DESARROLLO DE SOFTWARE PARA LA GESTIÓN Y CONTROL EFICIENTE DE LA CARTERA DE CRÉDITOS EDUCATIVOS, DE LA UNIVERSIDAD LIBRE DE PEREIRA

Autores:

MAURICIO RAMIREZ GIRALDO

DIEGO ALEJANDRO MADRID ROMAN

UNIVERSIDAD LIBRE

FACULTAD DE INGENIERÍA

PROGRAMA INGENIERIA DE SISTEMAS

PEREIRA

2013

DESARROLLO DE SOFTWARE PARA LA GESTIÓN Y CONTROL EFICIENTE DE LA CARTERA DE CRÉDITOS EDUCATIVOS, DE LA UNIVERSIDAD LIBRE DE PEREIRA

Autores:

MAURICIO RAMIREZ GIRALDO

DIEGO ALEJANDRO MADRID ROMAN

Presentación de proyecto de grado, para optar al título de Ingeniería de Sistemas

Asesor:

Ing. Carlos Alberto Ocampo S.

UNIVERSIDAD LIBRE

FACULTAD DE INGENIERÍAS

PROGRAMA INGENIERIA DE SISTEMAS

PEREIRA

2013

CONTENIDO

1 TITULO 11

2 ANTECEDENTES 12

3 DESCRIPCIÓN DEL PROBLEMA 14

4 JUSTIFICACION 15

5 OBJETIVOS 16

5.1 OBJETIVO GENERAL 16
5.2 OBJETIVOS ESPECIFICOS 16

6 HIPÓTESIS 17

6.1 DETERMINACIÓN DE VARIABLES 17
6.1.1 Variable Independiente. 17
6.1.2 Variable Dependiente 17

7 DELIMITACIÓN Y ALCANCE DEL PROYECTO 18

7.1 DELIMITACIÓN ESPACIAL 18
7.2 DELIMITACIÓN CRONOLÓGICA 18
7.3 DELIMITACIÓN CONCEPTUAL 18

8 MARCO REFERENCIAL 20

8.1 MARCO TEORICO 20
8.1.1 Programación Extrema O Extreme Programming (Xp) 20
8.1.2 UML (Lenguaje Unificado de Modelado) 24
8.1.3 StarUML 24
8.1.4 Herramientas RAD 24
8.1.5 Oracle Database 11g Express Edition (Oracle Database XE) 25
8.1.6 Oracle SQL Data Modeler 25
8.1.7 Oracle SQL Developer 25
8.1.8 Herramienta CASE 26
8.2 MARCO CONCEPTUAL 26
8.2.1 Software 26
8.2.2 Ingeniería del software 27
8.2.3 Lenguajes de programación 28
8.2.4 Base de datos 28
8.2.5 Programación Orientada a Objetos (POO) 28
8.2.6 RAD (Desarrollo Rápido de Aplicaciones): 30
8.2.7 UML 30
8.2.8 StarUML 30
8.2.9 Herramientas CASE 30
8.2.10 ERP 31
8.2.11 .NET 31
8.2.12 Visual Basic (VB.NET) 31

8.2.13 Oracle Database....31
8.2.14 DBMS....31
8.2.15 RDBMS....31

9 MARCO METODOLÓGICO....32

9.1 TIPO DE INVESTIGACIÓN....32
9.2 MÉTODO DE INVESTIGACIÓN....32
9.2.1 Planificación....32
9.2.2 Diseño....35
9.2.3 Codificación....36
9.2.4 Pruebas....38
9.3 MARCO LEGAL Y NORMATIVO....39

10 LEVANTAMIENTO DE REQUERIMIENTOS....40

10.1 REQUERIMIENTOS FUNCIONALES....40
10.2 REQUERIMIENTOS NO-FUNCIONALES....42

11 ANALISIS DE REQUERIMIENTOS....46

11.1 DIAGRAMAS DE CASOS DE USO....46
11.1.1 Generalización de los actores....46
11.1.2 Diagrama Contextual....47
11.2 ANALISIS DE CASOS DE USO....48
11.2.1 Caso de uso Ingresar al Sistema....48
11.2.2 Caso de uso Liquidar Cartera....52
11.2.3 Caso de uso liquidar intereses....56
11.2.4 Caso de uso Administrar Codeudores....60
11.2.5 Caso de uso administrar usuario....65

