vehículo

- **Interfaz Administración de Kilometraje.-**En esta interfaz el usuario administrará el kilometraje, teniendo en cuenta que cada kilometraje establece parámetros para el respectivo mantenimiento de los vehículos.

Figura 11. Interfaz Administración de Kilometraje

- **Interfaz de Administración de Accesorios.-**Esta interfaz permite al usuario registrar los accesorios que bien incluido el vehículo.

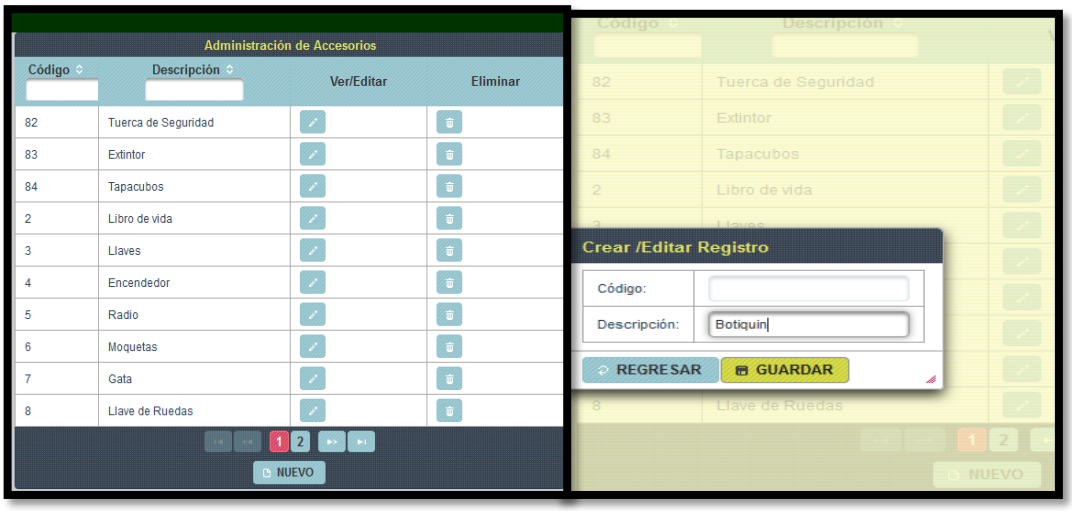


Figura 12. Interfaz Administración de Accesorios

- **Interfaz Administración de Sistemas.-**Administración de sistema permite al usuario registrar los diferentes sistemas que posee un automotor.

