FACULTAD DE INGENIERÍA Y CIENCIAS AGROPECUARIAS

DESARROLLO DE UN SISTEMA DE DOCUMENTACION OPERACIONAL EN LAS AERONAVES DE LAN ECUADOR

Trabajo de Titulación presentado en conformidad con los requisitos para obtener el Título de Ingeniero en Sistemas de Computación e Informática

Profesor guía

Ing. Edgar Fernando Solís Acosta

Autor

Daniel Alejandro Jurado Carrera

Año

2013

DECLARACIÓN PROFESOR GUÍA

“Declaro haber dirigido este trabajo a través de reuniones periódicas con el estudiante, orientando sus conocimientos y competencias para un eficiente desarrollo del tema escogido y dando cumplimiento a todas las disposiciones vigentes que regulan los Trabajo de Titulación”

Edgar Fernando Solís Acosta

Ingeniero de Sistemas

180300507-1

DECLARACIÓN DE AUDITORÍA DEL ESTUDIANTE

“Declaro que este trabajo es original, de mi autoría, que se han citados las fuentes correspondientes y que en su ejecución se respetaron las disposiciones legales que protegen los derechos de autor vigentes”

Daniel Alejandro Jurado Carrera

171602149-6

DEDICATORIA

A Dios.

Por ser el pilar fundamental en mi vida y haberme permitido llegar hasta este punto con salud, por darme la posibilidad de cumplir mis sueños y lograr mis objetivos, además de su infinito amor.

A mi Familia.

Por haberme apoyado en todo momento, por su motivación constante, sus consejos, sus valores que me ha permitido ser una persona de bien, pero más que nada, por su amor.

RESUMEN

LAN Ecuador es una aerolínea creada por LAN Airlines en el año 2001, holding chileno de aerolíneas, probablemente el más importante de América Latina y uno de los mejor catalogados en el mundo, dedicada al servicio de transporte aéreo de pasajeros y carga, dentro del país como fuera del país.

El presente proyecto tiene como objetivo lograr el desarrollo de un software para la gestión de información documental de la aerolínea LAN, Este sistema se encontrara en las aeronaves el cual se encargara de brindar servicios a su tripulantes ofreciendo una atención adecuada y oportuna.

El propósito de este desarrollo es dar facilidades tanto al personal encontrando de manera ordenada e inmediata la información una vez ingresado los datos correspondientes de los manuales. Para lograr con este objetivo, por lo tanto se utilizara la siguiente tecnología, Microsoft C# como lenguaje principal, Microsoft Access como Base De Datos.

El proceso de desarrollo del proyecto será aplicado utilizando la metodología Extreme Programming, la misma que servirá para gestionar la implementación del sistema a través de sus fases e iteraciones, otro elemento esencial el mismo que se usara para el diseño del sistema.

Como resultado de este proyecto se obtiene un sistema de documentación operacional, con una interfaz grafica amigable de fácil utilización que permite que el usuario consultar y realizar la búsqueda correspondiente de la información de los manuales de avión.

Esta propuesta tiene un favorable impacto social ya que los usuarios se verán estimulados al ver reflejado que la información de los manuales se manejen de forma ordenada, organizada, lo cual contribuirá también a incentivar la investigación de nuevas tecnologías para ofrecer un mejor servicio a cada uno de los usuarios de la aerolínea LAN Ecuador.

ABSTRACT

LAN Ecuador is an airline created by LAN Airlines in the year 2001, Chilean holding of airlines, probably the most important in Latin America and one of the best classified in the world, dedicated to the service of passengers' air transport and it loads, inside the country like outside of the country.

The present project has as objective to achieve the development of software for the administration of documental information of the airline LAN, This system was in the airships which took charge of offering services to its crew offering an appropriate and opportune attention.

The purpose of this development is to give facilities so much to the personnel finding once in an orderly and immediate way the information entered the data corresponding of the manuals. To achieve with this objective, therefore the following technology, Microsoft C# was used as main language, Microsoft Access like database.

The process of development of the project will be applied using the methodology it carries to an extreme Programming, the same one that will be good to negotiate the implementation of the system through its phases and iterations, another essential element the same one that was used for the design of the system.

As a result of this project a system of operational documentation is obtained, with a friendly graphic interface of easy use that allows that the user to consult and to carry out the search corresponding of the information of the airplane manuals.

This proposal has a favorable social impact since the users they will be stimulated when being reflected that the information of the manuals is managed in an orderly, organized way, that which will also contribute to incentivate the investigation of new technologies to offer a better service to each one of the users of the airline LAN Ecuador.

INDICE

INTRODUCCIÓN 1

1. MARCO TEÓRICO..... 5

1.1 Fundamentación Teórica..... 5
1.1.1 Metodología Extreme Programming(XP) o Programación extrema 5
1.1.1.1 Fases Extreme Programming (XP) 6
1.1.1.2 Características Extreme Programming (XP) 7
1.1.1.3 Roles 8
1.1.1.4 Reglas y practicas 9
1.1.1.4.1 Reglas y prácticas para la planificación 9
1.1.1.4.2 Reglas y prácticas para el diseño 10
1.1.1.4.3 Reglas y prácticas para el desarrollo de código..... 10
1.1.1.4.4 Reglas y prácticas para las pruebas 11
1.1.1.5 Mejore prácticas 12
1.1.2 Lenguaje UML 13
1.1.2.1 Modelo, vistas y diagramas 13
1.1.2.1.1 Vista de casos de uso 14
1.1.2.1.2 Vista estática..... 15
1.1.2.1.3 Vista de implementación 16
1.1.2.1.4 Vista de despliegue..... 17
1.1.2.1.5 Vista de máquina de estados..... 18
1.1.2.1.6 Vista de actividades 19
1.1.2.1.7 Vista de interacción..... 20
1.1.3 Base de datos 24
1.1.3.1 Microsoft Access..... 25
1.1.3.2 Lenguaje de alto nivel..... 26
1.1.3.3 Lenguaje de Microsoft C# 26
1.1.3.4 Proceso de digitalización 29
1.1.3.5 Sistemas electrónicos de gestión documental o sistemas de gestión de documentos electrónicos..... 31
1.1.3.6 Herramienta case 33
1.1.3.7 SAP Sybase PowerDesigner 33

2. DISEÑO Y DESARROLLO SISTEMA DE DOCUMENTACION OPERACIONAL. 34

2.1 Presentación 34
2.2 Objetivo..... 34
2.3 Descripción 34
2.4 Fase de Planificación 34
2.4.1 Sentencia que define el problema 35
2.4.2 Historias de usuarios(Stakeholders)..... 36
2.4.2.1 Perfil de “stakeholders” 37
2.4.3 Descripción global del producto 40

2.4.3.1 Característica del producto 40
2.4.3.2 Requerimientos específicos 41
2.4.3.2.1 Requerimientos de interfaces externas 41
2.4.3.3 Diagramas de casos de uso de sistemas 41
2.4.3.3.1 Casos de uso del negocio 41
2.5 Fase de Diseño 42
2.5.1 Caso de uso del sistema 43
2.5.1.1 Identificación de actores del sistema 43
2.5.1.2 Caso de uso del administrador 43
2.5.2 Diagrama de secuencias 45
2.5.3 Diagrama de clases 47
2.5.3.1 Diagrama de clases ventana del sistema 48
2.5.4 Diagrama de componentes 50
2.5.4.1 Diagrama de componentes sistema documental 51
2.5.4.2 Diagrama de componentes de gestión de base de datos 51
2.5.5 Diagrama Arquitectura Sistema 52
2.5.6 Plan De Entregas 52
2.5.7 Diagrama De Base De Datos 54
2.5.7.1 Modelo Físico 54
2.5.7.2 Modelo Lógico 55
2.5.7.3 Modelo Orientado A Objetos 56
2.6 Fase de Implementación 57
2.6.1 Modelo de la interfaz de usuario 57
2.6.2 Ventana Inicio Del Sistema 57

3. ANALISIS COSTO/BENEFICIO. 73

3.1 Costo 73
3.1.1 Factibilidad Técnica 73
3.1.1.1 Estación de trabajo de la aplicación 74
3.1.2 Factibilidad Operacional 75
3.1.3 Factibilidad Económica 76
3.2 Categorías y Beneficios 79

4. CONCLUSIONES Y RECOMENDACIONES..... 81

4.1 Conclusiones 81
4.2 Recomendaciones 82
Referencias: 83
ANEXOS 85

INDICE DE FIGURAS

Figura 1. Metodología Extreme Programming..... 5
Figura 2. Ciclo De Vida Proyecto XP 6
Figura 3. Principios De La Metodología XP 12
Figura 4. Diagramas de Casos de Uso. 14
Figura 5. Diagrama De Clases. 15
Figura 6. Diagrama De Objetos..... 16
Figura 7. Diagrama De Componentes. 17
Figura 8. Diagrama De Despliegue. 17
Figura 9. Diagrama De Estados. 18
Figura 10. Diagrama De Actividades..... 19
Figura 11. Diagrama De Secuencia. 20
Figura 12. Diagrama De Colaboración. 21
Figura 13. Gestión Modelo. 22
Figura 14. Diagrama De Paquetes. 22
Figura 15. Diagrama De Componentes Del Sistema. 23
Figura 16. Diagrama De Componentes De Base De Datos. 23
Figura 17. Arquitectura Del Sistema. 24
Figura 18. Entorno De Microsoft Access. 25
Figura 19. Entorno De C#. 29
Figura 20. Proceso De Digitalización 31
Figura 21: Dispositivos De Hardware. 32
Figura 22. Entorno Power Designer. 33
Figura 23. Esquema del sistema. 40
Figura 24. Esquema del sistema. 42
Figura 25. Caso de uso del administrador. 43
Figura 26. Paquete De Caso Uso. 44
Figura 27. Acceso A La Documentación. 45
Figura 28. Interacción De Matrícula. 46
Figura 29. Logueo. 46
Figura 30. Menú Documentación. 47
Figura 31. Clase Detallado1. 48
Figura 32. Clase Detallado 2. 49
Figura 33. RptAppServidor. 50
Figura 34. Diagrama De Componentes. 50
Figura 35. Diagrama De Gestión Documentación Operacional. 51
Figura 36. Diagrama de componentes de gestión de base de datos. 51
Figura 37. Diagrama Arquitectura Del Sistema 52
Figura 38. Modelo Físico. 54
Figura 39. Modelo Lógico. 55
Figura 40. Modelo Orientado A Objetos. 56
Figura 41. Usuario y Contraseña. 57
Figura 42. Documentación Operacional. 58
Figura 43. Menú Principal. 58
Figura 44. StoryBoard. 59
Figura 45. Pantalla Login 60
Figura 46. Pantalla Inicio. 61

Figura 47. Pantalla Menu. 62
Figura 48. Pantalla Manuales del Avión. 63
Figura 49. Pantalla de Manuales del Avión Mantenimiento. 64
Figura 50. Ventana Árbol. 65
Figura 51. Ventana Operador. 65
Figura 52. Ventana Flota. 66
Figura 53. Ventana Modelo de Aeronave. 66
Figura 54. Ventana Matrícula. 67
Figura 55. Ventana Tipo. 67
Figura 56. Ventana Parámetro. 67
Figura 57. Ventana Usuario. 68
Figura 58. Ventana Consultar Bitácora. 68
Figura 59. Ventana Consultar Usuario. 69
Figura 60. Pantalla Acceso usuario limitado al Menú Principal. 70
Figura 61. Menú documentación Operacional I. 71
Figura 62. Arbol de Menú Operacional. 71
Figura 63. Se puede escoger del menú principal sus diferentes opciones. 72
Figura 64. Influencias. 74
Figura 65. Representación De Beneficios Cuantitativos. 78

INDICE DE TABLAS

Tabla 1. Manejo de los manuales en forma manual. 35
Tabla 2. Definición del problema lentitud en la consulta. 35
Tabla 3. Ingreso A La Información Por Personas No Autorizadas. 35
Tabla 4. Stakeholders del Proyecto. 36
Tabla 5. Stakeholders del sistema. 37
Tabla 6. Representante de desarrollo del área técnica. 37
Tabla 7. Representante de control del área técnica. 38
Tabla 8. Administrador. 38
Tabla 9. Usuario Piloto 39
Tabla 10. Usuario Auxiliar. 39
Tabla 11. Actores del negocio. 42
Tabla 12. Actor y roles del sistema. 43
Tabla 13. Plan De Entrega "Iteración" 52
Tabla 14. Diferentes Iteraciones. 53
Tabla 15. Caja negra de acceso al sistema. 60
Tabla 16. Caja negra de ingreso a la documentación operativa. 61
Tabla 17. Caja negra menú principal. 62
Tabla 18. Caja negra Manual del avión. 63
Tabla 19. Caja negra Manual de Avión/Opción Mantenimiento. 64
Tabla 20. Caja negra Acceso usuario limitado al Menú Principal. 70
Tabla 21. Influencias. 73
Tabla 22. Escala de valor. 74
Tabla 23. Requerimiento Del Sistema. 74
Tabla 24. Causas. 75
Tabla 25. Soluciones. 75
Tabla 26. Beneficios De Las Flotas. 75
Tabla 27. Aspecto Económico. 76
Tabla 28. Peso Actual vs uso Notebook. 77
Tabla 29. Beneficio/Ahorro. 77
Tabla 30. Estadística De Beneficios Cuantitativos. 78
Tabla 31. Beneficio Para La Empresa 79
Tabla 32. Beneficio Para El Personal 79
Tabla 33. Metodología y tecnología para el Sistema. 80

INTRODUCCIÓN

Título del Proyecto

Sistema de documentación operacional en las aeronaves de LAN Ecuador

Ámbito De La Empresa

LAN Ecuador es una empresa que ofrece servicios de transporte aéreo de pasajeros y carga, dentro como fuera del país, probablemente el más importante de América Latina, uno de los mejor catalogados en el mundo.

Por esta razón el holding posee alrededor de 10.500 empleados, mientras que LAN Ecuador está formado por alrededor de 1.800 empleados trabajando dentro del país.

Además se debe tomar en cuenta que los manuales operacionales se encuentran a bordo de las aeronaves, las cuales dan mucho beneficio al piloto durante el vuelo. El inconveniente que presenta es porque los manuales se encuentran en papel y da un peso extra en la aeronave; Estos son algunos de los obstáculos a los que se enfrenta la empresa día a día si no cuenta con un sistema de documentación operacional.

Es por ello que un sistema de documentación operacional en las aeronaves de LAN Ecuador van a permitir visualizar los manuales durante el vuelo de manera detalla, actualizada y confiable.

Problema a Resolver

Actualmente los manuales operacionales que van a bordo de las aeronaves (Boeing / Airbus) están escritos en papel, cada uno de ellos poseen información especificada acerca de los sistemas de las aeronaves los cuales son enviados por los fabricantes, estos manuales son de gran utilidad para los pilotos durante los vuelos.

El problema radica en el manejo de los manuales durante el vuelo, por el proceso manual que lo realiza, dando como resultados pérdida de tiempo en la búsqueda sobre la información requerida por los pilotos durante el vuelo.

Por este motivo es necesario un sistema en el cual se integre y automatice sus procesos operacionales permitiendo disminuir el peso de las aeronaves.

Pronostico

Con respecto a los manuales de documentación operacional en las aeronaves de LAN Ecuador, se puede apreciar que al no disponer de la herramienta necesaria para un buen manejo, tendrá un decrecimiento a nivel de funcionamiento, ocasionando demora en la utilización y visualización de lo primordial que necesita el piloto durante su vuelo.

Por esta razón existe un problema al no disponer de un sistema de documentación operacional adecuada y confiable, que permita la digitalización de la documentación en papel que se encuentran a bordo de las aeronaves, para poder visualizar de manera correcta los manuales durante el vuelo.

Alcance del proyecto

Es necesario que LAN Ecuador disponga de un sistema de documentación operacional, para facilitar su visualización durante sus vuelos, además se utilizara la metodología eXtremeProgramming (XP), que es ágil en el desarrollo de software.

El sistema realizará las funciones:

- Visualizar los manuales operacionales del fabricante de la aeronave a través del Sistema de Documentación Operacional, ofreciendo información actualizada, confiable y segura al piloto.
- Visualizar los manuales operacionales de la empresa LAN Ecuador a través del Sistema de Documentación Operacional, ofreciendo información actualizada, confiable y segura al piloto.
- Permite visualizar programas del fabricante Airbus y Boeing, con acceso directo a los programas para fácil navegación en el sistema de documentación operacional.

- Permitir visualizar el programa de Mercancías Peligrosas IATA, para fácil navegación en el sistema.

Pero se debe tomar en cuenta los departamentos involucrados que son: publicaciones operacionales, estándares operacionales, pilotos flota (Airbus/Boeing), operacionales estaciones nacionales (Quito, Guayaquil, Cuenca y Galápagos), Internacionales (Madrid Miami, New York, Santiago de Chile y Buenos Aires).

Objetivos

Objetivo General

- Desarrollar el sistema de Documentación Operacional utilizando las TIC (tecnología de información y comunicaciones), para las Aeronaves de LAN Ecuador.

Objetivo Específicos

- Describir las tecnologías de manejo documental, que se adapten a los requerimientos del sistema documental operacional.
- Ejecutar el proceso de digitalización y manejo de información a través de procedimientos automáticos para evitar pérdidas de tiempo con la finalidad de que su mejore los servicios.
- Diseñar el sistema de manejo Documental, con el fin de adaptarlos a la necesidad de la empresa LAN Ecuador.
- Realizar las pruebas de usuario del sistema de manejo documental y conocer su funcionamiento.
- Analizar el costo beneficio de la aplicación desarrollada.

Justificación

En LAN Ecuador no lleva un adecuado proceso de documentación operacional, se lo realiza mediante manuales en papel, dando como resultado demora en la visualización de la información al momento de solicitar el piloto en el vuelo.

Por consiguiente el desarrollo del sistema de documentación operacional permitirá a LAN Ecuador disminuir los errores, simplificar procesos, información segura y confiable aumentando la eficiencia de sus pilotos en el vuelo.

Además con la ayuda del sistema se tendrá el ahorro de combustible (fuel) en reproducción de papel para la empresa y se apoyará una iniciativa verde, contribuyendo al planeta y su conservación.

Descripción de la empresa

LAN ECUADOR, es una empresa que ofrece servicios de transporte aéreo de pasajeros y carga, dentro del país como fuera del país. Entre las rutas domésticas cubre las ciudades de Quito, Guayaquil, Cuenca, San Cristóbal y Santa Cruz. En las rutas internacionales, que cubre las ciudades de Nueva York, Miami en Estados Unidos, también de Madrid en España, Santiago y Buenos Aires en Sudamérica.

Se encuentra distribuida de la siguiente manera una sede en Guayaquil, en el Edificio las Cámaras, ubicado en la Avenida de las Américas, junto al Centro de Convenciones Simón Bolívar y en Quito está localizado en la Av. Orellana E11-28 y Coruña, Edificio Orellana 2do. Piso, Oficina 201.

Misión

La misión del holding es: “transportar sueños”

Visión

La visión del holding es: “Ser una de las diez principales líneas aéreas del mundo”.

Beneficio de la documentación operacional

Un sistema de documentación operacional es de fácil navegación, debido a que serán visualizados digitalmente, ofreciendo información actualizada, confiable como también segura, los mismos que son de mucha utilidad debido a que abarcan la información de la aeronave durante sus vuelos, disminuyendo el peso.

1. MARCO TEÓRICO

1.1 Fundamentación Teórica

En el desarrollo del marco teórico se dará a conocer conceptos generales que se va a describir a continuación:

1.1.1 Metodología Extreme Programming(XP) o Programación extrema

Es una metodología esencialmente fundamentada en la facilidad y la agilidad, está diseñada para entregar el software de calidad de acuerdo a las necesidades de los clientes, cuya característica es tener como parte en el equipo al usuario final, pues es uno de los requerimientos para alcanzar al éxito del proyecto.

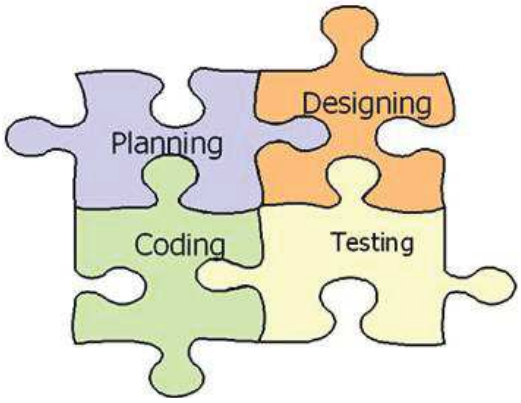


Figura 1. Metodología Extreme Programming
Tomado de María A. Mendoza Sanchez, 2004, http://www.informatizate.net/articulos/metodologias_de_desarrollo_de_software_07062004.html

El primer paso de cualquier proyecto que siga la metodología X.P es definir las historias de usuario con el cliente. Las historias de usuario tienen la misma finalidad que los casos de uso pero con algunas diferencias.
(Castillo Oswaldo, Figueroa Daniel, Sevilla Héctor, 14 de Septiembre del 2011, http://programacionextrema.tripod.com/fases.htm).
Además esta metodología XP está diseñada para adaptarse en forma inmediata a los cambios, con bajos costos asociados, en cualquier etapa del ciclo de vida, también se considera ágil para pequeños y medianos equipos,

consiguiendo productos más fiables y robustos. Asimismo promueve el trabajo en equipo, preocupándose en todo momento de la enseñanza de los desarrolladores y estableciendo responsabilidad como también un buen clima de trabajo.

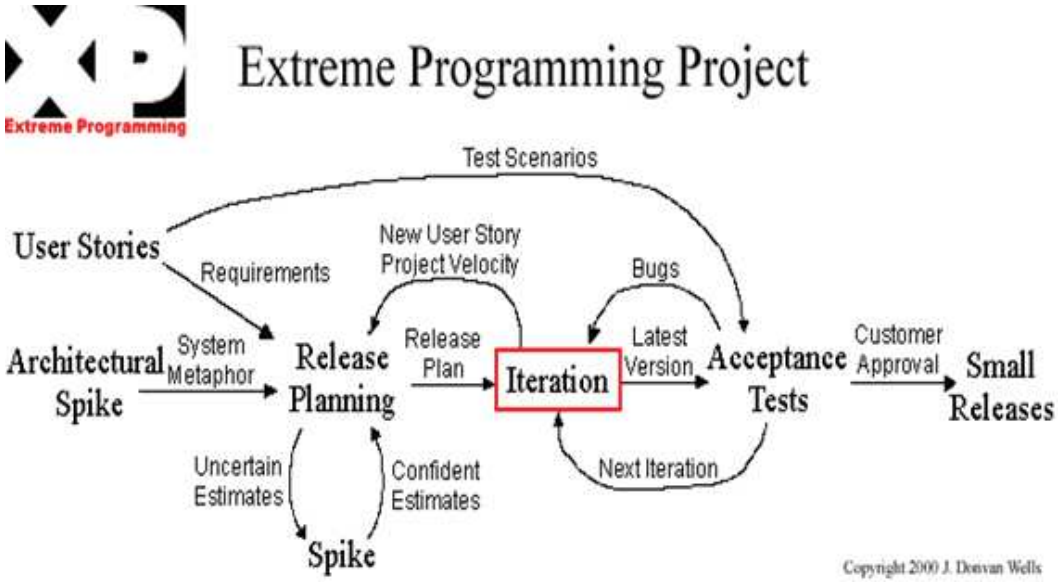


Figura 2. Ciclo De Vida Proyecto XP
Tomado de James Donovan Wells, (2009),
http://www.extremeprogramming.org/rules.html

Así, la XP se puede definir como un conjunto de pasos de diversas metodologías, acopladas de manera que sean pasos flexibles a seguir utilizadas con el uso común, para realizar un desarrollo más agradable y sencillo. (Universo, 11 de Junio del 2012).

1.1.1.1 Fases Extreme Programming (XP)

Las fases del Extreme Programming (XP), definen la duración y los periodos de entrega de un proyecto, a continuación se detalla cada uno de ellos.

- **Exploración**: En esta fase los clientes realizan las “Tarjetas de historia” (storycards) que describe una de las funcionalidades que el programa tendrá. De igual forma el equipo de desarrollo se adapta con las

herramientas, la tecnología y las prácticas a ser utilizadas durante el proyecto. La duración de esta fase puede extenderse desde semanas a meses dependiendo de la adaptación del equipo de desarrollo.

- **Planificación**: El objetivo de esta fase es fijar la prioridad de cada una de las “historias” (stories) y se establece cual va a ser el contenido de la primera entrega. Los programadores estiman cuanto esfuerzo requiere cada story y se establece el cronograma.
- **Iteraciones**: Esta etapa contiene varias iteraciones del sistema previamente a la entrega de la primera versión. El calendario es dividido en un número iteraciones de tal manera de que cada iteración tome de una a cuatro semanas de implementación.
- **Producción**: La fase de producción requiere realizar muchas más revisiones y pruebas antes que el sistema sea entregado al cliente. En esta fase aparecen nuevos cambios y se tiene que decidir si serán incorporados o no en dicha entrega.
- **Mantenimiento**: En esta fase por lo general se necesita un esfuerzo extra de los programadores para satisfacer los requerimientos del cliente.

1.1.1.2 Características Extreme Programming (XP)

Las características de la metodología Extreme Programming (XP) constituyen la esencia del proceso de desarrollo, a continuación se detalla a cada uno de ellos:

- **Pruebas Unitarias**: Se fundamenta en las pruebas efectuadas a los procesos más importantes, comprobando su correcto funcionamiento adelantándonos a posibles errores que podrían ocurrir en el futuro, asegurando la calidad del código entregado.
- **Re fabricación**: Se fundamenta en la reutilización de código, para lo cual se instauran patrones o modelos estándares, siendo más flexible al cambio.

- **Programación en pares**: Una característica de esta metodología es que plantea la programación en pares, la cual se fundamenta en que dos desarrolladores participen en un proyecto en una equivalente estación de trabajo. Cada miembro lleva a cabo la operación que el otro no está forjando en ese momento.

1.1.1.3 Roles

Los roles de la metodología Extreme Programming (XP) se detallan a continuación:

- **El Programador:** Escribe las pruebas y origina el código del sistema. Además define las tareas que conlleva cada historia de usuario y estima el tiempo que requerirá cada una.
- **El Cliente:** Escribe las historias de los usuarios y las pruebas funcionales para validar su implementación. El cliente da una preferencia a las historias de usuarios y resuelve cual implementar en cada iteración ajustándose en aportar mayor valor al negocio.
- **El encargado de pruebas:** Ayuda al cliente a escribir las pruebas funcionales, se encarga de ejecutar las pruebas con regularidad, además difunde las deducciones obtenidos al equipo y es responsable de las herramientas que dan soporte a las pruebas.
- **El encargado de seguimiento:** Es el que facilita la realimentación al equipo. Realiza el seguimiento del proceso de cada iteración e identifica el grado de acierto entre las estimaciones realizadas como también el tiempo real dedicado en ello para mejora de futuras estimaciones.
- **El entrenador:** Es el responsable del proceso integral. Se encarga de proporcionar guías al equipo de forma que se empleen las practicas XP y se vaya alcanzando el proceso correctamente.
- **El gestor:** Es el vínculo entre clientes y programadores, beneficia a que el equipo trabaje evidentemente creando las condiciones adecuadas, su labor principal es la coordinación.

- **Consultor:** Es un miembro externo del equipo con un conocimiento detallo en algún tema que es preciso para el proyecto, en el que tengamos inconvenientes. (Rayas, L. y J. L. Abreu, 2008, p. 720)

1.1.1.4 Reglas y practicas

1.1.1.4.1 Reglas y prácticas para la planificación

La metodología XP proyecta la planificación como un dialogo durable entre las partes involucradas en el proyecto, incluyendo al usuario, a los programadores y a los coordinadores o gerentes.

Las siguientes representaciones son primordiales de esta planificación:

- **Historia de usuarios:** Las Historias de usuarios o “User stories”, representan a los documentos de descripción funcional y a los casos de uso” Estas historias son escritas por el usuario, en su adecuado lenguaje, como descripciones cortas de lo que el sistema debe cumplir. La diferencia más importante entre estas historias y los tradicionales.

- **Plan de entregas:** El cronograma de entregas instituye qué historias de usuario serán concentradas para conformar una entrega, y el orden de las mismas. Este cronograma será la consecuencia de una reunión entre todos los actores del proyecto (cliente, desarrolladores, gerentes, etc.).

- **Plan de iteraciones:** Las historias de usuarios elegidas para cada entrega son desarrolladas y probadas en un ciclo de iteración, de acuerdo al orden prestablecido. Al inicio de cada ciclo se efectúa una reunión de planificación de la iteración.

- **Reuniones diarias de seguimiento:** El objetivo de realizar reuniones diarias es mantener la comunicación entre el equipo compartir inconvenientes y soluciones.

1.1.1.4.2 Reglas y prácticas para el diseño

La metodología XP hace respectivo énfasis en los diseños simples y claros. Las representaciones más significativas de diseño en esta metodología son los siguientes:

- **Simplicidad:** Un diseño simple que se efectúa más ligeramente que uno complejo.
- **Soluciones Spike:** Cuando surgen problemas técnicos, o cuando es dificultoso de estimar el tiempo para efectuar una historia de usuario, pueden utilizarse pequeños programas de prueba (llamados "spike"), para examinar diferentes soluciones.
- **Recodificación:** Consiste en escribir nuevamente fracción del código de un programa, sin cambiar su funcionalidad, a consecuencia de hacerlo más simple, conciso y/o entendible.
- **Metáforas:** Es algo que todos opinan, sin necesidad de mayores explicaciones. La metodología XP propone manejar este conocimiento como una manera natural de explicar el propósito del proyecto, guiar la estructura y arquitectura del mismo.

1.1.1.4.3 Reglas y prácticas para el desarrollo de código

Los conceptos más importantes para el desarrollo del código en esta metodología son los siguientes:

- **Disponibilidad del cliente:** Uno de los requerimientos de XP es tener al cliente disponible durante todo el proyecto. No simplemente como soporte a los desarrolladores, sino constituyendo parte del grupo. El involucramiento del cliente es primordial para que pueda desarrollarse el proyecto.
- **Uso de estándares:** La metodología XP comienza la programación establecida en estándares, de modo que sea fácilmente entendible por todo el equipo, y que proporcione la recodificación.

- **Programación dirigida por las pruebas**: En las metodologías tradicionales, la fase de pruebas, incluyendo la definición de los “test”, es comúnmente efectuada sobre el último del proyecto, o sobre el final del desarrollo de cada módulo.

- **Programación en pares:** La metodología XP propone que se desarrolle en pares de programadores, al trabajar en pares se empequeñecen los errores y se consiguen mejores diseños, ayudando la inversión en horas. El producto conseguido es por lo general de mejor calidad que cuando el progreso se efectúa por programadores individuales.

- **Propiedad colectiva del código:** Todo el equipo puede ayudar con nuevas ideas que utilicen a cualquier parte del proyecto. Además cualquier pareja de programadores puede cambiar el código que sea preciso para corregir problemas, adicionar funciones o recodificar.

- **Ritmo Sostenido:** La metodología XP indica que debe llevarse un ritmo sostenido de trabajo.

1.1.1.4.4 Reglas y prácticas para las pruebas

Los conceptos más importantes para las pruebas en esta metodología son los siguientes:

- **Pruebas Unitarias:** Los ensayos unitarios son una de las piedras angulares de la metodología XP. Todos los módulos deben pasar las pruebas unitarias antes de ser liberados o descubiertos.

- **Detección y corrección de errores:** Cuando se encuentra un error, éste debe ser corregido rápidamente, y se deben tener cautela para que errores relacionados no vuelvan a ocurrir. Igualmente se componen nuevas pruebas para comprobar que el error haya sido resuelto.

- **Pruebas de aceptación:** Las pruebas de aprobación son establecidas en base a las historias de usuarios, en cada período de la iteración del desarrollo. El cliente debe detallar uno o diversos escenarios para comprobar que una historia de usuario ha sido correctamente realizada.

Además las ventajas de metodología XP son las siguientes.

- Potenciar el trabajo en grupo durante el desarrollo de la aplicación.
- Entregar periódicamente avances del sistema.
- Realizar pruebas continuas hasta obtener la aplicación final.
- Los cambios que se realizan en la aplicación no deben aumentar con el tiempo.
- Realizar un desarrollo de manera eficiente, sencilla y clara; para facilitar el rendimiento y su actualización.
- Principios básicos.

Los principios básicos de esta metodología se representan en la siguiente figura 3.

Figura 3. Principios De La Metodología XP
Tomado de Ron Jeffries, 2001, http://xprogramming.com/book/whatisxp/

1.1.1.5 Mejore prácticas

- La metodología XP puede reutilizar el código al desarrollar un sistema unificado y entendible para todos los programadores, así como disponer de la información organizada y evitar errores o redundancias.

- La metodología XP es muy utilizada para desarrollar aplicaciones pequeñas o donde la aplicación deba ser actualizada constantemente.

- La metodología XP especifica que se debe realizar pruebas constantes para asegurar que la aplicación está realizando todas las funciones requeridas por el cliente y que no existan desviaciones en la aplicación, lo que permite reducir el riesgo de fallas que se puedan producir en ésta.

1.1.2 Lenguaje UML

Es el lenguaje Unificado de Modelado que permitirá realizar el diseño del software, reduce el costo y el tiempo empleado en la construcción de las piezas que constituyen el modelo del sistema; está compuesta por distintos diagramas, que representan las diferentes etapas del desarrollo de un proyecto. Es una herramienta estándar gráfica, fiable, eficiente y de calidad, que permite modelar, construir y documentar los elementos que forman un sistema software orientado a objetos.

1.1.2.1 Modelo, vistas y diagramas

- **Modelo:** Es una representación visual de los requisitos del usuario en el cual se describe completamente los aspectos y las funcionalidades del sistema automatizado.

- **Vistas:** Permite representar diferentes aspectos del sistema, cada uno contiene un conjunto de diagramas que ayudan a la construcción del sistema. El lenguaje Unificado de Modelado contiene las siguientes vistas:

    - Vista de caso de uso.
    - Vista estática.
    - Vista de implementación.
    - Vista de despliegue.
    - Vista de máquina de estados.
    - Vista de actividad.
    - Vista de interacción.
    - Vista de gestión del modelo.

- Diagramas: Permite representar gráficamente un conjunto de elementos que conforman el sistema que se desarrollará, el UML proporciona un conjunto de diagramas que se menciona a continuación.

(RUMBAUGH James, Jacobson I., Booch G., El Lenguaje de Modelado Manual de Referencia, 1999).

- Diagramas de casos de uso
- Diagrama de actividades
- Diagrama de clases
- Diagrama de objetos
- Diagramas de secuencia
- Diagramas de colaboración
- Diagramas de estado
- Diagrama de componentes
- Diagrama de despliegue

1.1.2.1.1 Vista de casos de uso

Capturan y describen los requerimientos funcionales del sistema mediante la relación entre un conjunto de actores que se encuentra vinculado con las funciones de casos de uso.

- **Diagramas de casos de uso:** Representan el comportamiento del sistema determina las características, límites y las relaciones entre el entorno además identifica los actores y sus diferentes roles, en la Figura 4 se muestra el esquema.

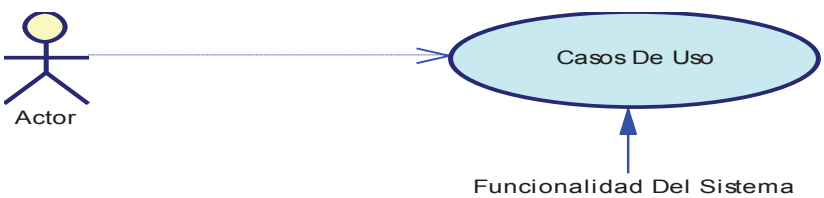


Figura 4. Diagramas de Casos de Uso.

En el diagrama de casos de uso intervienen los siguientes elementos.

- **Actor:** Un actor puede ser una persona, un proceso u otro sistema que interactúa dentro del entorno y se comunica con los casos de uso e intercambia mensajes.
- **Casos de uso:** Es una descripción que se produce entre un actor y el sistema para llevar a cabo un trabajo determinado.

1.1.2.1.2 Vista estática

La vista estática captura la estructura del objeto, describe las clases que forma el sistema y como se relacionan entre sí.

- **Diagrama de clases:** Permiten representar la estructura del sistema, el diseño conceptual mediante un conjunto de clases, objetos, interfaces que posee elementos que comparten en común, en la Figura 5 se muestra el diagrama. (Patricio Salinas, 30 de Octubre de1996).

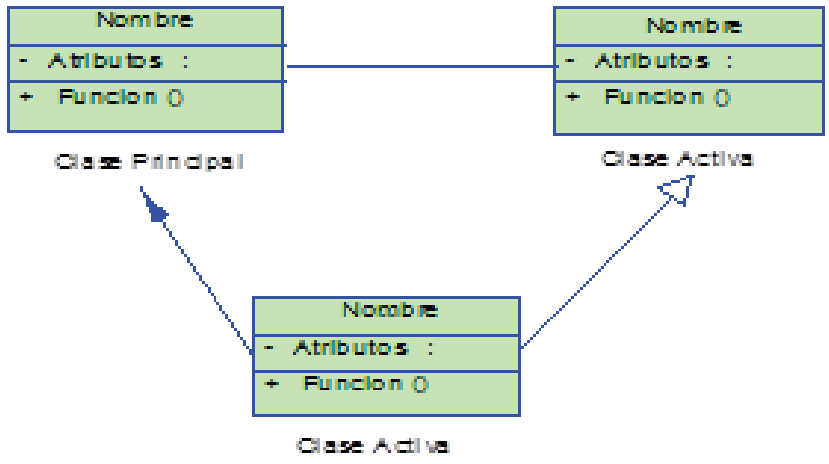


Figura 5. Diagrama De Clases.

En el diagrama de clases intervienen los siguientes elementos.

- **Nombre de clase:** Es una identificación única que posee cada clase para diferenciarse de las otras.

- **Atributo:** Es una propiedad, cualidad o características que posee la clase en el cual muestra su tipo y valor inicial.
- **Función:** Es una actividad, operación que realiza clase en el cual describe el comportamientos de los objetos.

- **Diagrama de objetos:** Describe la estructura de datos, muestra un subconjunto de los elementos del esquema de clases y sus relaciones entre ellos, este diagrama es diferente que el anterior porque en la parte superior va el nombre del objeto y de clases subrayado, en la Figura 6 se muestra la representación de los objetos.

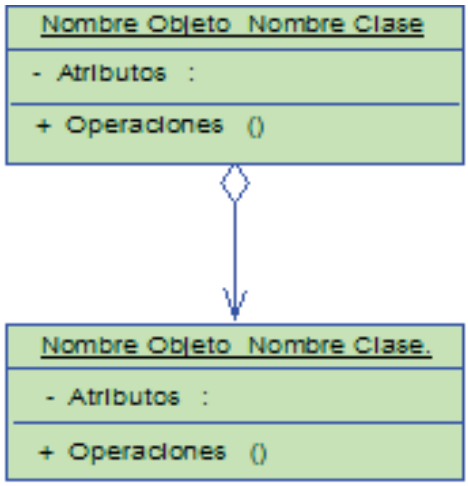


Figura 6. Diagrama De Objetos.

1.1.2.1.3 Vista de implementación

La vista de implementación muestra los elementos del diseño de sistema mediante los componentes.

- **Diagrama de componentes:** Los diagramas de componentes representan los componentes físicos del sistema cada uno de ellos realiza su propia función y se encuentra relacionados entre ellos, constituyen el empaquetamiento físico de los elementos lógicos y son aquellos que participan en la ejecución del sistema. En la Figura 7 se muestra el diagrama de componentes. (Pablo Figueroa, 1997).

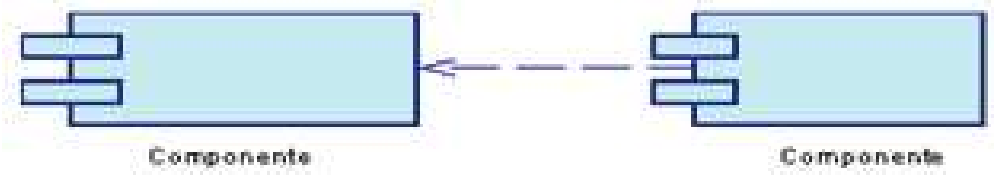


Figura 7. Diagrama De Componentes.

En el diagrama de componentes interviene el siguiente elemento:

- Nombre de clase: Es una identificación única que posee cada clase para diferenciarse de las otras.
- Componente: Es una parte física remplazable de un sistema. Puede estar formado por interfaces, clases y otros elementos.

1.1.2.1.4 Vista de despliegue

La vista de despliegue muestra los recursos de ejecución.

- **Diagrama de despliegue:** El diagrama de despliegue muestra las relaciones físicas entre los elementos, cada uno de los dispositivos se representa como un nodo que se utiliza para realizar la conexión con los demás elementos que forman parte del sistema. En la Figura 8 se muestra el esquema de despliegue. (CRAIG Larman, 2007).

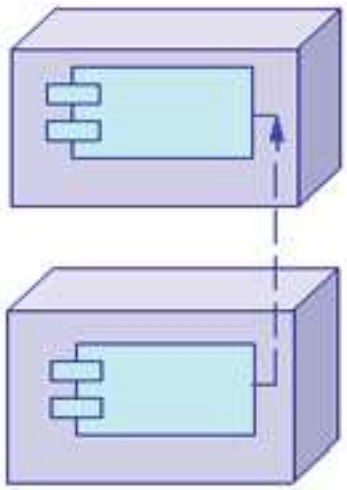


Figura 8. Diagrama De Despliegue.

En el diagrama de despliegue interviene los siguientes elementos:

- **Nodo:** Representa un cubo y es un elemento donde se ejecutan los elementos.
- **Componente:** Es la parte física del software, son los elementos que participan en la ejecución de un sistema.
- **Conexión:** Enlace entre los nodos.

1.1.2.1.5 Vista de máquina de estados

La vista de máquinas de estados describe la vida de un objeto y su comportamiento, se los representa en los esquemas de estados.

- **Diagrama de estados:** Los diagramas de estados muestran la secuencia de un objeto a lo largo de su vida mostrando los sucesos que realiza y las acciones que generan en el cual engloba todos los mensajes que un objeto logra enviar o recibir. En la Figura 9 se muestra el esquema de estados.

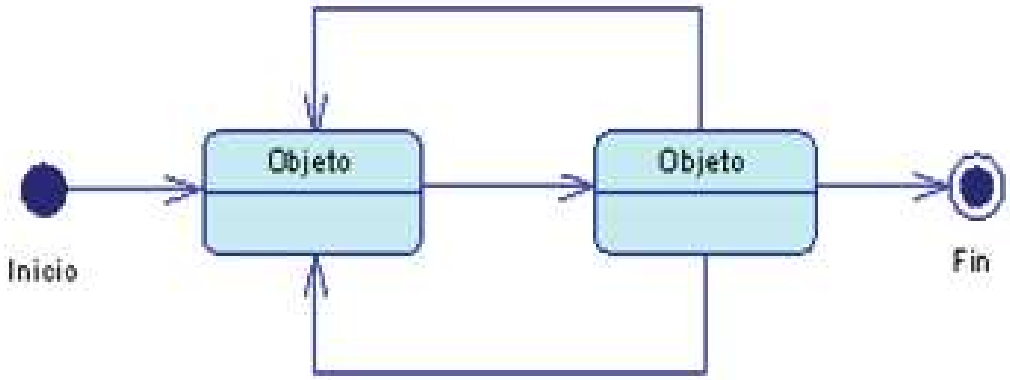


Figura 9. Diagrama De Estados.

En el diagrama de estado interviene los siguientes elementos.

- **Transición:** Es una relación entre dos estados que indican que un objeto en el primer estado puede entrar al segundo estado y ejecutar ciertas operaciones.

- **Objeto:** Se representa en un rectángulo y es una entidad que tiene un estado.
- **Estado:** Representa un conjunto de atributos.

1.1.2.1.6 Vista de actividades

La vista de máquina de actividades representa el flujo de acciones del sistema.

- **Diagrama de actividades:** En el diagrama de actividades muestra el comportamiento de ejecución del sistema. En la Figura 10 se muestra el esquema de actividades.

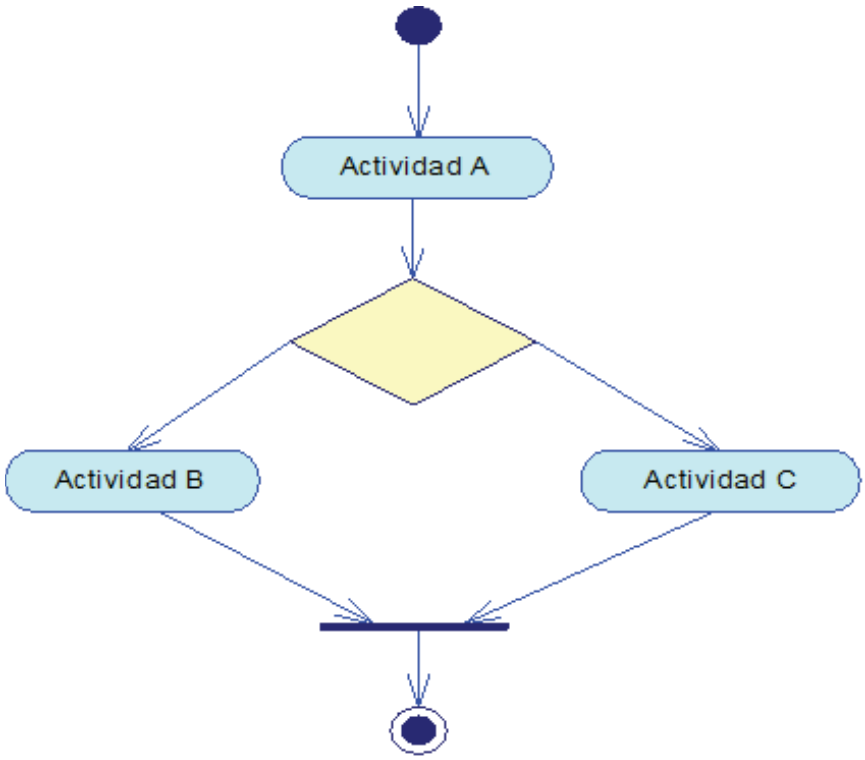


Figura 10. Diagrama De Actividades.