12 DISEÑO DEL PROTOTIPO DE LA APLICACIÓN....68

12.1 MODELO LÓGICO....68
12.2 MODELO RELACIONAL E/R....69
12.3 MODELO DE VISTAS....70
12.4 MODELO FÍSICO (DICCIONARIO DE DATOS)....70
12.4.1 Tabla Terceros....71
12.4.2 Tabla Estudiantes....72
12.4.3 Tabla Codeudores....73
12.4.5 Tabla Tipos de Crédito....74
12.4.6 Tabla Recibos de Caja....75
12.4.7 Tabla Amortizaciones....75
12.4.8 Tabla Facultades....77
12.4.9 Tabla Programas....77
12.4.10 Tabla Usuarios....78
12.4.11 Tabla Roles....79
12.4.12 Tabla Parámetros....79
12.5 DIAGRAMA DE CLASES....80

12.6 DISEÑO ARQUITECTURA DEL SISTEMA....81

13 IMPLEMENTACIÓN DEL PROTOTIPO....82

13.1 PANTALLA INICIAL DE ACCESO....82
13.1.1 Código fuente o lógica del negocio....83
13.2 PANTALLA CONTENEDORA O PRINCIPAL....83
13.2.1 Código fuente o lógica del negocio....84
13.3 PANTALLA LIQUIDAR CARTERA....85
13.3.1 Código fuente o lógica del negocio....86
13.4 PANTALLA LIQUIDAR INTERESES....90
13.4.1 Código fuente o lógica del negocio....91
13.5 PANTALLA ADMINISTRAR CODEUDORES....94
13.5.1 Código Fuente o lógica del negocio....94
13.6 PANTALLA ADMINISTRAR USUARIOS....100
13.6.1 Código fuente o lógica del negocio....101
13.7 PANTALLA ADMINISTRAR PARÁMETROS....103
13.7.1 Código fuente o lógica del negocio....103
13.8 OTRAS CLASES....104
13.8.1 Clase de Acceso a Datos....104
13.8.2 Clase varios métodos....105

14 PRUEBAS FINALES DE ACEPTACIÓN....107

14.1 PLAN DE PRUEBAS....107
14.2 METODOLOGÍA DE LAS PRUEBAS....107
14.3 ALCANCE DE LAS PRUEBAS....108
14.4 EVALUACIÓN DE LOS CASOS DE PRUEBA....109
14.4.1 Caso de prueba Ingresar al sistema....109
14.4.2 Caso de prueba liquidar cartera....110
14.4.3 Caso de prueba liquidar intereses....111
14.4.4 Caso de prueba administrar codeudores....113

15 MANUAL DEL USUARIO....117

15.1 INTRODUCCIÓN....117
15.1.1 Propósito y descripción de la aplicación....117
15.1.2 Propósito del documento....117
15.1.3 Esquema de funcionalidad....118
15.2 CONCEPTOS IMPORTANTES....119
15.2.1 ¿Cómo abrir la aplicación?....119
15.2.2 ¿Cómo Acceder al Sistema?....120
15.3 DESCRIPCION DE LA APLICACIÓN....121
15.3.1 Pantalla Inicial o Principal....121
15.3.2 Descripción de componentes....122
15.3.3 Barra de Menús....123
15.3.4 Barra de Herramientas....125
15.4 GUIA DE USO Y FUNCIONAMIENTO....127

15.4.1 Liquidar Cartera.....127
15.4.2 Liquidar Intereses.....130
15.4.3 Administración de Codeudores.....133
15.4.4 Administrar Usuarios del Sistema.....139
15.4.5 Administrar Parámetros.....144