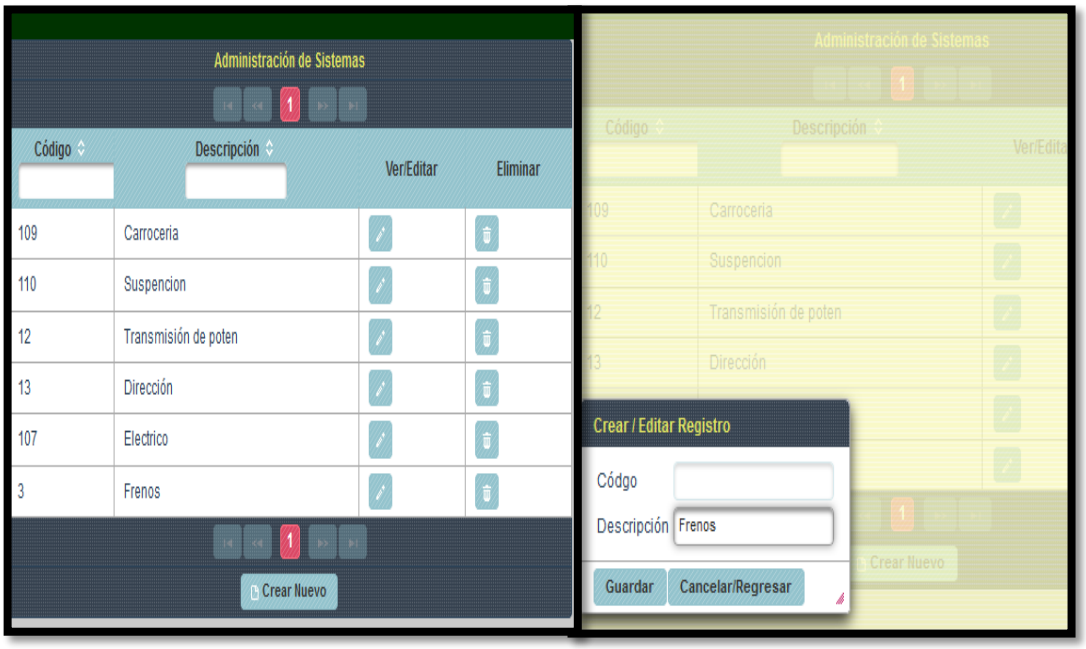


Figura 13. Interfaz Administración de Sistemas

- **Interfaz de Reporte de Mantenimiento.-** Esta interfaz permite al usuario visualizar los trabajos de mantenimiento realizados a los vehiculos.

Figura 14. Interfaz de Reporte de Mantenimiento

3.5.1. Diseño Conceptual de la Base de Datos

En el diseño de conceptual se describe la información de la bases de datos del prototipo de sistema, se identifica las entidades con sus respectivas características o atributos.

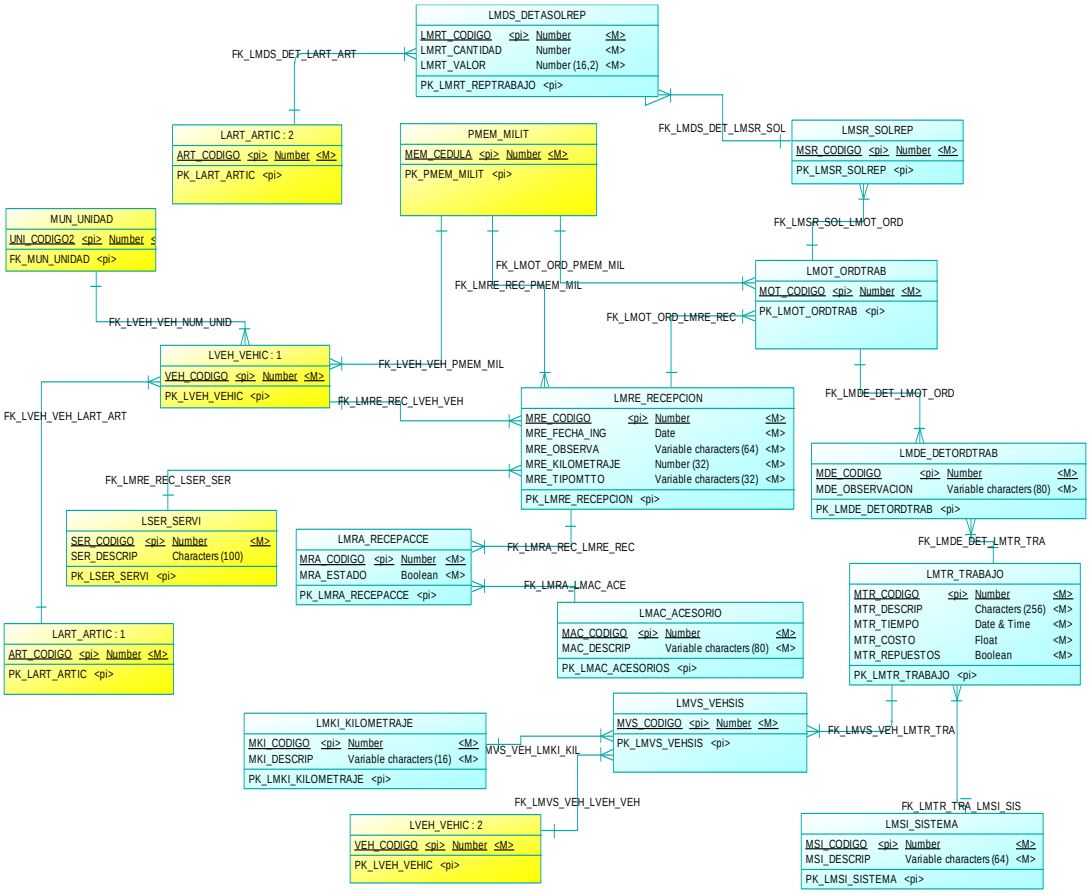


Figura 15. Diseño Conceptual de Base de Datos

3.5.2. Diseño Lógico de la Base de Batos

Partiendo del esquema conceptual, el diseño lógico describe la estructura de la base de datos que puede procesar un Sistema de Gestor de Base de Datos (SGBD).