En el diagrama de actividad interviene los siguientes modelos:

- **Inicio:** El inicio de un diagrama de actividad.
- **Actividad:** Representa la acción que será realizado por el sistema.
- **Fin:** Término de un diagrama de actividad.

1.1.2.1.7 Vista de interacción

La vista de interacción describe la secuencia de intercambio de mensajes entre los roles que implementa el comportamiento del sistema se representa en los diagramas de secuencias y colaboración.

(JACOBSON Ivar, Booch G., Rumbaugh J, 2000).

- **Diagrama de secuencia:** El diagrama de Secuencia muestra el conjunto de mensajes entre los diferentes objetos describen entidades y actores que participan en el sistema y representa la existencia a lo largo de un período de tiempo. En la Figura 11 se muestra el diagrama de secuencia.

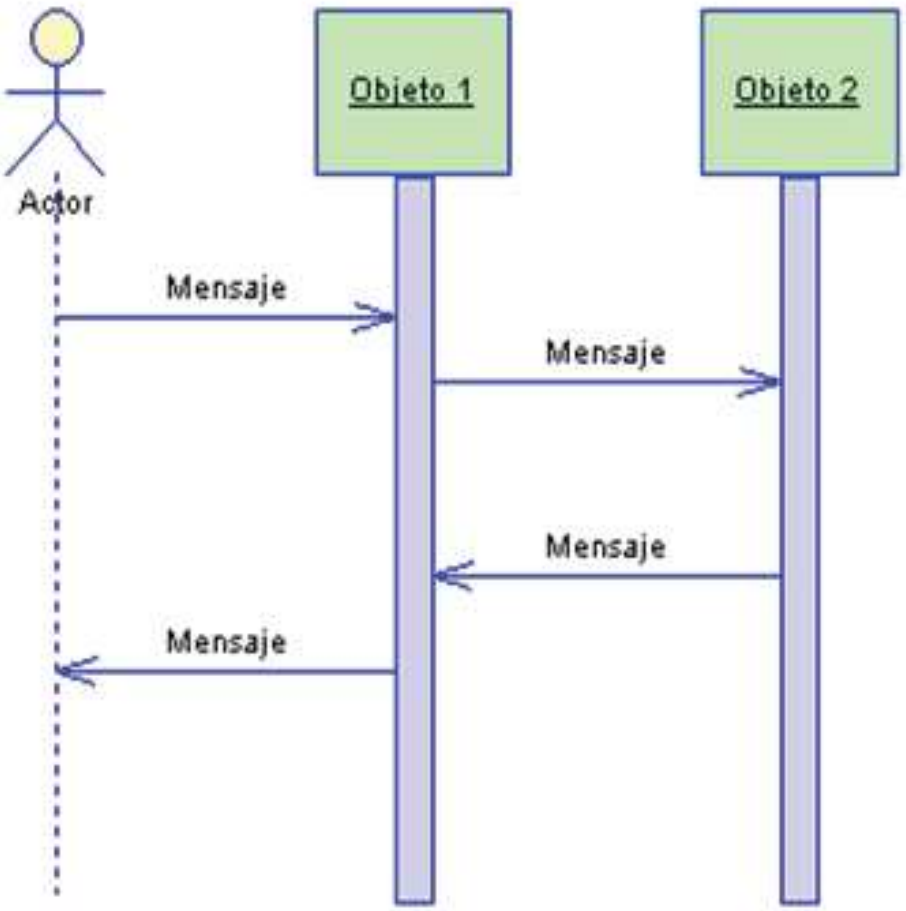


Figura 11. Diagrama De Secuencia.

En el diagrama de secuencia interviene los siguientes elementos.

- **Interacción:** Conducta que logra un conjunto de mensajes intercambiados entre un conjunto de objetos dentro de un argumento para alcanzar un propósito.
- **Mensajes:** Comunicación entre objetos que trasmiten información con la perspectiva de desencadenar una actividad.
- **Objetos:** Es una entidad que tiende un estado que representa un conjunto de atributos.
- **Rol:** Es la descripción de un objeto.

- **Diagrama de colaboración:** El diagrama de colaboración muestran un conjunto de objetos, enlaces y mensajes, se representa en cajas rectangulares para sus mensajes utilizan líneas que define una secuencia, en la Figura 12 se muestra el diagrama de colaboración.

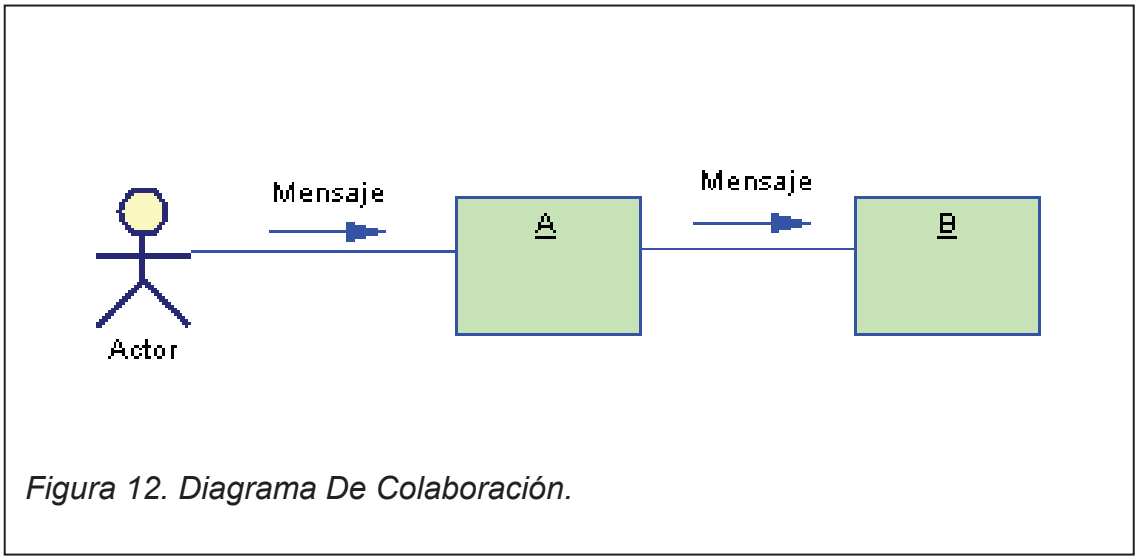


Figura 12. Diagrama De Colaboración.

En el Diagrama de Colaboración interviene los siguientes elementos.

- **Mensaje:** Comunicación entre objetos que trasmiten información.
- **Enlace:** Línea que indica relación estructural entre objetos.

Además la vista de gestión de modelo describe en forma visual los requisitos que utiliza el sistema están representados por un conjunto de paquetes que proporcionan servicios y se encuentra relacionados entre sí.

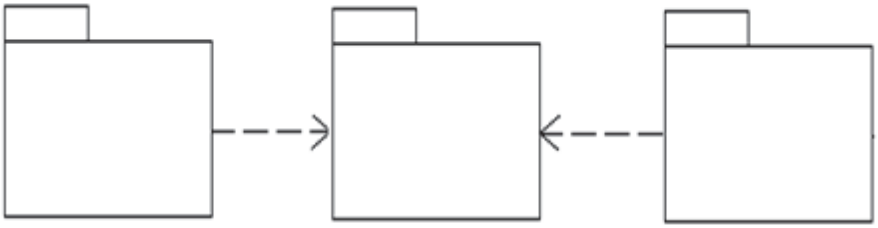


Figura 13. Gestión Modelo.

A continuación se muestra los diagramas de componentes los cuales identifican dos opciones:

- Sistema
- Base de datos

Figura 14. Diagrama De Paquetes.

En seguida se define el diagrama de componentes los cuales esta identificados por dos paquetes los cuales abarcan cada uno de ellos diferentes elementos.

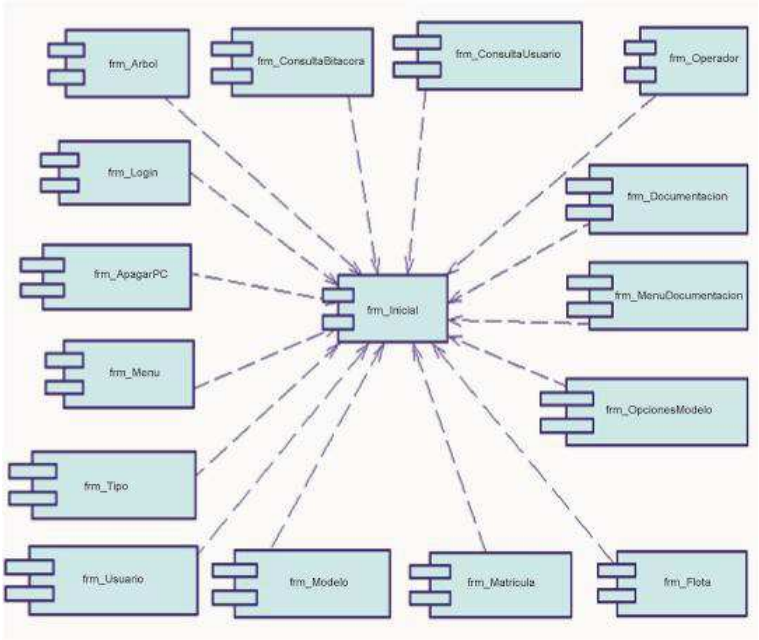


Figura 15. Diagrama De Componentes Del Sistema.

Además se puede identificar los componentes de la base de datos, identificando cada una de las tablas.

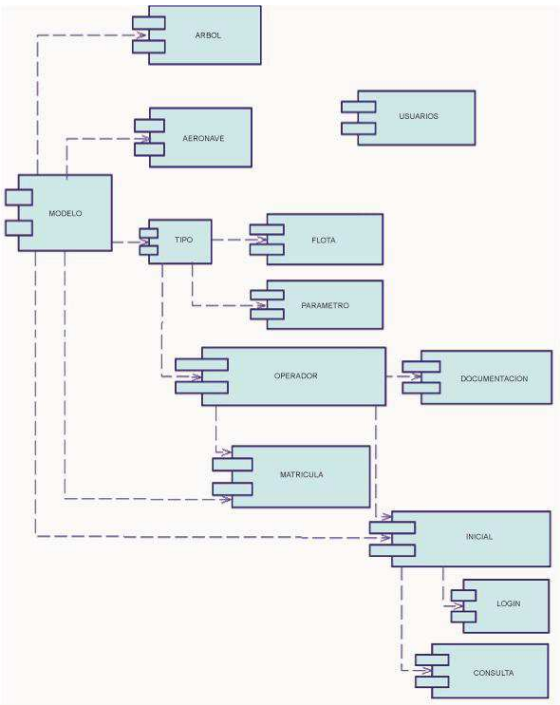


Figura 16. Diagrama De Componentes De Base De Datos.

Una vez finalizado los componentes se identifican la arquitectura del sistema que consta de dos paquetes, que es primordial para identificar las ventanas y tablas correspondientes de la base de datos.

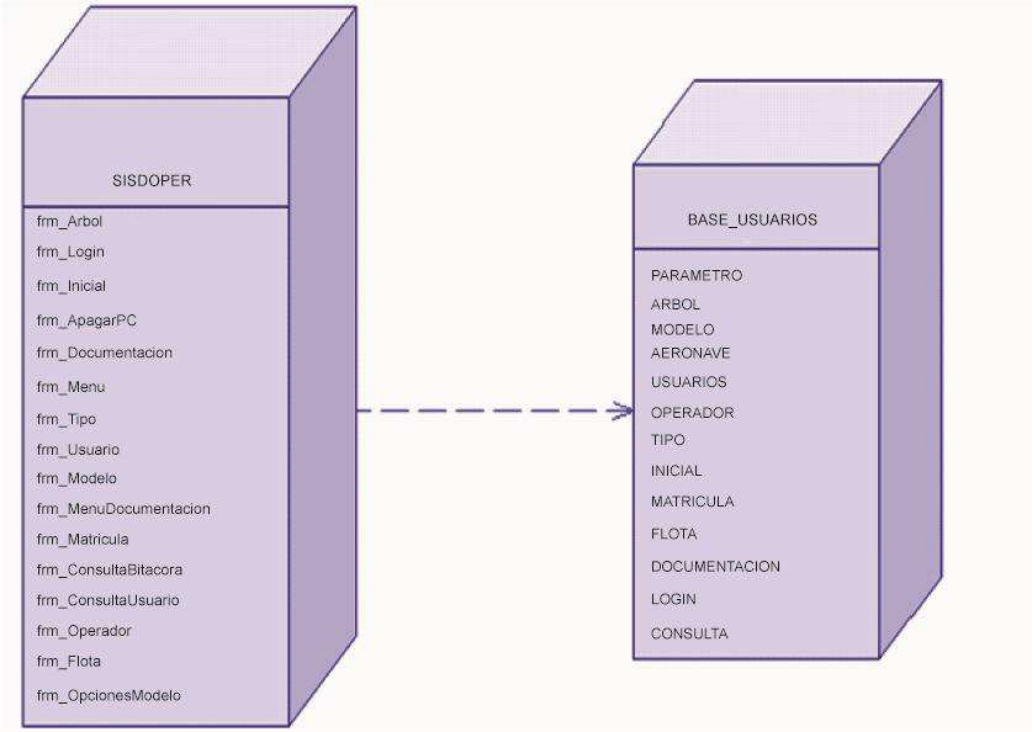


Figura 17. Arquitectura Del Sistema.

1.1.3 Base de datos

Las bases de datos son recursos que recopilan todo tipo de información, para atender las necesidades de un amplio grupo de usuarios. Su tipología es variada y se caracterizan por una alta estructuración y estandarización de la información. (Mariana Pinto, 26 de abril del 2011).

Además Una base de datos se define como un conjunto de datos relacionados que se almacenan de forma que se pueda acceder a ellos de manera sencilla, el objetivo principal es unificar toda la información del sistema para evitar redundancias además los datos se organizan y mantienen en un conjunto estructurado, permite almacenar grandes cantidades de datos y posee herramientas de consultas.

1.1.3.1 Microsoft Access

Es un software manipulado para el desarrollo de sistemas de Base de Datos, basado en el argumento de las bases de datos relacionales, en el cual el usuario conserva informaciones fundadas de forma tabular.

Además Microsoft Access crea un archivo de extensión .lbd. Este archivo tiene beneficio en habientes multiusuario donde una misma Base de datos puede estar siendo accedida paralelamente por varios usuarios.

También los sistemas de bases de datos se diseñan para manipular grandes cantidades de información. El manejo de datos envuelve tanto la definición de las estructuras para el almacenamiento de la información como los mecanismos para el manejo de la misma.
Algunas de las ventajas de usar las bases de datos son:

- Evitan la redundancia.
- Evitan la inconsistencia.
- Obligan al cumplimiento de las normas o requisitos para la adición y eliminación de datos a la base de datos.
- Es posible aplicar restricciones de seguridad para el acceso a datos.
- Se mantiene la integridad entre los datos.

Finalmente el sistema de gestión de bases de datos, contiene una colección de datos relacionados que permite de forma práctica y eficiente almacenar información en una base de datos.

Figura 18. Entorno De Microsoft Access.

1.1.3.2 Lenguaje de alto nivel

Un lenguaje de programación de alto nivel se caracteriza por expresar los algoritmos de una manera ordenada a la capacidad cognitiva humana además genera un código más natural y comprensible es válido para diversas máquinas.

Es más práctico y fácil de manipular para el programador puede concentrarse más en el programa en sí que en el lenguaje y, por otra parte, se reducen los tiempos de creación del programa, incluso en caso de que tenga que hacer modificaciones, son mucho más fáciles de hacer.

Otra característica importante de los lenguajes de alto nivel es que, para la mayoría de las instrucciones de estos lenguajes, se necesitarían varias instrucciones en un lenguaje ensamblador para indicar lo mismo. De igual forma que, la mayoría de las instrucciones de un lenguaje ensamblador, también agrupa a varias instrucciones de un lenguaje máquina (Carlos Javier Pes Rivas, 2002).

1.1.3.3 Lenguaje de Microsoft C#

C# leído es “C Sharp” y en español “C Almohadilla”, es el nuevo lenguaje de propósito frecuente diseñado por Microsoft para su plataforma .NET. Sus primordiales creadores son Scott Wiltamuth y Anders Hejlsberg.

(José Antonio González Seco, 2001).

C# es un lenguaje de propósito general que puede ser utilizado para la construcción de aplicaciones web, aplicaciones de escritorio, servicios web, aplicaciones para celulares y componentes. (Gabriel Bulfon, 2011).

La sintaxis y estructuración de C# es muy parecida a la de C++ o Java, puesto que la intención de Microsoft es proporcionar la migración de códigos escritos en estos lenguajes a C# y proveer su instrucción a los desarrolladores. Sin embargo, su simplicidad y el alto nivel de productividad son semejantes con los de Visual Basic. (José Antonio González Seco, 2001).

Las características del C#

- **Sencillez:** Elimina muchos elementos que otros lenguajes contienen y que son innecesarios en .NET. Por ejemplo:
  - El código escrito en C# es auto contenido, lo que representa que no requiere de ficheros agregados al propio fuente tales como ficheros de cabecera o ficheros IDL.
  - La dimensión de los tipos de datos básicos es fijo e independiente del compilador, sistema operativo o máquina para quienes se compile (no como en C++), lo que proporciona la portabilidad del código.
  - No se incluyen componentes poco útiles de lenguajes como C++ tales como macros, herencia múltiple o la necesidad de un operador desigual del punto (.) acceder a miembros de espacios de nombres (::).
- **Modernidad:** Incorpora en el adecuado lenguaje componentes que a lo largo de los años ha ido manifestándose son muy ventajosos para el desarrollo de aplicaciones y que en otros lenguajes como Java o C++ hay que simular, como un tipo básico decimal que permita efectuar operaciones de alta precisión con reales de 128 bits (muy útil en el mundo financiero).

  La inclusión de una instrucción foreach que permita recorrer colecciones con facilidad y es prolongable a tipos definidos por el usuario, la introducción de un tipo básico string para incorporar cadenas o la distinción de un tipo bool determinado para incorporar valores lógicos.

- **Orientación a objetos:** Es más puro en tanto que no aceptan ni funciones tampoco variables globales sino que todo el código y datos han de precisarse dentro de tipos de datos, lo que reduce inconvenientes por conflictos de nombres y proporciona la legibilidad del código.

También C# soporta todas las características propias del paradigma de programación orientada a objetos:

  - Encapsulación
  - Herencia
  - Polimorfismo.

- **Orientación a componentes:** La propia sintaxis de C# contiene componentes propios del diseño de componentes que otros lenguajes poseen que simular mediante construcciones más o menos complicadas.

- **Seguridad de tipos:** C# contiene mecanismos que permiten afirmar que los accesos a tipos de datos siempre se efectúen correctamente, lo que evita que se causen errores difíciles de manifestar por acceso a memoria no perteneciente a ningún objeto y principalmente necesario en un entorno gestionado por un recolector de basura.

- **Eficiente:** El código de C# encierra numerosas limitaciones para afirmar su seguridad y no permite el uso de punteros, en C# es permitido saltarse dichas restricciones manipulando objetos a través de punteros.

- **Compatible:** Para proveer la migración de programadores, C# no sólo mantiene una sintaxis muy similar a C, C++ o Java, que consiente incluir verdaderamente en código escrito en C# fragmentos de código escrito en estos lenguajes. (José Antonio González Seco, 2001).

C# combina los mejores elementos de múltiples lenguajes de amplia difusión como C++, Java, Visual Basic o Delphi. De hecho, su creador Anders Heljsberg

fue también el creador de muchos otros lenguajes y entornos como Turbo Pascal, Delphi o Visual J++. La idea principal detrás del lenguaje es combinar la potencia de lenguajes como C++ con la sencillez de lenguajes como Visual Basic, y que además la migración a este lenguaje por los programadores de C/C++/Java sea lo más inmediata posible. (José Antonio González Seco, 11 de Octubre del 2001).

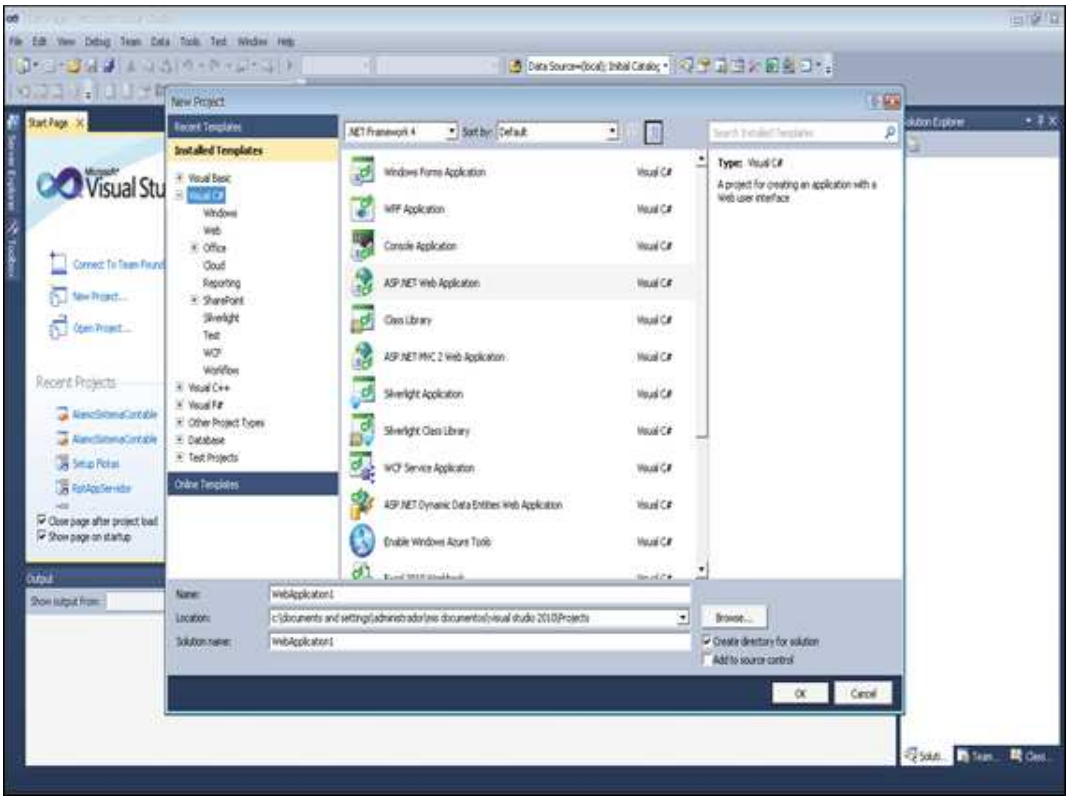


Figura 19. Entorno De C#.

Finalmente C# es un lenguaje de programación que toma las excelentes características de lenguajes prexistentes como Visual Basic, Java o C++ y las combina en uno solo.

1.1.3.4 Proceso de digitalización

Los procesos de digitalización, se fundamenta en la transformación de documentos físicos a imágenes digitales para su visualización instantánea a partir de cualquier equipo de cómputo.

Actualmente la digitalización es la herramienta de preservación más completa y agradable que nos brinda la tecnología, se debe tomar en consideración que el proceso de digitalización inicia con la depuración de los documentos.

El resultado de la digitalización es el archivo digital, el cual puede ser migrado a un gestor documental para consulta y búsqueda de información.

La información que se puede digitalizarse son:

- Documentos
- Planos
- Libros
- Fotos
- Mapas, etc.

En referencia a seguridad informática para la digitalización se debe tomar en consideración los siguientes puntos:

- Firmas digitales.
- Administración de registros documentarios.
- Seguridad en asistencia de procesos y procedimientos.
- Seguridad en los niveles de autorización del acceso a la información.
- Registro de uso de las líneas de producción.
- Control y registro de usuarios.

En cambio las tendencias para la digitalización que se debe tomar en consideración son los siguientes:

- Niveles de seguridad y encriptación.
- Protección de la información que solo debe visualizarse.
- Seguimientos y flujos de trabajo.

Figura 20. Proceso De Digitalización
Tomado de Rima Natural, (2012), http://www.rimanatural.mx/proceso.html

En definitiva un proceso de digitalización es una regla o conjunto de procedimientos que precisan el modo en que se digitaliza un documento, donde y a quien se le entrega en la red, y que usuarios o grupos puede aplicar la regla a los documentos digitalizados.

1.1.3.5 Sistemas electrónicos de gestión documental o sistemas de gestión de documentos electrónicos.

Un sistema de gestión de documentos electrónicos es un sistema computarizado, un conjunto de programas, utilizado para indagar y acumular documentos electrónicos o imágenes digitales de documentos inicialmente soportados en papel.

El término puede ser relacionado con nociones como sistemas de administración de contenido (CMS) y es usualmente visto como un sistema de administración de contenido corporativo y relacionado con la palabra Digital Asset Management.

Mientras que el flujo del sistema de gestión documental, alcanza los siguientes componentes:

- **Hardware**

  Escáneres y dispositivos de Digitalización: Los documentos originales, una vez preparados, son reformados en documentos digitales, los cuales estarán guardados o almacenados.

  Servidores: Contienen la información anticipadamente digitalizada. Los usuarios finales se enlazaran a un servidor para poder permitir a dicha información, para su modificación o revisión.

Figura 21: Dispositivos De Hardware.
Tomado de Unidad Educativa Nacional Bolivariana "Atapaima", 3 de Febrero del 2011, http://uenbatapaima1903.blogspot.com/2011/02/hardware-y-software.html

- **Software**

  Gestores documentales: Programas de soporte al proceso de gestión de la documentación que se maneja en la empresa. Existen soluciones informáticas para: la gestión documental, administración de archivos digitales y para administración de bibliotecas de diversa índole.

- **Usuarios**

  A través de una cuenta de Usuario, se lleva a cabo el acceso a los documentos digitalizados dentro del Sistema de Gestión Documental admitiendo así, efectuar la consulta electrónica de los mismos de acuerdo a los niveles de seguridad determinados a cada uno de los usuarios registrados en el Sistema.

- **Administradores**

  Los documentos digitalizados se codifican e indexan en la base de datos del sistema, igualando la ubicación física del documento original y determinando a cada documento lógico, las claves de acceso. Los documentos digitalizados se acumulan en su oportuno fichero.

1.1.3.6 Herramienta case

Las herramientas CASE de modelado como UML nos admiten aplicar la metodología de análisis y diseño orientados a objetos nos logran ayudar en todos los aspectos del ciclo de vida de desarrollo del software en tareas como el proceso de efectuar un diseño del proyecto.

1.1.3.7 SAP Sybase PowerDesigner

El programa PowerDesigner, nos proporciona un conjunto de herramientas estándar como procesos de negocios, datos conceptuales, modelos orientados a objetos y datos físicos además posee una interfaz gráfica altamente extensible que define el modelamiento UML de objetos también provee técnicas de análisis y diseño que permiten generar reportes completos.

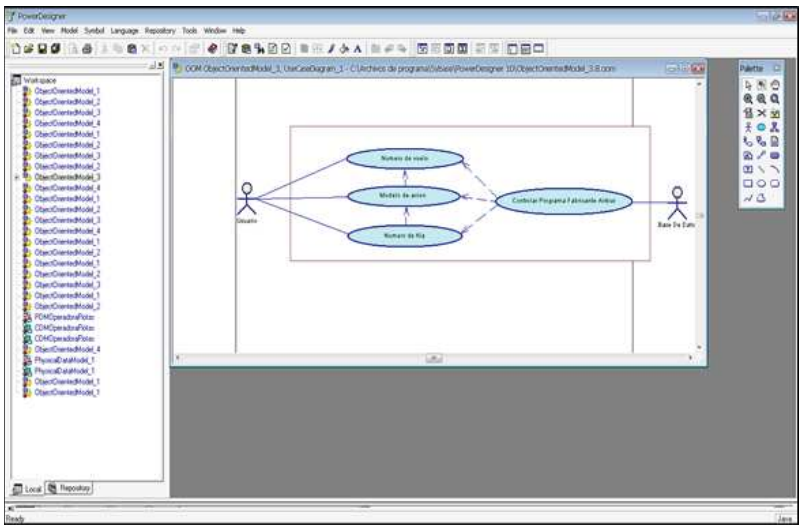


Figura 22. Entorno Power Designer.

2. DISEÑO Y DESARROLLO SISTEMA DE DOCUMENTACION OPERACIONAL

2.1 Presentación

El proceso de desarrollo del proyecto será aplicado utilizando la metodología Extreme (Programming XP), la misma que servirá para controlar y gestionar la implementación del sistema a través de sus fases, otro elemento esencial es el lenguaje UML el mismo que se usará para realizar el diseño del sistema además se obtendrá una base de datos que ayudará a tener rapidez y seguridad en el ingreso a la información.

2.2 Objetivo

- Se requiere desarrollar el sistema de documentación operacional utilizando tecnologías de información y comunicación, para el manejo de información a través de procedimientos automáticos.
- Se requiere diseñar una interfaz gráfica de manera que permita gestionar y organizar adecuadamente el manejo documental.

2.3 Descripción

La Ingeniería de Software es un enfoque sistemático de desarrollo la cual proporciona estabilidad, control y organización en las actividades además comprende la tecnología multicapas que contiene el proceso, métodos técnicos y herramientas automatizadas, el proceso de ingeniería aplicado en el presente trabajo aplica los criterios de la metodología Extreme Programming(XP), está constituido por las siguientes fases que son planificación, diseño, codificación y pruebas, el proceso de gestión de fases se describen en los puntos posteriores.

2.4 Fase de Planificación

En esta fase los clientes realizan las storycards que describe una de las funcionalidades que el programa tendrá además sirven para registrar los requerimientos de los clientes en base a lo que se estima es necesario para el sistema.

2.4.1 Sentencia que define el problema

En la tabla 1 se define la información sobre el problema del manejo de los manuales en forma manual.

Tabla 1. Manejo de los manuales en forma manual.

MANEJO DE LOS MANUALES	
El problema de	Manuales operacionales en forma manual.
afecta a	Al personal aéreo
El impacto asociado es	Tiempo de espera del personal aéreo.
una adecuada solución sería	El sistema automatizará los procesos operacionales de los manuales.

En la tabla 2 se define la información sobre el problema de la consulta de información en los manuales.

Tabla 2. Definición del problema lentitud en la consulta.

CONSULTA DE INFORMACION EN LOS MANUALES	
El problema de	Lentitud en la consulta de los manuales operacionales.
afecta a	Al personal aéreo
El impacto asociado es	Pérdida de tiempo en la búsqueda de datos.
una adecuada solución sería	El sistema automatizará el proceso y agilitara la búsqueda mediante consultas específicas.

En la tabla 3 se define el ingreso a la información por persona no autorizadas.

Tabla 3. Ingreso A La Información Por Personas No Autorizadas.

INGRESO A LA INFORMACION POR PERSONAS NO AUTORIZADAS	
El problema de	El ingreso a la información de manuales por personas no autorizadas.
afecta a	Al personal aéreo, La línea aérea.
El impacto asociado es	Riesgos de pérdida de información.
una adecuada solución sería	El sistema automatizará los perfiles de usuario para acceder a los datos.

2.4.2 Historias de usuarios(Stakeholders)

Para abastecer de una forma segura productos y servicios que se ajusten a las necesidades de los usuarios, es obligatorio identificar e involucrar a todos los colaboradores en el proyecto como parte del proceso de modelado de requerimientos. Igualmente es obligatorio igualar a los usuarios del sistema y afirmarse de que el conjunto de colaboradores en el proyecto los represente apropiadamente.

Esta sección muestra un perfil de los contribuyentes y de los usuarios implicados en el proyecto, así como los inconvenientes más importantes que éstos descubren para orientar la solución propuesta hacia ellos.

Representantes Del Proyecto

En esta tabla 4 se identificará las personas que formará parte del desarrollo del software durante su construcción, a continuación se describirá a cada uno de ellos.

Tabla 4. Stakeholders del Proyecto.

Nombre	Descripción	Responsabilidades
Daniel Jurado	Estudiante de la carrera de Ingeniería en Sistemas e Informática.	El “stakeholder” realiza: Levantamientos De Requerimientos. El análisis, diseño y documentación del sistema.
Ing. Fernando Solís	Ingeniero en Sistemas; tutor designado para el desarrollo del proyecto.	El “stakeholder” realiza: Control y seguimiento en el desarrollo del proyecto.

Representantes Del Sistema

En esta tabla se identificará las personas que formará parte en el funcionamiento del sistema, en donde consta el nombre la descripción y la responsabilidad, a continuación se detallará a cada uno de ellos en la tabla 5.

Tabla 5. Stakeholders del sistema.

Nombre	Descripción	Responsabilidades
Administrador	Privilegio con acceso total a las funciones del sistema documental	El “Stakeholder” realiza: El administrador en el Sistema puede crear, modificar, eliminar, actualizar, consultar datos de las aeronaves y sus procesos además crear usuario para el ingreso del sistema.
Piloto	Persona encargada de llevar el control de los manuales de las aeronaves y programas en su totalidad.	El “Stakeholde” realiza: Interactúa con el sistema para el ingreso y búsqueda de datos en el manual de operaciones.
Auxiliar de Vuelo	Persona que van a visualizar manuales de las aeronaves y programas del fabricante.	El “stakeholder” realiza: Visualiza y controla el plan de programas del fabricante.

2.4.2.1 Perfil de “stakeholders”

A continuación se detalla la información sobre la persona que se encarga del desarrollo y construcción del sistema.

Tabla 6. Representante de desarrollo del área técnica.

REPRESENTANTE DEL DESARROLLO Y CONSTRUCCION DEL SISTEMA	
Representante	Daniel Jurado
Descripción	Estudiante de la Universidad De Las Américas.
Tipo	Área de Sistemas.
Responsabilidades	Levantamiento de requerimientos, análisis, diseño y documentación del sistema.
Criterio de Éxito	A definir
Grado de participación	Se encargará del desarrollo y la construcción del Sistema.
Comentarios	Ninguno

En la tabla 7 se describe la información referente a la persona que controla el desarrollo del proyecto y la aprobación de requisitos y funcionalidades del sistema.

Tabla 7. Representante de control del área técnica.

REPRESENTANTE DEL CONTROL EN EL DESARROLLO DEL PROYECTO	
Representante	Fernando Solís.
Descripción	Ingeniero de la Universidad De Las Américas y tutor del proyecto.
Tipo	Área de Sistemas.
Responsabilidades	Control, seguimiento en el desarrollo del proyecto, aprobación de requisitos y funcionalidades del sistema.
Criterio de Éxito	A definir
Grado de participación	Revisará requerimientos y estructura del sistema.
Comentarios	Ninguno

En la tabla 8 se describe la información referente al perfil de usuario que representa el administrador.

Tabla 8. Administrador.

PERFIL USUARIO ADMINISTRADOR	
Representante	Administrador
Nombre Historia	Control del funcionamiento del sistema
Prioridad en negocio	Alta
Descripción	Persona que maneja el Sistema con privilegios de acceso total a sus funciones.
Tipo	Área De Sistemas
Responsabilidades	El administrador en el Sistema puede crear, modificar, eliminar, actualizar y consultar datos del paciente además crear usuario para el ingreso del sistema.
Criterio de Éxito	A definir
Grado de participación	Usuario que utilizará el sistema con acceso total.
Comentarios	Ninguno.

En la tabla 9 se define la información relacionado con el perfil de usuario piloto.

Tabla 9. Usuario Piloto

PERFIL DE USUARIO PILOTO	
Representante	Piloto(Usuario del sistema)
Descripción	Persona encargada del control de los manuales y programas.
Nombre Historia	Control del manual de las aeronaves
Prioridad en negocio	Alta
Tipo	Piloto Comercial
Responsabilidades	Interactúa con el sistema para el ingreso y búsqueda de datos en el los manuales operacionales.
Criterio de Éxito	A definir.
Grado de participación	Usuario que se encarga del control de los manuales de las aeronaves y programas,
Comentarios	Ninguno.

En la tabla 10 se define la información relacionado con el perfil de usuario auxiliar.

Tabla 10. Usuario Auxiliar.

PERFIL DE USUARIO AUXILIAR DE VUELO	
Representante	Auxiliar de Vuelo (Usuario del sistema)
Descripción	Persona encargada de visualizar, organizar y controlar los manuales.
Nombre Historia	Organización del manual de las aeronaves
Prioridad en negocio	Alta
Tipo	Piloto Comercial
Responsabilidades	Interactúa con el sistema para visualizar los manuales de operacionales.
Criterio de Éxito	A definir.
Grado de participación	Usuario que se encarga del control y organización de los manuales de las aeronaves y programas,
Comentarios	Ninguno.

2.4.3 Descripción global del producto

El Software gestionará información de los manuales de documentación en forma ordenada, facilitando el consultar datos de manera rápida y especifica adquiriendo información actualizada que obtendrá integridad a través de roles del usuario generando la visualización de manuales del fabricante, de la empresa Lan Ecuador además visualizar programas del fabricante Airbus y Boeing.

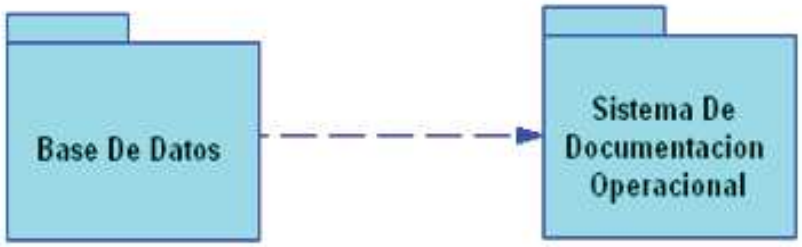


Figura 23. Esquema del sistema.

2.4.3.1 Característica del producto

El cumplimiento de las características del sistema será apoyado de la siguiente forma:

Gestionar la información los manuales operacionales del avión

La gestión de información, permitirá ingresar datos de manera automática obteniendo información organizada, facilitando la administración de datos, logrando tener una mejor visualización de los manuales operacionales.

Crear cuentas y roles de usuarios para el acceso al sistema

La creación de usuarios, proporcionará confidencialidad a la información, logrando tener integridad en los datos.

Automatizar el proceso de consulta

La automatización del proceso de consulta, permitirá obtener información específica, rápida y veraz.

Generar reportes

La generación de reportes, ayudara a tener información completa la cual permitirá obtener datos precisos y correspondientes de sus diferentes funciones.

2.4.3.2 Requerimientos específicos

2.4.3.2.1 Requerimientos de interfaces externas

Los requerimientos de interfaces externas permiten definir el entorno del producto del software con el usuario, a continuación se describen cada uno de las interfaces.

Interfaces de usuario

Las interfaces de usuario permitirán la iteración entre el usuario y el sistema, a continuación se describe las diferentes ventanas que tendrá el sistema.

- Ventana de contraseña
- Ventana de documentación operacional
- Ventana de manuales de avión
- • Ventana de manuales de compañía
- • Ventana de mantenimientos
- • Ventana de usuarios
- • Ventana de bitácora

2.4.3.3 Diagramas de casos de uso de sistemas

2.4.3.3.1 Casos de uso del negocio

Los diagramas de casos de uso del negocio permiten describir el ambiente del problema en estudio, en los apartados posteriores se describe el dominio de estudio realizado para la definición del proyecto.

Identificación de actores del negocio

Los principales actores que permiten definir el dominio de gestión de proyecto, se describe en la tabla.

Tabla 11. Actores del negocio.

Actor	Descripción
Pilotos	Se encarga de la visualización de la documentación operacional.
Auxiliar de vuelo	Se encarga de controlar los programas y de los reportes de datos.

Diagrama de negocio definición del sistema (Gestionar manuales)

Figura 24. Esquema del sistema.

Descripción:

- El piloto se encarga del manejo del sistema en el cual gestiona los manuales de avión y de la compañía.
- La base de datos guardar la información respectiva a los manuales en el cual se puede realizar diferente tipo de búsquedas.

2.5 Fase de Diseño

En esta fase se analizan los requisitos y se desarrolla un prototipo de arquitectura incluyendo las partes del sistema además se implementa todos los casos de uso correspondientes.

2.5.1 Caso de uso del sistema

Los diagramas de casos de uso del sistema permiten definir los requisitos funcionales del producto de software, con respecto a los usuarios que lo utilicen, a continuación se describe los casos de uso.

2.5.1.1 Identificación de actores del sistema

Para el desarrollo del sistema se utilizaron los actores los cuales desempeñan sus roles, el objetivo del sistema es automatizar los procesos que se realizan, los actores y sus roles se observa en la tabla.

Tabla 12. Actor y roles del sistema.

Actor del Negocio	Actor del Sistema	Descripción
Piloto o Auxiliar de vuelo	Usuario del Sistema acceso limitados	Persona que utiliza los servicios del sistema.
Usuario con permisos totales	Administrador	Persona que gestiona sus actividades y administra el sistema
Base De Datos		Interface del software donde se almacena la información.

2.5.1.2 Caso de uso del administrador

Figura 25. Caso de uso del administrador.

Descripción:

- El administrador accede al sistema para crear, modificar, eliminar, actualizar y consultar la documentación operacional.

El administrador y el usuario del sistema pueden acceder al menú de documentación

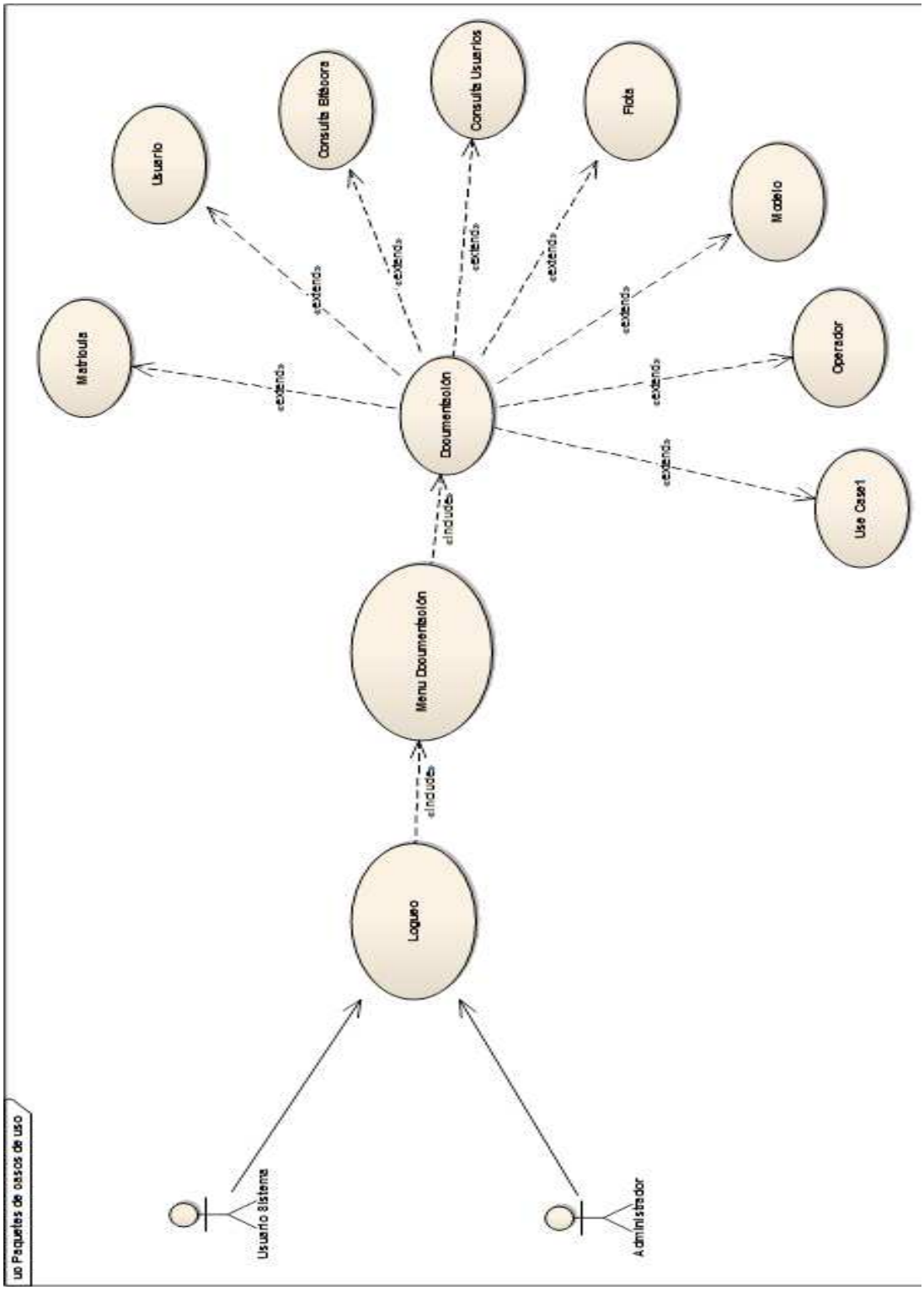


Figura 26 Paquete De Caso Uso.