16 CONCLUSIONES.....146

ANEXOS.....148

BIBLIOGRAFIA.....149

LISTADO DE ILUSTRACIONES

ILUSTRACIÓN 1. DIAGRAMA DE GENERALIZACIÓN DE LOS ACTORES. 46
ILUSTRACIÓN 2. DIAGRAMA DE CASOS DE USO CONTEXTUAL.....47
ILUSTRACIÓN 3. DIAGRAMA DE SECUENCIA INGRESAR AL SISTEMA 50
ILUSTRACIÓN 4. DIAGRAMA DE ACTIVIDADES INGRESAR AL SISTEMA. 51
ILUSTRACIÓN 5. DIAGRAMA DE SECUENCIA LIQUIDAR CARTERA. 54
ILUSTRACIÓN 6. DIAGRAMA DE ACTIVIDADES LIQUIDAR CARTERA.....55
ILUSTRACIÓN 7. DIAGRAMA DE SECUENCIA LIQUIDAR INTERESES. 58
ILUSTRACIÓN 8. DIAGRAMA DE ACTIVIDADES LIQUIDAR INTERESES. 59
ILUSTRACIÓN 9. DIAGRAMA DE SECUENCIA ADMINISTRAR CODEUDORES. 63
ILUSTRACIÓN 10. DIAGRAMA DE ACTIVIDADES ADMINISTRAR CODEUDORES.....64
ILUSTRACIÓN 11. DIAGRAMA DE SECUENCIA ADMINISTRAR USUARIOS.....67
ILUSTRACIÓN 12. MODELO LÓGICO DE LA BASE DE DATOS.....68
ILUSTRACIÓN 13. MODELO RELACIONAL E/R.....69
ILUSTRACIÓN 14. MODELO DE VISTAS DE SISTEMA.....70
ILUSTRACIÓN 15. DIAGRAMAS DE CLASES DEL SISTEMA. FUENTE: EL AUTOR. 80
ILUSTRACIÓN 16. DISEÑO ARQUITECTÓNICO DEL SISTEMA.....81
ILUSTRACIÓN 17. PANTALLA INICIAL DE LA APLICACIÓN.....82
ILUSTRACIÓN 18. PANTALLA O FORMULARIO PRINCIPAL, CONTENEDORA. 84
ILUSTRACIÓN 19. PANTALLA O FORMULARIO LIQUIDAR CARTERA. 86
ILUSTRACIÓN 20. PANTALLA O FORMULARIO LIQUIDAR INTERESES. 90
ILUSTRACIÓN 21. PANTALLA O FORMULARIO ADMINISTRAR CODEUDORES. 94
ILUSTRACIÓN 22. PANTALLA O FORMULARIO DE ADMINISTRACIÓN DE USUARIOS. 101
ILUSTRACIÓN 23. PANTALLA O FORMULARIO DE ADMINISTRACIÓN DE PARÁMETROS. 103
ILUSTRACIÓN 24. ESQUEMA DE FUNCIONALIDAD DEL SISTEMA. 118
ILUSTRACIÓN 25: UBICACIÓN DEL ACCESO DIRECTO A LA APLICACIÓN 119
ILUSTRACIÓN 26. ACCESO DIRECTO MENÚ DE INICIO.....119
ILUSTRACIÓN 27: FORMULARIO DE ACCESO AL SISTEMA.....120
ILUSTRACIÓN 28: MENSAJE DE ACCESO DENEGADO.....121
ILUSTRACIÓN 29: PANTALLA PRINCIPAL DE LA APLICACIÓN. 122
ILUSTRACIÓN 30: DESCRIPCIÓN DE COMPONENTES PRINCIPALES DE LA APLICACIÓN. 122
ILUSTRACIÓN 31: DESCRIPCIÓN DEL MENÚ "APLICACIONES".....123
ILUSTRACIÓN 32: DESCRIPCIÓN DEL MENÚ "PROCESOS”.....123
ILUSTRACIÓN 33: DESCRIPCIÓN DEL MENÚ "VENTANAS". 124
ILUSTRACIÓN 34: DESCRIPCIÓN DEL MENÚ "AYUDA".....125
ILUSTRACIÓN 35: DESCRIPCIÓN DE LA BARRA DE HERRAMIENTAS. 125
ILUSTRACIÓN 36: PANTALLA O FORMULARIO PARA LIQUIDAR CARTERA. 127
ILUSTRACIÓN 37: DESCRIPCIÓN DEL PROCESO DE BÚSQUEDA DE UN CRÉDITO 128
ILUSTRACIÓN 38: DESCRIPCIÓN DEL PROCESO DE SELECCIÓN DEL CRÉDITO. 128
ILUSTRACIÓN 39: MENSAJE DE INFORMACIÓN DEL ESTADO DEL CRÉDITO. 129
ILUSTRACIÓN 40: DESCRIPCIÓN DE LA SELECCIÓN DE "FECHA DE LIQUIDACIÓN" 129
ILUSTRACIÓN 41: DESCRIPCIÓN DEL BOTÓN PARA REALIZAR LA LIQUIDACIÓN. 130