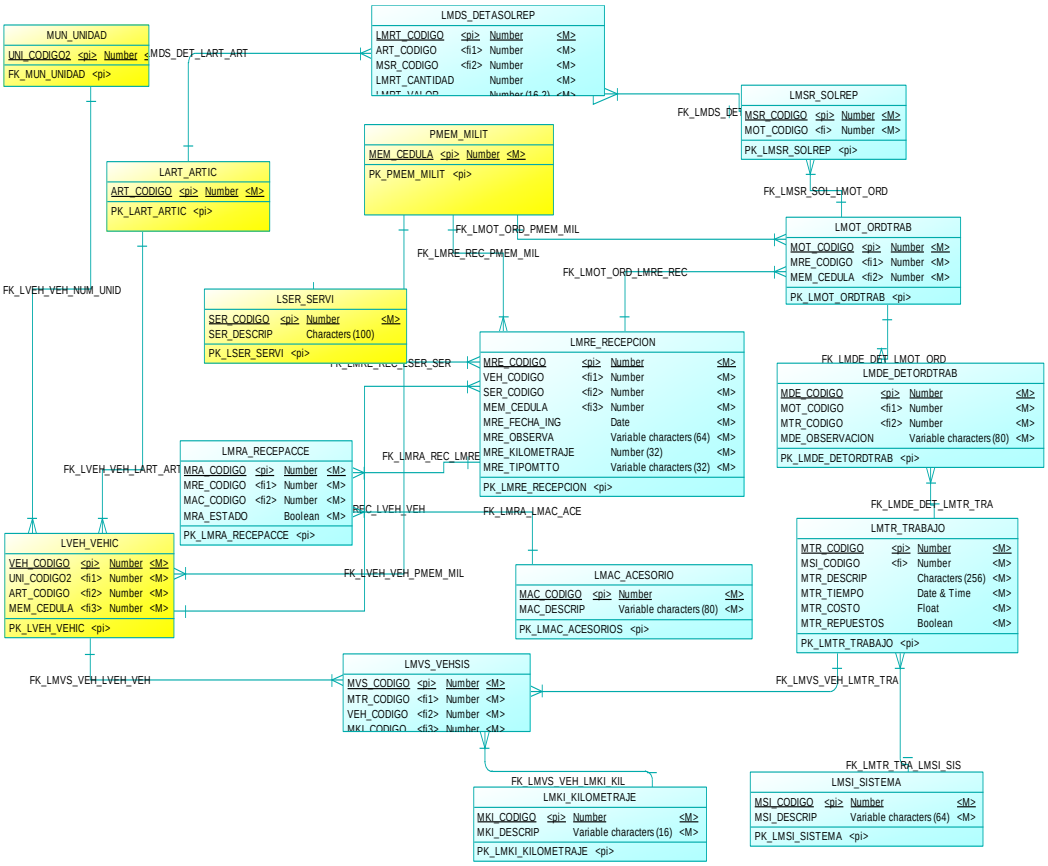


Figura 16. Diseño Lógico de la Base de Datos

3.5.3. Diseño Físico de la Base de Datos

Describe la implementación de la base de datos, la estructura de almacenamiento y los métodos de acceso a los datos se trasforma las entidades en tablas y los atributos en columnas.