2.5.2 Diagrama de secuencias

Los diagramas de secuencia permitirán definir la interacción entre los elementos que conforman el sistema, a continuación se detalla los esquemas del sistema

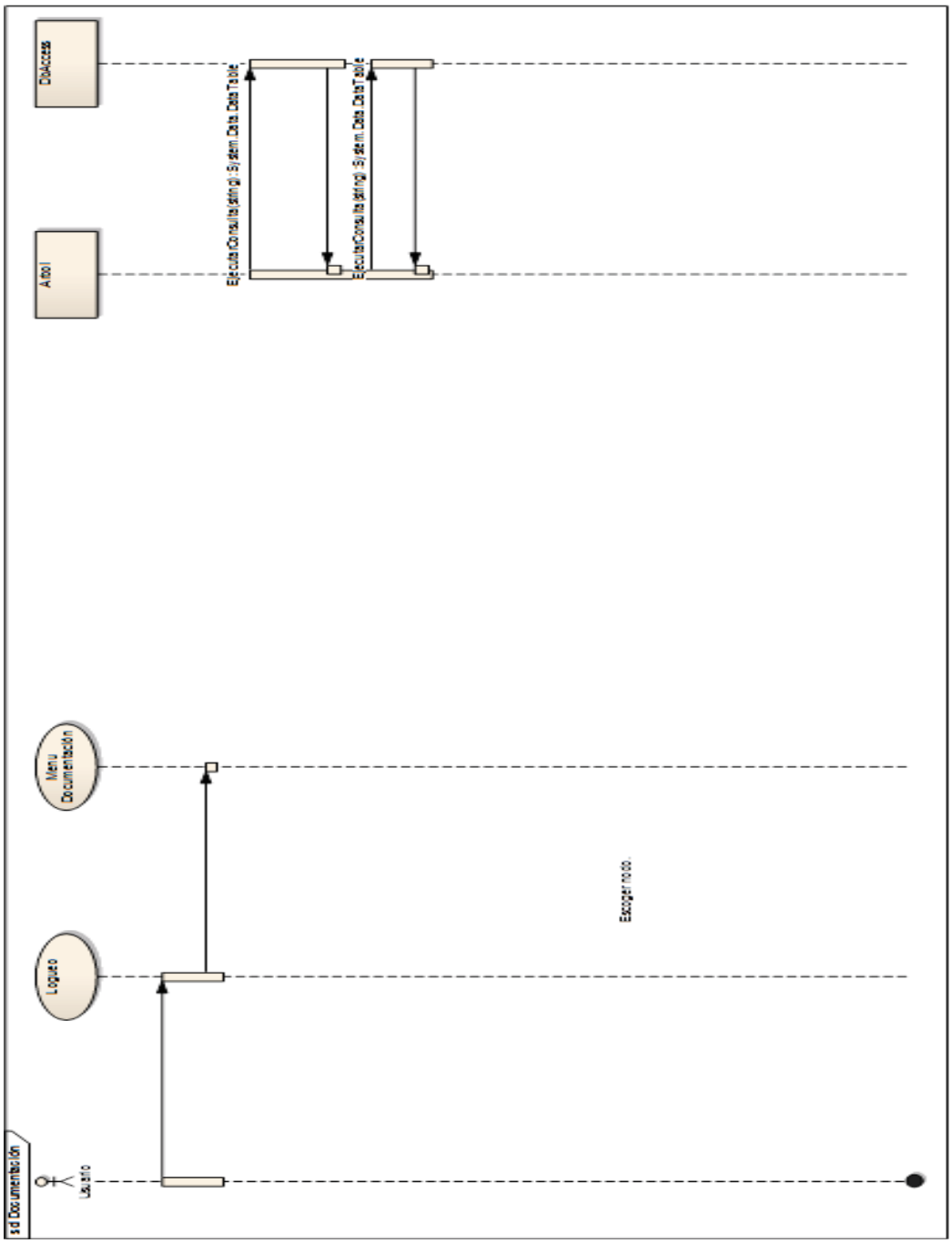


Figura 27. Acceso A La Documentación.

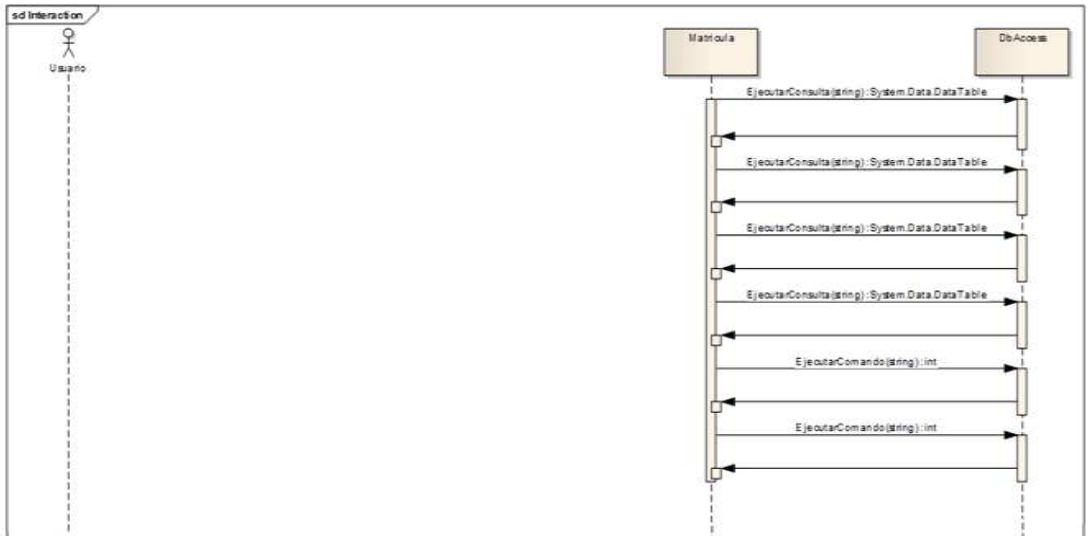


Figura 28. Interacción De Matrícula.

Descripción:

- El usuario accede al Sistema de Documentación Operacional.

El usuario ingresa las respectivas contraseñas para el acceso a las opciones del sistema.

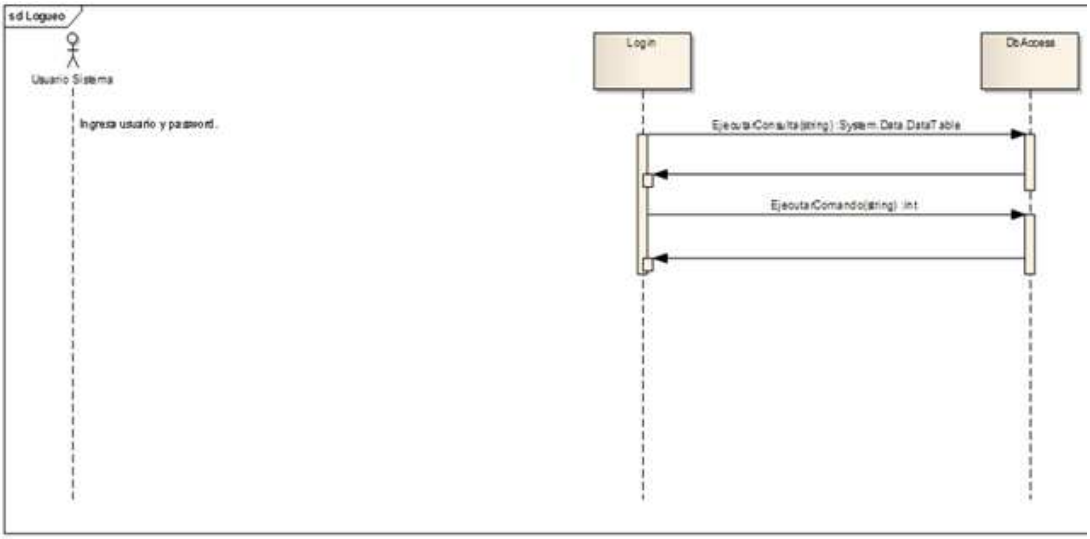


Figura 29. Logueo.

El usuario ingresa al menú de documentación para escoger sus diferentes opciones.

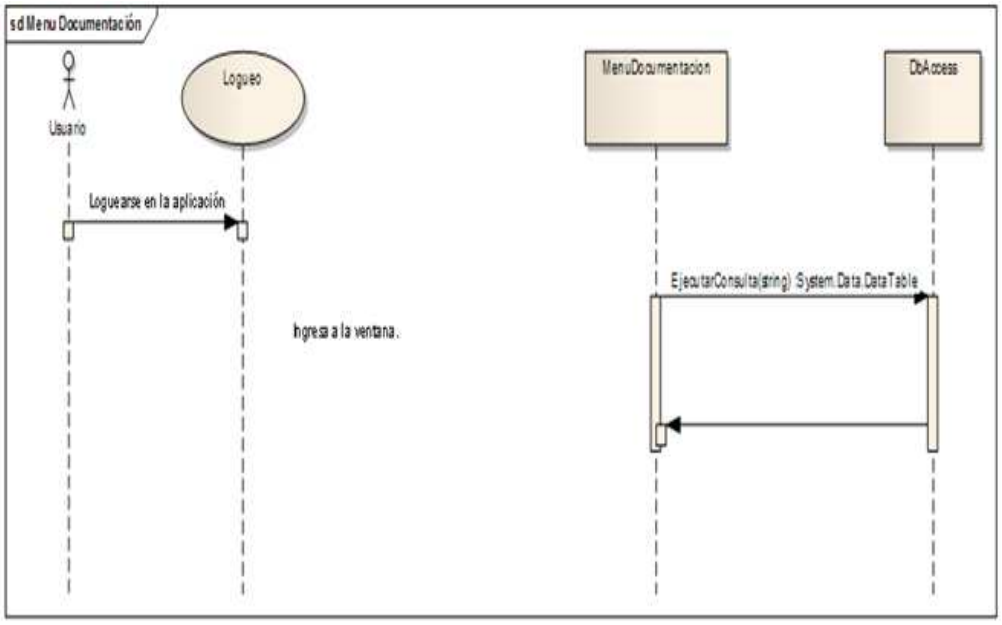


Figura 30. Menú Documentación.

2.5.3 Diagrama de clases

Los Diagramas de clases permitirán establecer el esquema lógico de cada uno de las ventanas requeridas para el sistema a continuación se describe a cada uno de ellos.

2.5.3.1 Diagrama de clases ventana del sistema

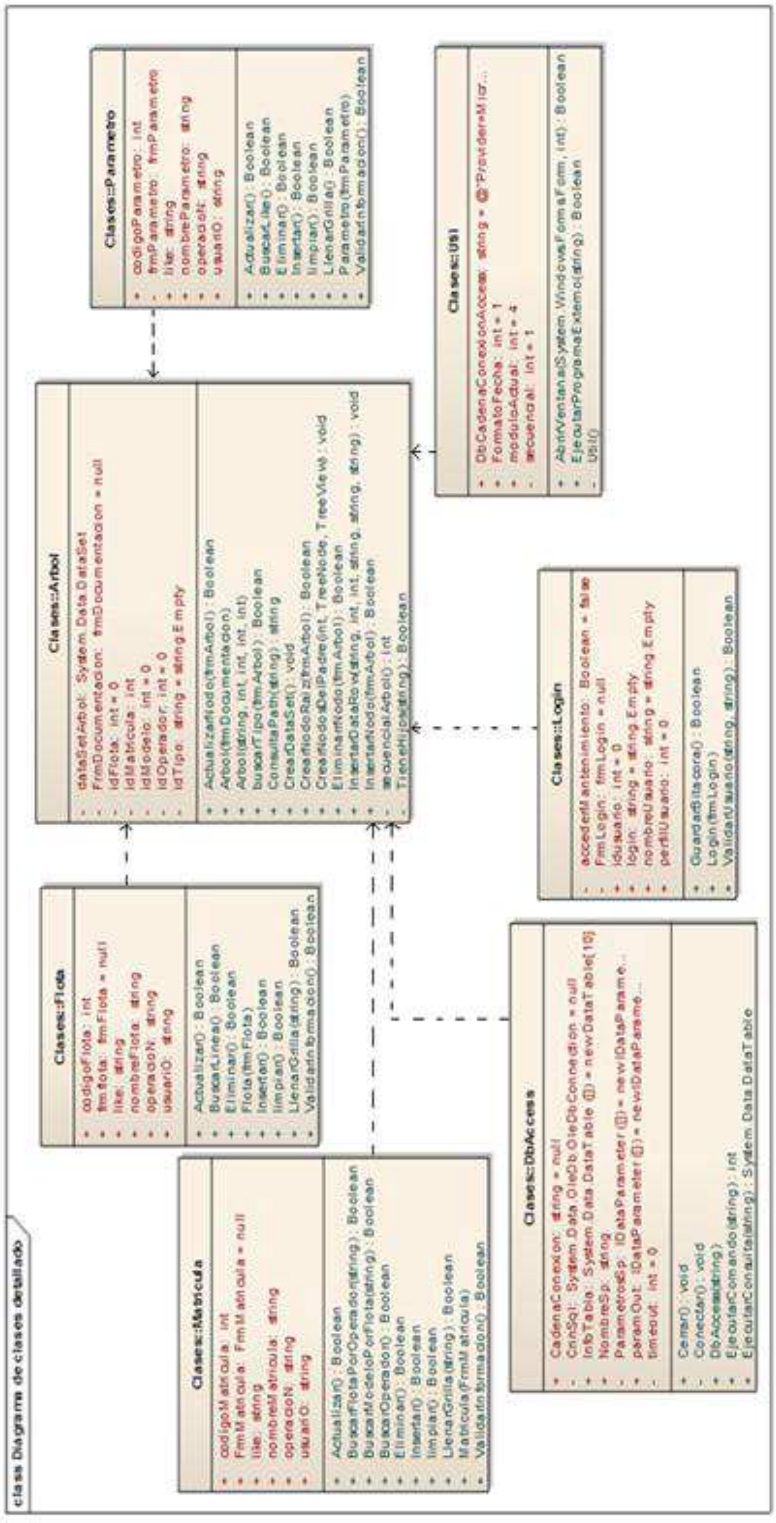


Figura 31. Clase Detallado1.

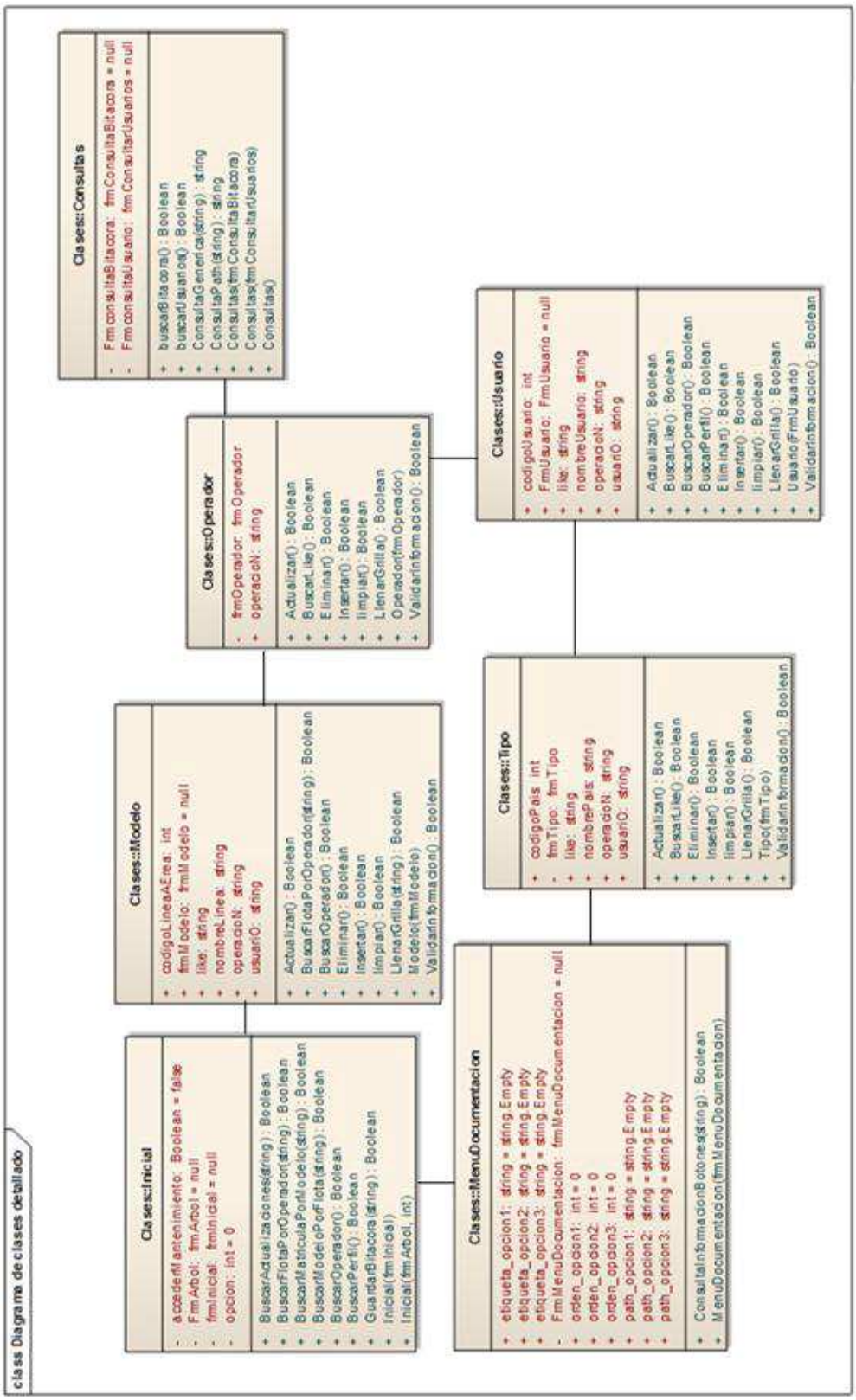


Figura 32. Clase Detallado 2.

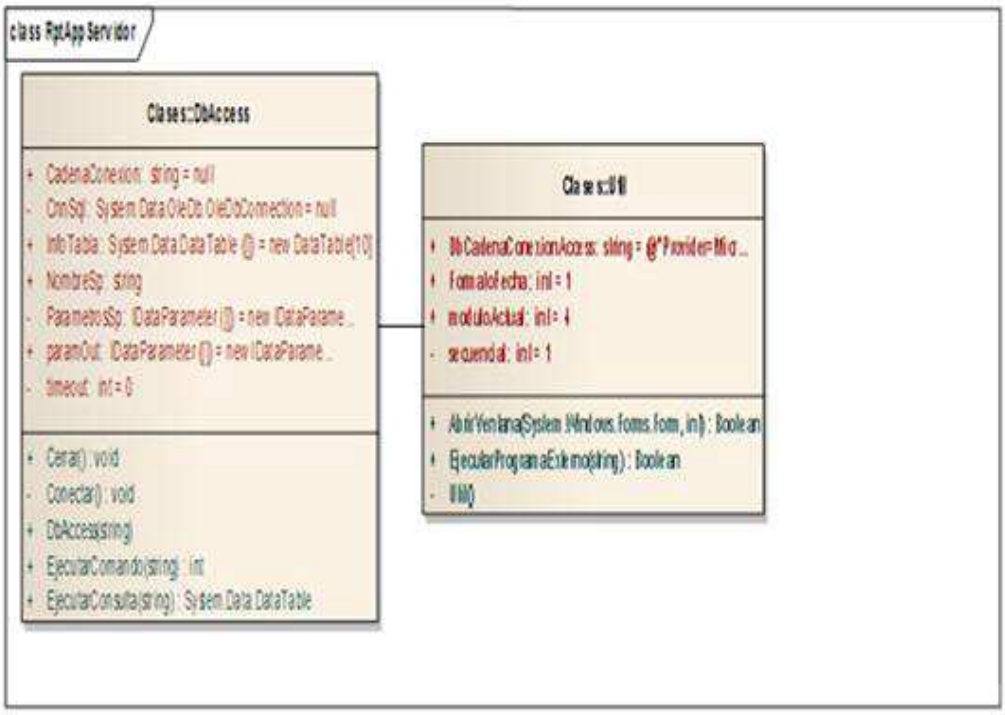


Figura 33. RptAppServidor.

2.5.4 Diagrama de componentes

Figura 34. Diagrama De Componentes.

Descripción:

- Gestión De Documentación Operacional: Este componente corresponde al Sistema.
- Gestión De Base De Datos: Este componente corresponde a la base de datos.

2.5.4.1 Diagrama de componentes sistema documental

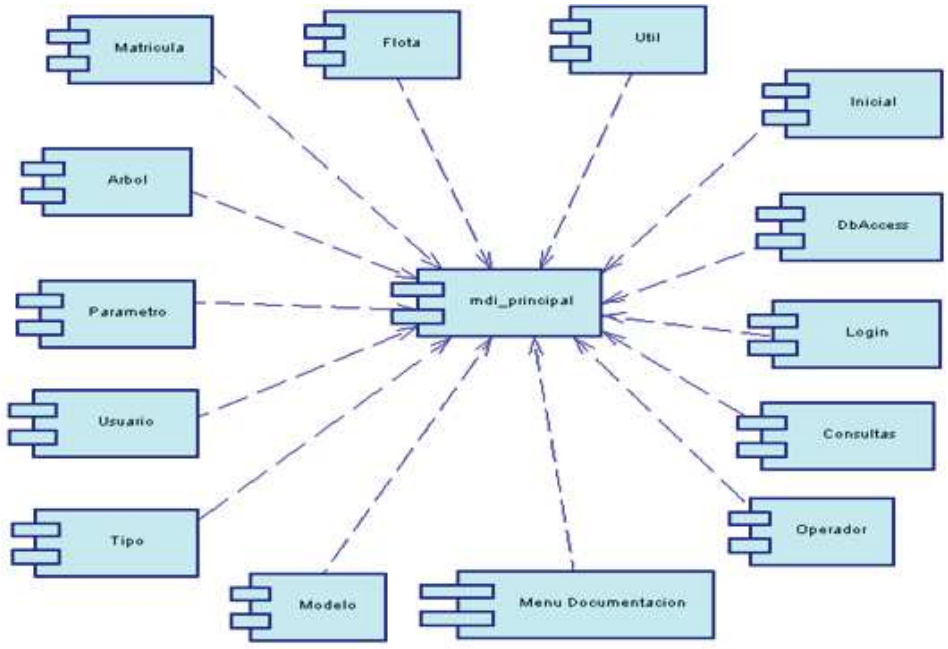


Figura 35. Diagrama De Gestión Documentación Operacional.

2.5.4.2 Diagrama de componentes de gestión de base de datos

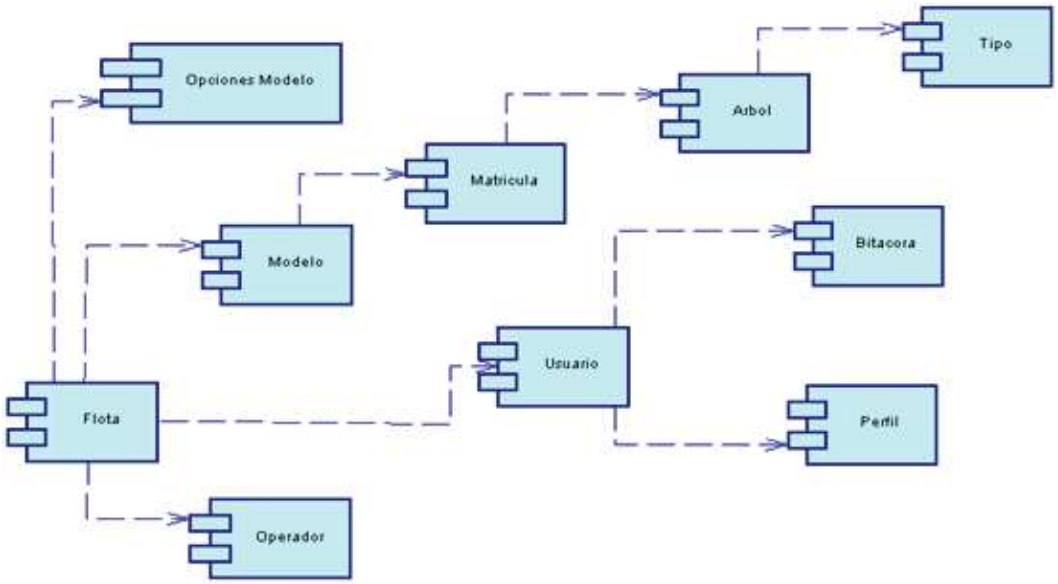


Figura 36. Diagrama de componentes de gestión de base de datos.

2.5.5 Diagrama Arquitectura Sistema

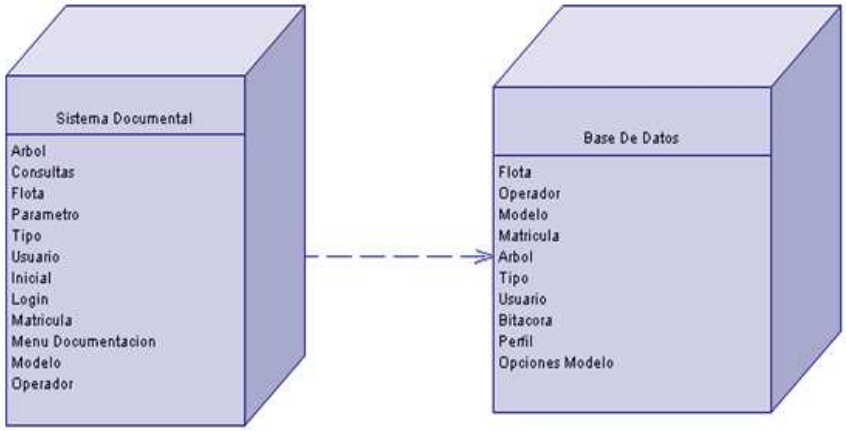


Figura 37. Diagrama Arquitectura Del Sistema

Descripción:

- Sistema Documental, significa que se encuentra todas las ventanas que se encuentra para el manejo del sistema para la gestión de información documental.
- Base De Datos, en el cual consta las tablas correspondientes para el funcionamiento del sistema.

2.5.6 Plan De Entregas

Tabla 13. Plan De Entrega "Iteración"

ITERACION	DESCRIPCION
Primera Iteración	Se desarrollan las funciones básicas del proyecto y que sean presentados al cliente.
Segunda Iteración	El proyecto debe pequeños prototipos de los proyectos con los requerimientos del cliente.
Tercera Iteración	Se determina los últimos ajustes del proyecto para entregar el producto final,

En la Tabla 14 se describe qué se va entregar en las diferentes iteraciones.

Tabla 14. Diferentes Iteraciones

Iteración	Nombre Tarea	Tipo Tarea	Descripción
Primera	Ingreso del nombre y la clave.	Desarrollo	Se debe verificar que el usuario se encuentre registrado en la base de datos para su ingreso.
Primera	Escoger el modelo y matrícula de la flota,	Desarrollo	Se debe verificar que se pueda escoger diferentes opciones de modelo y matrícula.
Primera	Interfaz Principal del usuario.	Desarrollo	Se verifica que se encuentre las opciones correspondientes en la lista, con los accesos correctos.
Primera	Búsqueda Información	Desarrollo	Se verifica la búsqueda de información en los diferentes manuales.
Segunda	Interfaz de la búsqueda del respectivo manual operacional.	Desarrollo	Se analiza el listado de ítems a consultar y las respectivas opciones del menú.
Segunda	Funcionamiento de los programas.	Desarrollo	Se debe realizar las pruebas necesarias a los programas, para verificar el funcionamiento.
Tercera	´Validar la base de datos.	Desarrollo	El proceso de la validación de datos consiste en verificar el ingreso de la información

2.5.7 Diagrama De Base De Datos

2.5.7.1 Modelo Físico

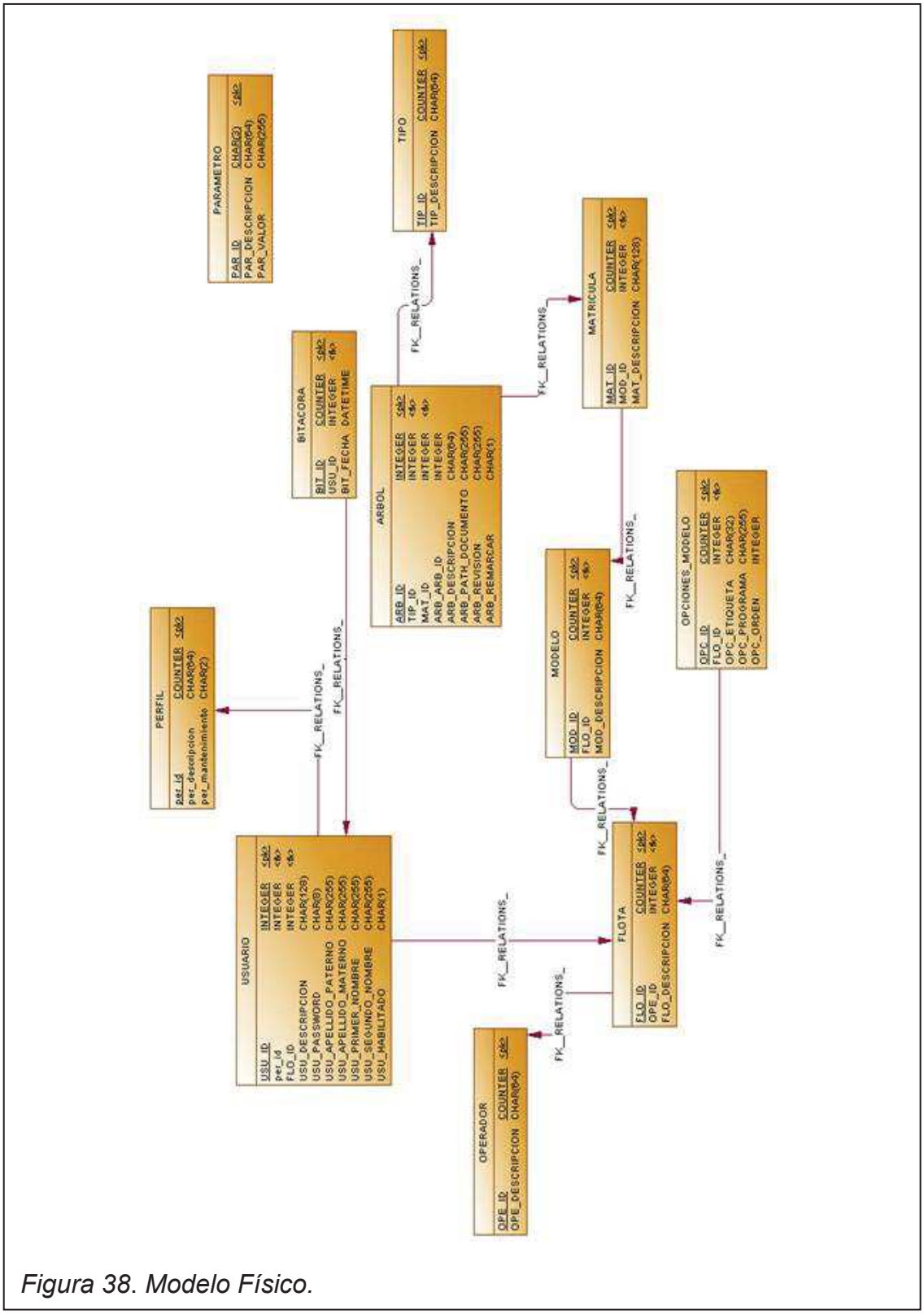


Figura 38. Modelo Físico.

2.5.7.2 Modelo Lógico

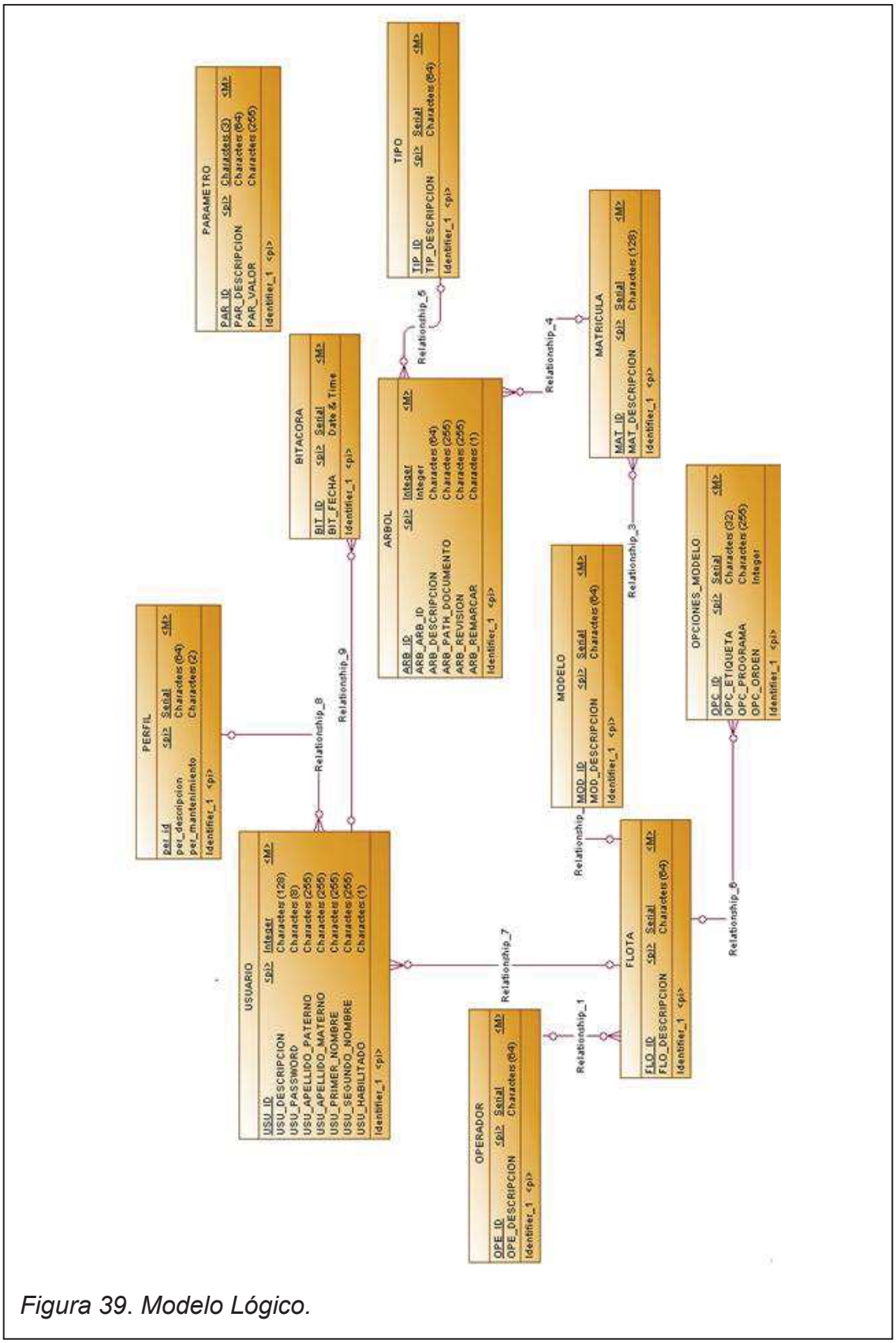


Figura 39. Modelo Lógico.

2.5.7.3 Modelo Orientado A Objetos

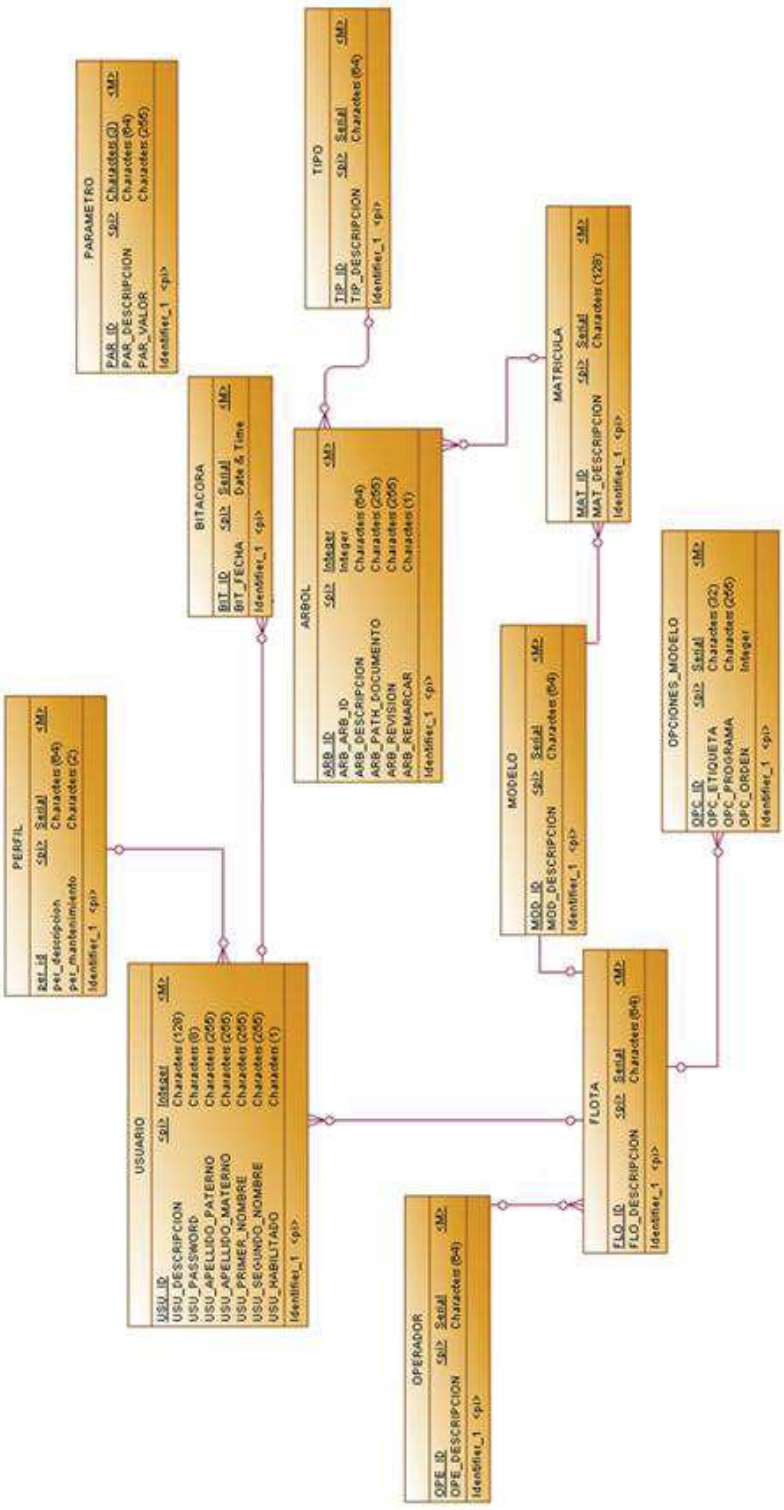


Figura 40. Modelo Orientado A Objetos.

2.6 Fase de Implementación

2.6.1 Modelo de la interfaz de usuario

El diseño de la interfaz de usuario permitirá demostrar los casos de uso y las actividades establecidas en los apartados posteriores.

2.6.2 Ventana Inicio Del Sistema

Como plantea en los requerimientos funcionales, aparecerá la pantalla de documentación operacional, para elegir la opción iniciar sesión, seguido de la contraseña, permitiendo ingresar el usuario y la contraseña de usuario que se utilizará para poder ingresar al sistema Documental.

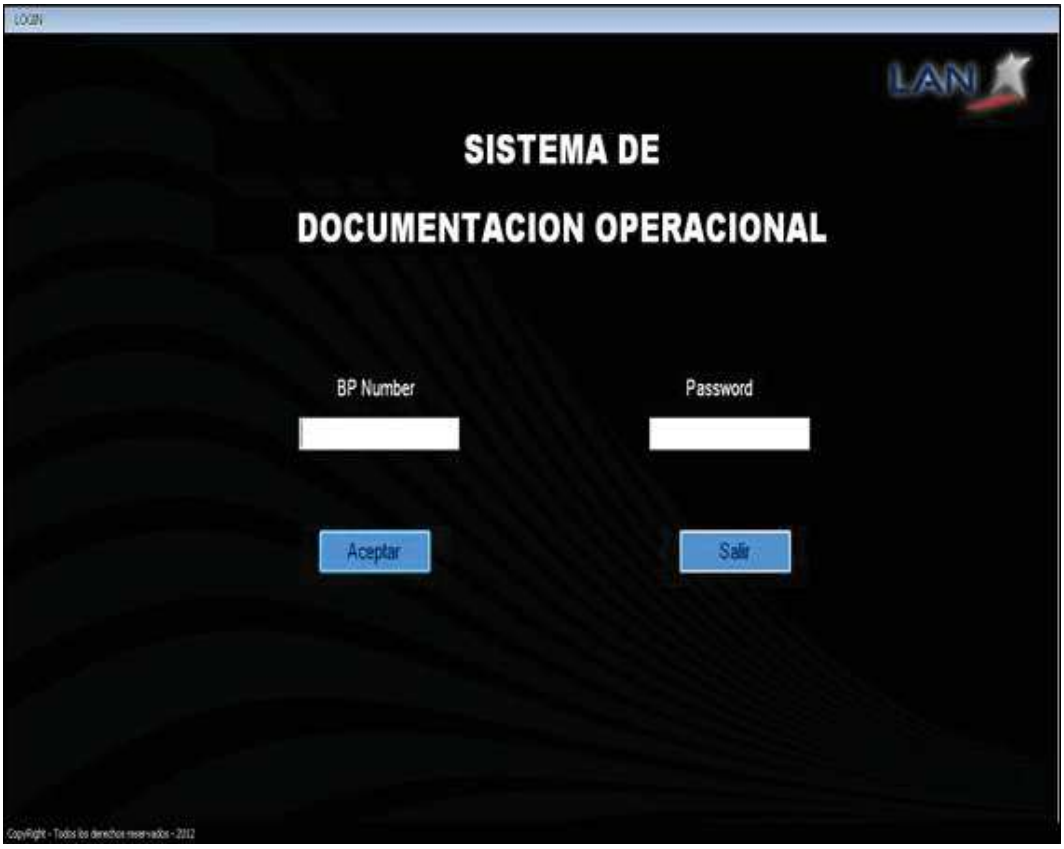


Figura 41. Usuario y Contraseña.

La ventana de documentación operacional, en el cual permite escoger el operador, flota, modelo, matrícula.

Figura 42. Documentación Operacional.

La ventana principal es la que permite escoger sus diversas funciones de cada flota sea esta Airbus o Boeing, está diseñada con botones el cual permite su fácil acceso a sus ventanas.

Figura 43. Menú Principal.

El “StoryBoard” permite desplegar todas las ventanas que tiene el sistema.

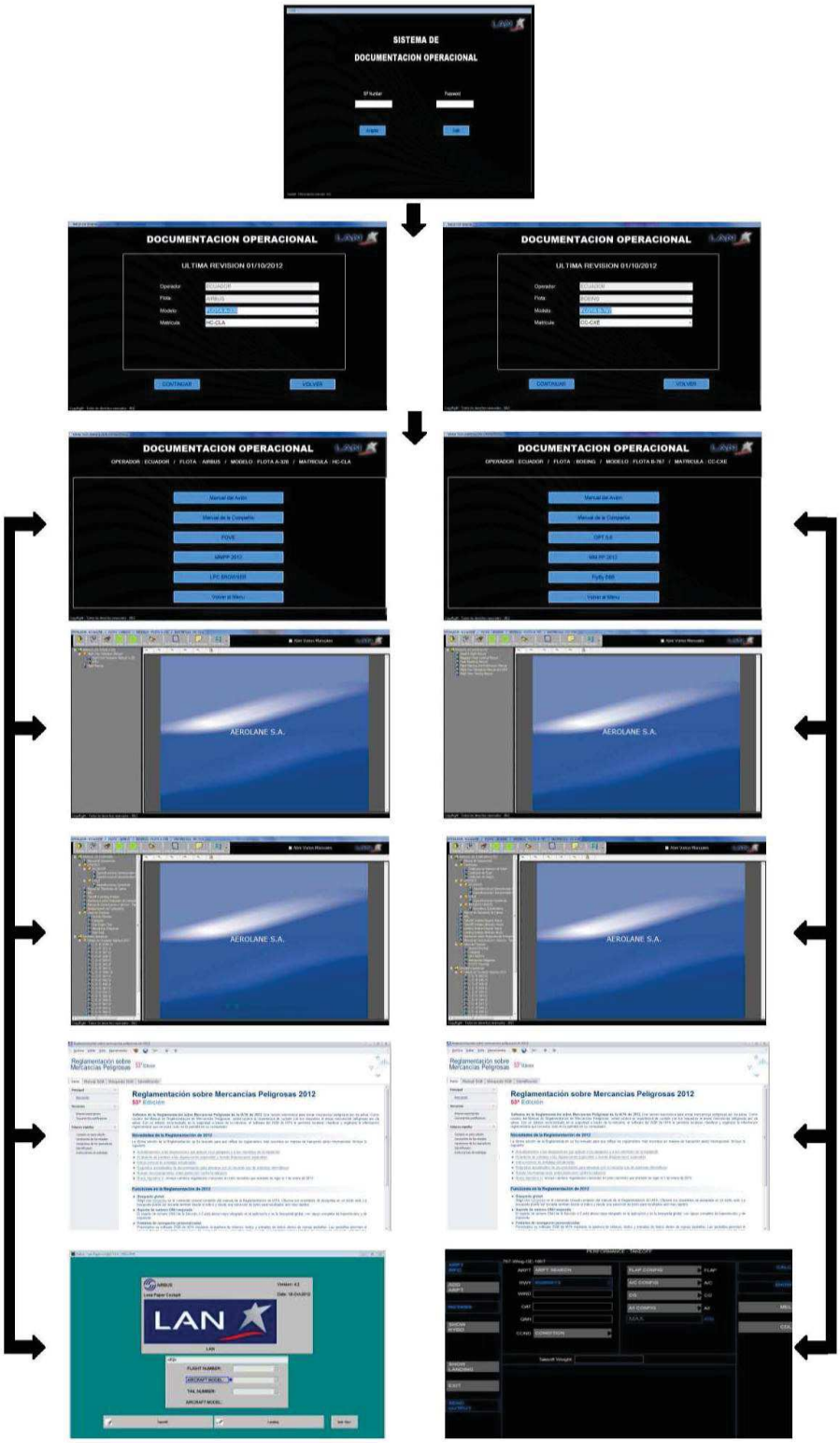


Figura 44. StoryBoard.