ILUSTRACIÓN 42: PANTALLA O FORMULARIO DONDE SE LIQUIDAN INTERESES. 130
ILUSTRACIÓN 43: DESCRIPCIÓN DEL PROCESO DE BÚSQUEDA DE UN CRÉDITO. 131
ILUSTRACIÓN 44: DESCRIPCIÓN DEL PROCESO DE SELECCIÓN DEL CRÉDITO. 131
ILUSTRACIÓN 45: PANTALLAZO CON UN EJEMPLO RESULTADO DE LIQUIDACIÓN. 132
ILUSTRACIÓN 46: DESCRIPCIÓN DEL PASO PARA IMPRIMIR LA LIQUIDACIÓN. 132
ILUSTRACIÓN 47: PANTALLA O FORMULARIO ADMINISTRAR CODEUDORES. 133
ILUSTRACIÓN 48: DESCRIPCIÓN DE LOS PASOS Y LAS OPCIONES DE BÚSQUEDA. 134
ILUSTRACIÓN 49: DESCRIPCIÓN DEL PASO PARA SELECCIONAR UN CRÉDITO. 135
ILUSTRACIÓN 50: MENSAJE DE CRÉDITO CONSULTADO NO TIENE UN CODEUDOR. 135
ILUSTRACIÓN 51: DESCRIPCIÓN DEL PROCESO PARA ADICIONAR NUEVO CODEUDOR.....136
ILUSTRACIÓN 52: MENSAJE DE INFORMACIÓN DE ÉXITO EN LA ADICIÓN CODEUDOR.....136
ILUSTRACIÓN 53: DESCRIPCIÓN DEL PROCESO Y PASOS MODIFICAR UN CODEUDOR.....137
ILUSTRACIÓN 54: DESCRIPCIÓN DEL PROCESO GUARDAR O CANCELAR MODIFICACIÓN..137
ILUSTRACIÓN 55: DESCRIPCIÓN DEL BOTÓN A SELECCIONAR PARA QUITAR CODEUDOR.138
ILUSTRACIÓN 56: MENSAJE DE ADVERTENCIA Y CONFIRMACIÓN QUITAR CODEUDOR. 138
ILUSTRACIÓN 57: DESCRIPCIÓN DEL BOTÓN A SELECCIONAR ELIMINAR UN CODEUDOR..139
ILUSTRACIÓN 58: MENSAJE DE ADVERTENCIA O CONFIRM ANTES ELIMINAR CODEUDOR. 139
ILUSTRACIÓN 59: DESCRIPCIÓN DE CONTROLES FORMULARIO ADMON USUARIOS. 140
ILUSTRACIÓN 60: DESCRIPCIÓN DE LOS PASOS HACER LA BÚSQUEDA DE UN USUARIO. .141
ILUSTRACIÓN 61: DESCRIPCIÓN DEL PROCESO PARA AGREGAR UN NUEVO USUARIO. 142
ILUSTRACIÓN 62: DESCRIPCIÓN DEL PROCESO PARA ELIMINAR UN USUARIO.....142
ILUSTRACIÓN 63: MENSAJE DE ADVERTENCIA Y CONFIRMACIÓN ELIMINAR CODEUDOR. .143
ILUSTRACIÓN 64: DESCRIPCIÓN DEL BOTÓN A SELECCIONAR PARA GUARDAS CAMBIOS. 143
ILUSTRACIÓN 65: DESCRIPCIÓN BOTONES PARA DESPLAZARSE POR LOS REGISTROS....144
ILUSTRACIÓN 66: DESCRIPCIÓN DE LOS BOTONES DE COMANDOS. 144
ILUSTRACIÓN 67: PANTALLA O FORMULARIO ADMINISTRAR PARÁMETROS DEL SISTEMA .145
ILUSTRACIÓN 68: DESCRIPCIÓN DE LOS PASOS PARA CAMBIAR UN PARÁMETRO. 146