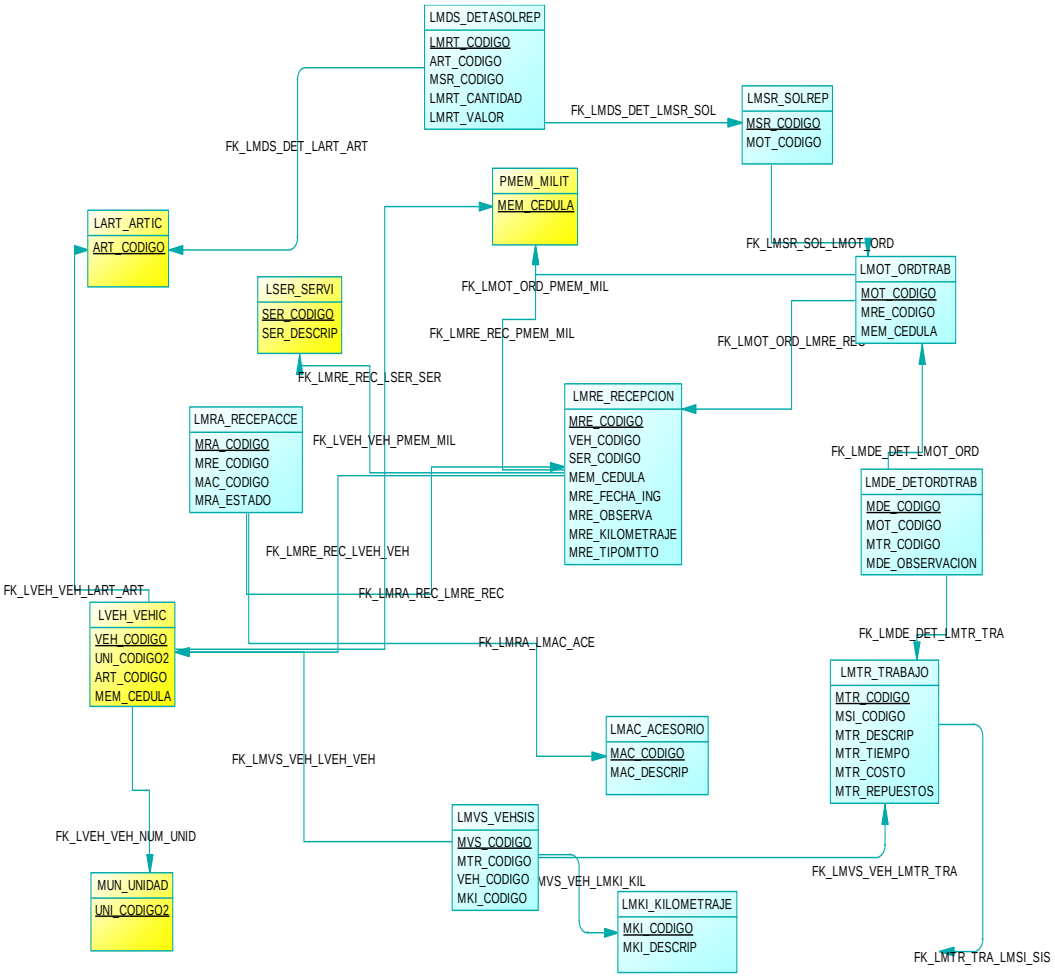


Figura 17. Diseño Físico de la Base de Datos

CAPITULO IV

CONCLUSIONES Y RECOMENDACIONES

4.1. CONCLUSIONES

- Se recopiló información técnica del proceso administrativo de mantenimiento de vehículos del Comando Logístico Nº 25 Reino de Quito, alcanzando el modelamiento de la Base de Datos que sirve de base para el desarrollo de versiones futuras del prototipo de sistema de mantenimiento de vehículos, misma que facilitara al desarrollador la sustentabilidad teórica del nuevo proyecto.

- Se Implementó el prototipo de sistema informático de mantenimiento de vehículos del departamento de Mando y Control en el Comando Logístico Nº25 Reino de Quito, el cual controla el proceso administrativo de recepción, tiempo de reparación y reporte de mantenimiento de vehículos, optimizando los recursos humanos y económicos que dispone esta unidad.

- Con la implementación del de software se logró obtener datos estadísticos relacionados al proceso de mantenimiento de vehículos, para determinar las necesidades en el tiempo de dotar de materiales y equipo de trabajo de mantenimiento, consiguiendo obtener el óptimo funcionamiento del parque automotor del Comando Logístico Nº 25 Reino de Quito.

4.2. RECOMENDACIONES

- Dentro de este proyecto de titulación tan ambicioso como lo fue: el Prototipo de Sistema, se requiere la actualización del sistema con nuevas Especiaciones de Requisitos de Software e historiales del usuario para fortalecer el software de mantenimiento de vehículos.

- Una vez concluida el presente proyecto, se considera interesante investigar sobre otros aspectos relacionados y se propone:

  - Extender el trabajo practico con la inclusión de los diferentes roles de cada uno de los usuarios del sistema dentro de la jerarquía militar.