PRUEBAS (Tarjetas CRC)

Tabla 15. Caja negra de acceso al sistema.

Proyecto	
Actividad	Plan de Pruebas
Técnica	Caja Negra
Caso de Uso	Ingreso Al Acceso Al Sistema con un usuario administrador.

Acceso Inicial

- Ingresar Login
- Ingresar Password
- Pulsar Aceptar

Propósito

Verificar que los datos se han ingresados en forma completa y correcta, en los dos campos tanto el usuario como contraseña.

Datos De Entrada

- Login
- Password

Salidas Esperadas

Una vez ingresado el usuario y contraseña, permitirá ingresar a la ventana principal o dará un mensaje de error en el caso de haber mal ingresado.

Resultado:

Figura 45. Pantalla Login

Para complementar el proceso de desarrollo se ha elaborado los casos de pruebas requeridas.

Tabla 16. Caja negra de ingreso a la documentación operativa.

Proyecto	
Actividad	Plan de Pruebas
Técnica	Caja Negra
Caso de Uso	Ingreso A La Documentación Operativa.

Acceso Inicial

- Ingresar Operador
- Ingresar Flota
- Ingresar Modelo
- Ingresar Matrícula
- Pulsar Continuar

Propósito

Verificar que los datos se han escogidos en forma completa y correcta, existe algunos algunas opciones que se puede escoger tanto en el país como en la matrícula.

Datos De Entrada

- Operador
- Flota
- Modelo
- Matrícula

Salidas Esperadas

- El Sistema devuelve una lista de países y matriculas para poder escoger.

Resultado:

Figura 46. Pantalla Inicio.

Tabla 17. Caja negra menú principal.

Proyecto	
Actividad	Plan de Pruebas
Técnica	Caja Negra
Caso de Uso	Menú Principal

Acceso Inicial

- Escoger manual de avión
- Escoger manual de compañía
- Escoger opciones

Propósito

Hacer uso de las diferentes opciones, de acuerdo a la necesidad del usuario y flota.

Datos De Entrada

- Elegir de la lista las diferentes opciones.

Salidas Esperadas

- Visualiza la opción elegida con sus diferentes funciones.

Resultado:

Figura 47. Pantalla Menú.

Tabla 18. Caja negra Manual del avión.

Proyecto	
Actividad	Plan de Pruebas
Técnica	Caja Negra
Caso de Uso	Manual de Avión

Acceso Inicial

- Elegir la opción Volver.
- Elegir la opción Ocultar.
- Elegir la opción Refrescar.
- Elegir la opción Adelante.
- Elegir la opción Atrás.
- Elegir la opción Ins. Manuales.
- Elegir la opción Inst Programas.
- Elegir Prg.Aerolane.
- Elegir Mantenimiento.

Propósito

- Se debe escoger una de las diferentes opciones, de acuerdo a la necesidad del usuario.

Datos De Entrada

- Elegir del menú las diferentes opciones.

Salidas Esperadas

- Visualiza la opción elegida con sus diferentes funciones.

Resultado:

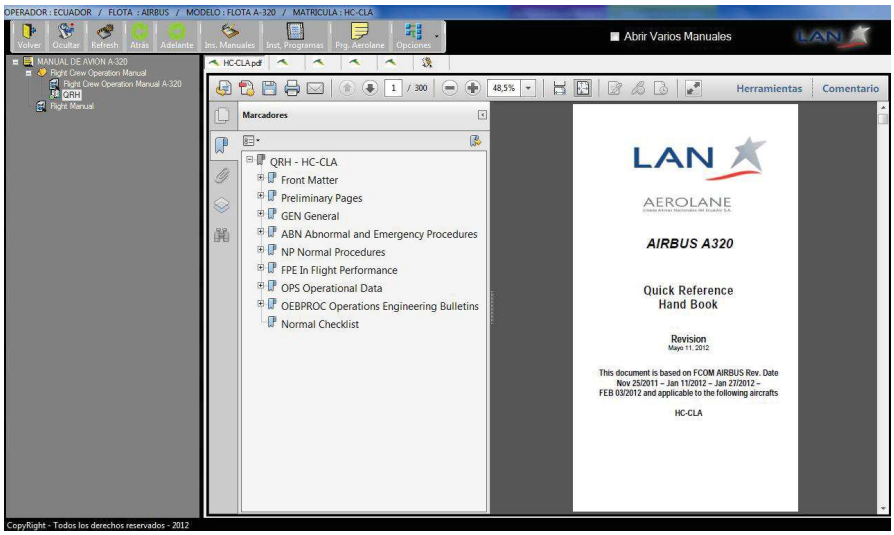


Figura 48. Pantalla Manuales del Avión.

Tabla 19. Caja negra Manual de Avión/Opción Mantenimiento.

Proyecto	
Actividad	Plan de Pruebas
Técnica	Caja Negra
Caso de Uso	Manual de Avión/Opción Mantenimiento

Acceso Inicial

- Elegir la opción árbol.
- Elegir la opción operador.
- Elegir la opción flota.
- Elegir la opción modelo de aeronave.
- Elegir la opción matrícula.
- Elegir la opción tipo.
- Elegir la opción parámetro.
- Elegir la opción usuario.
- Elegir la opción consultar bitácora.
- Elegir la opción consultar usuarios.

Propósito

- Se debe escoger una de las diferentes opciones, de acuerdo a la necesidad del usuario.

Datos De Entrada

- Elegir del menú las diferentes opciones.

Salidas Esperadas

- Visualiza la opción elegida con sus diferentes ventanas

Resultado:

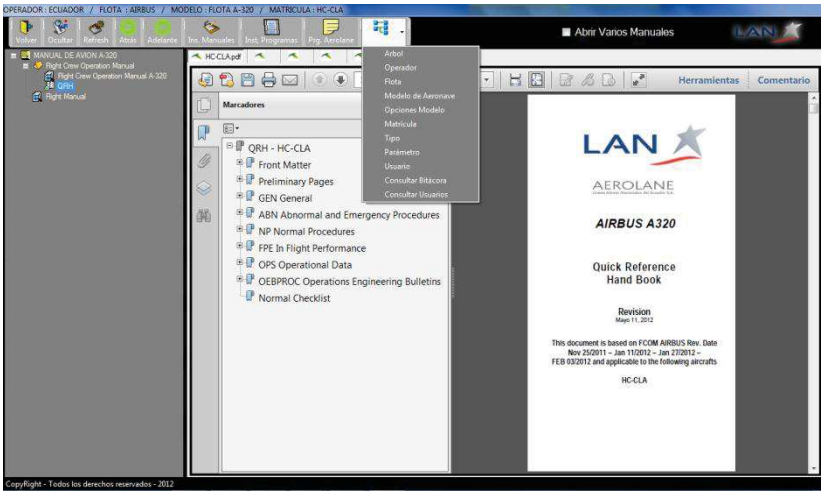


Figura 49. Pantalla de Manuales del Avión Mantenimiento.

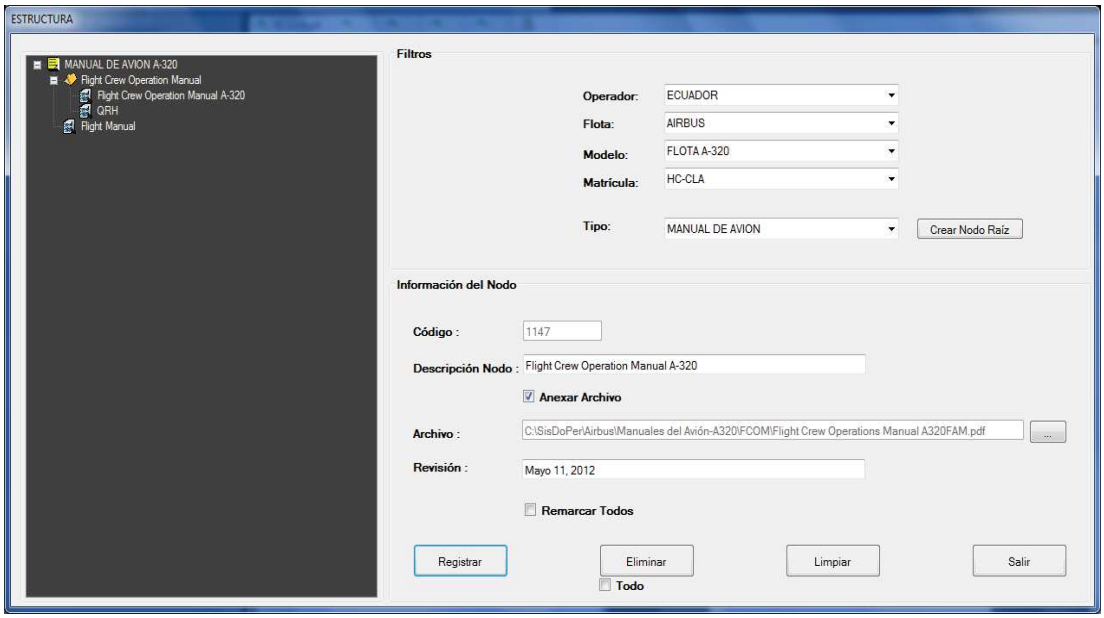


Figura 50. Ventana Árbol.

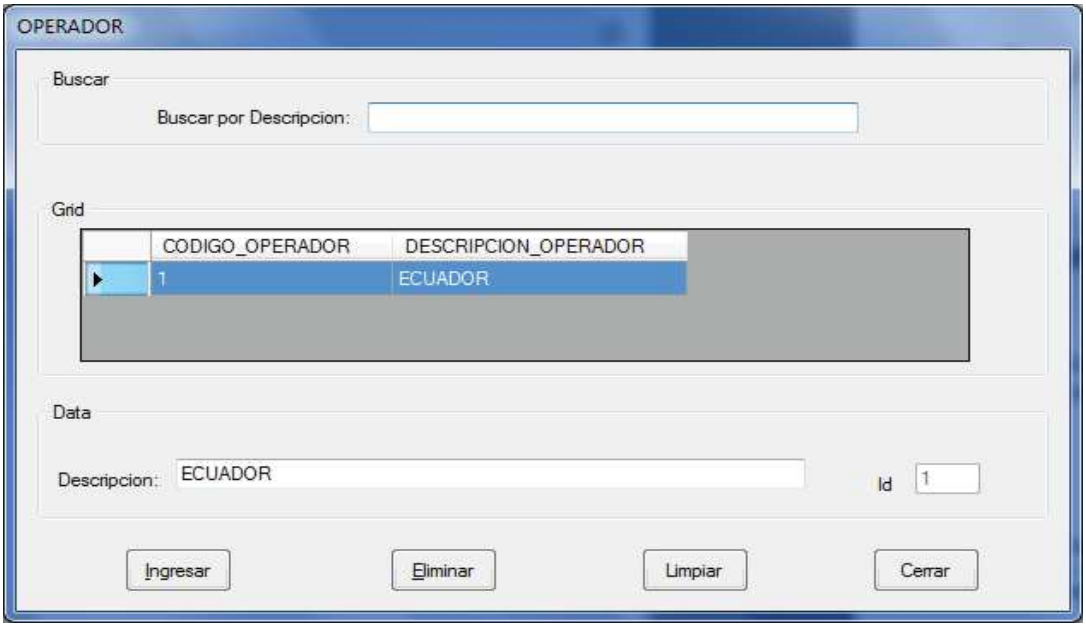


Figura 51. Ventana Operador.

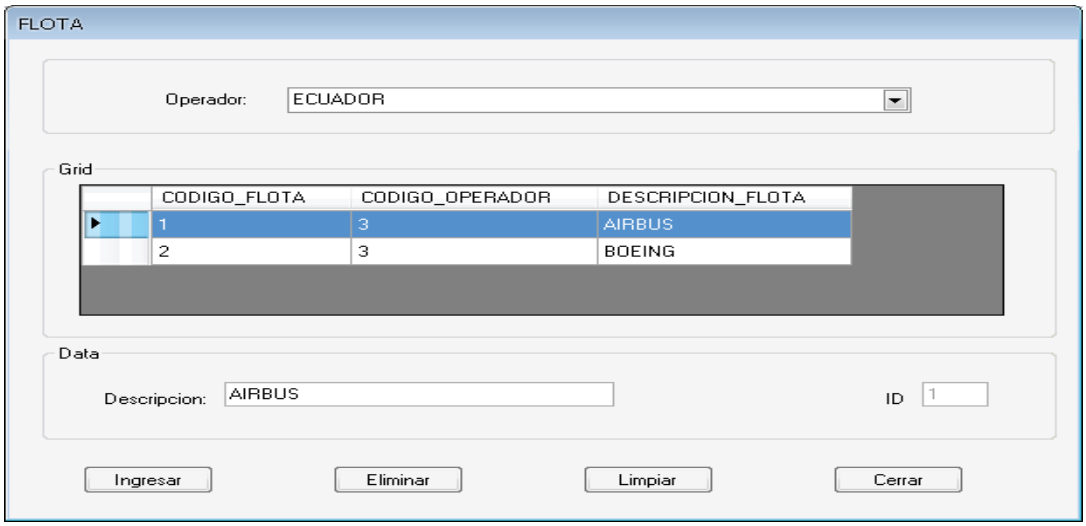


Figura 52. Ventana Flota.

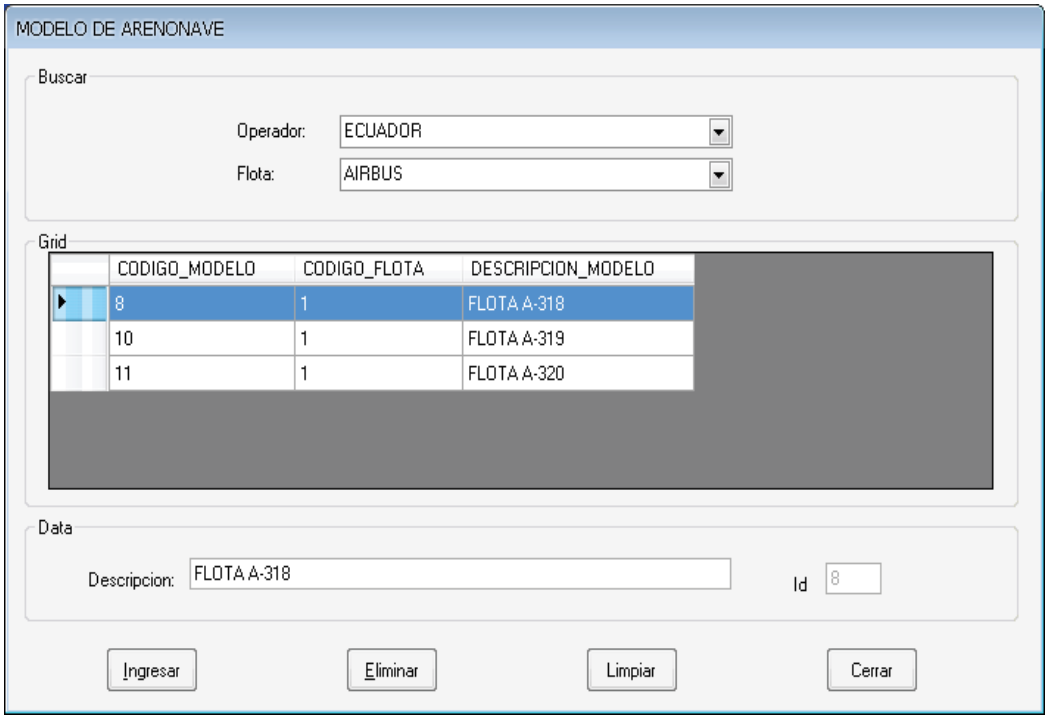


Figura 53. Ventana Modelo de Aeronave.

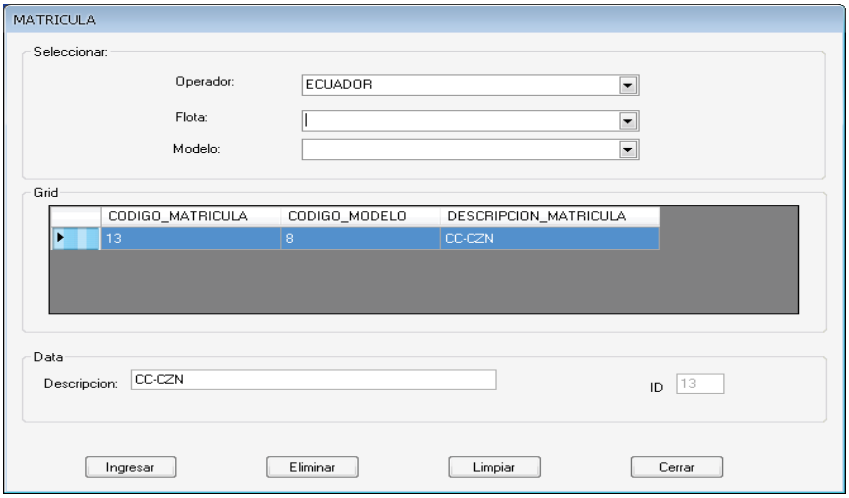


Figura 54. Ventana Matrícula.

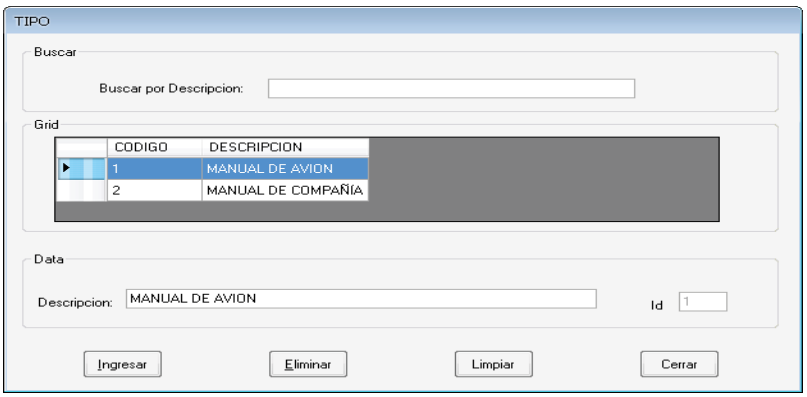


Figura 55. Ventana Tipo.

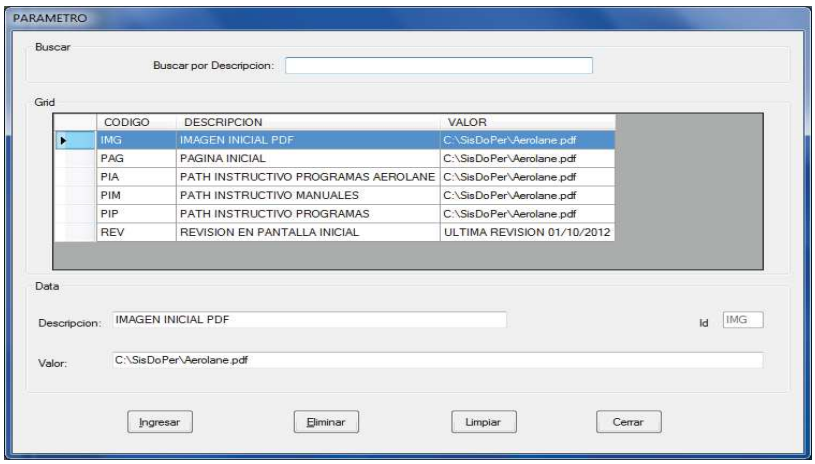


Figura 56. Ventana Parámetro.

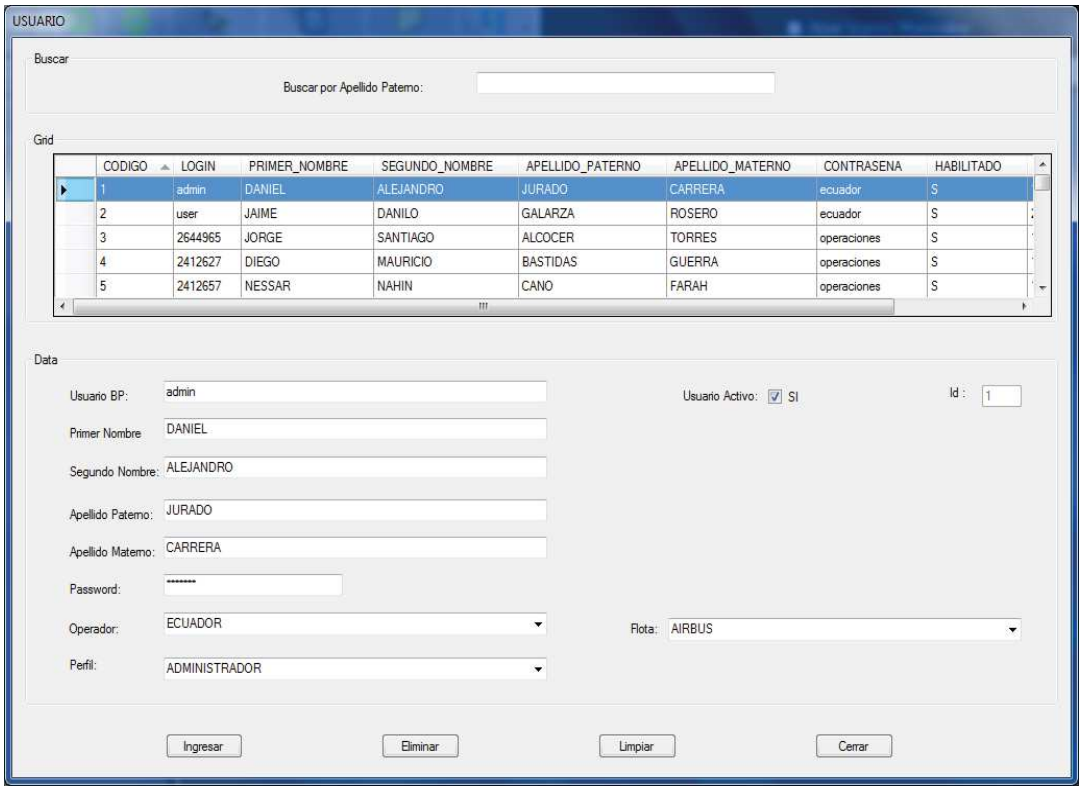


Figura 57. Ventana Usuario.

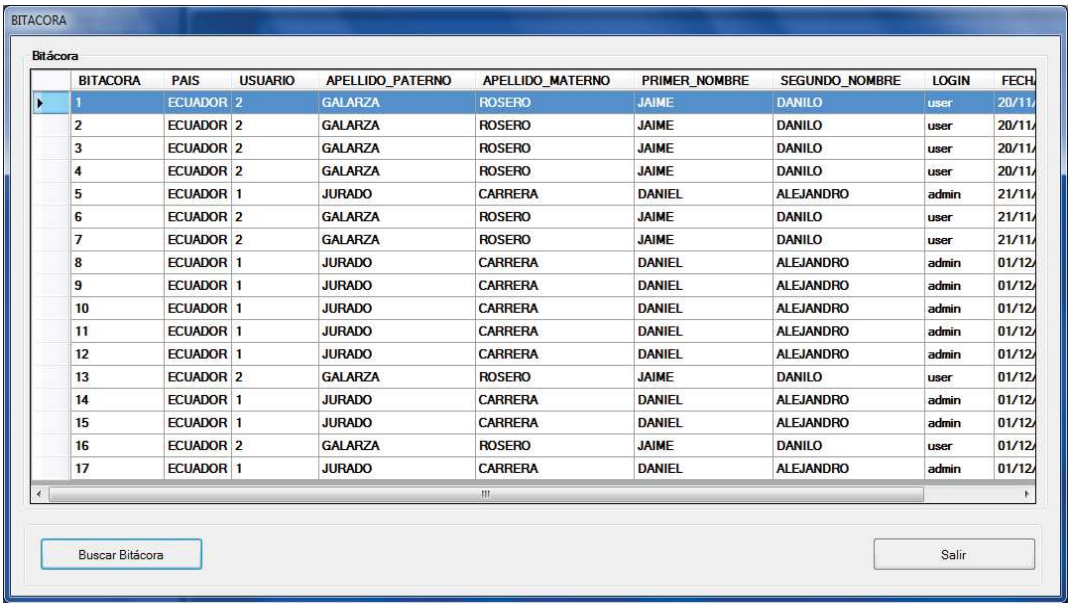


Figura 58. Ventana Consultar Bitácora.

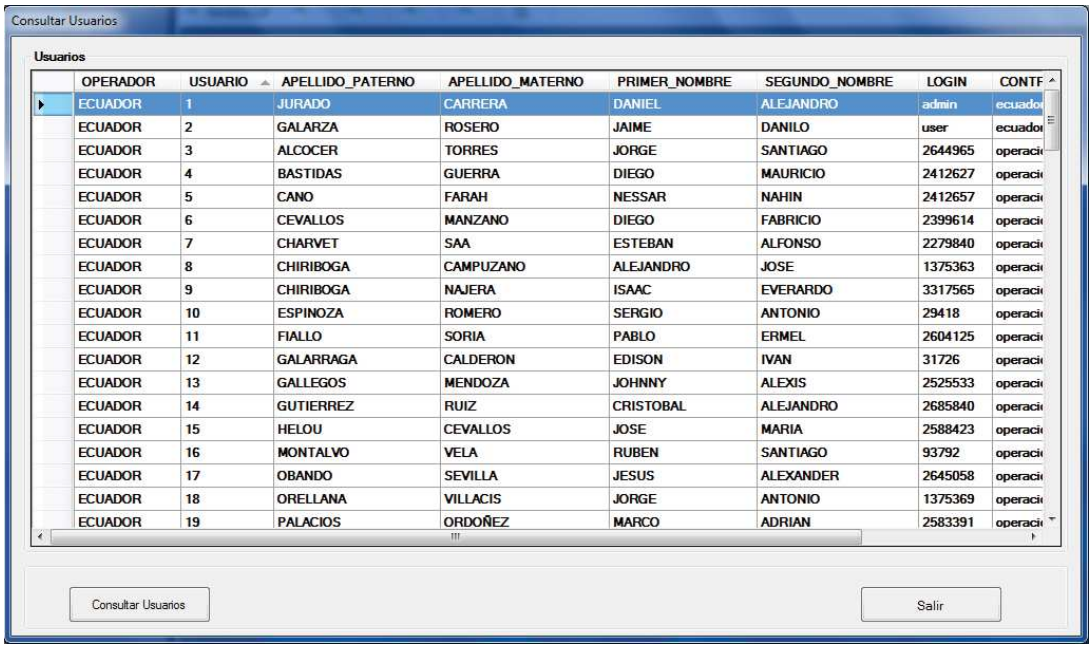


Figura 59. Ventana Consultar Usuario.

En la tabla 20 se accede al sistema con un usuario con accesos limitados

Tabla 20. Caja negra Acceso usuario limitado al Menú Principal.

Proyecto	
Actividad	Plan de Pruebas
Técnica	Caja Negra
Caso de Uso	Acceso usuario limitado al Menú Principal

Acceso Inicial

- Ingresar el usuario
- Escoger el modelo.
- Escoger manual de avión
- Escoger manual de compañía
- OPT 5.6, FlyByE6B
- Escoger opciones

Propósito

Hacer uso de las diferentes opciones, de acuerdo a la necesidad del usuario.

Datos De Entrada

- Elegir de la lista las diferentes opciones.

Salidas Esperadas

- Visualiza la opción elegida con sus diferentes funciones.

Resultado:

Al momento de ingresar con usuario limitado aparece el único modelo a escoger y sus diferentes matriculas.

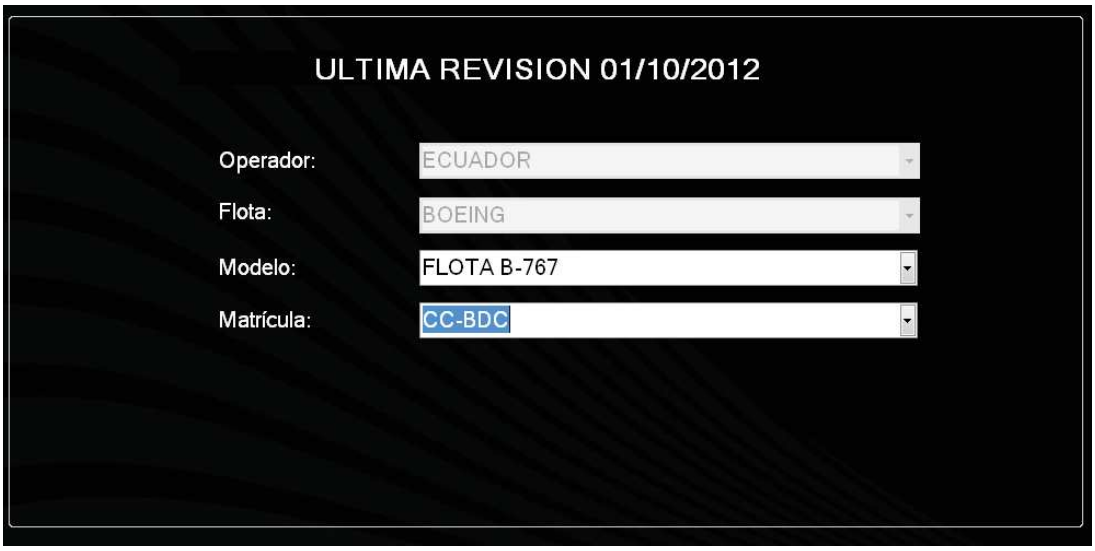


Figura 60. Pantalla Acceso usuario limitado al Menú Principal.

Figura 61. Menú documentación Operacional.

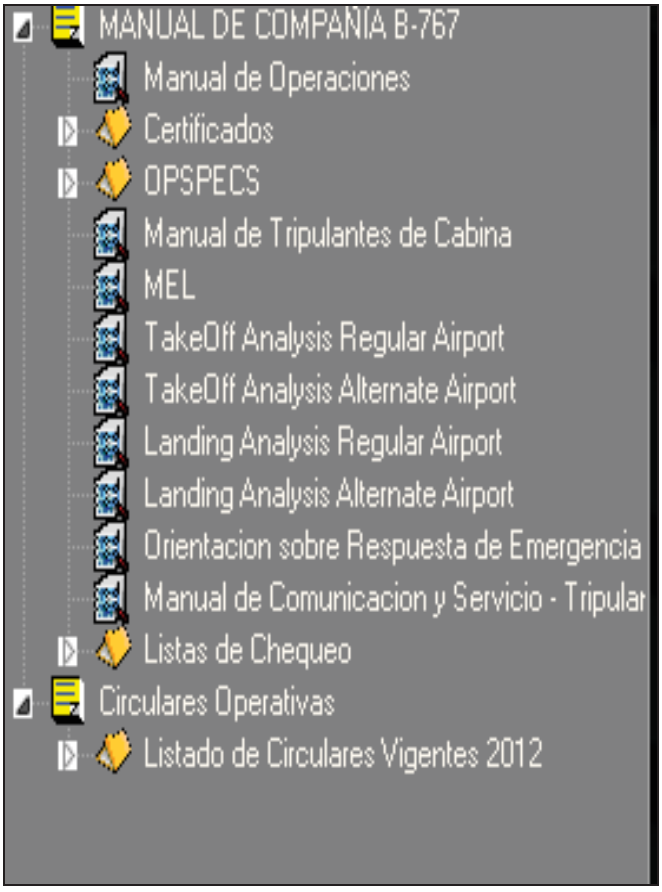


Figura 62. Arbol de Menú Operacional

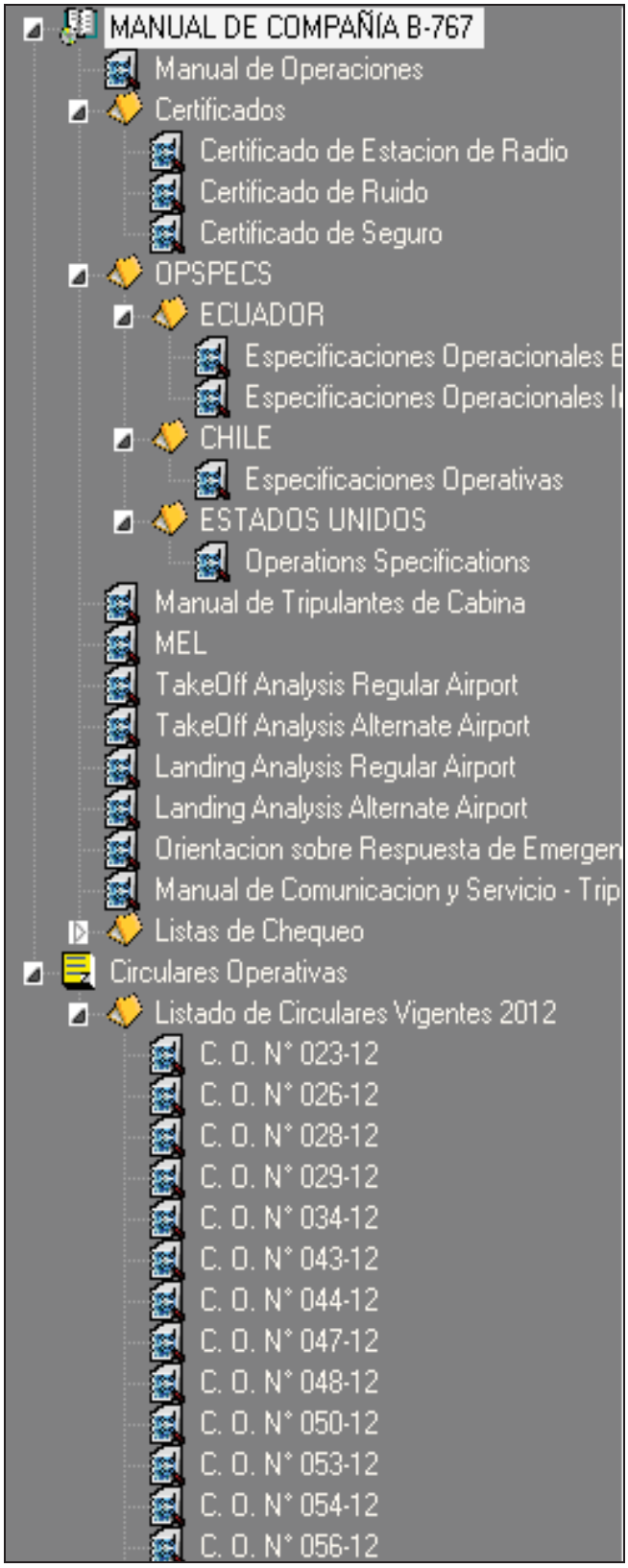


Figura 63. Se puede escoger del menú principal sus diferentes opciones.

3. ANALISIS COSTO/BENEFICIO.

3.1 Costo

Dentro del costo, se va a utilizar tres tipos de factibilidad el cual nos permite realizar un análisis técnico, operacional y económico, los cuales se van a detallar en los siguientes puntos.

- Técnico.
- Operacional
- Económico

3.1.1 Factibilidad Técnica

La sección de la factibilidad técnica, se utilizará para definir los componentes necesarios e indispensables con sus respectivas características las cuales formaran parte en la construcción y ejecución del sistema además es muy importante conocer los elementos como estaciones de trabajo las cuales nos permitirán realizar la ejecución del proyecto.

Además se va a mostrar los grados de influencias respectivas donde se va a detallar el nivel de complejidad técnica desarrollado en este proyecto.

Tabla 21. Influencias.

Numero	Influencia	Valor
1	Diseño para el usuario final	4
2	Actualización de datos	4
3	Facilidad de instalación	4
4	Facilidad de operación	5
5	Facilidad de cambios	5
6	Seguridad	5
7	Documentación	5

En la siguiente tabla se representa los valores que representa cada influencia.

Tabla 22. Escala de valor.

CALIFICACIÓN	ESCALA
Nada	1
Poco	2
Medianamente	3
Importante	4
Muy importante	5

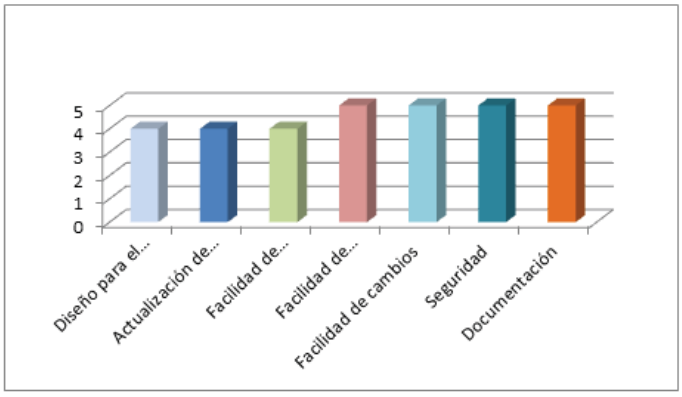


Figura 64. Influencias.

3.1.1.1 Estación de trabajo de la aplicación

Para realizar la ejecución del proyecto se utilizará los siguientes requerimientos que se plantean en la tabla 23.

Tabla 23. Requerimiento Del Sistema.

Requerimientos Del Sistema
PROCESADOR
800 MHZ (mínimo)
RAM
512 (mínimo)
DISCO DURO
4 GB (libres)
SISTEMA OPERATIVO
Windows XP

3.1.2 Factibilidad Operacional

En la sección de la factibilidad operacional se describe las causas que presenta LAN Ecuador., en la tabla 24 se mostrará las causas.

Tabla 24. Causas.

CAUSAS
Demora en el registro de datos de los clientes, servicios.
Tardanza en la consulta y búsqueda de información.
Utilización de manuales de las aeronaves en forma manual.

En la tabla 25 se describirá como se dará la solución de la causas.

Tabla 25. Soluciones.

SOLUCIONES
Automatización en el registro de datos de los clientes, servicios.
Automatización en el proceso de búsqueda de datos.
Automatización del proceso de información.

Además la implementación de flotas trae beneficios, a continuación se describe en la tabla 26,

Tabla 26. Beneficios De Las Flotas.

BENEFICIOS DE LAS FLOTAS
Simplicidad de procesos
Calidad de servicio.
Confiabilidad.
Minimización de errores.

3.1.3 Factibilidad Económica

Los aspectos administrativos se dividen en costo operativo, materiales, gasto del proyecto y un imprevisto del 10%, se detalla en la tabla 27.

Tabla 27. Aspecto Económico.

COSTO OPERATIVO			**SUBTOTAL**
Transporte			100,00
Internet			130,00
Total Costos Operativos			230,00
MATERIALES			**SUBTOTAL**
Copias			30,00
Impresora			120,00
Cartuchos Impresora			60,00
Resmas de Papel			25,00
Total Materiales			235,00
GASTO DEL PROYECTO			SUBTOTAL
Recursos Humanos			
	Costo Hora	**Número Total De horas**	
Ingeniero	*4,00*	*250*	1000,00
Recursos Técnicos			
Licencia de Microsoft Access			700,00
Licencia de Microsoft C#			950,00
Total Gastos del Proyecto			2650,00
COSTOS TOTALES			
Operativos			230,00
Materiales			235,00
Proyecto			2650,00
Costo Parcial			3115,00
Imprevistos 10%			311,50
Costo Total del Proyecto			**3426,50**

Se obtendrá un 10% de ganancia en la construcción del sistema Operacional. el costo del proyecto es de USD 3426,50 que debe ser entregado al cliente de la empresa para su consideración.

A continuación se establecerá en que ayudaría el cambiar la documentación manual, en cada una de las flotas, uno de ellos es el peso, el cual va a disminuir el 50% en manuales de papel a bordo.

Tabla 28. Peso Actual vs uso Notebook.

PESO ACTUAL VERSUS USO DE NOTEBOOK		
Aeronave	Sin Notebook	Con Notebook
A320 FAM	13.2OO K	12.280 K
B767 Pax	22.500 K	22.500 K

En la tabla nos representa el peso actual con la documentación manual y el cambio de peso que se obtiene al utilizar la Notebook con el sistema.

Además se debe tomar en cuenta las inversiones del proyecto en las flotas, se obtendrá beneficio y ahorro.

Tabla 29. Beneficio/Ahorro.

BENEFICIOS	AHORRO
Simplicidad de procesos	Ahorro anual fuel por disminución de peso en las aeronaves más de 180 mil dólares.
Reducción del 90% documentación en papel.	Disminución de reproducción de manuales en papel por 30 mil dólares.
Confiabilidad y liberación de espacios en cabina.	
Minimización de errores.	

Asimismo el beneficio será cuantitativo debido a 2 aspectos importantes que se detalla a continuación en la tabla 30.

Tabla 30. Estadística De Beneficios Cuantitativos.

BENEFICIOS CUANTITATIVO	AHORRO
Ahorro reproducción de manuales en papel aproximado.	USD 27,700
Ahorro Fuel por disminución de peso en cockpit aproximado.	USD 189,400
Ahorro Total	**USD 217,200**

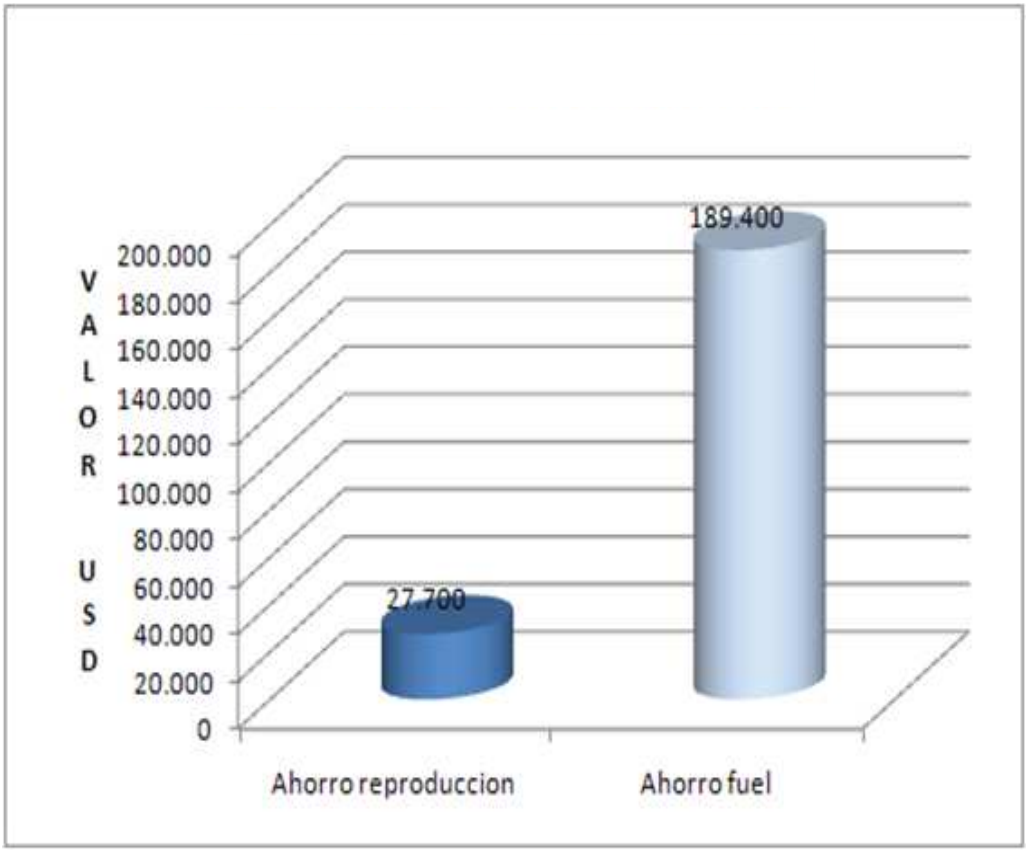


Figura 65. Representación De Beneficios Cuantitativos.

3.2 Categorías y Beneficios

Los beneficios que ofrece al realizar el sistema se ha dividido en las siguientes categorías:

- Para la Empresa
- Para el Personal
- Tecnológico

El sistema una vez implantado en la empresa permite que sus procesos sean automatizados para obtener como resultado la organización, integridad en su información y a su vez dar un mejor servicio a la clientela.

Tabla 31. Beneficio Para La Empresa

BENEFICIOS PARA LA EMPRESA
La organización de información (Documentación, flota, matrícula, modelo, operador, tipo).
La integridad en el acceso a los datos (El manejo de diferentes perfiles de usuario).
La gestión de la información (Consultar, insertar, búsqueda, actualización).

En cambio el sistema ayuda a su personal en la gestión de la información en forma rápida, optimizando el tiempo, obteniendo datos actualizados y veraces. A continuación se detalla los beneficios del sistema para el personal.

Tabla 32. Beneficio Para El Personal

BENEFICIOS PARA EL PERSONAL
La gestión de consulta y búsqueda de datos es rápida.
Los datos son detallados referentes a la información respectiva del manual de usuario.
El diseño de la interfaz es amigable con el usuario.

Además el sistema utiliza tecnología y metodología que permite que el desarrollo de sus procesos sea de manera completa e iterativa. A continuación se detalla la metodología y tecnología que se utilizó para el beneficio en el desarrollo del sistema.

Tabla 33. Metodología y tecnología para el Sistema.

METODOLOGIA Y TECNOLOGIA PARA EL SISTEMA
Metodología Extreme Programming (XP), utilización de sus fases, para el desarrollo de sus procesos.
UML, utilización de los diagramas para la realización de las funciones.
La tecnología de información y comunicaciones.

4. CONCLUSIONES Y RECOMENDACIONES.

4.1 Conclusiones

- La automatización de los procesos de registro y búsqueda que gestiona el sistema, permitió tener un acceso eficaz a la información, reduciendo los problemas encontrados con pérdidas de datos, tiempo. La seguridad para el acceso y los perfiles de usuario determinados permitió garantizar la integridad y confidencialidad de los datos almacenados, evitando problemas para errores humanos. Además el diseño de una interfaz permitió gestionar ordenadamente los datos de los respectivos manuales mejorando la calidad de servicio.