LISTADO DE TABLAS

TABLA 1. REQUERIMIENTOS FUNCIONALES.....41
TABLA 2. REQUERIMIENTO NO-FUNCIONAL. USABILIDAD.....43
TABLA 3. REQUERIMIENTO NO-FUNCIONAL. PORTABILIDAD.....43
TABLA 4. REQUERIMIENTO NO-FUNCIONAL. SEGURIDAD.....43
TABLA 5. REQUERIMIENTOS NO-FUNCIONALES. RENDIMIENTO 44
TABLA 6. REQUERIMIENTOS NO-FUNCIONALES. IMPLEMENTACIÓN.....44
TABLA 7. REQUERIMIENTO NO-FUNCIONAL. SOPORTE.....45
TABLA 8. DESCRIPCIÓN CASO DE USO INGRESAR AL SISTEMA. 48
TABLA 9. DESCRIPCIÓN CASO DE USO LIQUIDAR CARTERA.....52
TABLA 10. DESCRIPCIÓN CASO DE USO LIQUIDAR INTERESES. 56
TABLA 11. DESCRIPCIÓN DEL CASO DE USO ADMINISTRAR CODEUDORES.....60
TABLA 12. DESCRIPCIÓN CASO DE USO ADMINISTRAR USUARIOS. 65
TABLA 13. DICCIONARIO DE DATOS TABLA TERCEROS.....71
TABLA 14. DICCIONARIO DE DATOS. TABLA ESTUDIANTES. 72
TABLA 15. DICCIONARIO DE DATOS. TABLA CODEUDORES. 73
TABLA 16. DICCIONARIO DE DATOS. TABLA CRÉDITOS. 73
TABLA 17. DICCIONARIO DE DATOS. TABLA TIPOS CRÉDITO. 74
TABLA 18. DICCIONARIO DE DATOS, TABLA RECIBOS DE CAJA 75
TABLA 19. DICCIONARIO DE DATOS TABLA AMORTIZACIONES.....76
TABLA 20. DICCIONARIO DE DATOS, TABLA FACULTADES.....77
TABLA 21. DICCIONARIO DE DATOS, TABLA PROGRAMAS. 77
TABLA 22. DICCIONARIO DE DATOS, TABLA USUARIOS. 78
TABLA 23. DICCIONARIO DE DATOS, TABLA ROLES.....79
TABLA 24. DICCIONARIO DE DATOS, TABLA PARÁMETROS. 79
TABLA 25. FICHA DE LA APLICACIÓN. 107
TABLA 26. METODOLOGÍA DE PRUEBAS. 107
TABLA 27. CASOS DE PRUEBAS FINALES.....109
TABLA 28. PRUEBA INICIO O AUTENTICACIÓN. 109
TABLA 29. PRUEBA CONSULTAR CRÉDITO. 110
TABLA 30. PRUEBA LIQUIDAR CARTERA 111
TABLA 31. PRUEBA CONSULTAR CRÉDITO. 112
TABLA 32. PRUEBA LIQUIDAR INTERERES.....112
TABLA 33. PRUEBA CONSULTAR POR CRÉDITO 113
TABLA 33. PRUEBA CONSULTAR POR CRÉDITO 113
TABLA 34. PRUEBA ADICIONAR CODEUDOR. 114
TABLA 35. PRUEBA MODIFICAR CODEUDOR. 115
TABLA 36. PRUEBA ELIMINAR CODEUDOR. 115
TABLA 37. HARDWARE REQUERIDO PARA EL PROYECTO. ..... **¡ERROR! MARCADOR NO DEFINIDO.**
TABLA 38. SOFTWARE REQUERIDO PARA EL PROYECTO (LICENCIA DE USO LIBRE) .**¡ERROR! MARCADOR NO DEFINIDO.**

TABLA 39. PRESUPUESTO DEL PROYECTO. ..... **¡ERROR! MARCADOR NO DEFINIDO.**
TABLA 40. CRONOGRAMA DE ACTIVIDADES EN LA ETAPA DE PLANIFICACIÓN. ..... **¡ERROR! MARCADOR NO DEFINIDO.**
TABLA 41. CRONOGRAMA DE ACTIVIDADES EN LA ETAPA DE DISEÑO.. **¡ERROR! MARCADOR NO DEFINIDO.**
TABLA 42. CRONOGRAMA DE ACTIVIDADES EN LA ETAPA DE PRUEBAS. ..... **¡ERROR! MARCADOR NO DEFINIDO.**
TABLA 43. CRONOGRAMA DE ACTIVIDADES ETAPA DE CODIFICACIÓN. **¡ERROR! MARCADOR NO DEFINIDO.**

1 TITULO

DESARROLLO DE SOFTWARE PARA LA GESTIÓN Y CONTROL EFICIENTE DE LA CARTERA DE CRÉDITOS EDUCATIVOS, DE LA UNIVERSIDAD LIBRE DE PEREIRA

2 ANTECEDENTES

La Universidad Libre Seccional Pereira fue fundada hace 40 años, actualmente cuenta con 2 sedes y 4 facultades: Ingeniería, ciencias de la salud, ciencias económicas y contables, derecho. [1]

La Universidad tiene un CPD (Centro de procesamiento de datos) con sistema de seguridad, aire acondicionado, racks e inversores para la protección de servidores en instalaciones adecuadas. En general el CPD cuenta con los sistemas de respaldo, servicios Web y demás servicios para la comunidad académica sobre una buena infraestructura tecnológica para las necesidades actuales con sistema de seguridad, antivirus, firewall; para soportar estos aspectos. Cuenta con personal de mantenimiento, administrador de la red, administrador del sitio Web y el LMS, todos los servicios se proveen internamente.

Se beneficia de un servicio de Internet, en toda la Universidad, con acceso inalámbrico y desde las salas de computo, se dispone también de la plataforma Moodle y aulas virtuales en algunos cursos para el apoyo al proceso formativo presencial, además de un sistema de información académico y financiero tipo Web llamado SIUL (SINU) e implementado en una arquitectura del DBMS (Sistema de Gestión de Bases de Datos) objeto-Relacional Oracle.