  - Trabajar en mejorar el modelo XP para la comunicación de todos los participantes del proyecto en futuras versiones del sistema informático.

  - La utilización de herramientas de desarrollo de software existentes en el mercado informático, que permita realizar trabajos más interactivos entre el cliente y el servidor.

REFERENCIAS BIBLIOGRÁFICAS

Bernal Bermúdez , J. (01 de 09 de 2012). *http://miw.eui.upm.es/master-ingenieria-web.html.* Obtenido de https://www.etsisi.upm.es/sites/default/files/curso_2013_14/MASTER/MIW.JEE.POOJ.pdf

Enríquez Fuel , G. G., & Ayala Rosero , E. J. (2011). Obtenido de http://repositorio.espe.edu.ec/bitstream/21000/5661/1/AC-SISTEMA-ESPE-033734.pdf

Hernández Orallo, E. (s.f.). El Lenguaje Unificado de Modelado (UML)., (págs. 2,3,4,5).

Aprigliano, D. (1999). *2D Comunicación Interactiva*. Obtenido de http://www.dosd.com.ar/que-es-un-hosting-o-servidor-web-para-que-sirve/

Ejército, D. d. (s.f.). *Estandares Para Crear Objetos De Base De Datos Para Las Aplicaciones Del Sifte.* Quito.

Fortinet. (13 de Noviembre de 2016). Obtenido de Fortinet: http://proyectobasesdedatos.galeon.com/enlaces1407194.html

Frick, C. (28 de Diciembre de 2007). *The Fricky*. Obtenido de El Aprendiz: https://thefricky.wordpress.com/2007/12/28/programacion-orientada-a-objetos-caracteristicas-de-la-poo/

Gonzales, D. (s.f.). Obtenido de www.fraktalweb.com: http://fraktalweb.com/blog/sistemas-web-para-que-sirven/

http://proyectobasesdedatos.galeon.com/enlaces1407194.html. (s.f.). Obtenido de http://proyectobasesdedatos.galeon.com/enlaces1407194.html

Joskowicz , J. (2008). *Reglas y Prácticas en eXtreme Programming .* España.

Latorre, G. (22 de Marzo de 2010). *Programacion II JAVA*. Obtenido de http://gl-epn-programacion-ii.blogspot.com/2010/03/jvm-jdk-jre-conceptos-fundamentales-de.html

Letelier, P. (2003). *Metodologías Ágiles en el Desarrollo de Software.* Alicante.

Metodologías Tradicionales. (15 de diciembre de 2016). *Metodologías Tradicionales.* Obtenido de Metodologías Tradicionales: https://tallerinf281.wikispaces.com/file/view/METODOLOG%C3%8DAS+TRADICIONALES.pdf

Monferrer Agut, R. (2001). Obtenido de http://www.academia.edu/6647065/Especificaci%C3%B3n_de_Requisitos_Software_seg%C3%BAn_el_est%C3%A1ndar_de_IEEE_830

Obando, C. (s.f.). *Manual General de Mantenimiento.* Quito: Mta-tasea8-00.

Olivares, R. (14 de Diciembre de 2010). *EcuaRed*. Obtenido de https://www.ecured.cu/JDK

Oracle. (s.f.). *www.oraclebddepn.blogspot.com.* Obtenido de http://oraclebddepn.blogspot.com/2013/05/acerca-de-oracle_6479.html

Piattini Velthuis, M. (1996). *Análisis y Diseño detallado de aplicaciones Informáticas de Gestión.* Madrid: 1 ed.

Roger S, P. (2010). *Igeniería del Software.* México, D.F.: Mexicana.

Rouse, M. (16 de Enero de 2015). *TechTarget*. Obtenido de http://searchdatacenter.techtarget.com/es/definicion/Open-Database-Connectivity-ODBC

Software, L. N. (marzo de 2009). Obtenido de http://datateca.unad.edu.co/contenidos/301569/guia_de_ingenieria_del_software.pdf

Torres Letelier , P., & Sánchez López, E. (2003). *Metodologías Ágiles en el Desarrollo de Software.* España.

www.conmasfuturo.com. (20 de Marzo de 2014). Obtenido de http://www.conmasfuturo.es/los-entornos-de-desarrollo-java-los-mejores-entornos-de-desarrollo-java-para-ninos-y-adolescentes/

HOJA DE VIDA

DATOS PERSONALES

NOMBRE: GUANOLUISA GUALLICHICO ESTEBAN T.