- Para el Análisis de la Situación Actual de LAN ECUADOR se pudo observar los problemas que existían su causa y efecto negativo para la organización. Como resultado se pudo observar que el problema tiene una raíz que es la falta de procesos-procedimientos automatizados, el mismo que no estaba bien estructurado; no hay una correcta administración de los casos hacía el acceso a los diferentes manuales, debido a que el personal ingresaba a buscar de forma manual.

- Después del análisis de LAN ECUADOR, se plantea una alternativa de selección del sistema de documentación operacional, corrigiendo todo lo malo y rescatando lo que sea posible. Donde el diseño será el encargado de dar valor a cada una de las Áreas de la Organización y que se convertirá en el punto único de consulta de información para lograr que la organización camine a paso firme.

- El análisis de los costos directos e indirectos permite tener una idea clara dela factibilidad y viabilidad del proyecto, lo que permitirá a la Empresa tomar la decisión de invertir y empezar a tener utilidades.

- La utilización y enlace a los diferentes programas desde el sistema de documentación operacional permite tener mayor desempeño diario en cada uno de las actividades que realizan cada uno de los usuarios dentro de la organización LAN ECUADOR.

4.2 Recomendaciones

- Establecer una capacitación del manejo del sistema documental, hacia todos los usuarios para mantenerlos informados sobre las funcionalidades, ventajas y beneficios que nos ofrece el software. Haciendo reflexionar que con lo que cuentan al momento pueden cumplir con las expectativas del usuario, solo se necesita seguir un trabajo ordenado realizando un registro, seguimiento y cierre de los incidentes en un tiempo límite, alimentando a la Base de Conocimientos para que vaya incrementado de información.

- Es recomendable continuar un control de los procesos por parte de los coordinadores, notificar a cada una de las áreas los objetivos que se planearon y corresponde cumplir para obtener la satisfacción del usuario.

- En casos de incorporar nuevas funciones al sistema se recomienda tener en cuenta los modelos del sistema y los manuales técnicos como también de usuarios para comprender los elementos que conforman. Aprovechando al máximo todas las ventajas que ofrece el Sistema a todos los usuarios y actualizándolo de manera correcta con información real, manteniendo la Base de conocimientos actualizada.

- El administrador encargado de crear los usuarios, se recomienda establecer los usuarios indispensables y necesarios para ingresar al sistema.

- Se recomienda a la empresa LAN ECUADOR, hacer uso de los resultados del diseño para la implementación, realizar pruebas pilotos y determinar el nivel de señal para verificar si cumple con los requerimientos de confiabilidad y desempeño de la misma.

- Para fortalecer las competencias profesionales en este campo, se recomienda realizar un análisis ordenado de cada uno de las funciones del sistema de documentación operacional, permitiendo tener una idea más clara de la magnitud del proyecto.

Referencias

Carlos Javier Pes Rivas. (2002). Lenguaje de alto nivel. Carlos Javier Pes Rivas. (2002). Valencia, España: Recuperado el 12 de Agosto del 2012 de http://www.carlospes.com/ minidiccionario/lenguaje_de_alto_nivel.php

Castillo Oswaldo, Figueroa Daniel, Sevilla Héctor. (14 de Septiembre del 2011). Programación Extrema. Castillo Oswaldo, Figueroa Daniel, Sevilla Héctor. (14 de Septiembre del 2011). Perú: Recuperado el 11 de Abril del 2012 de http://programacionextrema.tripod.com/fases.htm

Craig Larman (2004). Introducción al análisis y diseño orientado a objetos, UML y Patrones Versión en español. Craig Larman (2004). México: Recuperado el 09 de Junio del 2012

Gabriel Bulfon (22 de Abril del 2011). Introduccion a C#. Gabriel Bulfon (22 de Abril del 2011). Argentina: Recuperado el 15 de Mayo del 2012 de http://www.dotech.com.ar/notes/CSharp1.htm

Jacobson Ivar, Booch G., Rumbaugh J. (2000). El Proceso Unificado De Desarrollo de Software. Jacobson Ivar, Booch G., Rumbaugh J. (2000). Madrid, España: Recuperado el 29 de Mayo del 2012

James Donovan Wells. (28 de Septiembre de 2009). Extreme Programming. James Donovan Wells. (2009). Estados Unidos: Recuperado el 21 de Abril del 2012 de http://www.extremeprogramming.org/rules.html.

José Antonio González Seco (2006). El lenguaje de programación C#. José Antonio González Seco (2006). España: Recuperado el 14 de Junio del 2012 de http://dis.um.es/~bmoros/privado/bibliografia/LibroCsharp.pdf

María A. Mendoza Sánchez. (7 de Junio del 2004). Metodologías De Desarrollo De Software. María A. Mendoza Sánchez. (2004). Perú: Recuperado el 11 de Mayo del 2012 de http://www.informatizate.net/articulos/metodologias_de_desarrollo_de_software_07062004.html

María Pinto. (26 de Abril del 2011). Bases de datos. María Pinto. (26 de Abril del 2011). Granada, España: Recuperado el 08 de Junio del 2012 de http://www.mariapinto.es/e-coms/bases_datos.htm

Pablo Figueroa. (1997). Conceptos en un Diagrama de Implementación. Pablo Figueroa. (1997). Bogotá, Colombia: Recuperado el 05 de Julio del 2012 de http://webdocs.cs.ualberta.ca/~pfiguero/soo/uml/implementacion01.html

Patricio Salinas. (30 de Octubre de 1996). Modelo de Clases. Patricio Salinas. (30 de Octubre de 1996). Santiago, Chile: Recuperado el 02 de Mayo del 2012 de http://www.dcc.uchile.cl/~psalinas/uml/modelo.html

Rayas, L. y J. L. Abreu. (Marzo del 2008) “Modelo para la exposición de la materia de ingeniería de software”. Revista Daena (International Journal of Good Conscience: 3(1) 701-750. Marzo 2008, p.720). México: Recuperado el 20 de Mayo del 2012

Rima Natural. (2012). Proceso de digitalización. Rima Natural. (2012). México: Recuperado el 19 de Junio del 2012 de http://www.rimanatural.mx/proceso.html

Ron Jeffries. (8 de Noviembre del 2001). Extreme Programming. Ron Jeffries. (8 de Noviembre del 2001). Estados Unidos: Recuperado el 11 de Abril del 2012 http://xprogramming.com/book/whatisxp/

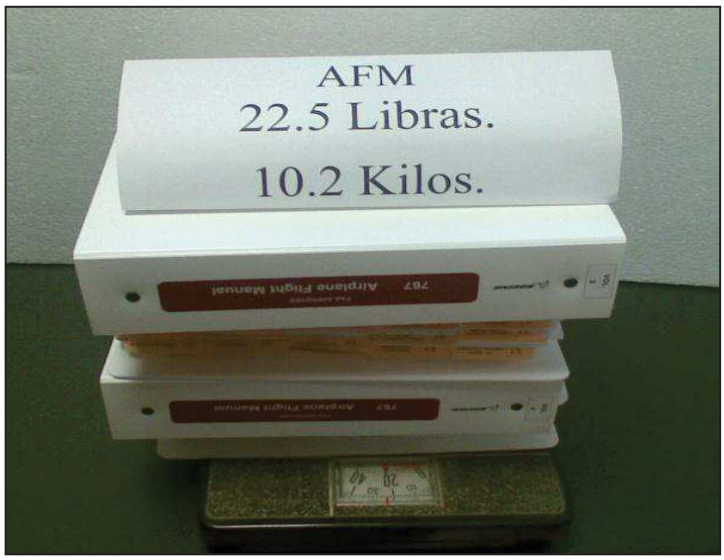
Rumbaugh James, Jacobson I., Booch G. (1999). El Lenguaje de Modelado Manual de Referencia. James, Jacobson I., Booch G. (1999) 2da edición. Madrid, España: Recuperado el 25 de Abril del 2012

Unidad Educativa Nacional Bolivariana "Atapaima". (3 de Febrero del 2011), Hardware y Software. Unidad Educativa Nacional Bolivariana "Atapaima". (3 de Febrero del 2011). Portuguesa, Venezuela: Recuperado el 22 de Julio del 2012 de http://uenbatapaima1903.blogspot.com/2011/02/hardware-y-software.html

Universo. (11 de Junio del 2012). Programación extrema: “Metodología para desarrollo ágil de aplicaciones”. Universo el periódico de los universitarios (año 11 N° 486). México: Universidad Veracruz Recuperado el 11 de Abril del 2012 de http://www.uv.mx/universo/486/infgral/infgral_15.html

ANEXOS I

Cambio Documentación Manual a Notebook



AFM

JEPPESEN

Maletín Lan Ecuador

Manuales

ANEXOS II

INSTRUCTIVO PARA EL USO DEL SISTEMA DE DOCUMENTACION OPERACIONAL

Este programa ha sido desarrollado con el objetivo de facilitar la búsqueda y el manejo de la documentación operacional por parte de la tripulación. El programa ha sido realizado en el lenguaje de programación C# y su despliegue será por medio del aplicativo Acrobat Reader.

Ejecución del Programa

El programa Documentación Operacional se ejecuta automáticamente al encender el Notebook. La pantalla principal del software queda maximizada sobre el escritorio.

Opciones de Ingreso

El menú principal (o primera pantalla) del programa nos muestra la opción de ingresar el usuario con su respectiva contraseña.

Elección de la flota

La siguiente pantalla permite elegir la Flota a la que se ingresará, con su respectivo modelo y matrícula.

Ingresara al menú principal

Regresara a la opción de ingreso

Elección de los manuales

Para la selección de los Manuales programa muestra una pantalla, en donde las opciones son las siguientes:

Regresara a La Opción Elección De Flota

Manuales del Avión: Se visualizan todos los manuales específicos del avión.

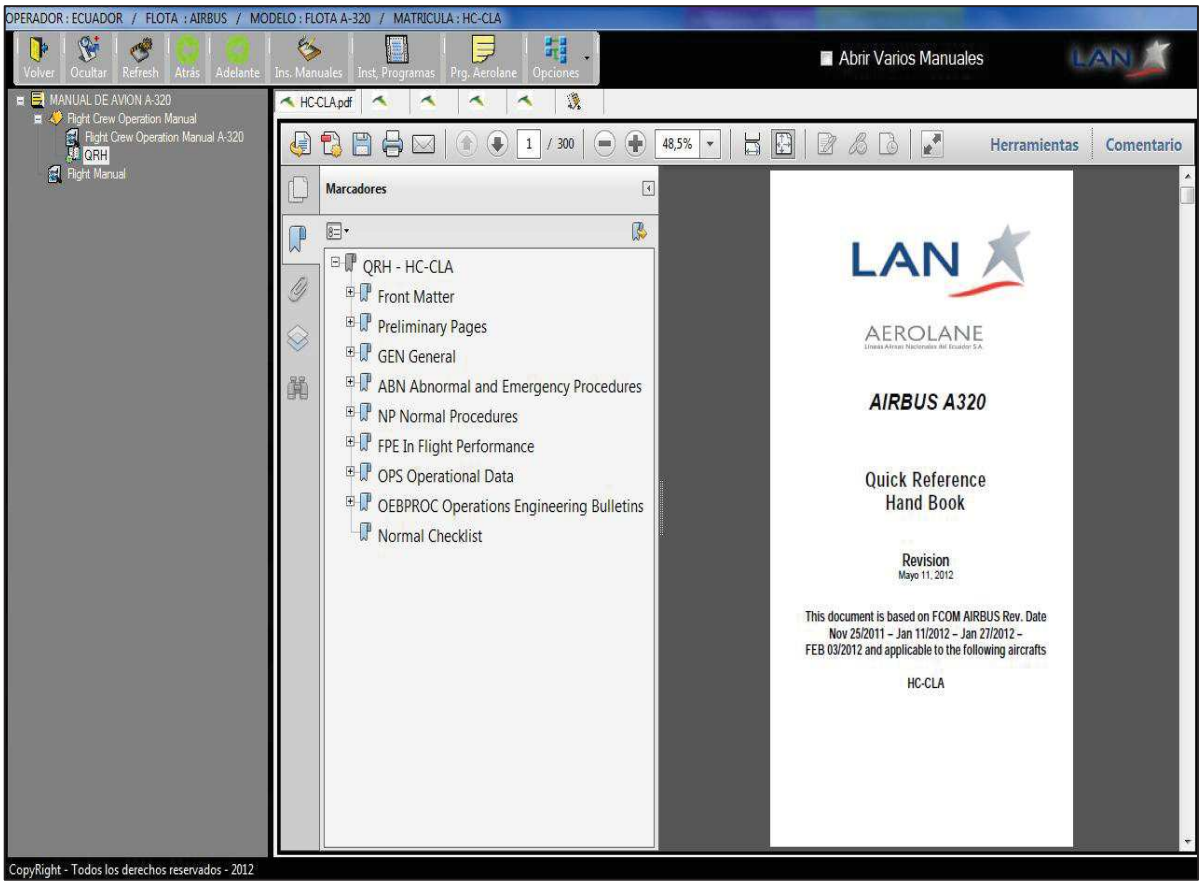
Manuales de la Compañía: se visualizan los manuales de la empresa, tales como:

MEL, Especificaciones Operativas, Manual de Operaciones, etc.

FOVE: ejecuta el programa FOVE.

LPC BROWSER: ejecuta el programa BROWSER.

Visualización de los manuales



El sistema de Documentación Operacional tiene una pantalla en donde se visualiza el manual requerido. Esta área es manipulada por el aplicativo Adobe Reader y sus características y funcionalidades se detallan a continuación.

El Área donde se elige el manual a visualizar. Donde se visualiza el manual elegido

La pantalla de representación de los manuales la podemos dividir en tres partes como a continuación se señala:

Barra de herramientas: ubicada en la parte superior de la pantalla y que contiene los siguientes botones:

- ✓ ***Volver:*** Sale del Browser y vuelve a la pantalla anterior.
- ✓ ***Ocultar:*** Aumenta el tamaño del área del browser ocupando el aérea de los
- ✓ ***Refresh:*** Actualiza la pagina
- ✓ ***Adelante:*** Cuando se han abierto más de un manual, va hacia el siguiente.
- ✓ ***Atrás:*** Cuando se han abierto más de un manual vuelve al anterior.
- ✓ tópicos con el fin de visualizar la información de mejor forma.
- ✓ ***Inst Manuales:*** Muestra instructivo de manuales.
- ✓ ***Inst Programas:*** Muestra instructivo de programas.
- ✓ ***Prg.Aerolane:*** Muestra instructivo de programas aerolane.
- ✓ ***Opciones:*** Se despliega las opciones de consultas, parámetros, modelos, opciones de modelos, etc.

Área de Tópicos: es la ubicada en la parte izquierda de la pantalla y es donde se listan todos los manuales que fueron solicitados por el usuario. Para su llamado y visualización en el área del browser basta con hacer clic con el botón derecho del mouse sobre su posición.

Área del Browser: que corresponde a la parte central donde el programa carga el aplicativo Acrobat Reader para desplegar el manual solicitado por el usuario. Todas las opciones (Imprimir, Guardar, Enviar por E-Mail, Zoom, etc.), pertenecen a este software y su descripción se encuentra en Instructivo de uso de Acrobat Reader.

Salir del Programa

El Botón Salir, ubicado en la pantalla principal nos permite salir del programa de Documentación Operacional y apagar el Notebook. Al presionar este botón el programa nos señalará si realmente deseamos salir.

Salir del programa

Regresar a la opción de usuario

ANEXOS III

CODIGO FUENTE EN C#

Implementación:

Se debe realizar la implementación del código fuente en Microsoft C# para su correcto funcionamiento A continuación se presenta el código fuente de los “froms” y posterior de las clases.

frmArbol.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using RptAppServidor.Clases;

namespace RptAppServidor.LINEA_AEREA
{
publicpartialclassfrmArbol : Form
   {
privateint idOperador = 0;
privateint idFlota = 0;
privateint idModelo = 0;
privateint idMatricula;


public frmArbol( int idOperador, int flota, int idModelo, int idMatricula)
{

this.idOperador = idOperador;
this.idFlota = flota;
this.idModelo = idModelo;
this.idMatricula = idMatricula;
InitializeComponent();
      }

privatevoid frmArbol_Load(object sender, EventArgs e)
{

Arbol objArbol = newArbol(null);
          objArbol.buscarTipo(this);
DialogoOpen.Filter = "PDF Files (*.pdf;*.pdf)|";
this.arbolDoc.ExpandAll();

Inicial objInicial = newInicial(this,1);
          objInicial.BuscarOperador();



      }
```

```
privatevoid cboTipo_SelectedIndexChanged(object sender, EventArgs e)
        {
            cboLinea_SelectedIndexChanged(sender, e);
        }

privatevoid cboLinea_SelectedIndexChanged(object sender, EventArgs e)
        {
if (this.cboOperador.SelectedIndex >= 0 &&this.cboFlota.SelectedIndex >= 0
             &&this.cboModelo.SelectedIndex >= 0 &&this.cboMatricula.SelectedIndex >= 0
&&this.cboTipo.SelectedIndex >= 0)
            {
this.arbolDoc.Nodes.Clear();
this.arbolDoc.ShowNodeToolTips = true;
Arbol objArbol = newArbol(  this.cboTipo.SelectedValue.ToString(),
int.Parse( this.cboOperador.SelectedValue.ToString() ) ,
int.Parse( this.cboFlota.SelectedValue.ToString() ) ,
int.Parse( this.cboModelo.SelectedValue.ToString() ) ,
int.Parse( this.cboMatricula.SelectedValue.ToString())  ) ;
                objArbol.CrearDataSet();
                objArbol.CrearNodosDelPadre(0, null, this.arbolDoc);
                arbolDoc.ExpandAll();
            }
        }

privatevoid cmdBuscar_Click(object sender, EventArgs e)
        {
            DialogoOpen.ShowDialog(this);
this.txtArchivo.Text = DialogoOpen.FileName;
        }

privatevoid arbolDoc_AfterSelect(object sender, TreeViewEventArgs e)
        {
if (this.arbolDoc.SelectedNode != null)
            {
string[] datos = arbolDoc.SelectedNode.Tag.ToString().Split('|');
this.txtArchivo.Text = datos[0];
this.txtDecripcionNodo.Text = this.arbolDoc.SelectedNode.Text;
this.txtidNodo.Text = datos[1];
this.txtRevision.Text = datos[2];
if (datos[3] == "S")
this.chkRemarcar.Checked = true;
else
this.chkRemarcar.Checked = false;
            }
        }

privatevoid cmdLimpiar_Click(object sender, EventArgs e)
        {
this.txtidNodo.Text = string.Empty;
this.txtArchivo.Text = string.Empty;
this.txtDecripcionNodo.Text = string.Empty;
this.txtRevision.Text = string.Empty;
this.chkRemarcar.Checked = false;
this.chkAnexarArchivo.Checked = true;
        }

privatevoid button1_Click(object sender, EventArgs e)
        {
if (this.arbolDoc.SelectedNode == null
                || this.cboOperador.SelectedIndex < 0
                || this.cboFlota.SelectedIndex < 0
```

```
                || this.cboModelo.SelectedIndex < 0
                || this.cboMatricula.SelectedIndex < 0
                || this.cboTipo.SelectedIndex < 0
                 || this.cboOperador.Text == string.Empty
                 || this.cboFlota.Text == string.Empty
                 || this.cboModelo.Text == string.Empty
                 || this.cboMatricula.Text == string.Empty
                 || this.cboTipo.Text == string.Empty)
{
MessageBox.Show("No se ha ingresado toda la información necesaria.");
return ;
        }

Arbol objArbol =
          newArbol(this.cboTipo.SelectedValue.ToString(),int.Parse(this.cboOperador.SelectedValue.
          ToString()),
          int.Parse(this.cboFlota.SelectedValue.ToString()),int.Parse(this.cboModelo.SelectedValue.T
          oString()),int.Parse(this.cboMatricula.SelectedValue.ToString()));
Boolean respuesta = false;

if (this.txtidNodo.Text == string.Empty)
respuesta = objArbol.InsertarNodo(this);
else
            respuesta = objArbol.ActualizarNodo(this);
if (respuesta)
        {
this.arbolDoc.Nodes.Clear();
this.arbolDoc.ShowNodeToolTips = true;
            objArbol.CrearDataSet();
objArbol.CrearNodosDelPadre(0, null, this.arbolDoc);
this.arbolDoc.ExpandAll();
        }
      }

privatevoid cmdEliminar_Click(object sender, EventArgs e)
      {
Arbol objArbol = newArbol(this.cboTipo.SelectedValue.ToString(), this.idOperador, this.idFlota,
          this.idModelo, this.idMatricula);

if (objArbol.EliminarrNodo(this))
        {
this.arbolDoc.Nodes.Clear();
this.arbolDoc.ShowNodeToolTips = true;
            objArbol.CrearDataSet();
objArbol.CrearNodosDelPadre(0, null, this.arbolDoc);
this.arbolDoc.ExpandAll();
            cmdLimpiar_Click(sender, e);
        }


      }

privatevoid cmdSalir_Click(object sender, EventArgs e)
      {
this.Close();
      }

privatevoid chkAnexarArchivo_CheckedChanged(object sender, EventArgs e)
{
if (this.chkAnexarArchivo.Checked)
        {
```

```
this.cmdBuscar.Enabled = true;

        }
else
        {
this.cmdBuscar.Enabled = false;

        }

    }

privatevoid groupBox3_Enter(object sender, EventArgs e)
    {

    }

privatevoid cboOperador_SelectedIndexChanged(object sender, EventArgs e)
    {
if (this.cboOperador.SelectedIndex >= 0)
        {
Inicial objInicial = newInicial(this,1);
objInicial.BuscarFlotaPorOperador(this.cboOperador.SelectedValue.ToString());
this.cboFlota.Text = string.Empty;
this.cboModelo.Text = string.Empty;
this.cboMatricula.Text = string.Empty;
        }
this.arbolDoc.Nodes.Clear();

    }

privatevoid cboFlota_SelectedIndexChanged(object sender, EventArgs e)
    {
if (this.cboFlota.SelectedIndex >= 0)
        {
Inicial objInicial = newInicial(this,1);
            objInicial.BuscarModeloPorFlota(this.cboFlota.SelectedValue.ToString());
this.cboModelo.Text = string.Empty;
this.cboMatricula.Text = string.Empty;
        }

    }

privatevoid cboModelo_SelectedIndexChanged(object sender, EventArgs e)
    {
if (this.cboModelo.SelectedIndex >= 0)
        {
Inicial objInicial = newInicial(this,1);
objInicial.BuscarMatriculaPorModelo(this.cboModelo.SelectedValue.ToString());
this.cboMatricula.Text = string.Empty;

        }
this.arbolDoc.Nodes.Clear();
    }

privatevoid cboMatricula_SelectedIndexChanged(object sender, EventArgs e)
    {
        cboTipo_SelectedIndexChanged(sender, e);
    }

privatevoid cmdNodoRaiz_Click(object sender, EventArgs e)
    {
```

```
if (this.cboOperador.SelectedIndex < 0
            || this.cboFlota.SelectedIndex < 0
            || this.cboModelo.SelectedIndex < 0
            || this.cboMatricula.SelectedIndex < 0
            || this.cboTipo.SelectedIndex < 0
            || this.cboOperador.Text == string.Empty
            || this.cboFlota.Text == string.Empty
            || this.cboModelo.Text == string.Empty
            || this.cboMatricula.Text == string.Empty
            || this.cboTipo.Text == string.Empty)
{
MessageBox.Show("No se ha ingresado toda la información necesaria.");
return;
          }

Arbol objArbol = newArbol(this.cboTipo.SelectedValue.ToString(),
            int.Parse(this.cboOperador.SelectedValue.ToString()),
            int.Parse(this.cboFlota.SelectedValue.ToString()),
            int.Parse(this.cboModelo.SelectedValue.ToString()),
            int.Parse(this.cboMatricula.SelectedValue.ToString()));
Boolean respuesta = false;
          respuesta = objArbol.CrearNodoRaiz(this);

if (respuesta)
          {
this.arbolDoc.Nodes.Clear();
this.arbolDoc.ShowNodeToolTips = true;
              objArbol.CrearDataSet();
objArbol.CrearNodosDelPadre(0, null, this.arbolDoc);
this.arbolDoc.ExpandAll();
          }

      }
//funcion para capturar evento y no se pueda cambiar de tamaño la forma
protectedoverridevoid WndProc(refMessage m)
      {
if (m.Msg == 0xa3 && m.WParam == newIntPtr(2))
          {
return;
          }
base.WndProc(ref m);
}
   }
}
```

frmConsultaBitacora.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using RptAppServidor.Clases;

namespace RptAppServidor.LINEA_AEREA
```

```
{
publicpartialclassfrmConsultaBitacora : Form
{
public frmConsultaBitacora()
      {
         InitializeComponent();
Util.lookGrilla(this.grdDataBitacora);
}

privatevoid cmdSalir_Click(object sender, EventArgs e)
      {
this.Close();
}

privatevoid cmdBuscarBitacora_Click(object sender, EventArgs e)
{
Consultas objConsultas = newConsultas(this);
         objConsultas.buscarBitacora();
      }
//funcion para capturar evento y no se pueda cambiar de tamaño la forma
protectedoverridevoid WndProc(refMessage m)
      {
if (m.Msg == 0xa3 && m.WParam == newIntPtr(2))
         {
return;
         }
base.WndProc(ref m);
}
   }
}
```

frmDocumentacion:

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Security.Policy;
using RptAppServidor.Clases;

namespace RptAppServidor.LINEA_AEREA
{
publicpartialclassfrmDocumentacion : Form
   {
//System.Data.DataSet dataSetArbol;
//string idLinea = string.Empty;
string idTipo = string.Empty;
privateint idusuario = 0;
privatestring login = string.Empty;
privatestring nombreUsuario = string.Empty;
privateint perfilUsuario = 0;
privateint idOperador = 0;
privateint idFlota = 0;
privateint idModelo = 0;
```

```
privateint idMatricula;
privatestring pathInstructivoManuales = string.Empty;
privatestring pathInstructivoProgramas = string.Empty;
privatestring pathInstructivoProgramasAerolane = string.Empty;
privatestring imagenInicial = string.Empty;
privatestring paginaInicial = string.Empty;


public frmDocumentacion(string idTipo, int idusuario, string login, string nombreUsuario, int
          perfilUsuario, int idOperador, int flota, int idModelo, int idMatricula, string info)
{
         InitializeComponent();
this.idusuario =  idusuario;
this.login = login;
this.nombreUsuario = nombreUsuario;
this.perfilUsuario = perfilUsuario;
this.idOperador = idOperador;
this.idFlota = flota;
this.idModelo = idModelo;
this.idMatricula = idMatricula;
this.idTipo = idTipo;
this.arbolDoc.ShowNodeToolTips = true;
Arbol objArbol = newArbol(this.idTipo,this.idOperador,this.idFlota,this.idModelo,this.idMatricula);
          objArbol.CrearDataSet();
          objArbol.CrearNodosDelPadre(0, null, this.arbolDoc);
this.arbolDoc.ExpandAll();
this.Text = info;
Consultas objConsultas = newConsultas();
string puedeHacerMantenimiento = objConsultas.ConsultaGenerica(" select per_mantenimiento from
           perfil where per_id = " + this.perfilUsuario.ToString());
if (puedeHacerMantenimiento == "SI")
this.toolBarMantenimiento.Visible = true;
else
this.toolBarMantenimiento.Visible = false;

try
         {
             pathInstructivoManuales = objArbol.ConsultaPath("PIM");
             pathInstructivoProgramas = objArbol.ConsultaPath("PIP");
pathInstructivoProgramasAerolane = objArbol.ConsultaPath("PIA");
             imagenInicial = objArbol.ConsultaPath("IMG");
paginaInicial = objArbol.ConsultaPath("PAG");


Uri uri1 = newUri(paginaInicial);
this.webBrowser1.Url = uri1;
this.webBrowser2.Url = uri1;
this.webBrowser3.Url = uri1;
this.webBrowser4.Url = uri1;
this.webBrowser5.Url = uri1;
this.webBrowser6.Url = uri1;

Size objSize = newSize(this.Size.Width - (this.arbolDoc.Size.Width + 15), this.Size.Height - 100);
this.tabVisor.Size = objSize;

Size objSizeArbol = newSize(this.arbolDoc.Size.Width, this.tabVisor.Size.Height);
this.arbolDoc.Size = objSizeArbol;

         }
catch (Exception)
{
```

```
MessageBox.Show("Error al recuperar información de Manuales");
}
      }


privatevoid frmDocumentacion_Load(object sender, EventArgs e)
      {


      }


privatevoid treeView1_AfterSelect(object sender, TreeViewEventArgs e)
      {


      }

privatevoid barraHerramientas_ItemClicked(object sender, ToolStripItemClickedEventArgs e)
      {

      }


privatevoid arbolDoc_AfterSelect(object sender, TreeViewEventArgs e)
      {
string documento = string.Empty;
string[] datosNodo = arbolDoc.SelectedNode.Tag.ToString().Split('|');
documento = datosNodo[0];


if (arbolDoc.SelectedNode != null)
if (arbolDoc.SelectedNode.Tag.ToString().Trim() != string.Empty)
            {


string[] datos = arbolDoc.SelectedNode.Tag.ToString().Split('|');
string[] pathArchivo = datos[0].Split('\\');
                  documento = pathArchivo[pathArchivo.Length - 1];

for (int i = 0; i <this.tabVisor.TabCount; i++)
if (this.tabVisor.TabPages[i].Text == documento)
                  {
this.tabVisor.TabIndex = i;
this.tabVisor.SelectedIndex = i;
return;
                  }


if (datos[0] == string.Empty)
return;

Uri uri1 = newUri(datos[0]);

if (!this.chkAbrirMas.Checked)
                  {
this.webBrowser1.Url = uri1;
this.tabVisor.SelectedIndex = 0;
this.tabVisor.TabPages[0].Text = documento;
return;
                  }
```

```
if (this.tabVisor.TabPages[0].Text == "")
                {
this.webBrowser1.Url = uri1;
this.tabVisor.SelectedIndex = 0;
this.tabVisor.TabPages[0].Text = documento;
return;
                }
if (this.tabVisor.TabPages[1].Text == "")
                {
this.webBrowser2.Url = uri1;
this.tabVisor.SelectedIndex = 1;
this.tabVisor.TabPages[1].Text = documento;
return;
                }

if (this.tabVisor.TabPages[2].Text == "")
                {
this.webBrowser3.Url = uri1;
this.tabVisor.SelectedIndex = 2;
this.tabVisor.TabPages[2].Text = documento;
return;
                }

if (this.tabVisor.TabPages[3].Text == "")
                {
this.webBrowser4.Url = uri1;
this.tabVisor.SelectedIndex = 3;
this.tabVisor.TabPages[3].Text = documento;
return;
                }

if (this.tabVisor.TabPages[4].Text == "")
                {
this.webBrowser5.Url = uri1;
this.tabVisor.SelectedIndex = 4;
this.tabVisor.TabPages[4].Text = documento;
return;
                }


            }
        }

privatevoid btnOcultar_Click(object sender, EventArgs e)
        {
if (this.arbolDoc.Visible)
            {
this.arbolDoc.Visible = false;
//this.arbolDoc.Location.
Point objLocation = newPoint(this.arbolDoc.Location.X, this.arbolDoc.Location.Y);
this.tabVisor.Location = objLocation;

Size objSize = newSize(this.Size.Width - 15,this.Size.Height - 100 );
this.tabVisor.Size = objSize;
//1152, 753
            }
else
            {
this.arbolDoc.Visible = true;
Point objLocation = newPoint(288, 50);
```

```
this.tabVisor.Location = objLocation;

Size objSize = newSize(this.Size.Width - (this.arbolDoc.Size.Width + 15), this.tabVisor.Size.Height);
this.tabVisor.Size = objSize;

Size objSizeArbol = newSize(this.arbolDoc.Size.Width, this.tabVisor.Size.Height);
this.arbolDoc.Size = objSizeArbol;

          }
      }

privatevoid cmdBuscarUsuarios_Click(object sender, EventArgs e)
      {
//Arbol objArbol = new Arbol(this);
//objArbol.buscarUsuarios();

      }

privatevoid cmdBuscarBitacora_Click(object sender, EventArgs e)
{
//Arbol objArbol = new Arbol(this);
//objArbol.buscarBitacora();
      }

privatevoid btnVolver_Click(object sender, EventArgs e)
      {
this.Close();
      }

privatevoid btnLineaSub_Click(object sender, EventArgs e)
      {

      }

privatevoid btnSubArbol_Click(object sender, EventArgs e)
      {
//this.Visible = false;
frmArbol objFrmArbol = newfrmArbol(this.idOperador, this.idFlota, this.idModelo, this.idMatricula);
Util.AbrirVentana(objFrmArbol,1);
this.arbolDoc.Nodes.Clear();
this.arbolDoc.ShowNodeToolTips = true;
Arbol objArbol = newArbol(this.idTipo,this.idOperador,this.idFlota,this.idModelo,this.idMatricula);
          objArbol.CrearDataSet();
          objArbol.CrearNodosDelPadre(0, null, this.arbolDoc);
this.arbolDoc.ExpandAll();
//this.Visible = true;
      }

privatevoid grdUsuarios_CellContentClick(object sender, DataGridViewCellEventArgs e)
{

      }

privatevoid btnSubConsultaUsuarios_Click(object sender, EventArgs e)
{
frmConsultarUsuarios objFrmConsultaUsuarios = newfrmConsultarUsuarios();
Util.AbrirVentana(objFrmConsultaUsuarios,1);
      }

privatevoid btnSubConsultaBitacora_Click(object sender, EventArgs e)
      {
```

```
frmConsultaBitacora objFrmConsultaBitacora = newfrmConsultaBitacora();
Util.AbrirVentana(objFrmConsultaBitacora,1);
}

privatevoid btnRefresh_Click(object sender, EventArgs e)
      {
this.arbolDoc.Nodes.Clear();
this.arbolDoc.ShowNodeToolTips = true;
Arbol objArbol = newArbol(this.idTipo, this.idOperador, this.idFlota, this.idModelo, this.idMatricula);
          objArbol.CrearDataSet();
          objArbol.CrearNodosDelPadre(0, null, this.arbolDoc);
this.arbolDoc.ExpandAll();

for (int i = 0; i<this.tabVisor.TabCount; i++)
this.tabVisor.TabPages[i].Text = string.Empty;

          paginaInicial = objArbol.ConsultaPath("PAG");


Uri uri1 = newUri(paginaInicial);
this.webBrowser1.Url = uri1;
this.webBrowser2.Url = uri1;
this.webBrowser3.Url = uri1;
this.webBrowser4.Url = uri1;
this.webBrowser5.Url = uri1;
this.webBrowser6.Url = uri1;

this.tabVisor.SelectedIndex = 0;
this.chkAbrirMas.Checked = false;

      }

privatevoid toolStripMenuItem2_Click(object sender, EventArgs e)
{
frmModelo objLIneaAerea = newfrmModelo(1, 1);
Util.AbrirVentana(objLIneaAerea, 1);
}

privatevoid toolStripMenuItem3_Click(object sender, EventArgs e)
{
frmFlota objFrmFlota = newfrmFlota(1,1);
Util.AbrirVentana(objFrmFlota, 1);
}

privatevoid toolStripMenuItem4_Click(object sender, EventArgs e)
{
frmOperador objFrmPais = newfrmOperador(1, 1);
Util.AbrirVentana(objFrmPais, 1);
}

privatevoid toolStripMenuItem5_Click(object sender, EventArgs e)
{
FrmMatricula objFrmMatricula = newFrmMatricula(1,1);
Util.AbrirVentana(objFrmMatricula, 1);
}

privatevoid toolStripMenuItem6_Click(object sender, EventArgs e)
{
frmParametro objFrmParametro = newfrmParametro(1, 1);
Util.AbrirVentana(objFrmParametro, 1);
}
```

```
privatevoid toolStripMenuItem7_Click(object sender, EventArgs e)
{
frmTipo objFrmTipo = newfrmTipo(1, 1);
Util.AbrirVentana(objFrmTipo, 1);
}

privatevoid toolStripMenuItem8_Click(object sender, EventArgs e)
{
FrmUsuario objFrmUsuario = newFrmUsuario(1, 1);
Util.AbrirVentana(objFrmUsuario, 1);
}

privatevoid cmdInsProgramas_Click(object sender, EventArgs e)
        {
this.tabVisor.SelectedIndex = 5;
Uri uri = newUri(pathInstructivoProgramas);
this.webBrowser6.Url = uri;

        }

privatevoid cmdInstManuales_Click(object sender, EventArgs e)
        {
this.tabVisor.SelectedIndex = 5;
Uri uri = newUri(pathInstructivoManuales);
this.webBrowser6.Url = uri;

        }

privatevoid cmdPrgAerolane_Click(object sender, EventArgs e)
        {
this.tabVisor.SelectedIndex = 5;
Uri uri = newUri(pathInstructivoProgramasAerolane);
this.webBrowser6.Url = uri;
        }

privatevoid cmdBack_Click(object sender, EventArgs e)
        {
if (this.tabVisor.SelectedIndex == 0)
this.webBrowser1.GoBack();
if (this.tabVisor.SelectedIndex == 1)
this.webBrowser2.GoBack();
if (this.tabVisor.SelectedIndex == 2)
this.webBrowser3.GoBack();
if (this.tabVisor.SelectedIndex == 3)
this.webBrowser4.GoBack();
if (this.tabVisor.SelectedIndex == 4)
this.webBrowser5.GoBack();
        }

privatevoid cmdFoward_Click(object sender, EventArgs e)
        {
if (this.tabVisor.SelectedIndex == 0)
this.webBrowser1.GoForward();
if (this.tabVisor.SelectedIndex == 1)
this.webBrowser2.GoForward();
if (this.tabVisor.SelectedIndex == 2)
this.webBrowser3.GoForward();
if (this.tabVisor.SelectedIndex == 3)
this.webBrowser4.GoForward();
if (this.tabVisor.SelectedIndex == 4)
```

```
this.webBrowser5.GoForward();

      }

privatevoid arbolDoc_NodeMouseHover_1(object sender, TreeNodeMouseHoverEventArgs e)
      {
this.toolTipTextArbol.ToolTipTitle = string.Empty; //e.Node.ToolTipText;
this.toolTipTextArbol.Show(e.Node.ToolTipText, this);
// this.toolTipTextArbol.ShowAlways = true;

      }

privatevoid consultarUsuariosToolStripMenuItem_Click(object sender, EventArgs e)
{
frmConsultarUsuarios objFrmConsultaUsuarios = newfrmConsultarUsuarios();
Util.AbrirVentana(objFrmConsultaUsuarios, 1);
      }

//funcion para capturar evento y no se pueda cambiar de tamaño la forma
protectedoverridevoid WndProc(refMessage m)
      {
if (m.Msg == 0xa3 && m.WParam == newIntPtr(2))
         {
return;
         }
base.WndProc(ref m);
      }

privatevoid opcionesModeloToolStripMenuItem_Click(object sender, EventArgs e)
{
frmOpcionesModelo objFrmOpcionesModelo = newfrmOpcionesModelo(1, 1);
Util.AbrirVentana(objFrmOpcionesModelo,1);
}

   }


}
```

frmFlotas.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using RptAppServidor.Clases;

namespace RptAppServidor
{
publicpartialclassfrmFlota : Form
   {
publicint usuarioActual = 0;
int PefilActual = 0;
publicbool flag = true;
public frmFlota(int p_usuario, int p_perfil)
      {
```

```
            InitializeComponent();
this.usuarioActual = p_usuario;
this.PefilActual = p_perfil;
Util.lookGrilla(this.grdData);
        }


privatevoid frmFlota_Load(object sender, EventArgs e)
        {
/*Turno objTurno = new Turno(this);
objTurno.BuscarTurnoCombos();*/
Flota objFlota = newFlota(this);
            objFlota.BuscarLinea();


        }


privatevoid cmdIngresar_Click(object sender, EventArgs e)
        {
Flota objFlota = newFlota(this);

if (this.txtCodigo.Text.Trim() == string.Empty)
{
                objFlota.operacioN = "I";
if (!objFlota.ValidarInformacion())
return;
              objFlota.Insertar();
            }
else
            {
              objFlota.operacioN = "U";
if (!objFlota.ValidarInformacion())
return;
              objFlota.Actualizar();
}
this.cboLineaAerea_SelectedIndexChanged(sender, e);
        }

privatevoid txtBuscar_TextChanged(object sender, EventArgs e)
        {

        }


privatevoid cmdEliminar_Click(object sender, EventArgs e)
{
Flota objflota = newFlota(this);
            objflota.operacioN = "D";
objflota.Eliminar();
this.cboLineaAerea_SelectedIndexChanged(sender, e);
        }
privatevoid grdData_CellClick(object sender, DataGridViewCellEventArgs e)
        {
if (this.grdData.Rows.Count > 0 &&this.grdData.SelectedRows.Count > 0)
            {
this.txtCodigo.Text = this.grdData.SelectedRows[0].Cells[0].Value.ToString();
this.txtDescripcion.Text = this.grdData.SelectedRows[0].Cells[2].Value.ToString();
            }
        }
```

```
privatevoid grdData_SelectionChanged(object sender, EventArgs e)
        {
            grdData_CellClick(sender, null);
        }


privatevoid grdData_CellContentClick(object sender, DataGridViewCellEventArgs e)
        {

        }

privatevoid cmdLimpiar_Click(object sender, EventArgs e)
{
Flota objflota = newFlota(this);
objflota.limpiar();
}

privatevoid cmdCerrar_Click(object sender, EventArgs e)
        {
this.Close();
        }

privatevoid groupBox1_Enter(object sender, EventArgs e)
        {

        }

privatevoid cboLineaAerea_SelectedIndexChanged(object sender, EventArgs e)
        {
if (this.cboOperador.SelectedIndex >= 0)
            {
Flota objFlota = newFlota(this);
                objFlota.LlenarGrilla(this.cboOperador.SelectedValue.ToString());
            }
        }

privatevoid groupBox2_Enter(object sender, EventArgs e)
        {

        }

privatevoid groupBox3_Enter(object sender, EventArgs e)
        {

        }

privatevoid txtCodigo_TextChanged(object sender, EventArgs e)
        {

        }

privatevoid label3_Click(object sender, EventArgs e)
{

        }

//funcion para capturar evento y no se pueda cambiar de tamaño la forma
protectedoverridevoid WndProc(refMessage m)
        {
if (m.Msg == 0xa3 && m.WParam == newIntPtr(2))
```

```
        {
return;
        }
base.WndProc(ref m);
}


   }
}
```

frmInicial.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using RptAppServidor.Clases;

namespace RptAppServidor.LINEA_AEREA
{
publicpartialclassfrmInicial : Form
   {
privateint idusuario = 0;
privatestring login = string.Empty;
privatestring nombreUsuario = string.Empty;
privateint perfilUsuario = 0;
publicBoolean accionaMantenerse = false;

public frmInicial( int idusuario, string login, string nombreUsuario, int perfilUsuario )
      {
this.idusuario =  idusuario;
this.login = login;
this.nombreUsuario = nombreUsuario;
this.perfilUsuario = perfilUsuario;
         InitializeComponent();
      }

privatevoid frmInicial_Load(object sender, EventArgs e)
      {
Inicial objInicial = newInicial(this);
         objInicial.BuscarOperador();

Consultas objConsultas = newConsultas();


//this.cboOperator.Text = string.Empty;
this.cboFlota.Text = string.Empty;

this.cboOperator.SelectedValue = objConsultas.ConsultaGenerica("select o.ope_id from operador o,
          flota f, usuario u where o.ope_id = f.ope_id and f.flo_borrado = 0 and o.ope_borrado = 0 and
          f.flo_id = u.flo_id and u.usu_id = " + this.idusuario.ToString());
         cboPais_SelectedIndexChanged(sender, e);
```

```
this.cboFlota.SelectedValue = objConsultas.ConsultaGenerica("select f.flo_id from usuario u, flota f
          where f.flo_id = u.flo_id  and f.flo_borrado = 0 and u.usu_borrado = 0 and usu_id = " +
          this.idusuario.ToString());
this.cboOperator.Enabled = false;
this.cboFlota.Enabled = false;
//objInicial.BuscarActualizaciones(this.cboFlota.SelectedValue.ToString());

this.txtRevision.Text = objConsultas.ConsultaPath("REV");

      }

privatevoid cboLinea_SelectedIndexChanged(object sender, EventArgs e)
      {
if (cboFlota.SelectedIndex >= 0)
         {
Inicial objInicial = newInicial(this);
            objInicial.BuscarModeloPorFlota(this.cboFlota.SelectedValue.ToString());
this.cboModelo.Text = string.Empty;
this.cboMatricula.Text = string.Empty;

         }

      }

privatevoid cboPais_SelectedIndexChanged(object sender, EventArgs e)
      {
if (this.cboOperator.SelectedIndex >= 0)
         {
Inicial objInicial = newInicial(this);
            objInicial.BuscarFlotaPorOperador(this.cboOperator.SelectedValue.ToString());
//this.cboFlota.Text = string.Empty;
this.cboModelo.Text = string.Empty;
this.cboMatricula.Text = string.Empty;
         }
      }

privatevoid cmdEntrar_Click(object sender, EventArgs e)
{
if (this.cboOperator.SelectedIndex < 0)
         {
MessageBox.Show("No se ha seleccionado una flota válida");
this.cboOperator.Focus();
return;
         }
if (this.cboFlota.SelectedIndex < 0 )
{
MessageBox.Show("No se ha seleccionado una flota válida");
this.cboFlota.Focus();
return;
         }
if (this.cboModelo.SelectedIndex < 0 || this.cboModelo.Text.Trim() == string.Empty)
         {
MessageBox.Show("No se ha seleccionado un modelo válido");
this.cboModelo.Focus();
return;
         }
if (this.cboMatricula.SelectedIndex < 0 || this.cboMatricula.Text.Trim() == string.Empty)
{
MessageBox.Show("No se ha seleccionado una matrícula válida");
this.cboMatricula.Focus();
return;
```