Actualmente la universidad ofrece un servicio de financiación de la matrícula para aquellos estudiantes de pregrado, posgrado; trabajadores, administrativos y docentes de la universidad. Los cuales deben cumplir unos requisitos específicos para ser beneficiarios para un crédito directo y en aquellos casos especiales, cuando el estudiante ha agotado otras instancias de financiación la universidad se reserva el derecho de otorgar el crédito directo.[2]

Objetivo del servicio: Brindar facilidades de crédito a los estudiantes para estudiar cualquiera de los programas de pregrado y postgrado que ofrece la Universidad, en casos especiales cuando estos han agotado otras instancias de financiación.

Alcance del servicio: Toda la Comunidad Unilibrista en todas las Seccionales a excepto la Seccional Barranquilla.

[1] Página web Universidad Libre Seccional Pereira. Nuestra Universidad – Reseña histórica. Disponible en: www.unilibrepereira.edu.co

[2] Tesoreria – Financiacion de matricula. Disponible en: www.unilibrepereira.edu.co

Manera de acceder: Solamente se otorgaran créditos directos en casos especiales, cuando el estudiante ha agotado otras instancias de financiación la universidad se reserva el derecho de otorgar el crédito directo.[3]

[3] Página web Universidad Libre Seccional Pereira. Tesoreria – financiación de matricula. Disponible en: www.unilibrepereira.edu.co/acuerdo_financiacion.pdf

3 DESCRIPCIÓN DEL PROBLEMA

Actualmente la Universidad libre seccional Pereira, cuenta con varias modalidades de crédito educativo a los estudiantes, entre los que se encuentran el crédito directo financiado por la misma universidad y el crédito otorgado por el ICETEX. Estos créditos se registran inicialmente en el software ERP de la universidad llamado SINU y posteriormente en el mismo se van registrando los pagos o abonos que los estudiantes realizan a su deuda. El área de tesorería es la que se encarga de gestionar dicha cartera, pero actualmente se encuentra con varios inconvenientes y retrasos a la hora de realizar la amortización y la liquidación de intereses de dichos créditos, ya que el software SINU no tiene el modulo con los requisitos funcionales y específicos que requiere el área, para que de manera eficiente se pueda tener una correcta administración de dicha cartera, de manera ágil y automatizada. Igualmente el software SINU no calcula o no tiene en cuenta los intereses que se generan en aquellos créditos donde los estudiantes incumplen con sus obligaciones crediticias. Otros inconvenientes que se evidencian, es que el software no consolida los créditos directos junto con los ICETEX en aquellos casos que un mismo estudiante es beneficiario de los dos créditos; como de igual forma el software no maneja información de los codeudores de los créditos. Por consiguiente se genera un gran traumatismo al momento de hacer la consolidación individual y general de esta cartera; con lo que específicamente en este proceso, el software no brinda un apoyo real y eficiente para su correcta gestión, dando lugar a retrasos, pérdida de tiempo en re-procesos y trabajos adicionales que origina que se deban crear procesos alternos de forma manual y poco agiles, llevando el registro de cada pago y efectuando el cálculo de los periodos o número de días en mora del crédito, en una plantilla de Excel por cada crédito y por cada estudiante, para poder efectuar la correcta amortización del crédito y la respectiva liquidación de sus intereses, conllevando a crear en esta plantilla de Excel doble digitación de la información que inicialmente ya fue ingresada en el software SINU.

4 JUSTIFICACION

Con el pasar del tiempo, los métodos de ingreso y almacenamiento de la información han ido evolucionando, pero en algunos casos las empresas no comprenden de la necesidad de este cambio y se quedan poco a poco relegadas. Este es el caso de la Universidad Libre Seccional Pereira, que logró comprender el retraso y la necesidad de evolucionar.

Actualmente el proceso de liquidación de intereses y amortización de los créditos educativos genera muchos retrasos e inconvenientes en el área de tesorería de la Universidad Libre, porque el software SINU donde se administran actualmente los créditos educativos, no tiene un manejo eficiente y acorde a los requerimientos del área para la correcta gestión de dichos créditos. Con lo que esta carencia en el software actual conlleva a generar por cada deudor un archivo en Excel con la respectiva liquidación de intereses y la respectiva amortización del crédito; archivos cuales se deben ir actualizando periódicamente y de forma manual a medida que los deudores abonan a su deuda, con lo que origina un nuevo recalculo de toda la información. Con este inconveniente no sólo se ven afectados los empleados de tesorería responsables del todo el proceso de crédito y manejo de la cartera en sus tiempos de respuesta, sino también toda la comunidad estudiantil beneficiaria de dichos créditos.