GRADO: CBOS. DE COM

NACIONALIDAD: ECUATORIANA

FECHA DE NACIMIENTO: 4-ABR-1988

CÉDULA DE CIUDADANÍA: 1718428491

CORREO ELECTRÓNICO: estebanteo1718@gmail.com

TELÉFONO: 0983351860

DIRECCIÓN: PICHINCHA –QUITO– SANTA ROSA DE CHILLOGALLO

ESTUDIOS REALIZADOS

PRIMARIA:

Escuela Cristo Salvador

Pichincha –Quito– Santa Rosa De Chillogallo

SECUNDARIA:

Colegio Militar No. 10 Abdón Calderón

Pichincha –Quito– Santa Rosa De Chillogallo

SUPERIOR:

Escuela De Formación De Soldados De La Fuerza Terrestre.

Unidad De Gestión De Tecnologías – ESPE

Cotopaxi - Latacunga – Ecuador

TÍTULO OBTENIDO

Bachiller en Ciencias “Comercio y Administración” Especialidad Informática

Tecnólogo en Ciencias Militares

Tecnólogo en la Carrera de Computación

Suficiencia en el Idioma Ingles ESPE

Conductor Profesional

EXPERIENCIA PROFESIONAL O PRÁCTICAS PRE PROFESIONALES

Dirección de Sistemas y Comunicación (COMANDANCIA) Pasantías ESPE

Grupo de telecomunicaciones (COMANDO CONJUNTO) Pasantías ESPE

Unidad Educativa Ana Páez Vinculación ESPE

CURSOS Y SEMINARIOS

Curso de derechos humanos

Curso de derecho internacional humanitario

HOJA DE VIDA

DATOS PERSONALES

NOMBRE: CAÑAR SIZA MARCO EDUARDO

GRADO: CBOS. DE I.M

NACIONALIDAD: ECUATORIANA

FECHA DE NACIMIENTO: 12-DIC-1987

CÉDULA DE CIUDADANÍA: 1804277257

CORREO ELECTRÓNICO: eduart.ca24@hotmail.es

TELÉFONO: 0984962684

DIRECCIÓN: TUNGURAHUA - PILLARO - SAN ANDRES – SAN ANTONIO

ESTUDIOS REALIZADOS

PRIMARIA:

Escuela Manuela Cañizares

Tungurahua - Pìllaro - San Andrés – San Pedro

SECUNDARIA:

Instituto Superior Tecnológico “los Andes”

Tungurahua - Pìllaro

SUPERIOR:

Escuela De Formación De Soldados De La Fuerza Terrestre.

Unidad De Gestión De Tecnologías – ESPE

Cotopaxi - Latacunga – Ecuador

TÍTULO OBTENIDO

Bachiller en Ciencias “Comercio y Administración” Especialidad Informática

Tecnólogo en Ciencias Militares

Tecnólogo en la Carrera de Computación

Suficiencia en el Idioma Ingles ESPE

Conductor Profesional

EXPERIENCIA PROFESIONAL O PRÁCTICAS PRE PROFESIONALES

Dirección de Sistemas y Comunicación (COMANDANCIA) Pasantías ESPE

Escuela De Formación De Soldados De La Fuerza Terrestre Pasantías ESPE

Unidad Educativa Ana Páez Vinculación ESPE

CURSOS Y SEMINARIOS

Curso de derechos humanos

Curso de derecho internacional humanitario

HOJA DE LEGALIZACIÓN DE FIRMAS

DEL CONTENIDO DE LA PRESENTE INVESTIGACIÓN SE RESPONSABILIZAN LOS AUTORES

CAÑAR S. MARCO E.
CBOS. DE I.M.
CI: 1804277257

GUANOLUISA G. ESTEBAN T.
CBOS. DE COM.
CI: 1718428491

DIRECTOR DE LA CARRERA DE COMPUTACIÓN

ING. JORGE PARDO

Latacunga, 06 junio de 2017