```
        }


frmMenuDocumentacion objFrmMenuDocumentacion = newfrmMenuDocumentacion(this.idusuario,
            this.login, this.nombreUsuario, this.perfilUsuario,
int.Parse(this.cboOperator.SelectedValue.ToString()),
int.Parse(this.cboFlota.SelectedValue.ToString()),
int.Parse(this.cboModelo.SelectedValue.ToString()),
int.Parse(this.cboMatricula.SelectedValue.ToString())
);
this.Visible = false;
Util.AbrirVentana(objFrmMenuDocumentacion,0);
this.Visible = true;


      }

privatevoid button1_Click(object sender, EventArgs e)
      {

      }

privatevoid cboModelo_SelectedIndexChanged(object sender, EventArgs e)
      {
if (this.cboModelo.SelectedIndex >= 0)
          {
Inicial objInicial = newInicial(this);
objInicial.BuscarMatriculaPorModelo(this.cboModelo.SelectedValue.ToString());
this.cboMatricula.Text = string.Empty;

          }
      }

privatevoid cboMatricula_SelectedIndexChanged(object sender, EventArgs e)
      {
//if (this.cboModelo.SelectedIndex >= 0)
//{
//    Inicial objInicial = new Inicial(this);
//    objInicial.BuscarMatriculaPorModelo(this.cboModelo.SelectedValue.ToString());
//}
      }

privatevoid timUpd_Tick(object sender, EventArgs e)
      {
//if (this.lstCambios.BackColor == System.Drawing.Color.AliceBlue)
//    this.lstCambios.BackColor = System.Drawing.Color.AntiqueWhite;
//else if (this.lstCambios.BackColor == System.Drawing.Color.Aquamarine)
//    this.lstCambios.BackColor = System.Drawing.Color.AntiqueWhite;
//else if (this.lstCambios.BackColor == System.Drawing.Color.AntiqueWhite)
//    this.lstCambios.BackColor = System.Drawing.Color.AliceBlue;
//else
//    this.lstCambios.BackColor = System.Drawing.Color.AntiqueWhite;

      }

privatevoid label6_Click(object sender, EventArgs e)
      {

      }

privatevoid label4_Click(object sender, EventArgs e)
```

```
        {

        }

privatevoid frmInicial_Activated(object sender, EventArgs e)
        {
//frmInicial_Load(sender, e);
this.cboModelo.Text = string.Empty;
this.cboMatricula.Text = string.Empty;

}
//funcion para capturar evento y no se pueda cambiar de tamaño la forma
protectedoverridevoid WndProc(refMessage m)
        {
if (m.Msg == 0xa3 && m.WParam == newIntPtr(2))
            {
return;
            }
base.WndProc(ref m);
        }

privatevoid cmdVolver_Click(object sender, EventArgs e)
{
Util.ApagarPC("Salir y apagar el computador?", "Regresar");

}
    }
}
```

frmLogin.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
using RptAppServidor.Clases;
using RptAppServidor.LINEA_AEREA;

namespace RptAppServidor
{
publicpartialclassfrmLogin : Form
    {

publicBoolean mantenerLogin = false;


public frmLogin()
        {
            InitializeComponent();
this.FormBorderStyle = FormBorderStyle.Fixed3D;
this.WindowState = FormWindowState.Maximized;
this.ControlBox = false;
        }

privatevoid cmdAceptar_Click(object sender, EventArgs e)
        {
StringBuilder strBuilder = newStringBuilder();
```

```
if (this.txtLogin.Text.Trim() == string.Empty)
{
MessageBox.Show("El usuario es mandatorio.");
this.txtLogin.Focus();
return;
        }
if (this.txtPassword.Text.Trim() == string.Empty)
        {
MessageBox.Show("El password es mandatorio.");
this.txtPassword.Focus();
return;
        }

Login objLogin = newLogin(this);
if (!objLogin.ValidarUsuario(this.txtLogin.Text, this.txtPassword.Text))
{
MessageBox.Show("Información incorrecta, no se puede iniciar sesión.");
return;
        }
//else
//this.Close();


    }



privatevoid frmLogin_Load(object sender, EventArgs e)
    {

this.txtLogin.Focus();
    }

privatevoid cmdCancelar_Click(object sender, EventArgs e)
{
if (Util.ApagarPC("Salir y apagar el computador?", "Cancelar"))
{
this.mantenerLogin = false;
this.Close();
}
else
this.mantenerLogin = true;
    }

//funcion para capturar evento y no se pueda cambiar de tamaño la forma
protectedoverridevoid WndProc(refMessage m)
    {
if (m.Msg == 0xa3 && m.WParam == newIntPtr(2))
        {
return;
        }
base.WndProc(ref m);
    }

privatevoid frmLogin_Activated(object sender, EventArgs e)
    {
this.txtLogin.Text = string.Empty;
this.txtPassword.Text = string.Empty;
    }
```

```
privatevoid toolStripStatusLabel1_Click(object sender, EventArgs e)
{

      }
   }
}
```

frmMatricula.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using RptAppServidor.Clases;

namespace RptAppServidor
{
publicpartialclassFrmMatricula : Form
   {

publicint usuarioActual = 0;
int PefilActual = 0;
publicbool flag = true;
public FrmMatricula(int p_usuario, int p_perfil)
      {

          InitializeComponent();
this.usuarioActual = p_usuario;
this.PefilActual = p_perfil;
Util.lookGrilla(this.grdData);
      }


privatevoid cmdIngresar_Click(object sender, EventArgs e)
      {
Matricula objMatricula = newMatricula(this);

if (this.txtCodigo.Text.Trim() == string.Empty)
{
              objMatricula.operacioN = "I";
if (!objMatricula.ValidarInformacion())
return;
              objMatricula.Insertar();
          }
else
          {
              objMatricula.operacioN = "U";
if (!objMatricula.ValidarInformacion())
return;
              objMatricula.Actualizar();
          }
this.cboModelo_SelectedIndexChanged(sender, e);
      }
```

```
privatevoid cmdEliminar_Click(object sender, EventArgs e)
{
Matricula objMatricula = newMatricula(this);
        objMatricula.operacioN = "D";
objMatricula.Eliminar();
this.cboModelo_SelectedIndexChanged(sender, e);
    }
privatevoid grdData_CellClick(object sender, DataGridViewCellEventArgs e)
    {
if (this.grdData.Rows.Count > 0 &&this.grdData.SelectedRows.Count > 0)
        {
//this.cboLineaAerea.Text = this.grdData.SelectedRows[0].Cells[2].Value.ToString();
this.txtCodigo.Text = this.grdData.SelectedRows[0].Cells[0].Value.ToString();
//this.cboDiaSemana.SelectedValue   = this.grdData.SelectedRows[0].Cells[8].Value.ToString();
this.txtDescripcion.Text = this.grdData.SelectedRows[0].Cells[2].Value.ToString();
        }
    }

privatevoid grdData_SelectionChanged(object sender, EventArgs e)
    {
        grdData_CellClick(sender, null);
    }


privatevoid grdData_CellContentClick(object sender, DataGridViewCellEventArgs e)
    {

    }

privatevoid cmdLimpiar_Click(object sender, EventArgs e)
{
Matricula objMatricula = newMatricula(this);
        objMatricula.limpiar();
}

privatevoid cmdCerrar_Click(object sender, EventArgs e)
    {
this.Close();
    }

privatevoid groupBox1_Enter(object sender, EventArgs e)
    {

    }

privatevoid cboPais_SelectedIndexChanged(object sender, EventArgs e)
    {

    }

privatevoid groupBox2_Enter(object sender, EventArgs e)
    {

    }

privatevoid groupBox3_Enter(object sender, EventArgs e)
    {

    }
```

```
privatevoid txtCodigo_TextChanged(object sender, EventArgs e)
        {

        }

privatevoid label3_Click(object sender, EventArgs e)
        {

        }

privatevoid FrmMatricula_Load(object sender, EventArgs e)
{
Matricula objMatricula = newMatricula(this);
            objMatricula.BuscarOperador();
}

privatevoid cboOperador_SelectedIndexChanged(object sender, EventArgs e)
        {
if (this.cboOperador.SelectedIndex >= 0)
            {
Matricula objMatricula = newMatricula(this);
                objMatricula.BuscarFlotaPorOperador(this.cboOperador.SelectedValue.ToString());
this.cboFlota.Text = string.Empty;
if (this.cboFlota.SelectedIndex < 0)
                {
this.cboFlota.Text = string.Empty;
                    objMatricula.LlenarGrilla("-1");
                }

            }
        }

privatevoid cboFlota_SelectedIndexChanged(object sender, EventArgs e)
        {
if (cboFlota.SelectedIndex >= 0)
            {
Matricula objMatricula = newMatricula(this);
                objMatricula.BuscarModeloPorFlota(this.cboFlota.SelectedValue.ToString());
this.cboModelo.Text = string.Empty;
if (this.cboModelo.SelectedIndex < 0)
                {
this.cboModelo.Text = string.Empty;
                    objMatricula.LlenarGrilla("-1");
                }
            }

        }

privatevoid cboModelo_SelectedIndexChanged(object sender, EventArgs e)
        {
if (cboFlota.SelectedIndex >= 0)
            {
Matricula objModelo = newMatricula(this);
                objModelo.LlenarGrilla(this.cboModelo.SelectedValue.ToString());
}
}

//funcion para capturar evento y no se pueda cambiar de tamaño la forma
protectedoverridevoid WndProc(refMessage m)
        {
if (m.Msg == 0xa3 && m.WParam == newIntPtr(2))
```

```
        {
return;
        }
base.WndProc(ref m);
}

   }
}
```

frmMenuDocumentacion.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using RptAppServidor.Clases;

namespace RptAppServidor.LINEA_AEREA
{
publicpartialclassfrmMenuDocumentacion : Form
   {
privateint idusuario = 0;
privatestring login = string.Empty;
privatestring nombreUsuario = string.Empty;
privateint perfilUsuario = 0;
privateint idOperador = 0;
privateint idFlota = 0;
privateint idModelo = 0;
privateint idMatricula;
privatestring info = string.Empty;


public frmMenuDocumentacion(int idusuario, string login, string nombreUsuario, int perfilUsuario,int
          idOperador, int flota, int idModelo, int idMatricula)
{
this.idusuario =  idusuario;
this.login = login;
this.nombreUsuario = nombreUsuario;
this.perfilUsuario = perfilUsuario;
this.idOperador = idOperador;
this.idFlota = flota;
this.idModelo = idModelo;
this.idMatricula = idMatricula;
          InitializeComponent();
      }


privatevoid frmMenuDocumentacion_Load(object sender, EventArgs e)
      {
try
        {
```

```
MenuDocumentacion objMenuDocumentacion = newMenuDocumentacion(this);
if (!objMenuDocumentacion.ConsultaInformacionBotones(this.idFlota.ToString()))
thrownewException();
this.cmdOpcion1.Text = objMenuDocumentacion.etiqueta_opcion1;
this.lblPath1.Text = objMenuDocumentacion.path_opcion1;

this.cmdOpcion2.Text = objMenuDocumentacion.etiqueta_opcion2;
this.lblPath2.Text = objMenuDocumentacion.path_opcion2;

this.cmdOpcion3.Text = objMenuDocumentacion.etiqueta_opcion3;
this.lblPath3.Text = objMenuDocumentacion.path_opcion3;

Consultas objConsultas = newConsultas();
this.txtFlota.Text = "OPERADOR : " + objConsultas.ConsultaGenerica("select ope_descripcion from
        operador where ope_id = " + this.idOperador.ToString());
this.txtFlota.Text += "   /   FLOTA  : " + objConsultas.ConsultaGenerica("select flo_descripcion from
        flota where flo_id = " + this.idFlota.ToString());
this.txtFlota.Text += "   /   MODELO : " + objConsultas.ConsultaGenerica("select mod_descripcion
        from modelo where mod_id = " + this.idModelo.ToString());
this.txtFlota.Text += "   /   MATRICULA : " + objConsultas.ConsultaGenerica("select mat_descripcion
        from matricula where mat_id = " + this.idMatricula.ToString());
this.info = this.txtFlota.Text;

this.cmdManualAvion.Focus();
        }
catch (Exception)
{
MessageBox.Show("Error al recuperar información de las opciones en esta pantalla.");
}

    }

privatevoid cmdManualAvion_Click(object sender, EventArgs e)
{
frmDocumentacion objfrmDocumentacion = newfrmDocumentacion("1", this.idusuario,
this.login,
this.nombreUsuario,
this.perfilUsuario,
this.idOperador,
this.idFlota,
this.idModelo,
this.idMatricula, this.info);//manual del avion
this.Visible = false;
Util.AbrirVentana(objfrmDocumentacion,0);
this.Visible = true;
    }

privatevoid cmdManualCompania_Click(object sender, EventArgs e)
{
frmDocumentacion objfrmDocumentacion = newfrmDocumentacion("2", this.idusuario,
this.login,
this.nombreUsuario,
this.perfilUsuario,
this.idOperador,
this.idFlota,
this.idModelo,
this.idMatricula,this.info);//manual del avion
this.Visible = false;
Util.AbrirVentana(objfrmDocumentacion,0);
this.Visible = true;
    }
```

```
privatevoid cmdRegresar_Click(object sender, EventArgs e)
        {
this.Close();
        }

privatevoid cmdOpt_Click(object sender, EventArgs e)
        {
Util.EjecutarProgramaExterno(this.lblPath1.Text);
        }

privatevoid cmdMMPP_Click(object sender, EventArgs e)
        {
Util.EjecutarProgramaExterno(this.lblPath2.Text);
        }

privatevoid cmdFlyE6B_Click(object sender, EventArgs e)
{
Util.EjecutarProgramaExterno(this.lblPath3.Text);
        }
//funcion para capturar evento y no se pueda cambiar de tamaño la forma
protectedoverridevoid WndProc(refMessage m)
        {
if (m.Msg == 0xa3 && m.WParam == newIntPtr(2))
            {
return;
            }
base.WndProc(ref m);
}
    }
}
```

frmModelo.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
using RptAppServidor.Clases;

namespace RptAppServidor
{
publicpartialclassfrmModelo : Form
    {
publicint usuarioActual = 0;
int PefilActual = 0;
publicbool flag = true;
public frmModelo(int p_usuario, int p_perfil)
        {
            InitializeComponent();
this.usuarioActual = p_usuario;
this.PefilActual = p_perfil;
Util.lookGrilla(this.grdData);
        }
```

```
privatevoid frmModulo_Load(object sender, EventArgs e)
      {
/*Turno objTurno = new Turno(this);
objTurno.BuscarTurnoCombos();*/
Modelo objModelo = newModelo(this);
objModelo.BuscarOperador();
      }

publicvoid cmdBuscar_Click(object sender, EventArgs e)
      {

      }

privatevoid grdData_CellContentClick(object sender, DataGridViewCellEventArgs e)
      {
      }

privatevoid grdData_CellClick(object sender, DataGridViewCellEventArgs e)
      {
if (this.grdData.Rows.Count > 0 &&this.grdData.SelectedRows.Count > 0 )
         {


this.txtCodigo.Text = this.grdData.SelectedRows[0].Cells[0].Value.ToString();

this.txtDescripcion.Text = this.grdData.SelectedRows[0].Cells[2].Value.ToString();


         }
      }

privatevoid cmdIngresar_Click(object sender, EventArgs e)
      {


Modelo objLineaAerea = newModelo(this);

if (this.txtCodigo.Text.Trim() == string.Empty)
         {
            objLineaAerea.operacioN = "I";
if (!objLineaAerea.ValidarInformacion())
return;
objLineaAerea.Insertar();
         }
else
         {
            objLineaAerea.operacioN = "U";
if (!objLineaAerea.ValidarInformacion())
return;
            objLineaAerea.Actualizar();
         }
this.cboFlota_SelectedIndexChanged(sender, e);
      }



privatevoid button4_Click(object sender, EventArgs e)
      {

      }
```

```
privatevoid cmdSalir_Click(object sender, EventArgs e)
        {
this.Close();
        }
publicvoid cmdLimpiar_Click(object sender, EventArgs e)
{
Modelo objLinea = newModelo(this);
            objLinea.limpiar();


}

privatevoid grdData_SelectionChanged(object sender, EventArgs e)
        {
            grdData_CellClick(sender, null);
        }

privatevoid grdData_CurrentCellChanged(object sender, EventArgs e)
        {

        }

privatevoid lblBuscarPor_Click(object sender, EventArgs e)
        {

        }

privatevoid label4_Click(object sender, EventArgs e)
        {

        }

privatevoid comboBox2_SelectedIndexChanged(object sender, EventArgs e)
        {

        }

privatevoid fraData_Enter(object sender, EventArgs e)
        {

        }

privatevoid cboTurno_SelectedValueChanged(object sender, EventArgs e)
{
this.cmdBuscar_Click(sender, e);
}

privatevoid cmdEliminar_Click_1(object sender, EventArgs e)
{
Modelo objlinea = newModelo(this);
            objlinea.operacioN = "D";
objlinea.Eliminar();
this.cboFlota_SelectedIndexChanged(sender, e);
        }

privatevoid cboTurno_SelectedIndexChanged(object sender, EventArgs e)
        {

        }
```

```
privatevoid cboEstado_SelectedIndexChanged(object sender, EventArgs e)
      {

      }

privatevoid txtDescripcion_TextChanged(object sender, EventArgs e)
      {

      }

privatevoid fraGrid_Enter(object sender, EventArgs e)
{

      }

privatevoid cboOperador_SelectedIndexChanged(object sender, EventArgs e)
      {
if (this.cboOperador.SelectedIndex >= 0)
         {
Modelo objModelo = newModelo(this);
objModelo.BuscarFlotaPorOperador(this.cboOperador.SelectedValue.ToString());
if (this.cboFlota.SelectedIndex < 0)
            {
this.cboFlota.Text = string.Empty;
               objModelo.LlenarGrilla("-1");
            }

         }

      }

privatevoid cboFlota_SelectedIndexChanged(object sender, EventArgs e)
      {
if (cboFlota.SelectedIndex >= 0)
         {
Modelo objModelo = newModelo(this);
            objModelo.LlenarGrilla(this.cboFlota.SelectedValue.ToString());
}
      }

//funcion para capturar evento y no se pueda cambiar de tamaño la forma
protectedoverridevoid WndProc(refMessage m)
      {
if (m.Msg == 0xa3 && m.WParam == newIntPtr(2))
         {
return;
         }
base.WndProc(ref m);
}
   }
}
```

frmOpcionesModelo.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
```

```
using System.Linq;
using System.Text;
using System.Windows.Forms;
using RptAppServidor.Clases;

namespace RptAppServidor.LINEA_AEREA
{
publicpartialclassfrmOpcionesModelo : Form
   {
publicint usuarioActual = 0;
int PefilActual = 0;
publicbool flag = true;
public frmOpcionesModelo(int p_usuario, int p_perfil)
{
          InitializeComponent();
this.usuarioActual = p_usuario;
this.PefilActual = p_perfil;
Util.lookGrilla(this.grdData);
      }


privatevoid cmdBuscar_Click(object sender, EventArgs e)
      {
         DialogoOpen.ShowDialog(this);
this.txtArchivo.Text = DialogoOpen.FileName;
      }

privatevoid cboOperador_SelectedIndexChanged(object sender, EventArgs e)
      {
if (this.cboOperador.SelectedIndex >= 0)
          {
OpcionesModelo objModelo = newOpcionesModelo(this);
             objModelo.BuscarFlotaPorOperador(this.cboOperador.SelectedValue.ToString());
if (this.cboFlota.SelectedIndex < 0)
             {
this.cboFlota.Text = string.Empty;
                objModelo.LlenarGrilla("-1");
             }

          }
      }

privatevoid cboFlota_SelectedIndexChanged(object sender, EventArgs e)
{
if (cboFlota.SelectedIndex >= 0)
          {
OpcionesModelo objModelo = newOpcionesModelo(this);
             objModelo.LlenarGrilla(this.cboFlota.SelectedValue.ToString());
}
      }

privatevoid cmdIngresar_Click(object sender, EventArgs e)
{
OpcionesModelo objOpcionesModelo = newOpcionesModelo(this);

if (this.txtCodigo.Text.Trim() == string.Empty)
{
             objOpcionesModelo.operacioN = "I";
if (!objOpcionesModelo.ValidarInformacion())
return;
             objOpcionesModelo.Insertar();
```

```
        }
else
        {
            objOpcionesModelo.operacioN = "U";
if (!objOpcionesModelo.ValidarInformacion())
return;
            objOpcionesModelo.Actualizar();
}
this.cboFlota_SelectedIndexChanged(sender, e);

      }

privatevoid cmdEliminar_Click(object sender, EventArgs e)
{
OpcionesModelo objlinea = newOpcionesModelo(this);
objlinea.operacioN = "D";
objlinea.Eliminar();
this.cboFlota_SelectedIndexChanged(sender, e);
      }

privatevoid cmdLimpiar_Click(object sender, EventArgs e)
{
OpcionesModelo objLinea = newOpcionesModelo(this);
objLinea.limpiar();
}

privatevoid cmdSalir_Click(object sender, EventArgs e)
      {
this.Close();
      }

privatevoid grdData_CellClick(object sender, DataGridViewCellEventArgs e)
      {
if (this.grdData.Rows.Count > 0 &&this.grdData.SelectedRows.Count > 0)
        {

this.txtCodigo.Text = this.grdData.SelectedRows[0].Cells[0].Value.ToString();
this.txtDescripcion.Text = this.grdData.SelectedRows[0].Cells[1].Value.ToString();
this.txtArchivo.Text = this.grdData.SelectedRows[0].Cells[2].Value.ToString();
this.cboOrden.Text = this.grdData.SelectedRows[0].Cells[4].Value.ToString();
}
      }

privatevoid frmOpcionesModelo_Load(object sender, EventArgs e)
      {
OpcionesModelo objModelo = newOpcionesModelo(this);
objModelo.BuscarOperador();
      }

privatevoid grdData_SelectionChanged(object sender, EventArgs e)
{
        grdData_CellClick(sender, null);
}

//funcion para capturar evento y no se pueda cambiar de tamaño la forma
protectedoverridevoid WndProc(refMessage m)
      {
if (m.Msg == 0xa3 && m.WParam == newIntPtr(2))
        {
return;
        }
```

```
base.WndProc(ref m);
}

   }
}
```

frmOperador.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using RptAppServidor.Clases;

namespace RptAppServidor
{
publicpartialclassfrmOperador : Form
   {
publicint usuarioActual = 0;
int PefilActual = 0;
publicbool flag = true;
public frmOperador(int p_usuario, int p_perfil)


      {
         InitializeComponent();
this.usuarioActual = p_usuario;
this.PefilActual = p_perfil;
Util.lookGrilla(this.grdData);
      }

privatevoid cmdIngresar_Click(object sender, EventArgs e)
      {
Operador objPais= newOperador(this);

if (this.txtCodigo.Text.Trim() == string.Empty)
         {
            objPais.operacioN = "I";
if (!objPais.ValidarInformacion())
return;
            objPais.Insertar();
         }
else
{
            objPais.operacioN = "U";
if (!objPais.ValidarInformacion())
return;
            objPais.Actualizar();
         }
         objPais.LlenarGrilla();
      }

privatevoid frmPais_Load(object sender, EventArgs e)
      {
```

```
/*Turno objTurno = new Turno(this);
objTurno.BuscarTurnoCombos();*/
Operador objLineaAerea = newOperador(this);
objLineaAerea.LlenarGrilla();
        }

privatevoid cmdEliminar_Click(object sender, EventArgs e)
{
Operador objPais = newOperador(this);
objPais.operacioN = "D";
            objPais.Eliminar();
            objPais.LlenarGrilla();
}

privatevoid cmdLimpiar_Click(object sender, EventArgs e)
        {
Operador objPais = newOperador(this);
            objPais.limpiar();
}

privatevoid button1_Click(object sender, EventArgs e)
        {
this.Close();
        }

privatevoid grdData_CellClick(object sender, DataGridViewCellEventArgs e)
        {
if (this.grdData.Rows.Count > 0 &&this.grdData.SelectedRows.Count > 0)
            {
//this.grdData.Rows.
//this.grdData.Rows[this.grdData.Rows][0]
//this.grdUsuarios.SelectedItems[0].Cells[2].Text;

this.txtCodigo.Text = this.grdData.SelectedRows[0].Cells[0].Value.ToString();
//this.cboDiaSemana.SelectedValue   = this.grdData.SelectedRows[0].Cells[8].Value.ToString();
this.txtDescripcion.Text = this.grdData.SelectedRows[0].Cells[1].Value.ToString();


            }
        }

privatevoid grdData_CellContentClick(object sender, DataGridViewCellEventArgs e)
        {

        }

privatevoid grdData_SelectionChanged(object sender, EventArgs e)
        {
            grdData_CellClick(sender, null);
        }

privatevoid txtBuscar_TextChanged(object sender, EventArgs e)
        {
Operador objPais = newOperador(this);
            objPais.BuscarLike();
        }

privatevoid label2_Click(object sender, EventArgs e)
        {

        }
```

```
privatevoid txtDescripcion_TextChanged(object sender, EventArgs e)
      {

      }

privatevoid txtCodigo_TextChanged(object sender, EventArgs e)
      {

      }

privatevoid label3_Click(object sender, EventArgs e)
      {

      }

privatevoid fraData_Enter(object sender, EventArgs e)
      {

      }

privatevoid fraGrid_Enter(object sender, EventArgs e)
      {

      }

privatevoid label1_Click(object sender, EventArgs e)
      {

      }

privatevoid groupBox1_Enter(object sender, EventArgs e)
{

      }

//funcion para capturar evento y no se pueda cambiar de tamaño la forma
protectedoverridevoid WndProc(refMessage m)
      {
if (m.Msg == 0xa3 && m.WParam == newIntPtr(2))
         {
return;
         }
base.WndProc(ref m);
}

   }
}
```

frmParametro.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
```

```
using System.Windows.Forms;
using RptAppServidor.Clases;

namespace RptAppServidor
{
publicpartialclassfrmParametro : Form
   {
publicint usuarioActual = 0;
int PefilActual = 0;
publicbool flag = true;
public frmParametro(int p_usuario, int p_perfil)

      {
         InitializeComponent();
this.usuarioActual = p_usuario;
this.PefilActual = p_perfil;
Util.lookGrilla(this.grdData);
      }

privatevoid cmdIngresar_Click(object sender, EventArgs e)
{
Parametro objParametro= newParametro(this);

if (this.txtCodigo.Enabled)
{
            objParametro.operacioN = "I";
if (!objParametro.ValidarInformacion())
return;
            objParametro.Insertar();
         }
else
         {
            objParametro.operacioN = "U";
if (!objParametro.ValidarInformacion())
return;
           objParametro.Actualizar();
         }
        objParametro.LlenarGrilla();
}



privatevoid cmdEliminar_Click(object sender, EventArgs e)
{
Parametro objParametro = newParametro(this);
         objParametro.operacioN = "D";
         objParametro.Eliminar();
         objParametro.LlenarGrilla();
      }

privatevoid cmdLimpiar_Click(object sender, EventArgs e)
      {
Parametro objParametro = newParametro(this);
         objParametro.limpiar();
}

privatevoid button1_Click(object sender, EventArgs e)
{
this.Close();
      }
```

```
privatevoid grdData_CellClick(object sender, DataGridViewCellEventArgs e)
        {
if (this.grdData.Rows.Count > 0 &&this.grdData.SelectedRows.Count > 0)
            {

this.txtCodigo.Text = this.grdData.SelectedRows[0].Cells[0].Value.ToString();
this.txtDescripcion.Text = this.grdData.SelectedRows[0].Cells[1].Value.ToString();
this.txtValor.Text = this.grdData.SelectedRows[0].Cells[2].Value.ToString();
this.txtCodigo.Enabled = false;


            }
        }

privatevoid grdData_CellContentClick(object sender, DataGridViewCellEventArgs e)
        {

        }

privatevoid grdData_SelectionChanged(object sender, EventArgs e)
        {
            grdData_CellClick(sender, null);
        }

privatevoid txtBuscar_TextChanged(object sender, EventArgs e)
{
Parametro objParametro = newParametro(this);
            objParametro.BuscarLike();
}

privatevoid frmParametro_Load(object sender, EventArgs e)
{

Parametro objParametro = newParametro(this);
            objParametro.LlenarGrilla();
        }

//funcion para capturar evento y no se pueda cambiar de tamaño la forma
protectedoverridevoid WndProc(refMessage m)
        {
if (m.Msg == 0xa3 && m.WParam == newIntPtr(2))
            {
return;
            }
base.WndProc(ref m);
        }

    }
}
```

frmQApagarPC.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
```

```
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace RptAppServidor.LINEA_AEREA
{
publicpartialclassfrmQApagarPC : Form
   {
publicBoolean accionIrLogin = false;
publicBoolean accionApagarPC = false;
publicBoolean accionNo = false;

public frmQApagarPC(string msgEtiqueta, string msgBoton)
      {
         InitializeComponent();
this.lblApagar.Text = msgEtiqueta;
this.cmdCancelar.Text = msgBoton;
      }

privatevoid cmdSi_Click(object sender, EventArgs e)
      {
         accionApagarPC = true;
this.Close();
      }

privatevoid cmdNo_Click(object sender, EventArgs e)
      {
         accionIrLogin = false;
         accionNo = true;
this.Close();
      }

privatevoid cmdCancelar_Click(object sender, EventArgs e)
{
         accionIrLogin = true;
this.Close();
      }

//funcion para capturar evento y no se pueda cambiar de tamaño la forma
protectedoverridevoid WndProc(refMessage m)
      {
if (m.Msg == 0xa3 && m.WParam == newIntPtr(2))
         {
return;
         }
base.WndProc(ref m);
}


   }
}
```

frmTipo.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
```

```
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using RptAppServidor.Clases;

namespace RptAppServidor
{
publicpartialclassfrmTipo : Form
   { publicint usuarioActual = 0;
int PefilActual = 0;
publicbool flag = true;
public frmTipo(int p_usuario, int p_perfil)

      {
         InitializeComponent();
this.usuarioActual = p_usuario;
this.PefilActual = p_perfil;
Util.lookGrilla(this.grdData);
      }

privatevoid frmTipo_Load(object sender, EventArgs e)
{
Tipo objTipo = newTipo(this);
objTipo.LlenarGrilla();
}

privatevoid grdData_CellClick(object sender, DataGridViewCellEventArgs e)
      {
if (this.grdData.Rows.Count > 0 &&this.grdData.SelectedRows.Count > 0)
         {


this.txtCodigo.Text = this.grdData.SelectedRows[0].Cells[0].Value.ToString();

this.txtDescripcion.Text = this.grdData.SelectedRows[0].Cells[1].Value.ToString();

         }
      }

privatevoid grdData_SelectionChanged(object sender, EventArgs e)
      {
         grdData_CellClick(sender, null);
      }

privatevoid cmdIngresar_Click(object sender, EventArgs e)
{
Tipo objTipo = newTipo(this);

if (this.txtCodigo.Text.Trim() == string.Empty)
{
            objTipo.operacioN = "I";
if (!objTipo.ValidarInformacion())
return;
            objTipo.Insertar();
         }
else
         {
            objTipo.operacioN = "U";
if (!objTipo.ValidarInformacion())
return;
```

```
            objTipo.Actualizar();
        }
        objTipo.LlenarGrilla();
}

privatevoid cmdEliminar_Click(object sender, EventArgs e)
{
Tipo objTipo = newTipo(this);
        objTipo.operacioN = "D";
        objTipo.Eliminar();
        objTipo.LlenarGrilla();
    }

privatevoid cmdLimpiar_Click(object sender, EventArgs e)
    {
Tipo objTipo = newTipo(this);
objTipo.limpiar();
}

privatevoid button1_Click(object sender, EventArgs e)
    {
this.Close();
    }

privatevoid txtBuscar_TextChanged(object sender, EventArgs e)
{
Tipo objTipo = newTipo(this);
        objTipo.BuscarLike();
    }

//funcion para capturar evento y no se pueda cambiar de tamaño la forma
protectedoverridevoid WndProc(refMessage m)
    {
if (m.Msg == 0xa3 && m.WParam == newIntPtr(2))
        {
return;
        }
base.WndProc(ref m);
}
  }
}
```

frmUsuario.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using RptAppServidor.Clases;

namespace RptAppServidor
```

```
{
publicpartialclassFrmUsuario : Form
   {

publicint usuarioActual = 0;
int PefilActual = 0;
publicbool flag = true;
public FrmUsuario(int p_usuario, int p_perfil)
      {
         InitializeComponent();
this.usuarioActual = p_usuario;
this.PefilActual = p_perfil;
Util.lookGrilla(this.grdData);
      }



publicvoid cmdBuscar_Click(object sender, EventArgs e)
      {
/* Turno objTurno = new Turno(this);
         objTurno.BuscarTurnoLike(); */
      }

privatevoid grdData_CellContentClick(object sender, DataGridViewCellEventArgs e)
      {
      }

privatevoid grdData_CellClick(object sender, DataGridViewCellEventArgs e)
      {
if (this.grdData.Rows.Count > 0 &&this.grdData.SelectedRows.Count > 0 )
         {

this.txtCodigo.Text = this.grdData.SelectedRows[0].Cells[0].Value.ToString();
this.txtDescripcion.Text = this.grdData.SelectedRows[0].Cells[1].Value.ToString();
this.txtNombre.Text = this.grdData.SelectedRows[0].Cells[2].Value.ToString();
this.txtSegundoNombre.Text = this.grdData.SelectedRows[0].Cells[3].Value.ToString();
this.txtApellidoPaterno.Text = this.grdData.SelectedRows[0].Cells[4].Value.ToString();
this.txtApellidoMaterno.Text = this.grdData.SelectedRows[0].Cells[5].Value.ToString();
this.txtPasword.Text = this.grdData.SelectedRows[0].Cells[6].Value.ToString();
if (this.grdData.SelectedRows[0].Cells[7].Value.ToString() == "S")
this.cbvalidado.Checked = true;
else
this.cbvalidado.Checked = false;
//this.cbvalidado.Text = this.grdData.SelectedRows[0].Cells[6].Value.ToString();
this.cboFlota.SelectedValue = this.grdData.SelectedRows[0].Cells[8].Value.ToString();
Consultas objConsultas = newConsultas();
string sqlOperador = "select ope_id from flota where flo_id = " +
          this.grdData.SelectedRows[0].Cells[8].Value.ToString();
this.cboOperador.SelectedValue = objConsultas.ConsultaGenerica(sqlOperador);
this.cboPerfil.SelectedValue = this.grdData.SelectedRows[0].Cells[9].Value.ToString();

         }
      }

privatevoid cmdIngresar_Click(object sender, EventArgs e)
{


Usuario objUsuario = newUsuario(this);

if (this.txtCodigo.Text.Trim() == string.Empty)
```

```
{
            objUsuario.operacioN = "I";
if (!objUsuario.ValidarInformacion())
return;
            objUsuario.Insertar();
        }
else
        {
            objUsuario.operacioN = "U";
if (!objUsuario.ValidarInformacion())
return;
            objUsuario.Actualizar();
        }
        objUsuario.limpiar();
        objUsuario.VaciarGrilla();
        objUsuario.LlenarGrilla();
}



privatevoid button4_Click(object sender, EventArgs e)
    {

    }

privatevoid cmdSalir_Click(object sender, EventArgs e)
{
this.Close();
    }
publicvoid cmdLimpiar_Click(object sender, EventArgs e)
{
Usuario objUsuario = newUsuario(this);
objUsuario.limpiar();
objUsuario.LlenarGrilla();

    }

privatevoid grdData_SelectionChanged(object sender, EventArgs e)
    {
        grdData_CellClick(sender, null);
    }

privatevoid grdData_CurrentCellChanged(object sender, EventArgs e)
    {

    }

privatevoid lblBuscarPor_Click(object sender, EventArgs e)
    {

    }

privatevoid label4_Click(object sender, EventArgs e)
    {

    }

privatevoid comboBox2_SelectedIndexChanged(object sender, EventArgs e)
    {

    }
```

```
privatevoid fraData_Enter(object sender, EventArgs e)
        {

        }

privatevoid cboTurno_SelectedValueChanged(object sender, EventArgs e)
{
this.cmdBuscar_Click(sender, e);
}

privatevoid cmdEliminar_Click_1(object sender, EventArgs e)
{
Usuario objUsuario = newUsuario(this);
            objUsuario.operacioN = "D";
objUsuario.Eliminar();
objUsuario.LlenarGrilla();
        }

privatevoid cboTurno_SelectedIndexChanged(object sender, EventArgs e)
        {

        }

privatevoid txtBuscar_TextChanged(object sender, EventArgs e)
{
Usuario objUsuario = newUsuario(this);
objUsuario.BuscarLike();
}
privatevoid cbvalidado_CheckedChanged(object sender, EventArgs e)
        {

        }

privatevoid FrmUsuario_Load(object sender, EventArgs e)
{
Usuario objUsuario = newUsuario(this);
            objUsuario.LlenarGrilla();
            objUsuario.BuscarOperador();
            objUsuario.BuscarPerfil();
        }

//funcion para capturar evento y no se pueda cambiar de tamaño la forma
protectedoverridevoid WndProc(refMessage m)
        {
if (m.Msg == 0xa3 && m.WParam == newIntPtr(2))
            {
return;
            }
base.WndProc(ref m);
        }

privatevoid cboOperador_SelectedIndexChanged(object sender, EventArgs e)
        {
this.cboFlota.Text = string.Empty;
if (this.cboOperador.SelectedIndex >= 0)
            {
Usuario objUsuario = newUsuario(this);
                objUsuario.BuscarFlota(this.cboOperador.SelectedValue.ToString());
            }
else
```

```
this.cboFlota.Text = string.Empty;
}
   }
}
```

Arbol.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using RptAppServidor.LINEA_AEREA;
using System.Data;
using System.Windows.Forms;
using System.Security.Policy;

namespace RptAppServidor.Clases
{
classArbol
   {
//atributos de la clase
privatefrmDocumentacion FrmDocumentacion = null;
private System.Data.DataSet dataSetArbol;
string idTipo = string.Empty;
privateint idOperador = 0;
privateint idFlota = 0;
privateint idModelo = 0;
privateint idMatricula;

//Primer constuctor de la clase
public Arbol(frmDocumentacion pfrmDocumentacion)
      {
this.FrmDocumentacion = pfrmDocumentacion;
      }

//Sobrecarga del constructor
public Arbol(string tipo, int idOperador, int flota, int idModelo, int idMatricula)
      {
//this.idLinea = linea;
this.idTipo = tipo;
this.idOperador = idOperador;
this.idFlota = flota;
this.idModelo = idModelo;
this.idMatricula = idMatricula;
      }

//Inserta un nuevo nodo
publicBoolean InsertarNodo(frmArbol FrmArbol)
{
try
         {
if (FrmArbol.arbolDoc.SelectedNode == null
               || FrmArbol.cboOperador.SelectedIndex < 0
               || FrmArbol.cboFlota.SelectedIndex < 0
               || FrmArbol.cboModelo.SelectedIndex < 0
               || FrmArbol.cboMatricula.SelectedIndex < 0
               || FrmArbol.cboTipo.SelectedIndex < 0
```

```
                || (FrmArbol.txtArchivo.Text.Trim() == string.Empty  &&
            FrmArbol.chkAnexarArchivo.Checked)
                || FrmArbol.txtDecripcionNodo.Text.Trim() == string.Empty)
{
MessageBox.Show("No se ha ingresado toda la información necesaria.");
returnfalse;
            }
DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
string[] datos = FrmArbol.arbolDoc.SelectedNode.Tag.ToString().Split('|');
int idArbol = this.secuencialArbol();
            FrmArbol.txtidNodo.Text = idArbol.ToString();
string remarcar = string.Empty;
if (FrmArbol.chkRemarcar.Checked)
                remarcar = "S";
else
                remarcar = "N";


                objAccess.EjecutarComando(" insert into arbol ( arb_id, arb_arb_id, tip_id,
            arb_descripcion, arb_path_documento, mat_id, arb_revision,arb_remarcar) "
                                    + " values ( "
                                    + idArbol.ToString() + ","
                                    + datos[1].ToString() + ","
                                    + FrmArbol.cboTipo.SelectedValue.ToString() + ","
+ "'" + FrmArbol.txtDecripcionNodo.Text + "',"
+ "'" + FrmArbol.txtArchivo.Text + "',"
                                    + FrmArbol.cboMatricula.SelectedValue.ToString() + ","
                                    + "'" + FrmArbol.txtRevision.Text + "',"
+ "'" + remarcar + "'"
                                    + "  ) ");



            objAccess.Cerrar();

returntrue;
          }
catch (Exception)
          {
returnfalse;
}

      }

//Creación de nodo raíz
publicBoolean CrearNodoRaiz(frmArbol FrmArbol)
{

try{
if (FrmArbol.cboOperador.SelectedIndex < 0
                || FrmArbol.cboFlota.SelectedIndex < 0
                || FrmArbol.cboModelo.SelectedIndex < 0
                || FrmArbol.cboMatricula.SelectedIndex < 0
                || FrmArbol.cboTipo.SelectedIndex < 0
                || FrmArbol.cboOperador.Text == string.Empty
                || FrmArbol.cboFlota.Text == string.Empty
                || FrmArbol.cboModelo.Text == string.Empty
                || FrmArbol.cboMatricula.Text == string.Empty
                || FrmArbol.cboTipo.Text == string.Empty)
{
MessageBox.Show("No se ha ingresado toda la información necesaria.");
```

```
returnfalse;
            }

if (FrmArbol.arbolDoc.Nodes.Count > 0 || FrmArbol.arbolDoc.SelectedNode != null )
            {
MessageBox.Show("El árbol actual ya tiene nodos creados.");
returnfalse;
            }


DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
int idArbol = this.secuencialArbol();
string desTipo = FrmArbol.cboTipo.Text.Trim();



        objAccess.EjecutarComando(" insert into arbol ( arb_id, arb_arb_id, tip_id, arb_descripcion,
         arb_path_documento, mat_id, arb_revision,arb_remarcar) "
                                + " values ( "
                                + idArbol.ToString() + ","
                                + "0" + ","
                                + FrmArbol.cboTipo.SelectedValue.ToString() + ","
                                + "'" + desTipo + "',"
                                + "'',"
                                + FrmArbol.cboMatricula.SelectedValue.ToString() + ","
                                + "'NODO RAIZ',"
                                + "'N'"
                                + "  ) ");

        objAccess.Cerrar();
returntrue;
         }
catch (Exception)
        {
returnfalse;
}

      }

//Actualizar información de nodo
publicBoolean ActualizarNodo(frmArbol FrmArbol)
{
try
        {
if (FrmArbol.arbolDoc.SelectedNode == null
                || FrmArbol.cboOperador.SelectedIndex < 0
                || FrmArbol.cboFlota.SelectedIndex < 0
                || FrmArbol.cboModelo.SelectedIndex < 0
                || FrmArbol.cboMatricula.SelectedIndex < 0
                || FrmArbol.cboTipo.SelectedIndex < 0
                || (FrmArbol.txtArchivo.Text.Trim() == string.Empty &&
           FrmArbol.chkAnexarArchivo.Checked)
                || FrmArbol.txtDecripcionNodo.Text.Trim() == string.Empty
                || FrmArbol.txtidNodo.Text.Trim() == string.Empty)
{
MessageBox.Show("No se ha ingresado toda la información necesaria.");
returnfalse;
            }
DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
string[] datos = FrmArbol.arbolDoc.SelectedNode.Tag.ToString().Split('|');
```

```
string remarcar = string.Empty;
if (FrmArbol.chkRemarcar.Checked)
                remarcar = "S";
else
                remarcar = "N";
objAccess.EjecutarComando(" update arbol "
                                + " set "
                                + " arb_descripcion = '" + FrmArbol.txtDecripcionNodo.Text + "',"
                                + " arb_path_documento = '" + FrmArbol.txtArchivo.Text + "',"
                                + " arb_revision = '" + FrmArbol.txtRevision.Text + "',"
                                + " arb_remarcar = '" + remarcar + "'"
+ " where arb_id = " + FrmArbol.txtidNodo.Text);

string[] info = FrmArbol.txtArchivo.Text.ToString().Split('\\');
string nombreArchivo = info[info.Length - 1];

            objAccess.EjecutarComando(" update arbol "
                                + " set "
                                + " arb_revision = '" + FrmArbol.txtRevision.Text + "',"
                                + " arb_remarcar = '" + remarcar + "'"
                                + " where arb_path_documento like  '%" + nombreArchivo + "'");

            objAccess.Cerrar();

returntrue;
          }
catch (Exception)
{
returnfalse;
          }

       }

//Eliminación de un nodo
publicBoolean EliminarrNodo(frmArbol FrmArbol)
{
try
          {
if (FrmArbol.txtidNodo.Text.Trim() == string.Empty)
{
MessageBox.Show("No se ha ingresado toda la información necesaria.");
returnfalse;
              }
if (this.TieneHijos(FrmArbol.txtidNodo.Text))
{
MessageBox.Show("No se puede eliminar, el nodo tiene hijos.");
returnfalse;
              }

DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
string[] datos = FrmArbol.arbolDoc.SelectedNode.Tag.ToString().Split('|');
//objAccess.EjecutarComando(" delete from arbol where arb_id = " + FrmArbol.txtidNodo.Text);
if (!FrmArbol.chkEliminarTodo.Checked)
                objAccess.EjecutarComando(" update arbol set arb_borrado = 1 where arb_id = " +
            FrmArbol.txtidNodo.Text);
else
              {
string[] info = FrmArbol.txtArchivo.Text.ToString().Split('\\');
string nombreArchivo = info[info.Length - 1];
                objAccess.EjecutarComando(" update arbol set arb_borrado = 1 " + " where
            arb_path_documento like  '%" + nombreArchivo + "%'");
```

```
            }

            objAccess.Cerrar();

returntrue;
        }
catch (Exception)
{
returnfalse;
        }

    }


//Llenar  lista desplegable de Tipo de documento
publicBoolean buscarTipo(frmArbol FrmArbol )
{
try
        {
DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
            objAccess.EjecutarConsulta(" select tip_id, tip_descripcion from tipo where tip_borrado =
         0");
            FrmArbol.cboTipo.ValueMember = "tip_id";
            FrmArbol.cboTipo.DisplayMember = "tip_descripcion";
            FrmArbol.cboTipo.DataSource = objAccess.InfoTabla[0];

returntrue;
}
catch (Exception)
        {
returnfalse;
        }
    }

//Mapeado recursivo en el objeto Treeview
publicvoid CrearNodosDelPadre(int indicePadre, TreeNode nodePadre, TreeView arbolDoc)
    {
// Crear un DataView con los Nodos que dependen del Nodo padre pasado como parámetro.
DataView dataViewHijos = newDataView(dataSetArbol.Tables["TablaArbol"]);
        dataViewHijos.RowFilter =
         dataSetArbol.Tables["TablaArbol"].Columns["IdentificadorPadre"].ColumnName + " = " +
         indicePadre;

// Agregar al TreeView los nodos Hijos que se han obtenido en el DataView.
foreach (DataRowView dataRowCurrent in dataViewHijos)
        {
TreeNode nuevoNodo = newTreeNode();

            nuevoNodo.Text = dataRowCurrent["NombreNodo"].ToString().Trim();
            nuevoNodo.Tag = dataRowCurrent["Path"].ToString().Trim() + "|" +
          dataRowCurrent["IdentificadorNodo"].ToString().Trim()
              + "|" + dataRowCurrent["Revision"].ToString().Trim()
              + "|" + dataRowCurrent["Remarcar"].ToString().Trim();
            nuevoNodo.ToolTipText = dataRowCurrent["Revision"].ToString().Trim();

if (dataRowCurrent["Remarcar"].ToString().Trim() == "S")
            {
                System.Drawing.Font letra = new System.Drawing.Font(arbolDoc.Font.FontFamily, 9,
          System.Drawing.FontStyle.Underline);
                nuevoNodo.NodeFont = letra;
                nuevoNodo.ForeColor = System.Drawing.Color.GreenYellow;
```

```
            }
else
            {
                System.Drawing.Font letra = new System.Drawing.Font(arbolDoc.Font.FontFamily,8);
                nuevoNodo.NodeFont = letra;
                nuevoNodo.ForeColor = System.Drawing.Color.FloralWhite;
            }

DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
            objAccess.EjecutarConsulta(" select count(1) from arbol where arb_arb_id = " +
          dataRowCurrent["IdentificadorNodo"].ToString().Trim());
int indiceImagen = 0;
if (int.Parse(objAccess.InfoTabla[0].Rows[0][0].ToString()) > 0)
indiceImagen = 2;
else
                indiceImagen = 1;

// si el parámetro nodoPadre es nulo es porque es la primera llamada, son los Nodos
// del primer nivel que no dependen de otro nodo.
if (nodePadre == null)
            {
                nuevoNodo.ImageIndex = 3;
                arbolDoc.Nodes.Add(nuevoNodo);

            }
// se añade el nuevo nodo al nodo padre.
else
            {
                nuevoNodo.ImageIndex = indiceImagen;
                nodePadre.Nodes.Add(nuevoNodo);
            }

// Llamada recurrente al mismo método para agregar los Hijos del Nodo recién agregado.
            CrearNodosDelPadre(Int32.Parse(dataRowCurrent["IdentificadorNodo"].ToString()),
          nuevoNodo,arbolDoc);
        }
    }

//Crear dataset con estructura para mapeo en treeview
publicvoid CrearDataSet()
    {
        dataSetArbol = newDataSet("DataSetArbol");

DataTable tablaArbol = dataSetArbol.Tables.Add("TablaArbol");
        tablaArbol.Columns.Add("NombreNodo", typeof(string));
        tablaArbol.Columns.Add("IdentificadorNodo", typeof(Int32));
tablaArbol.Columns.Add("IdentificadorPadre", typeof(Int32));
        tablaArbol.Columns.Add("Path", typeof(string));
        tablaArbol.Columns.Add("Revision", typeof(string));
        tablaArbol.Columns.Add("Remarcar", typeof(string));

DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
        objAccess.EjecutarConsulta("select
         arb_descripcion,arb_id,arb_arb_id,arb_path_documento,arb_revision,arb_remarcar from
         arbol where mat_id = " + this.idMatricula.ToString() + " and tip_id = " + this.idTipo + " and
         arb_borrado = 0 order by arb_id");
for (int i = 0; i < objAccess.InfoTabla[0].Rows.Count; i++)
        {
            InsertarDataRow(objAccess.InfoTabla[0].Rows[i][0].ToString(),
int.Parse(objAccess.InfoTabla[0].Rows[i][1].ToString()),
int.Parse(objAccess.InfoTabla[0].Rows[i][2].ToString()),
```

```
                objAccess.InfoTabla[0].Rows[i][3].ToString(),
                objAccess.InfoTabla[0].Rows[i][4].ToString(),
objAccess.InfoTabla[0].Rows[i][5].ToString()
                );

            }
        }

//Inserción en estructura de nodos para treeview
publicvoid InsertarDataRow(string column1, int column2, int column3, string column4, string
          column5, string column6)
{
DataRow nuevaFila = dataSetArbol.Tables["TablaArbol"].NewRow();
            nuevaFila["NombreNodo"] = column1;
            nuevaFila["IdentificadorNodo"] = column2;
            nuevaFila["IdentificadorPadre"] = column3;
            nuevaFila["Path"] = column4;
            nuevaFila["Revision"] = column5;
            nuevaFila["Remarcar"] = column6;
            dataSetArbol.Tables["TablaArbol"].Rows.Add(nuevaFila);
        }

//Obtener clave primaria para nuevo nodo de árbol
privateint secuencialArbol()
        {
try
            {
DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
                objAccess.EjecutarConsulta("select max(arb_id) + 1 from arbol ");
int secuencial = -1;
if (objAccess.InfoTabla[0].Rows.Count > 0)
secuencial = int.Parse(objAccess.InfoTabla[0].Rows[0][0].ToString());
else
                secuencial = 1;

if (secuencial <= 0)
                secuencial++;
return secuencial;
            }
catch (Exception)
            {
return 1;
}
        }

//Verificación para saber si un nodo tiene hijos y validar la eliminación
privateBoolean TieneHijos(string idNodo)
        {
try
            {
DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
                objAccess.EjecutarConsulta("select count(1) from arbol where arb_arb_id = " + idNodo);
if (int.Parse(objAccess.InfoTabla[0].Rows[0][0].ToString()) > 0)
returntrue;
else
returnfalse;

}
catch (Exception)
{
returnfalse;
```

```
        }
    }

//Obtener valor de un parámetro dado
publicstring ConsultaPath(string idParameto)
    {
try
        {
DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
            objAccess.EjecutarConsulta("select par_valor from parametro where par_id = '" +
          idParameto + "' and par_borrado = 0");
return objAccess.InfoTabla[0].Rows[0][0].ToString();
        }
catch (Exception)
        {
returnstring.Empty;
}
    }
  }


}
```