Lo que se pretende al desarrollar este aplicativo es remplazar la necesidad de diseñar estos archivos en Excel por cada deudor y aportar un software que realice todo este proceso de forma automática los más ágil y eficiente posible para el usuario final. Y a su vez suplir las carencias comprobadas que posee el software ERP de la universidad (SINU) para administrar dicha cartera de manera útil, ágil y eficiente.

5 OBJETIVOS

5.1 OBJETIVO GENERAL

Diseñar una solución a nivel de software que resuelva la problemática evidenciada en el área de tesorería, en cuanto al manejo y gestión de la cartera de los créditos educativos directos financiados por la misma Universidad. Mediante el análisis del proceso actual, planteando la mejor solución posible de optimización.

5.2 OBJETIVOS ESPECIFICOS

- Determinar y analizar los requerimientos iniciales del software utilizando la técnica de casos de uso y sus respectivos diagramas de secuencia y actividades comprendidas en el estándar de modelado de sistemas de software UML y Emplearla conjuntamente con la metodología de desarrollo rápido de aplicaciones (RAD) .

- Diseñar una aplicación de prototipo con una interface gráfica de usuario, que consulte o extraiga la información necesaria de los créditos y pagos que inicialmente ya fueron registrados en la base de datos del software ERP de la universidad (SINU) y que permita de manera amigable el control del proceso y la interacción amigable con el usuario.

- Utilizar las ventajas de las nuevas herramientas para el desarrollo de software como .NET y sus potentes lenguajes C++ o Visual Basic y la herramienta de integración Oracle Developer Tools para Visual Studio.

6 HIPÓTESIS

Con el diseño de un software resolveremos el inconveniente evidenciado en el manejo y gestión de la cartera de los créditos educativos dando paso a un sistema que agilice el proceso, logrando conseguir un manejo eficiente en el momento de liquidar los intereses de los créditos otorgados a los estudiantes. Y además con el diseño del aplicativo y la base de datos, conseguiremos mejorar la apreciación que se tiene actualmente de los proyectos de desarrollo de software diseñados por los mismos estudiantes de la universidad.

Con la información recopilada sobre los sistemas de información existentes en la universidad se determinará los pasos a seguir para llevar a cabo el desarrollo de un programa más práctico. Para llevar a cabo el proyecto se utilizara una metodología de desarrollo rápido de aplicaciones donde en la fase inicial de planificación se obtendrán los requerimientos funcionales y no funcionales de los usuarios y se definirá un plan de pruebas inicial. En la parte de diseño se establecerán las partes y funcionalidades de la aplicación a través de diagramas UML, pasando a la codificación en la cual se desarrollara la aplicación, definiendo estándares y parámetros de codificación utilizando el paradigma de programación orientada a objetos. Y ya en la fase final se realizaran las pruebas unitarias analizando los resultados obtenidos si se puede continuar iterando o no, para terminar con las pruebas y el producto final.

6.1 Determinación de variables

6.1.1 **Variable Independiente:** Desarrollo de un software que calcule los intereses que se generan por el no pago de las obligaciones crediticias para gestionar la cartera administrada por el área de tesorería de la Universidad Libre Seccional Pereira.

6.1.2 **Variable Dependiente:** Agilizar los procesos operativos para la gestión de la cartera de créditos directos otorgados a los estudiantes de la Universidad, que permita una administración más efectiva en la liquidación de esta cartera.

7 DELIMITACIÓN Y ALCANCE DEL PROYECTO

Los Límites de estudio del presente proyecto, están involucrados dentro de las necesidades expresadas por el área de tesorería y la autorización previa de los administrativos de la universidad para tener acceso a toda la información concerniente a los créditos educativos directos almacenados en la base de datos del SINU.

Este proyecto se desarrollara bajo la asesoría de tutores calificados por la misma Universidad que exhibe la problemática, y será propuesto para ser aplicado en el centro educativo.

7.1 Delimitación espacial

El presente proyecto objeto de estudio abarcará el área de tesorería de la Universidad Libre en la Seccional de la ciudad de Pereira del departamento de Risaralda - Colombia.

Viendo como referencia la problemática que se analizará se pretende favorecer a toda la comunidad educativa de la Universidad y los empleados de tesorería encargados de la gestión de dichos créditos educativos directos.