Consultas.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using RptAppServidor.LINEA_AEREA;

namespace RptAppServidor.Clases
{
classConsultas
  {
//Referencias locales
frmConsultaBitacora FrmconsultaBitacora = null;
frmConsultarUsuarios FrmconsultaUsuario = null;

//Sobrecarga Constructor
public Consultas(frmConsultaBitacora p_frmConsultaBitacora)
    {
this.FrmconsultaBitacora = p_frmConsultaBitacora;
    }

//Sobrecarga Constructor
public Consultas(frmConsultarUsuarios consultaUsuario)
    {
this.FrmconsultaUsuario = consultaUsuario;
    }

//Constructor estandar
public Consultas()
    {
    }

//consulta para pantalla Consulta Usuarios
publicBoolean buscarUsuarios()
```

```
{
try
        {
DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
objAccess.EjecutarConsulta(" select b.ope_descripcion as OPERADOR, a.usu_id as USUARIO,
          a.usu_apellido_paterno as APELLIDO_PATERNO, a.usu_apellido_materno as
          APELLIDO_MATERNO, a.usu_primer_nombre as PRIMER_NOMBRE,
          a.usu_segundo_nombre as SEGUNDO_NOMBRE, a.usu_descripcion as LOGIN,
          a.usu_password as CONTRASENA, d.per_descripcion as PERFIL from usuario a, operador
          b, flota c, perfil d where a.flo_id = c.flo_id and  c.ope_id = b.ope_id and a.per_id = d.per_id
          and usu_borrado = 0 order by ope_descripcion, a.usu_descripcion  ");
this.FrmconsultaUsuario.grdData.DataSource = objAccess.InfoTabla[0];
            objAccess.Cerrar();
returntrue;
        }
catch (Exception)
{
returnfalse;
        }

      }

//Consulta para pantalla Consulta Bitácora
publicBoolean buscarBitacora()
      {
try
        {
DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
objAccess.EjecutarConsulta(" select bit_id as BITACORA,   c.ope_descripcion  as PAIS ,a.usu_id as
          USUARIO, usu_apellido_paterno as APELLIDO_PATERNO, usu_apellido_materno as
          APELLIDO_MATERNO, usu_primer_nombre as PRIMER_NOMBRE, usu_segundo_nombre
          as SEGUNDO_NOMBRE , b.usu_descripcion as LOGIN, bit_fecha as FECHA_REGISTRO
          from bitacora a, usuario b, operador c, flota d "
+ " where a.usu_id = b.usu_id and b.flo_id = d.flo_id  "
                                  + "  and d.ope_id = c.ope_id "
                                  + "  order by bit_fecha ");
this.FrmconsultaBitacora.grdDataBitacora.DataSource = objAccess.InfoTabla[0];
            objAccess.Cerrar();
returntrue;
        }
catch (Exception)
{
returnfalse;
        }

      }

//Consulta de parámetros generales
publicstring ConsultaPath(string idParameto)
      {
try
        {
DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
            objAccess.EjecutarConsulta("select par_valor from parametro where par_id = '" +
          idParameto + "' and par_borrado = 0");
return objAccess.InfoTabla[0].Rows[0][0].ToString();
}
catch (Exception)
        {
returnstring.Empty;
}
```

```
        }

//Consulta genérica para devolver un valor escalar
publicstring ConsultaGenerica(string pSQL)
        {
try
            {
DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
                objAccess.EjecutarConsulta(pSQL);
return objAccess.InfoTabla[0].Rows[0][0].ToString();
            }
catch (Exception)
            {
returnstring.Empty;
            }
        }


    }
}
```

DBacsess.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Data;
using System.Data.SqlClient;
using System.Windows.Forms;

namespace RptAppServidor
{
classDbAccess
    {
//atributos de la clase
private System.Data.OleDb.OleDbConnection CnnSql = null;
publicstring CadenaConexion = null;
privateIDataParameter[] ParametrosSp = newIDataParameter[50];
publicIDataParameter[] paramOut = newIDataParameter[20];
public System.Data.DataTable[] InfoTabla = newDataTable[10];
publicstring NombreSp = string.Empty;

//Constructor
public DbAccess(string p_cadena_conexion)
{
this.CadenaConexion = p_cadena_conexion;
this.Conectar();
        }

//Método de conexión
privatevoid Conectar()
{

this.CnnSql = new System.Data.OleDb.OleDbConnection(this.CadenaConexion);
this.CnnSql.Open();
        }
```

```
//Ejecutar sentencia DML
publicint EjecutarComando(string TextoComando)
      {
DateTime HoraInicio = DateTime.Now;
int NumeroRegistos = -1;
         System.Data.OleDb.OleDbCommand ComandoSql = null;
try
         {
            ComandoSql = new System.Data.OleDb.OleDbCommand();
            ComandoSql.Connection = this.CnnSql;
            ComandoSql.CommandType = System.Data.CommandType.Text;
ComandoSql.CommandText = TextoComando;
            NumeroRegistos = ComandoSql.ExecuteNonQuery();
}
catch (Exception ex)
         {
MessageBox.Show(ex.Message, "Error");
throw;
         }
finally
         {
if (ComandoSql != null)
{
               ComandoSql.Dispose();
            }
         }
return NumeroRegistos;
      }

//Cerrar conexión activa
publicvoid Cerrar()
      {
if (this.CnnSql != null)
         {
this.CnnSql.Close();
this.CnnSql.Dispose();
}
      }

//Realizar una consulta sobre la base de datos y devolver en un datatable par su uso posterior
public System.Data.DataTable EjecutarConsulta(string p_sql)
      {
DateTime HoraInicio = DateTime.Now;
         System.Data.DataTable dtRegistros = null;
         System.Data.OleDb.OleDbCommand ComandoSql = null;
try
         {
            ComandoSql = new System.Data.OleDb.OleDbCommand();
            ComandoSql.Connection = this.CnnSql;
            ComandoSql.CommandType = System.Data.CommandType.Text;
            ComandoSql.CommandText = p_sql;

            System.Data.OleDb.OleDbDataAdapter objDataAdapter = new
          System.Data.OleDb.OleDbDataAdapter(ComandoSql);
            System.Data.DataSet dsResultado = new System.Data.DataSet();
objDataAdapter.Fill(dsResultado);
            dtRegistros = dsResultado.Tables[0];
            InfoTabla[0] = dsResultado.Tables[0];
         }
catch (Exception ex)
{
```

```
MessageBox.Show(ex.Message);
throw;
        }
finally
        {
if (ComandoSql != null)
            {
                ComandoSql.Dispose();
            }
        }
return dtRegistros;
      }
   }
}
```

Flota.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Windows.Forms;


namespace RptAppServidor.Clases
{
classFlota
   {
//atributos locales
publicstring operacioN = string.Empty;
publicfrmFlota frmflota = null;

//Constructor con referencia a pantalla Flotas
public Flota(frmFlota p_frmFlota)
      {
this.frmflota = p_frmFlota;
      }

//LIenado de Grilla con informaciáon de flotas
publicBoolean LlenarGrilla(string idOperador)
      {
try
        {
DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
objAccess.EjecutarConsulta("select FLO_ID as CODIGO_FLOTA, ope_id AS
          CODIGO_OPERADOR, FLO_DESCRIPCION as DESCRIPCION_FLOTA from FLOTA
          where flo_borrado = 0 and ope_id = " + idOperador); //traer datos
frmflota.grdData.DataSource = objAccess.InfoTabla[0];

objAccess.Cerrar();
returntrue;
        }
catch (Exception)
{
returnfalse;
        }
      }
```

```
//Crear una nueva flota
publicBoolean Insertar()
{
try
        {

DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
        objAccess.EjecutarComando(" insert into flota(FLO_DESCRIPCION,OPE_ID) values ('" +
         this.frmflota.txtDescripcion.Text + "' ," + this.frmflota.cboOperador.SelectedValue + " ) ");
         //insertar nuevo
        objAccess.Cerrar();
returntrue;

        }
catch (Exception)
{
returnfalse;
        }

    }

//Borrado lógico de Flota
publicBoolean Eliminar()
{
//try
//{

//DbAccess objAccess = new DbAccess(Util.DbCadenaConexionAccess);
//objAccess.EjecutarComando(" delete from flota where FLO_ID = " + this.frmflota.txtCodigo.Text);
//objAccess.Cerrar();
//return true;
//}
//catch (Exception)
//{
//    return false;
//}
try
        {

DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
          objAccess.EjecutarComando(" update Flota set  flo_borrado = 1 where flo_id = " +
         this.frmflota.txtCodigo.Text);
          objAccess.Cerrar();
returntrue;
        }
catch (Exception)
{
returnfalse;
        }

    }

//Llenado de lista desplegable con Operadores en el formulario flotas
publicBoolean BuscarLinea()
    {

try
        {

DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
```

```
            objAccess.EjecutarConsulta("select ope_id, ope_descripcion from operador where
             ope_borrado = 0 ");

            frmflota.cboOperador.DisplayMember = "ope_descripcion";
            frmflota.cboOperador.ValueMember = "ope_id";
frmflota.cboOperador.DataSource = objAccess.InfoTabla[0];
            objAccess.Cerrar();

returntrue;
            }
catch (Exception)
            {
returnfalse;
}

      }

//Actualizar descripción de Flotas
publicBoolean Actualizar()
{
try
            {

DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
objAccess.EjecutarComando(" update Flota set  flo_descripcion = '" +
              this.frmflota.txtDescripcion.Text
+ "' where flo_id = " + this.frmflota.txtCodigo.Text);
            objAccess.Cerrar();
returntrue;
            }
catch (Exception)
{
returnfalse;
            }

      }

//Limpiar formulario asociado de flotas
publicBoolean limpiar()
      {
            frmflota.txtCodigo.Text = string.Empty;
            frmflota.txtDescripcion.Text = string.Empty;

returntrue;
      }


//Validar información de entrada
publicBoolean ValidarInformacion()
{
if (this.operacioN == "I")
            {

if (frmflota.txtDescripcion.Text.Trim() == string.Empty)
{
MessageBox.Show("La descripciòn es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
                  }
            }
elseif (this.operacioN == "U")
            {
```

```
if (frmflota.txtCodigo.Text.Trim() == string.Empty)
{
MessageBox.Show("El còdigo es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
            }

if (frmflota.txtDescripcion.Text.Trim() == string.Empty)
{
MessageBox.Show("La descripciòn es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
            }
        }
elseif (this.operacioN == "D")
        {
if (frmflota.txtCodigo.Text.Trim() == string.Empty)
{
MessageBox.Show("El còdigo es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
            }
        }
returntrue;
      }

   }
}
```

Inicial.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using RptAppServidor.LINEA_AEREA;
using System.Windows.Forms;

namespace RptAppServidor.Clases
{
classInicial
   {
//Atributos locales
frmInicial frmInicial = null;
frmArbol FrmArbol = null;
privateint opcion = 0;

//Constructor
public Inicial(frmInicial p_frmInicial)
      {
this.frmInicial = p_frmInicial;
      }

//Sobrecarga Constructor
public Inicial(frmArbol p_frmArbol, int p_opcion)
      {
this.FrmArbol = p_frmArbol;
this.opcion = p_opcion;
      }

//Llenar lista desplegable de flota por operador según formulario asociado
```

```
publicBoolean BuscarFlotaPorOperador(string idOperador)
      {
try
         {
DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
            objAccess.EjecutarConsulta("select flo_id, flo_descripcion from flota where ope_id = " +
          idOperador + " and flo_borrado = 0 order by ope_id, flo_id");
if (this.opcion == 0)
            {
this.frmInicial.cboFlota.ValueMember = "flo_id";
this.frmInicial.cboFlota.DisplayMember = "flo_descripcion";
this.frmInicial.cboFlota.DataSource = objAccess.InfoTabla[0];
            }
else
            {
this.FrmArbol.cboFlota.ValueMember = "flo_id";
this.FrmArbol.cboFlota.DisplayMember = "flo_descripcion";
this.FrmArbol.cboFlota.DataSource = objAccess.InfoTabla[0];

            }

            objAccess.Cerrar();
         }
catch (Exception)
         {
returnfalse;
}
returntrue;
      }

//Llenar lista desplegable operador según formulario asociado
publicBoolean BuscarOperador()
      {
try
         {
DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
            objAccess.EjecutarConsulta("select  ope_id, ope_descripcion from operador where
          ope_borrado = 0 order by ope_descripcion");
if (this.opcion == 0)
            {

this.frmInicial.cboOperator.ValueMember = "ope_id";
this.frmInicial.cboOperator.DisplayMember = "ope_descripcion";
this.frmInicial.cboOperator.DataSource = objAccess.InfoTabla[0];
            }
else
            {
this.FrmArbol.cboOperador.ValueMember = "ope_id";
this.FrmArbol.cboOperador.DisplayMember = "ope_descripcion";
this.FrmArbol.cboOperador.DataSource = objAccess.InfoTabla[0];
            }
            objAccess.Cerrar();
         }
catch (Exception)
         {
returnfalse;
}
returntrue;
      }

//Llenar lista desplegable operador según formulario asociado
```

```
publicBoolean BuscarPerfil()
      {
try
        {
DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
            objAccess.EjecutarConsulta("select  per_id, per_descripcion,per_mantenimiento from perfil
          where per_borrado = 0 order by per_descripcion");
this.frmInicial.cboOperator.ValueMember = "pai_id";
this.frmInicial.cboOperator.DisplayMember = "pai_descripcion";
this.frmInicial.cboOperator.DataSource = objAccess.InfoTabla[0];

            objAccess.Cerrar();
        }
catch (Exception)
        {
returnfalse;
}
returntrue;
      }

//Llenar lista desplegable matricula por operador en formulario asociado
publicBoolean BuscarMatriculaPorModelo(string idModelo)
      {
try
        {
DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
            objAccess.EjecutarConsulta("select mat_id, mat_descripcion from matricula where
          mat_borrado = 0 and mod_id = " + idModelo + " order by mod_id, mat_descripcion");
if (this.opcion == 0)
            {
this.frmInicial.cboMatricula.ValueMember = "mat_id";
this.frmInicial.cboMatricula.DisplayMember = "mat_descripcion";
this.frmInicial.cboMatricula.DataSource = objAccess.InfoTabla[0];
            }
else
            {
this.FrmArbol.cboMatricula.ValueMember = "mat_id";
this.FrmArbol.cboMatricula.DisplayMember = "mat_descripcion";
this.FrmArbol.cboMatricula.DataSource = objAccess.InfoTabla[0];
            }

            objAccess.Cerrar();
        }
catch (Exception)
        {
returnfalse;
}
returntrue;
      }

//Llenar lista desplegable modelo por flota en formulario asociado
publicBoolean BuscarModeloPorFlota(string idFlota)
      {
try
        {
DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
objAccess.EjecutarConsulta("select mod_id, mod_descripcion from modelo where mod_borrado = 0
          and flo_id = " + idFlota + " order by flo_id, mod_descripcion");
if (this.opcion == 0)
            {
this.frmInicial.cboModelo.ValueMember = "mod_id";
```

```
this.frmInicial.cboModelo.DisplayMember = "mod_descripcion";
this.frmInicial.cboModelo.DataSource = objAccess.InfoTabla[0];
            }
else
            {
this.FrmArbol.cboModelo.ValueMember = "mod_id";
this.FrmArbol.cboModelo.DisplayMember = "mod_descripcion";
this.FrmArbol.cboModelo.DataSource = objAccess.InfoTabla[0];
            }

            objAccess.Cerrar();
        }
catch (Exception)
        {
returnfalse;
        }
returntrue;
      }


publicBoolean BuscarActualizaciones(string idFlota)
      {
try
        {
DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
            objAccess.EjecutarConsulta("select arb_id, arb_descripcion +  '  /  ' + arb_revision  + '  / ' +
          arb_path_documento  as DOCUMENTACION from arbol a, matricula b, modelo c where
          a.mat_id = b.mat_id  and c.mod_id = b.mod_id and c.flo_id = " + idFlota + " and
          arb_remarcar = 'S' and arb_borrado = 0 order by a.arb_descripcion");
//this.frmInicial.lstCambios.ValueMember = "arb_id";
//this.frmInicial.lstCambios.DisplayMember = "DOCUMENTACION";
//this.frmInicial.lstCambios.DataSource = objAccess.InfoTabla[0];

            objAccess.Cerrar();
        }
catch (Exception)
        {
returnfalse;
}
returntrue;
      }

//Guarda log de bitácora cuando un usuario se loguea.
publicBoolean GuardarBitacora(string usuario)
      {
try
        {
DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
            objAccess.EjecutarComando(" insert into bitacora(usu_id,bit_fecha) values('" + usuario +
         "','"   +  DateTime.Now.ToString() + "')");
objAccess.Cerrar();
        }
catch (Exception)
        {
returnfalse;
        }
returntrue;
}
```

```
    }

}
```

Login.cs

```
using System;
using System.Collections.Generic;
using System.Text;
using System.Data;
using RptAppServidor.LINEA_AEREA;
using RptAppServidor.Clases;
using System.Windows.Forms;

namespace RptAppServidor.Clases
{
classLogin
    {
//atributos locales
publicint idusuario = 0;
publicstring login = string.Empty;
publicstring nombreUsuario = string.Empty ;
publicint perfilUsuario = 0;
frmLogin FrmLogin = null;

//Constructor
public Login(frmLogin p_frmLogin)
{
this.FrmLogin = p_frmLogin;

      }

//validación de ingreso de usuario
publicBoolean ValidarUsuario(string usuario, string password)
      {
string idLogin = string.Empty;
try
        {
DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
objAccess.EjecutarConsulta("select usu_id, usu_descripcion, usu_primer_nombre,
          usu_segundo_nombre, usu_apellido_paterno, usu_apellido_materno, per_id from usuario
          where usu_descripcion = '" + usuario + "' and usu_password = '" + password  + "' and
          usu_borrado = 0 ");

objAccess.Cerrar();
if (objAccess.InfoTabla[0].Rows.Count > 0)
{

this.idusuario = int.Parse(objAccess.InfoTabla[0].Rows[0][0].ToString());
this.login = objAccess.InfoTabla[0].Rows[0][1].ToString();
this.nombreUsuario = objAccess.InfoTabla[0].Rows[0][2].ToString() + " " +
          objAccess.InfoTabla[0].Rows[0][3].ToString() + " "
                + objAccess.InfoTabla[0].Rows[0][4].ToString() + " " +
          objAccess.InfoTabla[0].Rows[0][5].ToString();
this.perfilUsuario = int.Parse(objAccess.InfoTabla[0].Rows[0][6].ToString());
```

```
this.GuardarBitacora();
frmInicial objFrmInicial = newfrmInicial(idusuario,login,nombreUsuario,perfilUsuario);
this.FrmLogin.Visible = false;
Util.AbrirVentana(objFrmInicial,0);

this.FrmLogin.Visible = true;

            }
else
MessageBox.Show("Usuario no válido para el pais ingresado / Usuario o Password no válido. ");
        }
catch (Exception)
        {
returnfalse;
        }
returntrue;
      }

//Guardar log de bitácora cuando un usuario se conecta.
publicBoolean GuardarBitacora()
      {
try
        {
DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
            objAccess.EjecutarComando(" insert into bitacora(usu_id,bit_fecha) values('" +
          this.idusuario.ToString() + "','"   +  DateTime.Now.ToString() + "')");
objAccess.Cerrar();
        }
catch (Exception)
        {
returnfalse;
}
returntrue;
      }
   }
}
```

Matricula.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace RptAppServidor.Clases
{
classMatricula
   {
//atributos locales
publicint codigoMatricula = 0;
publicstring nombreMatricula = string.Empty;
publicstring operacioN = string.Empty;
publicstring usuariO = string.Empty;
publicstring like = string.Empty;

//formulario asociado
publicFrmMatricula FrmMatricula = null;
```

```
//constructor
public Matricula(FrmMatricula p_FrmMatricula)
      {
this.FrmMatricula = p_FrmMatricula;
      }

//Llena grilla de matrículas por modelo
publicBoolean LlenarGrilla(string idModelo)
      {
DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
          objAccess.EjecutarConsulta("select mat_ID AS CODIGO_MATRICULA, MOD_ID AS
           CODIGO_MODELO ,mat_DESCRIPCION AS DESCRIPCION_MATRICULA from matricula
           where mat_borrado = 0 and mod_id = " + idModelo); //traer datos
FrmMatricula.grdData.DataSource = objAccess.InfoTabla[0];
objAccess.Cerrar();
returntrue;
      }


//crear nueva matrícula
publicBoolean Insertar()
      {

DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
          objAccess.EjecutarComando(" insert into matricula(mat_DESCRIPCION,mod_ID) values ('" +
           this.FrmMatricula.txtDescripcion.Text + "' ," +
           this.FrmMatricula.cboModelo.SelectedValue.ToString() + " ) "); //insertar nuevo//linid es
           entero
objAccess.Cerrar();
returntrue;
      }

//eliminado lógico de matrícula
publicBoolean Eliminar()
      {
//DbAccess objAccess = new DbAccess(Util.DbCadenaConexionAccess);
//objAccess.EjecutarComando(" delete from matricula where mat_ID = " +
          this.FrmMatricula.txtCodigo.Text);
//objAccess.Cerrar();
//return true;
DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
          objAccess.EjecutarComando(" update matricula set  mat_borrado = 1 "
                                + " where mat_id = " + this.FrmMatricula.txtCodigo.Text);
objAccess.Cerrar();
returntrue;
      }

//actualización de datos matrícula
publicBoolean Actualizar()
      {
DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
          objAccess.EjecutarComando(" update matricula set  mat_descripcion = '" +
           this.FrmMatricula.txtDescripcion.Text
+ "' where mat_id = " + this.FrmMatricula.txtCodigo.Text);
objAccess.Cerrar();
returntrue;
      }

//limpiar formulario asociado
publicBoolean limpiar()
```

```
        {
            FrmMatricula.txtCodigo.Text = string.Empty;
            FrmMatricula.txtDescripcion.Text = string.Empty;

returntrue;
}

//Validar información de acuerdo a acción a realizarse
publicBoolean ValidarInformacion()
        {
if (this.operacioN == "I")
            {
if (this.FrmMatricula.cboOperador.SelectedIndex < 0)
{
MessageBox.Show("El operador es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
                }

if (this.FrmMatricula.cboFlota.SelectedIndex < 0)
{
MessageBox.Show("La flota es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
                }

if (this.FrmMatricula.cboModelo.SelectedIndex < 0)
{
MessageBox.Show("el modelo es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
            }

if (FrmMatricula.txtDescripcion.Text.Trim() == string.Empty)
{
MessageBox.Show("La descripciòn es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
                }
            }
elseif (this.operacioN == "U")
            {
if (this.FrmMatricula.cboOperador.SelectedIndex < 0)
{
MessageBox.Show("El operador es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
                }

if (this.FrmMatricula.cboFlota.SelectedIndex < 0)
{
MessageBox.Show("La flota es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
                }

if (this.FrmMatricula.cboModelo.SelectedIndex < 0)
{
MessageBox.Show("el modelo es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
                }

if (FrmMatricula.txtCodigo.Text.Trim() == string.Empty)
{
MessageBox.Show("El còdigo es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
                }
```

```
if (FrmMatricula.txtDescripcion.Text.Trim() == string.Empty)
{
MessageBox.Show("La descripciòn es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
            }
        }
elseif (this.operacioN == "D")
        {
if (FrmMatricula.txtCodigo.Text.Trim() == string.Empty)
{
MessageBox.Show("El còdigo es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
            }
        }
returntrue;
    }

//LIenar lista desplegable operador
publicBoolean BuscarOperador()
    {
try
        {
DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
            objAccess.EjecutarConsulta("select  ope_id, ope_descripcion from operador where
          ope_borrado = 0 order by ope_descripcion");
this.FrmMatricula.cboOperador.ValueMember = "ope_id";
this.FrmMatricula.cboOperador.DisplayMember = "ope_descripcion";
this.FrmMatricula.cboOperador.DataSource = objAccess.InfoTabla[0];
            objAccess.Cerrar();
}
catch (Exception)
        {
returnfalse;
        }
returntrue;
}

//LIenar lista desplegable de flota por operador
publicBoolean BuscarFlotaPorOperador(string idOperador)
    {
try
        {
DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
            objAccess.EjecutarConsulta("select flo_id, flo_descripcion from flota where flo_borrado = 0
          and ope_id = " + idOperador + " order by ope_id, flo_id");

this.FrmMatricula.cboFlota.ValueMember = "flo_id";
this.FrmMatricula.cboFlota.DisplayMember = "flo_descripcion";
this.FrmMatricula.cboFlota.DataSource = objAccess.InfoTabla[0];

            objAccess.Cerrar();
        }
catch (Exception)
        {
returnfalse;
}
returntrue;
    }

//llenar lista desplegable para buscar Modelo por flota
```

```
publicBoolean BuscarModeloPorFlota(string idFlota)
      {
try
         {
DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
objAccess.EjecutarConsulta("select mod_id, mod_descripcion from modelo where mod_borrado = 0
          and flo_id = " + idFlota + " order by flo_id, mod_descripcion");

this.FrmMatricula.cboModelo.ValueMember = "mod_id";
this.FrmMatricula.cboModelo.DisplayMember = "mod_descripcion";
this.FrmMatricula.cboModelo.DataSource = objAccess.InfoTabla[0];

            objAccess.Cerrar();
         }
catch (Exception)
         {
returnfalse;
         }
returntrue;
      }


   }
}
```

MenuDocumentacion.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using RptAppServidor.LINEA_AEREA;

namespace RptAppServidor.Clases
{
classMenuDocumentacion
   {
//atributos localse
publicint orden_opcion1 = 0;
publicint orden_opcion2 = 0;
publicint orden_opcion3 = 0;

publicstring etiqueta_opcion1 = string.Empty;
publicstring etiqueta_opcion2 = string.Empty;
publicstring etiqueta_opcion3 = string.Empty;

publicstring path_opcion1 = string.Empty;
publicstring path_opcion2 = string.Empty;
publicstring path_opcion3 = string.Empty;

//formulario asociado
privatefrmMenuDocumentacion FrmMenuDocumentacion = null;

//Constructor
public MenuDocumentacion(frmMenuDocumentacion p_FrmMenuDocumentacion)
      {
this.FrmMenuDocumentacion = p_FrmMenuDocumentacion;
      }

//Mapear texto en etiquetas de botones
```

```
publicBoolean ConsultaInformacionBotones(string idFlota)
      {
try
         {
DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
            objAccess.EjecutarConsulta("select opc_orden,opc_etiqueta,opc_programa from
          opciones_modelo where flo_id = " + idFlota + " and opc_borrado = 0 order by opc_orden ");
this.orden_opcion1 = int.Parse(objAccess.InfoTabla[0].Rows[0]["opc_orden"].ToString());
this.etiqueta_opcion1 = objAccess.InfoTabla[0].Rows[0]["opc_etiqueta"].ToString();
this.path_opcion1 = objAccess.InfoTabla[0].Rows[0]["opc_programa"].ToString();

this.orden_opcion2 = int.Parse(objAccess.InfoTabla[0].Rows[1]["opc_orden"].ToString());
this.etiqueta_opcion2 = objAccess.InfoTabla[0].Rows[1]["opc_etiqueta"].ToString();
this.path_opcion2 = objAccess.InfoTabla[0].Rows[1]["opc_programa"].ToString();

this.orden_opcion3 = int.Parse(objAccess.InfoTabla[0].Rows[2]["opc_orden"].ToString());
this.etiqueta_opcion3 = objAccess.InfoTabla[0].Rows[2]["opc_etiqueta"].ToString();
this.path_opcion3 = objAccess.InfoTabla[0].Rows[2]["opc_programa"].ToString();

            objAccess.Cerrar();
returntrue;

         }
catch (Exception)
         {
returnfalse;
         }
      }
   }
}
```

Modelo.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace RptAppServidor.Clases
{
classModelo
   {
//atributos locales
publicint codigoLineaAErea = 0;
publicstring nombreLinea = string.Empty;
publicstring operacioN;
publicstring like = string.Empty;

//formulario asociado
publicfrmModelo frmModelo = null;

//constructor
public Modelo(frmModelo p_frmModelo)
      {
this.frmModelo = p_frmModelo;
      }

//llenar grilla de modelo por flota
publicBoolean LlenarGrilla(string idFlota)
```

```
        {
DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
objAccess.EjecutarConsulta("select mod_id as CODIGO_MODELO,flo_id as CODIGO_FLOTA,
            mod_descripcion as DESCRIPCION_MODELO from MODELO where mod_borrado = 0 and
            flo_id = " + idFlota); //traer datos
frmModelo.grdData.DataSource = objAccess.InfoTabla[0];


objAccess.Cerrar();
returntrue;
        }

//Crear nuevo modelo
publicBoolean Insertar()
        {

DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
            objAccess.EjecutarComando(" insert into modelo(mod_descripcion,flo_id) values ('" +
             this.frmModelo.txtDescripcion.Text + "'," + this.frmModelo.cboFlota.SelectedValue.ToString()
             + ") "); //insertar nuevo
objAccess.Cerrar();
returntrue;
        }

//eliminar lógicamente modelo
publicBoolean Eliminar()
        {
//DbAccess objAccess = new DbAccess(Util.DbCadenaConexionAccess);
//objAccess.EjecutarComando(" delete from modelo where mod_id = " +
            this.frmModelo.txtCodigo.Text);
//objAccess.Cerrar();
//return true;
DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
            objAccess.EjecutarComando(" update modelo set  mod_borrado = 1 where mod_id = " +
             this.frmModelo.txtCodigo.Text);
objAccess.Cerrar();
returntrue;
        }

//actualizar modelo
publicBoolean Actualizar()
{
DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
objAccess.EjecutarComando(" update modelo set  mod_descripcion = '" +
            this.frmModelo.txtDescripcion.Text
+ "' where mod_id = " + this.frmModelo.txtCodigo.Text );
objAccess.Cerrar();
returntrue;
        }

//limpiar formulario asociado
publicBoolean limpiar()
        {
            frmModelo.txtCodigo.Text = string.Empty ;
            frmModelo.txtDescripcion.Text = string.Empty;
//this.frmModelo.txtBuscar.Text = string.Empty;
//this.LlenarGrilla();

returntrue;
        }
```

```
//validar información de acuerdo a operación escogida
publicBoolean ValidarInformacion()
      {
if (this.operacioN == "I")
         {
if (this.frmModelo.cboOperador.SelectedIndex < 0)
{
MessageBox.Show("El operador es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
            }

if (this.frmModelo.cboFlota.SelectedIndex < 0)
{
MessageBox.Show("La flota es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
            }
if (frmModelo.txtDescripcion.Text.Trim() == string.Empty)
{
MessageBox.Show("La descripciòn es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
            }
         }
elseif (this.operacioN == "U")
         {
if (this.frmModelo.cboOperador.SelectedIndex < 0)
{
MessageBox.Show("El operador es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
            }

if (this.frmModelo.cboFlota.SelectedIndex < 0)
{
MessageBox.Show("La flota es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
            }

if (frmModelo.txtCodigo.Text.Trim() == string.Empty)
{
MessageBox.Show("El còdigo es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
            }

if (frmModelo.txtDescripcion.Text.Trim() == string.Empty)
{
MessageBox.Show("La descripciòn es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
            }
         }
elseif (this.operacioN == "D")
         {
if (frmModelo.txtCodigo.Text.Trim() == string.Empty)
{
MessageBox.Show("El còdigo es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
            }
         }
returntrue;
      }

//Buscar operador y mapearlo en lista desplegable
publicBoolean BuscarOperador()
```

```
        {
try
            {
DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
                objAccess.EjecutarConsulta("select  ope_id, ope_descripcion from operador where
              ope_borrado = 0 order by ope_descripcion");
this.frmModelo.cboOperador.ValueMember = "ope_id";
this.frmModelo.cboOperador.DisplayMember = "ope_descripcion";
this.frmModelo.cboOperador.DataSource = objAccess.InfoTabla[0];
                objAccess.Cerrar();
            }
catch (Exception)
            {
returnfalse;
}
returntrue;
        }

//buscar flota por operador y mapearlo en lista desplegable
publicBoolean BuscarFlotaPorOperador(string idOperador)
        {
try
            {
DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
                objAccess.EjecutarConsulta("select flo_id, flo_descripcion from flota where flo_borrado = 0
              and ope_id = " + idOperador + " order by ope_id, flo_id");

this.frmModelo.cboFlota.ValueMember = "flo_id";
this.frmModelo.cboFlota.DisplayMember = "flo_descripcion";
this.frmModelo.cboFlota.DataSource = objAccess.InfoTabla[0];

                objAccess.Cerrar();
            }
catch (Exception)
            {
returnfalse;
}
returntrue;
        }


    }
}
```

OpcionesModelo.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using RptAppServidor.LINEA_AEREA;

namespace RptAppServidor.Clases
{
classOpcionesModelo
{
//atributos locales
```

```
publicint codigoLineaAErea = 0;
publicstring nombreLinea = string.Empty;
publicstring operacioN;
publicstring usuariO = string.Empty ;
publicstring like = string.Empty;

//formulario asociado
publicfrmOpcionesModelo frmOpcionesModelo = null;

//constructor
public OpcionesModelo(frmOpcionesModelo p_frmOpcionesModelo)
      {
this.frmOpcionesModelo = p_frmOpcionesModelo;
      }

//lenar grilla de opciones por flota
publicBoolean LlenarGrilla(string idFlota)
      {
DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);

objAccess.EjecutarConsulta("SELECT OPC_ID AS CODIGO_OPCION, OPC_ETIQUETA AS
          ETIQUETA, OPC_PROGRAMA AS PROGRAMA_OPCION, FLO_ID AS CODIGO_FLOTA,
          OPC_ORDEN AS ORDEN_BOTON FROM OPCIONES_MODELO where opc_borrado = 0"
          + " and flo_id = " + idFlota); //traer datos
         frmOpcionesModelo.grdData.DataSource = objAccess.InfoTabla[0];

          objAccess.Cerrar();
returntrue;
      }

//insertar nueva opción
publicBoolean Insertar()
{

DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
        objAccess.EjecutarComando(" insert into
          opciones_modelo(OPC_ETIQUETA,FLO_ID,OPC_PROGRAMA,OPC_ORDEN) values ('" +
          this.frmOpcionesModelo.txtDescripcion.Text + "'," +
          this.frmOpcionesModelo.cboFlota.SelectedValue.ToString() + ","
                    + """ + this.frmOpcionesModelo.txtArchivo.Text  + "',"  +
          this.frmOpcionesModelo.cboOrden.Text + "  ) "); //insertar nuevo
objAccess.Cerrar();
returntrue;
      }

//eliminado lógico de una opción
publicBoolean Eliminar()
      {
//DbAccess objAccess = new DbAccess(Util.DbCadenaConexionAccess);
//objAccess.EjecutarComando(" delete from modelo where mod_id = " +
          this.frmModelo.txtCodigo.Text);
//objAccess.Cerrar();
//return true;
DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
          objAccess.EjecutarComando(" update opciones_modelo set  opc_borrado = 1 where opc_id
          = " + this.frmOpcionesModelo.txtCodigo.Text);
objAccess.Cerrar();
returntrue;
      }

//actualización lógica de una opción
```

```
publicBoolean Actualizar()
{
DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
objAccess.EjecutarComando(" update opciones_modelo set  opc_etiqueta = '" +
          this.frmOpcionesModelo.txtDescripcion.Text + "',"
                                + " OPC_PROGRAMA = '" + this.frmOpcionesModelo.txtArchivo.Text + "',"
                                + " OPC_ORDEN = " + this.frmOpcionesModelo.cboOrden.Text
                                + " where OPC_ID = " + this.frmOpcionesModelo.txtCodigo.Text);
          objAccess.Cerrar();
returntrue;
      }

//limpiar formulario asociado
publicBoolean limpiar()
      {
          frmOpcionesModelo.txtCodigo.Text = string.Empty;
          frmOpcionesModelo.txtDescripcion.Text = string.Empty;
          frmOpcionesModelo.txtArchivo.Text = string.Empty;
frmOpcionesModelo.cboOrden.Text = "1";


returntrue;
}


//validar información de acuerdo a operación escogida
publicBoolean ValidarInformacion()
      {
if (this.operacioN == "I")
          {
if (this.frmOpcionesModelo.cboOperador.SelectedIndex < 0)
{
MessageBox.Show("El operador es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
              }

if (this.frmOpcionesModelo.cboFlota.SelectedIndex < 0)
{
MessageBox.Show("La flota es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
              }
if (frmOpcionesModelo.txtDescripcion.Text.Trim() == string.Empty)
              {
MessageBox.Show("La descripciòn es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
              }
if (this.frmOpcionesModelo.cboOrden.SelectedIndex < 0)
{
MessageBox.Show("El orden es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
              }
          }
elseif (this.operacioN == "U")
          {
if (this.frmOpcionesModelo.cboOperador.SelectedIndex < 0)
{
MessageBox.Show("El operador es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
              }

if (this.frmOpcionesModelo.cboFlota.SelectedIndex < 0)
```

```
{
MessageBox.Show("La flota es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
            }

if (frmOpcionesModelo.txtCodigo.Text.Trim() == string.Empty)
            {
MessageBox.Show("El còdigo es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
            }

if (frmOpcionesModelo.txtDescripcion.Text.Trim() == string.Empty)
            {
MessageBox.Show("La descripciòn es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
            }
if (this.frmOpcionesModelo.cboOrden.SelectedIndex < 0)
{
MessageBox.Show("El orden es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
            }

        }
elseif (this.operacioN == "D")
        {
if (frmOpcionesModelo.txtCodigo.Text.Trim() == string.Empty)
{
MessageBox.Show("El còdigo es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
            }
        }
returntrue;
      }

//llenar lista desplegable de operador
publicBoolean BuscarOperador()
      {
try
        {
DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
            objAccess.EjecutarConsulta("select  ope_id, ope_descripcion from operador where
         ope_borrado = 0 order by ope_descripcion");
this.frmOpcionesModelo.cboOperador.ValueMember = "ope_id";
this.frmOpcionesModelo.cboOperador.DisplayMember = "ope_descripcion";
this.frmOpcionesModelo.cboOperador.DataSource = objAccess.InfoTabla[0];
            objAccess.Cerrar();
}
catch (Exception)
        {
returnfalse;
        }
returntrue;
}


//llenar lista desplegable de flota por operador
publicBoolean BuscarFlotaPorOperador(string idOperador)
      {
try
        {
DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
```

```
            objAccess.EjecutarConsulta("select flo_id, flo_descripcion from flota where flo_borrado = 0
          and ope_id = " + idOperador + " order by ope_id, flo_id");

this.frmOpcionesModelo.cboFlota.ValueMember = "flo_id";
this.frmOpcionesModelo.cboFlota.DisplayMember = "flo_descripcion";
this.frmOpcionesModelo.cboFlota.DataSource = objAccess.InfoTabla[0];

objAccess.Cerrar();
        }
catch (Exception)
        {
returnfalse;
}
returntrue;
      }

   }
}
```

Operador.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace RptAppServidor.Clases
{
classOperador
   {
//atributos locales
publicstring operacioN;
privatefrmOperador frmOperador;

//constructor local
public Operador(frmOperador p_frmOperador)
      {
this.frmOperador = p_frmOperador;
      }

//llenar grilla con operadores
publicBoolean LlenarGrilla()
      {
DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
objAccess.EjecutarConsulta("select ope_id as CODIGO_OPERADOR, ope_descripcion as
          DESCRIPCION_OPERADOR from operador where ope_borrado = 0"); //traer datos
frmOperador.grdData.DataSource = objAccess.InfoTabla[0];

objAccess.Cerrar();
returntrue;
      }

//método para búsqueda por ingreso en text box superior
publicBoolean BuscarLike()
```

```
        {
if (this.frmOperador.txtBuscar.Text.Trim() != string.Empty)
            {
DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
                objAccess.EjecutarConsulta("select ope_id as CODIGO_OPERADOR, ope_descripcion as
              DESCRIPCION_OPERADOR from operador where ope_descripcion like '%" +
              this.frmOperador.txtBuscar.Text + "%' and ope_borrado = 0"); //traer datos
                frmOperador.grdData.DataSource = objAccess.InfoTabla[0];
objAccess.Cerrar();
            }
else
this.LlenarGrilla();
returntrue;
}

//crear nuevo operador
publicBoolean Insertar()
{

DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
objAccess.EjecutarComando(" insert into operador(ope_descripcion) values ('" +
              this.frmOperador.txtDescripcion.Text + "' ) "); //insertar nuevo
            objAccess.Cerrar();
returntrue;
//this.frmlineAerea.cboEstado.SelectedItem.ToString() para tomar el valor a insertar o actualizar de
              una lista desplegable o combo
        }

//eliminación lógica de operador
publicBoolean Eliminar()
{
//DbAccess objAccess = new DbAccess(Util.DbCadenaConexionAccess);
//objAccess.EjecutarComando(" delete from operador where ope_id = " +
              this.frmOperador.txtCodigo.Text);
//objAccess.Cerrar();
//return true;
DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
            objAccess.EjecutarComando(" update operador set ope_borrado = 1 where ope_id = " +
              this.frmOperador.txtCodigo.Text);
            objAccess.Cerrar();
returntrue;
        }

//actualización de operador
publicBoolean Actualizar()
{
DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
objAccess.EjecutarComando(" update operador set  ope_descripcion = '" +
              this.frmOperador.txtDescripcion.Text
+ "' where ope_id = " + this.frmOperador.txtCodigo.Text);
objAccess.Cerrar();
returntrue;
        }

//limpiar formulario asociado
publicBoolean limpiar()
        {
            frmOperador.txtCodigo.Text = string.Empty;
            frmOperador.txtDescripcion.Text = string.Empty;
            frmOperador.txtBuscar.Text = string.Empty;
//this.LlenarGrilla();
```

```
returntrue;
      }


//validación de información a ingresar de acuero a operación escogida
publicBoolean ValidarInformacion()
      {
if (this.operacioN == "I")
         {

if (frmOperador.txtDescripcion.Text.Trim() == string.Empty)
{
MessageBox.Show("La descripciòn es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
            }
         }
elseif (this.operacioN == "U")
         {
if (frmOperador.txtCodigo.Text.Trim() == string.Empty)
{
MessageBox.Show("El còdigo es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
            }

if (frmOperador.txtDescripcion.Text.Trim() == string.Empty)
{
MessageBox.Show("La descripciòn es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
            }
         }
elseif (this.operacioN == "D")
         {
if (frmOperador.txtCodigo.Text.Trim() == string.Empty)
{
MessageBox.Show("El còdigo es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
            }
         }
returntrue;
      }


   }
   }
```

Parametro.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace RptAppServidor.Clases
{
classParametro
```

```
   {
//atributos locales
publicint codigoParametro = 0;
publicstring nombreParametro = string.Empty;
publicstring operacioN;
publicstring usuariO = string.Empty;
publicstring like = string.Empty;
privatefrmParametro frmParametro;

//constructor local
public Parametro(frmParametro p_frmParametro)
      {
this.frmParametro = p_frmParametro;
      }

//llenar grilla con operadores
publicBoolean LlenarGrilla()
      {
DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
objAccess.EjecutarConsulta("select PAR_ID as CODIGO, par_descripcion AS DESCRIPCION,
          par_valor AS VALOR from parametro where par_borrado = 0"); //traer datos
frmParametro.grdData.DataSource = objAccess.InfoTabla[0];

objAccess.Cerrar();
returntrue;
      }

//método para búsqueda por ingreso en text box superior
publicBoolean BuscarLike()
      {
if (this.frmParametro.txtBuscar.Text.Trim() != string.Empty)
         {
DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
objAccess.EjecutarConsulta("select PAR_ID as CODIGO, par_descripcion AS DESCRIPCION,
          par_valor AS VALOR from parametro where par_descripcion like '%" +
          this.frmParametro.txtBuscar.Text + "%' and par_borrado = 0"); //traer datos
frmParametro.grdData.DataSource = objAccess.InfoTabla[0];
            objAccess.Cerrar();
         }
else
this.LlenarGrilla();
returntrue;
}

//crear nuevo parámetro
publicBoolean Insertar()
{

DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
         objAccess.EjecutarComando(" insert into PARAMETRO(par_id, par_descripcion, par_valor)
          values ('" + this.frmParametro.txtCodigo.Text + "','" + this.frmParametro.txtDescripcion.Text
          + "','"  + this.frmParametro.txtValor.Text +  "' ) "); //insertar nuevo
objAccess.Cerrar();
returntrue;
      }

//eliminación lógica de parámetro
publicBoolean Eliminar()
      {
//DbAccess objAccess = new DbAccess(Util.DbCadenaConexionAccess);
```

```
//objAccess.EjecutarComando(" delete from parametro where par_id = '" +
            this.frmParametro.txtCodigo.Text + "'");
//objAccess.Cerrar();
//return true;
DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
objAccess.EjecutarComando(" update parametro set  par_descripcion = '" +
            this.frmParametro.txtDescripcion.Text + "',"
                                + " par_borrado = 1 "
                                + " where par_id = '" + this.frmParametro.txtCodigo.Text + "'");
          objAccess.Cerrar();
returntrue;
       }

//actualización de parámetro
publicBoolean Actualizar()
{
DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
objAccess.EjecutarComando(" update parametro set  par_descripcion = '" +
            this.frmParametro.txtDescripcion.Text + "',"
                                + " par_valor = '" + this.frmParametro.txtValor.Text + "'"
+ " where par_id = '" + this.frmParametro.txtCodigo.Text + "'");
objAccess.Cerrar();
returntrue;
       }

//limpiar formulario asociado
publicBoolean limpiar()
       {
          frmParametro.txtCodigo.Text = string.Empty;
          frmParametro.txtDescripcion.Text = string.Empty;
          frmParametro.txtBuscar.Text = string.Empty;
this.frmParametro.txtValor.Text = string.Empty;
this.frmParametro.txtCodigo.Enabled = true;


returntrue;
       }


//validación de información a ingresar de acuero a operación escogida
publicBoolean ValidarInformacion()
       {
if (this.operacioN == "I")
          {
if (frmParametro.txtCodigo.Text.Trim() == string.Empty)
{
MessageBox.Show("El còdigo es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
             }
if (frmParametro.txtDescripcion.Text.Trim() == string.Empty)
{
MessageBox.Show("La descripciòn es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
             }
if (frmParametro.txtValor.Text.Trim() == string.Empty)
{
MessageBox.Show("El valor es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
             }

          }
```

```
elseif (this.operacioN == "U")
        {
if (frmParametro.txtCodigo.Text.Trim() == string.Empty)
{
MessageBox.Show("El còdigo es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
            }

if (frmParametro.txtDescripcion.Text.Trim() == string.Empty)
{
MessageBox.Show("La descripciòn es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
            }
if (frmParametro.txtValor.Text.Trim() == string.Empty)
{
MessageBox.Show("El valor es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
            }

        }
elseif (this.operacioN == "D")
        {
if (frmParametro.txtCodigo.Text.Trim() == string.Empty)
{
MessageBox.Show("El còdigo es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
            }
        }
returntrue;
      }
   }
}
```

Tipo.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace RptAppServidor.Clases
{
classTipo
   {

//atributos locales
publicint codigoPais = 0;
publicstring nombrePais = string.Empty;
publicstring operacioN = string.Empty;
publicstring usuariO = string.Empty;
publicstring like = string.Empty;

//formulario asociado
privatefrmTipo frmTipo;

//constructor local
public Tipo(frmTipo p_frmTipo)
```

```
        {
this.frmTipo = p_frmTipo;
        }

//llenar grilla con operadores
publicBoolean LlenarGrilla()
        {
DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
objAccess.EjecutarConsulta("select tip_ID as CODIGO, tip_descripcion AS DESCRIPCION from tipo
            where tip_borrado = 0"); //traer datos
frmTipo.grdData.DataSource = objAccess.InfoTabla[0];

objAccess.Cerrar();
returntrue;
        }

//método para búsqueda por ingreso en text box superior
publicBoolean BuscarLike()
        {
if (this.frmTipo.txtBuscar.Text.Trim() != string.Empty)
            {
DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
                objAccess.EjecutarConsulta("select tip_id, tip_descripcion from tipo where tip_descripcion
              like '%" + this.frmTipo.txtBuscar.Text + "%' and tip_borrado = 0"); //traer datos
                frmTipo.grdData.DataSource = objAccess.InfoTabla[0];
                objAccess.Cerrar();
            }
else
this.LlenarGrilla();
returntrue;
}

//crear nuevo tipo
publicBoolean Insertar()
{

DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
objAccess.EjecutarComando(" insert into tipo(tip_descripcion) values ('" +
             this.frmTipo.txtDescripcion.Text + "' ) "); //insertar nuevo
            objAccess.Cerrar();
returntrue;
//this.frmlineAerea.cboEstado.SelectedItem.ToString() para tomar el valor a insertar o actualizar de
             una lista desplegable o combo
        }

//eliminación lógica de tipo
publicBoolean Eliminar()
{
/*DbAccess objAccess = new DbAccess(Util.DbCadenaConexionAccess);
            objAccess.EjecutarComando(" delete from tipo where tip_id = " +
             this.frmTipo.txtCodigo.Text);
            objAccess.Cerrar();
            return true;*/
DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
objAccess.EjecutarComando(" update tipo set  tip_descripcion = '" + this.frmTipo.txtDescripcion.Text
                                + "' , tip_borrado = 1"
                                + " where tip_id = " + this.frmTipo.txtCodigo.Text);
            objAccess.Cerrar();
returntrue;

        }
```

```
//actualización de tipo
publicBoolean Actualizar()
      {
DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
objAccess.EjecutarComando(" update tipo set  tip_descripcion = '" + this.frmTipo.txtDescripcion.Text
+ "' where tip_id = " + this.frmTipo.txtCodigo.Text);
objAccess.Cerrar();
returntrue;
      }

//limpiar formulario asociado
publicBoolean limpiar()
      {
         frmTipo.txtCodigo.Text = string.Empty;
         frmTipo.txtDescripcion.Text = string.Empty;
frmTipo.txtBuscar.Text = string.Empty;
//this.LlenarGrilla();

returntrue;
      }


//validación de información a ingresar de acuero a operación escogida
publicBoolean ValidarInformacion()
      {
if (this.operacioN == "I")
         {

if (frmTipo.txtDescripcion.Text.Trim() == string.Empty)
{
MessageBox.Show("La descripciòn es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
            }
         }
elseif (this.operacioN == "U")
         {
if (frmTipo.txtCodigo.Text.Trim() == string.Empty)
{
MessageBox.Show("El còdigo es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
            }

if (frmTipo.txtDescripcion.Text.Trim() == string.Empty)
{
MessageBox.Show("La descripciòn es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
            }
         }
elseif (this.operacioN == "D")
         {
if (frmTipo.txtCodigo.Text.Trim() == string.Empty)
{
MessageBox.Show("El còdigo es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
            }
         }
returntrue;
      }

   }
```

```
}
```

Usuarios.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace RptAppServidor.Clases
{
classUsuario
    {
//atributos locales
publicint codigoUsuario = 0;
publicstring nombreUsuario = string.Empty;
publicstring operacioN;
publicstring like = string.Empty;

//formulario asociado
publicFrmUsuario FrmUsuario = null;

//constructor local
public Usuario(FrmUsuario p_FrmUsuario)
        {
this.FrmUsuario = p_FrmUsuario;
        }

//vaciar grilla
publicBoolean VaciarGrilla()
        {
DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
objAccess.EjecutarConsulta("select USU_ID AS CODIGO, USU_descripcion AS
            LOGIN,USU_PRIMER_NOMBRE AS PRIMER_NOMBRE, USU_SEGUNDO_NOMBRE AS
            SEGUNDO_NOMBRE,USU_APELLIDO_PATERNO AS
            APELLIDO_PATERNO,USU_APELLIDO_MATERNO AS
            APELLIDO_MATERNO,USU_PASSWORD AS CONTRASENA,USU_HABILITADO AS
            HABILITADO, FLO_ID AS FLOTA, PER_ID AS PERFIL from USUARIO WHERE USU_id = -
            1"); //traer datos
FrmUsuario.grdData.DataSource = objAccess.InfoTabla[0];
objAccess.Cerrar();
returntrue;
        }


//llenar grilla con usuarios
publicBoolean LlenarGrilla()
        {
DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
objAccess.EjecutarConsulta("select USU_ID AS CODIGO, USU_descripcion AS
            LOGIN,USU_PRIMER_NOMBRE AS PRIMER_NOMBRE, USU_SEGUNDO_NOMBRE AS
            SEGUNDO_NOMBRE,USU_APELLIDO_PATERNO AS
            APELLIDO_PATERNO,USU_APELLIDO_MATERNO AS
            APELLIDO_MATERNO,USU_PASSWORD AS CONTRASENA,USU_HABILITADO AS
            HABILITADO, FLO_ID AS FLOTA, PER_ID AS PERFIL from USUARIO WHERE
            USU_BORRADO = 0  ORDER BY USU_APELLIDO_PATERNO"); //traer datos
```

```
FrmUsuario.grdData.DataSource = objAccess.InfoTabla[0];


objAccess.Cerrar();
returntrue;
      }

//método para búsqueda por ingreso en text box superior
publicBoolean BuscarLike()
      {
if (this.FrmUsuario.txtBuscar.Text.Trim() != string.Empty)
        {
DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
objAccess.EjecutarConsulta("select USU_ID AS CODIGO, USU_descripcion AS
          LOGIN,USU_PRIMER_NOMBRE AS PRIMER_NOMBRE, USU_SEGUNDO_NOMBRE AS
          SEGUNDO_NOMBRE,USU_APELLIDO_PATERNO AS
          APELLIDO_PATERNO,USU_APELLIDO_MATERNO AS
          APELLIDO_MATERNO,USU_PASSWORD AS CONTRASENA,USU_HABILITADO AS
          HABILITADO, FLO_ID AS FLOTA, PER_ID AS PERFIL from USUARIO where
          usu_apellido_paterno like '%" + this.FrmUsuario.txtBuscar.Text + "%' AND USU_BORRADO
          = 0 ORDER BY USU_APELLIDO_PATERNO");//traer datos

FrmUsuario.grdData.DataSource = objAccess.InfoTabla[0];
            objAccess.Cerrar();
        }
else
this.LlenarGrilla();
returntrue;
}

//crear nuevo usuario
publicBoolean Insertar()
{
string habilitado = string.Empty;
if (this.FrmUsuario.cbvalidado.Checked)
habilitado = "S";
else
             habilitado = "N";
DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
objAccess.EjecutarComando("insert into usuario( USU_descripcion,USU_PRIMER_NOMBRE,
          USU_SEGUNDO_NOMBRE,USU_APELLIDO_PATERNO,USU_APELLIDO_MATERNO,US
          U_PASSWORD,USU_HABILITADO,FLO_ID, PER_ID, USU_BORRADO) values ('"
           + this.FrmUsuario.txtDescripcion.Text + "' ,'" + this.FrmUsuario.txtNombre.Text + "' ,'" +
          this.FrmUsuario.txtSegundoNombre.Text + "' ,'"
           + this.FrmUsuario.txtApellidoPaterno.Text + "' ,'" + this.FrmUsuario.txtApellidoMaterno.Text
          + "' ,'" +
this.FrmUsuario.txtPasword.Text + "','" + habilitado + "','"
           + this.FrmUsuario.cboFlota.SelectedValue + "'," +
          FrmUsuario.cboPerfil.SelectedValue.ToString() + ",0  ) ");

returntrue;

      }

//eliminación lógica de usuarios
publicBoolean Eliminar()
{
//DbAccess objAccess = new DbAccess(Util.DbCadenaConexionAccess);
//objAccess.EjecutarComando(" delete from Usuario where usu_id = " +
          this.FrmUsuario.txtCodigo.Text);
//objAccess.Cerrar();
```

```
//return true;
string activo = string.Empty;

if (!this.ValidarInformacion())
returnfalse;

if (this.FrmUsuario.cbvalidado.Checked)
            activo = "S";
else
            activo = "N";
DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
objAccess.EjecutarComando(" update usuario set "
                                + " usu_borrado = 1 "
                                + " where usu_id = " + this.FrmUsuario.txtCodigo.Text);


objAccess.Cerrar();
returntrue;
      }

//llenar lista desplegable de operador
publicBoolean BuscarOperador()
      {

DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
          objAccess.EjecutarConsulta("select ope_id, ope_descripcion from operador where
           ope_borrado = 0");

          FrmUsuario.cboOperador.DisplayMember = "ope_descripcion";
FrmUsuario.cboOperador.ValueMember = "ope_id";
FrmUsuario.cboOperador.DataSource = objAccess.InfoTabla[0];
          objAccess.Cerrar();

returntrue;
      }

//llenar lista desplegable de flota por operador
publicBoolean BuscarFlota(string idOperador)
      {

DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
          objAccess.EjecutarConsulta("select flo_id, flo_descripcion from flota where flo_borrado = 0
           and ope_id = " + idOperador);

          FrmUsuario.cboFlota.DisplayMember = "flo_descripcion";
          FrmUsuario.cboFlota.ValueMember = "flo_id";
          FrmUsuario.cboFlota.DataSource = objAccess.InfoTabla[0];
objAccess.Cerrar();

returntrue;
      }

//Llenar lista desplegable de perfil
publicBoolean BuscarPerfil()
      {

DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
          objAccess.EjecutarConsulta("select per_id, per_descripcion from perfil");
          FrmUsuario.cboPerfil.DisplayMember = "per_descripcion";
          FrmUsuario.cboPerfil.ValueMember = "per_id";
FrmUsuario.cboPerfil.DataSource = objAccess.InfoTabla[0];
```

```
            objAccess.Cerrar();

returntrue;
        }

//actualización de usuario
publicBoolean Actualizar()
{
string activo = string.Empty;

if (this.FrmUsuario.cbvalidado.Checked)
                activo = "S";
else
                activo = "N";
DbAccess objAccess = newDbAccess(Util.DbCadenaConexionAccess);
objAccess.EjecutarComando(" update usuario set "
                                + " usu_descripcion = '" + this.FrmUsuario.txtDescripcion.Text + "' ,"
+ " USU_PRIMER_NOMBRE  = '" + this.FrmUsuario.txtNombre.Text + "' ,"
                                + " USU_SEGUNDO_NOMBRE = '" +
            this.FrmUsuario.txtSegundoNombre.Text + "' ,"
                                + " USU_APELLIDO_PATERNO = '" +
            this.FrmUsuario.txtApellidoPaterno.Text + "' ,"
                                + " USU_APELLIDO_MATERNO = '" +
            this.FrmUsuario.txtApellidoMaterno.Text + "' ,"
+ " USU_PASSWORD = '"  +  this.FrmUsuario.txtPasword.Text + "',"
+ " USU_HABILITADO = '" + activo +"',"
+ " FLO_ID  = " + this.FrmUsuario.cboFlota.SelectedValue.ToString() + ","
                                + " PER_ID = " + this.FrmUsuario.cboPerfil.SelectedValue.ToString()
                                + " where usu_id = " + this.FrmUsuario.txtCodigo.Text );


objAccess.Cerrar();
returntrue;
        }

//limpiar formulario asociado
publicBoolean limpiar()
        {
          FrmUsuario.txtCodigo.Text = string.Empty ;
          FrmUsuario.txtDescripcion.Text = string.Empty;
FrmUsuario.txtBuscar.Text = string.Empty;
          FrmUsuario.txtApellidoMaterno.Text = string.Empty;
          FrmUsuario.txtApellidoPaterno.Text = string.Empty;
          FrmUsuario.txtPasword.Text = string.Empty;
          FrmUsuario.txtNombre.Text = string.Empty;
FrmUsuario.txtSegundoNombre.Text = string.Empty;

//this.LlenarGrilla();

returntrue;
        }


//validación de información a ingresar de acuero a operación escogida
publicBoolean ValidarInformacion()
        {
if (this.operacioN == "I")
           {

if (FrmUsuario.txtDescripcion.Text.Trim() == string.Empty)
{
```

```
MessageBox.Show("La descripciòn es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
            }

if (FrmUsuario.txtApellidoMaterno.Text.Trim() == string.Empty)
{
MessageBox.Show("El apellido materno es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
            }

if (FrmUsuario.txtApellidoPaterno.Text.Trim() == string.Empty)
{
MessageBox.Show("El apellido paterno es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
            }

if (FrmUsuario.txtNombre.Text.Trim() == string.Empty)
{
MessageBox.Show("El nombre es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
            }
if (FrmUsuario.txtPasword.Text.Trim() == string.Empty)
{
MessageBox.Show("La clave es mandatoria", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
            }
Consultas objConsultas = newConsultas();
string sqlFlagExisteUsuarios = objConsultas.ConsultaGenerica("select count(1) from usuario where
          usu_descripcion = '" + this.FrmUsuario.txtDescripcion.Text + "'");

if (int.Parse(sqlFlagExisteUsuarios) > 0)
            {
MessageBox.Show("Nombre de Usuario BP ya existente", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
            }
if (FrmUsuario.cboOperador.SelectedIndex < 0)
{
MessageBox.Show("El Operador es mandatoria", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
            }
if (FrmUsuario.cboFlota.SelectedIndex < 0)
{
MessageBox.Show("La Flota es mandatoria", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
            }
        }
elseif (this.operacioN == "U")
        {
if (FrmUsuario.txtCodigo.Text.Trim() == string.Empty)
{
MessageBox.Show("El còdigo es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
            }

if (FrmUsuario.txtDescripcion.Text.Trim() == string.Empty)
{
MessageBox.Show("La descripciòn es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
            }

if (FrmUsuario.txtApellidoMaterno.Text.Trim() == string.Empty)
```

```
{
MessageBox.Show("El apellido materno es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
            }

if (FrmUsuario.txtApellidoPaterno.Text.Trim() == string.Empty)
{
MessageBox.Show("El apellido paterno es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
            }

if (FrmUsuario.txtNombre.Text.Trim() == string.Empty)
{
MessageBox.Show("El nombre es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
            }
if (FrmUsuario.txtPasword.Text.Trim() == string.Empty)
{
MessageBox.Show("La clave es mandatoria", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
            }
if (FrmUsuario.cboOperador.SelectedIndex < 0)
{
MessageBox.Show("El Operador es mandatoria", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
            }
if (FrmUsuario.cboFlota.SelectedIndex < 0)
{
MessageBox.Show("La Flota es mandatoria", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
            }

          }
elseif (this.operacioN == "D")
          {
if (FrmUsuario.txtCodigo.Text.Trim() == string.Empty)
{
MessageBox.Show("El còdigo es mandatorio", "AdverTencia", MessageBoxButtons.OK);
returnfalse;
            }

          }
returntrue;
      }

   }
}
```

ANEXOS IV

BASE DE DATOS MICROSOFT ACCESS

Implementación:

Se debe realizar la implementación de la base de datos código fuente en Microsoft C# para su correcto funcionamiento A continuación se presenta el código fuente de los "froms" y posterior de las clases.

TABLA ARBOL

ARBOL

ARB_	ARB_ARB_ID	TIP_I	ARB_DESCRIPCION	ARB_PATH_DOCUMENTO	ARB_REVISION	MAT_	ARB_REMARCAR	ARB_BORRADO
1	0	1	MANUAL DE AVION A-318	C:\SisDoPer\Airbus\Aerolane.pdf	Aerolane S.A	1	N	Falso
2	1	1	Flight Crew Operation Manual	C:\SisDoPer\Airbus\Aerolane.pdf	FCOM	1	N	Falso
3	2	1	Flight Crew Operation Manual A-318	C:\SisDoPer\Airbus\Manuales del Avión-A318\FCOM\Flight Crew Operations Manual A318.pdf	Mayo 11, 2012	1	N	Falso
4	2	1	QRH	C:\SisDoPer\Airbus\Manuales del Avión-A318\QRH CC\CC-CZN.pdf	Mayo 11, 2012	1	N	Falso
5	1	1	Flight Manual	C:\SisDoPer\Airbus\Manuales del Avión-A318\AFM\Flight Manual.pdf	24 JUL 12	1	N	Falso

ARBOL								
ARB_	ARB_ARB_ID	TIP_I	ARB_DESCRIPCION	ARB_PATH_DOCUMENTO	ARB_REVISION	MAT_	ARB_REMARCAR	ARB_BORRADO
6	0	2	MANUAL DE COMPAÑÍA	C:\SisDoPer\Airbus\Aerolane.pdf	Aerolane S.A.	1	N	Falso
7	6	2	Manual de Operaciones	C:\SisDoPer\Airbus\Documentacion Compartida A320Fam\Manual de Operaciones.pdf	Rev. 38	1	N	Falso
8	6	2	OPSPECS	C:\SisDoPer\Airbus\Aerolane.pdf	OPSPECS	1	N	Falso
9	8	2	ECUADOR	C:\SisDoPer\Airbus\Aerolane.pdf	Edc. Noviembre	1	N	Falso
10	9	2	Especificaciones Operacionales Español	C:\SisDoPer\Airbus\Documentacion Compartida A320Fam\OPSPECS\ECUADOR\Especificaciones Operacionales Español.pdf	Edc. Noviembre	1	N	Falso
11	9	2	Especificaciones Operacionales Ingles	C:\SisDoPer\Airbus\Documentacion Compartida A320Fam\OPSPECS\ECUADOR\Especificaciones Operacionales Inglés.pdf	Edc. Noviembre	1	N	Falso
12	8	2	CHILE	C:\SisDoPer\Airbus\Aerolane.pdf	Rev. 11/12	1	N	Falso
13	12	2	Especificaciones Operativas	C:\SisDoPer\Airbus\Documentacion Compartida A320Fam\OPSPECS\CHILE\Esp	Rev. 11/12	1	N	Falso

ARBOL								
ARB_	ARB_ARB_ID	TIP_I	ARB_DESCRIPCION	ARB_PATH_DOCUMENTO	ARB_REVISION	MAT_	ARB_REMARCAR	ARB_BORRADO
				ecificaciones Operativas.pdf				
14	6	2	Manual de Tripulantes de Cabina	C:\SisDoPer\Airbus\Documentacion Compartida A320Fam\Manual de Tripulantes de Cabina.pdf	Rev. 29	1	N	Falso
15	6	2	MEL	C:\SisDoPer\Airbus\Manuales del Avión-A318\MEL CC\MEL.pdf	Rev. 2	1	N	Falso
16	6	2	Takeoff & Landing Analysis	C:\SisDoPer\Airbus\Manuales del Avión-A318\TOA CC\TOA.pdf	Rev. 6	1	N	Falso
17	6	2	Orientacion sobre Respuesta de Emergencia Doc.9481 OACI	C:\SisDoPer\Airbus\Documentacion Compartida A320Fam\Orientacion sobre Respuesta de Emergencia Doc.9481 OACI.pdf	Edicion: 2012	1	N	Falso
18	6	2	Manual de Comunicacion y Servicio - Tripulantes de Mando	C:\SisDoPer\Airbus\Documentacion Compartida A320Fam\Manual de Comunicación y Servicio a Pasajeros - Tripulantes de Mando.pdf	Rev. 2	1	N	Falso
19	6	2	Abastecimiento de	C:\SisDoPer\Airbus\Documentacion Compartida	Presentación	1	N	Falso

ARBOL								
ARB_	ARB_ARB_ID	TIP_I	ARB_DESCRIPCION	ARB_PATH_DOCUMENTO	ARB_REVISION	MAT_	ARB_REMARCAR	ARB_BORRADO
			Combustible	A320Fam\Abastecimiento de Combustible.pdf				
20	6	2	Listas de Chequeo	C:\SisDoPer\Airbus\Aerolane.pdf	Listas de Chequeo	1	N	Falso
21	20	2	Normal Checklist	C:\SisDoPer\Airbus\Manuales del Avión-A318\Listas de Chequeo\Normal Checklist.pdf		1	N	Falso
22	20	1	RVSM	C:\SisDoPer\Airbus\Manuales del Avión-A318\Listas de Chequeo\RVSM.pdf		1	N	Falso
23	20	2	Categoria	C:\SisDoPer\Airbus\Manuales del Avión-A318\Listas de Chequeo\Categoria.pdf		1	N	Falso
24	20	2	One Engine Taxi	C:\SisDoPer\Airbus\Manuales del Avión-A318\Listas de Chequeo\One Engine Taxi.pdf		1	N	Falso
25	20	2	Mercancias Peligrosas	C:\SisDoPer\Airbus\Manuales del Avión-A318\Listas de Chequeo\Mercancías Peligrosas.pdf		1	N	Falso
26	20	2	Start Fault	C:\SisDoPer\Airbus\Manuales del Avión-A318\Listas de Chequeo\Start Fault.pdf		1	N	Falso
27	0	2	Circulares Operativa	C:\SisDoPer\Airbus\Aerolane.pdf	Circulares Opera	1	N	Falso

ARBOL								
ARB_	ARB_ARB_ID	TIP_I	ARB_DESCRIPCION	ARB_PATH_DOCUMENTO	ARB_REVISION	MAT_	ARB_REMARCAR	ARB_BORRADO
			s		tivas			
28	27	2	Listado de Circulares Vigentes 2012	C:\SisDoPer\Airbus\Documentacion Compartida A320Fam\Circulares Operativas\2012\Listado de Circulares Operativas Vigentes.pdf	2012	1	N	Falso
29	28	2	C. O. N° 019A-12	C:\SisDoPer\Airbus\Documentacion Compartida A320Fam\Circulares Operativas\2012\C. O. N° 019A-12.pdf		1	N	Falso
30	28	2	C. O. N° 023-12	C:\SisDoPer\Airbus\Documentacion Compartida A320Fam\Circulares Operativas\2012\C. O. N° 023-12.pdf		1	N	Falso
31	28	2	C. O. N° 027-12	C:\SisDoPer\Airbus\Documentacion Compartida A320Fam\Circulares Operativas\2012\C. O. N° 027-12.pdf		1	N	Falso
32	28	2	C. O. N° 028-12	C:\SisDoPer\Airbus\Documentacion Compartida A320Fam\Circulares Operativas\2012\C. O. N° 028-12.pdf		1	N	Falso
33	28	2	C. O. N° 029-12	C:\SisDoPer\Airbus\Documentacion Compartida A320Fam\Circulares Operativas\2012\C. O. N° 029-12.pdf		1	N	Falso
34	28	2	C. O. N° 043-12	C:\SisDoPer\Airbus\Documentacion		1	N	Falso

ARBOL								
ARB_	ARB_ARB_ID	TIP_I	ARB_DESCRIPCION	ARB_PATH_DOCUMENTO	ARB_REVISION	MAT_	ARB_REMARCAR	ARB_BORRADO
				Compartida A320Fam\Circulares Operativas\2012\C. O. N° 043-12.pdf				
35	28	2	C. O. N° 044-12	C:\SisDoPer\Airbus\Documentacion Compartida A320Fam\Circulares Operativas\2012\C. O. N° 044-12.pdf		1	N	Falso
36	28	2	C. O. N° 046A-12	C:\SisDoPer\Airbus\Documentacion Compartida A320Fam\Circulares Operativas\2012\C. O. N° 046A-12.pdf		1	N	Falso
37	28	2	C. O. N° 047-12	C:\SisDoPer\Airbus\Documentacion Compartida A320Fam\Circulares Operativas\2012\C. O. N° 047-12.pdf		1	N	Falso
38	28	2	C. O. N° 050-12	C:\SisDoPer\Airbus\Documentacion Compartida A320Fam\Circulares Operativas\2012\C. O. N° 050-12.pdf		1	N	Falso
39	28	2	C. O. N° 052-12	C:\SisDoPer\Airbus\Documentacion Compartida A320Fam\Circulares Operativas\2012\C. O. N° 052-12.pdf		1	N	Falso
40	28	2	C. O. N° 057-12	C:\SisDoPer\Airbus\Documentacion Compartida A320Fam\Circulares Operativas\2012\C. O. N° 057-12.pdf		1	N	Falso

ARBOL								
ARB_	ARB_ARB_ID	TIP_I	ARB_DESCRIPCION	ARB_PATH_DOCUMENTO	ARB_REVISION	MAT_	ARB_REMARCAR	ARB_BORRADO
41	28	2	C. O. N° 058-12	C:\SisDoPer\Airbus\Documentacion Compartida A320Fam\Circulares Operativas\2012\C. O. N° 058-12.pdf		1	N	Falso
42	28	2	C. O. N° 059-12	C:\SisDoPer\Airbus\Documentacion Compartida A320Fam\Circulares Operativas\2012\C. O. N° 059-12.pdf		1	N	Falso
43	28	2	C. O. N° 060-12	C:\SisDoPer\Airbus\Documentacion Compartida A320Fam\Circulares Operativas\2012\C. O. N° 060-12.pdf		1	N	Falso
44	28	2	C. O. N° 061-12	C:\SisDoPer\Airbus\Documentacion Compartida A320Fam\Circulares Operativas\2012\C. O. N° 061-12.pdf		1	N	Falso
45	28	2	C. O. N° 065-12	C:\SisDoPer\Airbus\Documentacion Compartida A320Fam\Circulares Operativas\2012\C. O. N° 065-12.pdf		1	N	Falso
46	28	2	C. O. N° 068-12	C:\SisDoPer\Airbus\Documentacion Compartida A320Fam\Circulares Operativas\2012\C. O. N° 068-12.pdf		1	N	Falso
47	28	2	C. O. N° 070-12	C:\SisDoPer\Airbus\Documentacion Compartida A320Fam\Circulares Operativas\2012\C. O. N° 070-		1	N	Falso

ARBOL								
ARB_	ARB_ARB_ID	TIP_I	ARB_DESCRIPCION	ARB_PATH_DOCUMENTO	ARB_REVISION	MAT_	ARB_REMARCAR	ARB_BORRADO
				12.pdf				
48	28	2	C. O. N° 071-12	C:\SisDoPer\Airbus\Documentacion Compartida A320Fam\Circulares Operativas\2012\C. O. N° 071-12.pdf		1	N	Falso
49	28	2	C. O. N° 074-12	C:\SisDoPer\Airbus\Documentacion Compartida A320Fam\Circulares Operativas\2012\C. O. N° 074-12.pdf		1	N	Falso
50	28	2	C. O. N° 076-12	C:\SisDoPer\Airbus\Documentacion Compartida A320Fam\Circulares Operativas\2012\C. O. N° 076-12.pdf		1	N	Falso
51	28	2	C. O. N° 078-12	C:\SisDoPer\Airbus\Documentacion Compartida A320Fam\Circulares Operativas\2012\C. O. N° 078-12.pdf		1	N	Falso
52	28	2	C. O. N° 079-13	C:\SisDoPer\Airbus\Documentacion Compartida A320Fam\Circulares Operativas\2012\C. O. N° 079-12.pdf		1	S	Falso

TABLA BITACORA

BITACORA		
BIT_ID	BIT_FECHA	USU_ID
1	20/11/2012 23:03:43	2
2	20/11/2012 23:08:20	2
3	20/11/2012 23:09:07	2
4	20/11/2012 23:09:57	2
5	21/11/2012 18:28:07	1
6	21/11/2012 18:42:14	2
7	21/11/2012 18:49:02	2
8	01/12/2012 21:44:35	1
9	01/12/2012 21:54:02	1
10	01/12/2012 21:54:51	1
11	01/12/2012 21:56:57	1
12	01/12/2012 22:04:07	1
13	01/12/2012 22:04:55	2
14	01/12/2012 22:16:34	1
15	01/12/2012 22:17:36	1
16	01/12/2012 22:21:50	2
17	01/12/2012 22:56:31	1
18	06/12/2012 21:27:50	1

TABLA FLOTA

FLOTA			
FLO_ID	FLO_DESCRIPCION	OPE_ID	FLO_BORRADO
1	AIRBUS	1	Falso
2	BOEING	1	Falso

TABLA MATRICULA

MATRICULA			
MAT_ID	MAT_DESCRIPCION	MOD_ID	MAT_BORRADO
1	CC-CZN	1	Falso
2	CC-CZQ	1	Falso
3	CC-CZS	1	Falso
4	CC-COU	2	Falso
5	CC-COX	2	Falso
6	CC-COZ	2	Falso
7	CC-CPE	2	Falso
8	CC-CPM	2	Falso
9	CC-CPO	2	Falso
10	CC-CYE	2	Falso
11	CC-CYF	2	Falso
12	CC-CYI	2	Falso
13	CC-CYL	2	Falso
14	CC-CYJ	2	Falso
15	CC-CPF	2	Falso

MATRICULA			
MAT_ID	MAT_DESCRIPCION	MOD_ID	MAT_BORRADO
16	CC-CPI	2	Falso
17	CC-BCA	2	Falso
18	CC-BCB	2	Falso
19	CC-BCC	2	Falso
20	CC-BCD	2	Falso
21	CC-BCE	2	Falso
22	CC-BCF	2	Falso
23	HC-CLA	3	Falso
24	HC-CLB	3	Falso
25	HC-CLC	3	Falso
26	HC-CLD	3	Falso
27	HC-CLE	3	Falso
28	CC-BAF	3	Falso
29	CC-BAG	3	Falso
30	CC-COF	3	Falso
31	CC-COM	3	Falso
32	CC-BAH	3	Falso
33	CC-BAJ	3	Falso
34	CC-BAK	3	Falso
35	CC-BAL	3	Falso
36	CC-BAM	3	Falso
37	CC-BAS	3	Falso
38	CC-BAT	3	Falso
39	CC-BAU	3	Falso
40	CC-BAV	3	Falso

MATRICULA			
MAT_ID	MAT_DESCRIPCION	MOD_ID	MAT_BORRADO
41	CC-BAZ	3	Falso
42	CC-BFB	3	Falso
43	CC-CDM	4	Falso
44	CC-CRH	4	Falso
45	CC-CDP	4	Falso
46	CC-CWF	4	Falso
47	CC-CWV	4	Falso
48	CC-CWY	4	Falso
49	CC-CXC	4	Falso
50	CC-CRV	4	Falso
51	CC-CZW	4	Falso
52	CC-CZU	4	Falso
53	CC-CZT	4	Falso
54	CC-CXG	4	Falso
55	CC-CXI	4	Falso
56	CC-CXJ	4	Falso
57	CC-CXL	4	Falso
58	CC-CXK	4	Falso
59	CC-BJA	4	Falso
60	CC-CML	4	Falso
61	CC-CRG	4	Falso
62	CC-CBJ	4	Falso
63	CC-BDB	4	Falso
64	CC-BDC	4	Falso
65	CC-CXF	4	Falso

MATRICULA			
MAT_ID	MAT_DESCRIPCION	MOD_ID	MAT_BORRADO
66	CC-CXH	4	Falso
67	CC-CXE	4	Falso
68	CC-CXD	4	Falso
69	CC-BDA	4	Falso
70	CC-BDD	4	Falso
71	CC-CWN	4	Falso
72	CC-BDE	4	Falso
73	CC-BDF	4	Falso
74	CC-BDG	4	Falso

TABLA MODELO

MODELO			
MOD_ID	FLO_ID	MOD_DESCRIPCION	MOD_BORRADO
1	1	FLOTA A-318	Falso
2	1	FLOTA A-319	Falso
3	1	FLOTA A-320	Falso
4	2	FLOTA B-767	Falso

TABLA OPCIONES MODELO

OPCIONES_MODELO					
OPC_ID	OPC_ETIQUETA	OPC_PROGRAMA	FLO_ID	OPC_ORDEN	OPC_BORRADO
1	FOVE	C:\SisDoPer\Acceso directo\Fove\FOVE.lnk	1	1	Falso
2	MMPP 2012	C:\SisDoPer\Acceso directo\MMPP\DGRES.lnk	1	2	Falso
3	LPC BROWSER	C:\Windows\NOTEPAD.EXE	1	3	Falso
4	OPT 5.6	C:\SisDoPer\Acceso directo\Opt\OPT.lnk	2	1	Falso
5	MM PP 2012	C:\SisDoPer\Acceso directo\MMPP\DGRES.lnk	2	2	Falso
6	FlyBy E6B	C:\SisDoPer\Acceso directo\FlyBy E6B\FlyBy E6B.lnk	2	3	Falso

TABLA OPERADOR

OPERADOR		
OPE_ID	OPE_DESCRIPCION	OPE_BORRADO
1	ECUADOR	Falso

TABLA PARAMETRO

PARAMETRO			
PAR_ID	PAR_DESCRIPCION	PAR_VALOR	PAR_BORRADO
BOR	asdfasdf	asdfsad	Verdadero

PARAMETRO			
PAR_ID	**PAR_DESCRIPCION**	**PAR_VALOR**	**PAR_BORRADO**
IMG	IMAGEN INICIAL PDF	C:\SisDoPer\Aerolane.pdf	Falso
PAG	PAGINA INICIAL	C:\SisDoPer\Aerolane.pdf	Falso
PIA	PATH INSTRUCTIVO PROGRAMAS AEROLANE	C:\SisDoPer\Aerolane.pdf	Falso
PIM	PATH INSTRUCTIVO MANUALES	C:\SisDoPer\Aerolane.pdf	Falso
PIP	PATH INSTRUCTIVO PROGRAMAS	C:\SisDoPer\Aerolane.pdf	Falso
REV	REVISION EN PANTALLA INICIAL	ULTIMA REVISION 01/10/2012	Falso

TABLA PERFIL

PERFIL			
per_id	**per_descripcion**	**per_mantenimiento**	**per_borrado**
1	ADMINISTRADOR	SI	Falso
2	PILOTO	NO	Falso
3	USUARIO	NO	Falso

TABLA TIPO

TIPO		
TIP_ID	**TIP_DESCRIPCION**	**TIP_BORRADO**
1	MANUAL DE AVION	Falso
2	MANUAL DE COMPAÑÍA	Falso

TABLA USUARIO

USUARIO										
FLO	USU_DESCRIPCION	USU_PASSWORD	USU_APELLIDO_PATERNO	USU_APELLIDO_MATERNO	USU_PRIMER_NOMBRE	USU_SEGUNDO_NOMBRE	USU_HABILITADO	PER	USU	USU_BORRADO
1	admin	ecuador	JURADO	CARRERA	DANIEL	ALEJANDRO	S	1	1	Falso
2	user	ecuador	GALARZA	ROSERO	JAIME	DANILO	S	1	2	Falso
1	2644965	operaciones	ALCOCER	TORRES	JORGE	SANTIAGO	S	2	3	Falso
1	2412627	operaciones	BASTIDAS	GUERRA	DIEGO	MAURICIO	S	2	4	Falso
1	2412657	operaciones	CANO	FARAH	NESSAR	NAHIN	S	2	5	Falso
1	2399614	operaciones	CEVALLOS	MANZANO	DIEGO	FABRICIO	S	2	6	Falso
1	2279840	operaciones	CHARVET	SAA	ESTEBAN	ALFONSO	S	2	7	Falso
1	1375363	operaciones	CHIRIBOGA	CAMPUZANO	ALEJANDRO	JOSE	S	2	8	Falso
1	3317565	operaciones	CHIRIBOGA	NAJERA	ISAAC	EVERARDO	S	2	9	Falso
1	29418	operaciones	ESPINOZA	ROMERO	SERGIO	ANTONIO	S	2	10	Falso
1	2604125	operaciones	FIALLO	SORIA	PABLO	ERMEL	S	2	11	Falso
1	31726	operaciones	GALARRAGA	CALDERON	EDISON	IVAN	S	2	12	Falso

USUARIO										
FLO	USU_DESCRIPCION	USU_PASSWORD	USU_APELLIDO_PATERNO	USU_APELLIDO_MATERNO	USU_PRIMER_NOMBRE	USU_SEGUNDO_NOMBRE	USU_HABILITADO	PER	USU	USU_BORRADO
1	2525533	operaciones	GALLEGOS	MENDOZA	JOHNNY	ALEXIS	S	2	13	Falso
1	2685840	operaciones	GUTIERREZ	RUIZ	CRISTOBAL	ALEJANDRO	S	2	14	Falso
1	2588423	operaciones	HELOU	CEVALLOS	JOSE	MARIA	S	2	15	Falso
1	93792	operaciones	MONTALVO	VELA	RUBEN	SANTIAGO	S	2	16	Falso
1	2645058	operaciones	OBANDO	SEVILLA	JESUS	ALEXANDER	S	2	17	Falso
1	1375369	operaciones	ORELLANA	VILLACIS	JORGE	ANTONIO	S	2	18	Falso
1	2583391	operaciones	PALACIOS	ORDOÑEZ	MARCO	ADRIAN	S	2	19	Falso
1	2687966	operaciones	PUGA	ESTÉVEZ	CRISTOBAL	MAURICIO	S	2	20	Falso