7.2 Delimitación cronológica

El proyecto pretende establecer un tiempo máximo de desarrollo y la implementación del mismo durante los tres meses siguientes. En donde se empezó a regir cada actividad a realizar y a cumplir en el cronograma de actividades diseñado para tal fin.

7.3 Delimitación conceptual

El desarrollo del proyecto tiene como finalidad hacer un análisis y plantear una solución a nivel de software tipo prototipo donde se abarcara la problemática actual que se presenta en el área de tesorería de la Universidad, en el proceso especifico de hacer la conciliación de la cartera y la liquidación de los intereses causados en aquellos créditos educativos que otorga directamente la Universidad a sus estudiantes.

El estudio de la problemática y un conocimiento previo de la información almacenada en las bases de datos referente a los créditos, deudores, codeudores y abonos, es uno de los factores más importantes que se debe perseguir para un mayor entendimiento del problema y así poder brindar la mejor alternativa a una solución óptima y eficiente, que conlleven a una culminación exitosa y satisfactoria del proyecto el cual quiere brindar una soluciona a una problemática existente.

Este proyecto abarca dos conceptos fundamentales como lo son la optimización y los tiempos de respuestas que toda organización quiere lograr, para un óptimo desempeño de sus empleados, procesos críticos y el tiempo empleado en sus labores; para alcanzar una calidad acorde y satisfactoria a los servicios ofrecidos.

8 MARCO REFERENCIAL

En este capítulo se dará a conocer la metodología utilizada, conceptos técnicos y herramientas relacionados con el desarrollo del proyecto para la gestión y control eficiente de la cartera de créditos educativos. Comenzando por la metodología de *Programación Extrema (XP)* para la ingeniería del software, basado en la técnica de *Lenguaje Unificado de Modelado (UML)*, utilizando herramientas de *Ingeniería del software asistida por computador (CASE)* y para el *Desarrollo Rápido de Aplicaciones (RAD).* Culminando con los conceptos más relevantes utilizados en el desarrollo del proyecto.

8.1 MARCO TEORICO

8.1.1 Programación Extrema O Extreme Programming (Xp)

En esta parte del documento se hace una introducción y una presentación teórica de XP como metodología de desarrollo. Luego exponiendo los principios teóricos que inspiraron esta metodología. Prosiguiendo con una conceptualización importante sobre XP la cual consta de los valores, principios y el alcance de la misma.

8.1.1.1 Introducción a XP

Es la metodología de desarrollo de la ingeniería de software formulada por Kent Beck, autor del primer libro sobre la materia. Esta fue la metodología de desarrollo elegida para el diseño de la aplicación o prototipo, por ser la más destacada de los procesos ágiles de desarrollo de software. “Al igual que éstos, la programación extrema se diferencia de las metodologías tradicionales principalmente en que pone más énfasis en la adaptabilidad que en la previsibilidad. Se considera que los cambios de requisitos sobre la marcha son un aspecto natural, inevitable e incluso deseable del desarrollo de proyectos que ser capaz de adaptarse a los cambios de requisitos en cualquier punto de la vida del proyecto es una aproximación mejor y más realista que intentar definir todos los requisitos al comienzo del proyecto e invertir esfuerzos después en controlar los cambios en los requisitos”. [4]

[4] Weitzenfeld, Alfredo. Ingenieria del software orientada a objetos con UML, Java e internet. Editorial THOMSON. 2011. P. 55.

8.1.1.2 Principios Teóricos de XP

"Se puede considerar la programación extrema como la adopción de las mejores metodologías de desarrollo de acuerdo a lo que se pretende llevar a cabo con el proyecto, y aplicarlo de manera dinámica durante el ciclo de vida del software. Para el desarrollo de aplicaciones se requiere de un grupo de programadores pequeño, dónde la comunicación sea más factible, adoptando únicamente las etapas y/o características que se acoplen a nuestras necesidades a la hora de la realización de la herramienta".[5]

8.1.1.3 Conceptualización

XP resalta una serie de valores y principios que deben tenerse en cuenta y practicarlos durante el tiempo de desarrollo que dure el proyecto. Al final de este apartado se enuncian algunas de las características que deben tener los proyectos que se realicen con XP.

8.1.1.4 Valores

Más que una metodología, XP se considera una disciplina, la cual está sostenida por valores y principios propios de las metodologías ágiles. Existen cuatro valores que cumplen su papel como pilares en el desarrollo de las metodologías livianas:

- **La comunicación:** En la metodología XP es muy importante que exista un ambiente de colaboración y comunicación al interior del equipo de desarrollo, así como en la interacción de éste con el cliente. En XP la interacción con el cliente es tan estrecha, que es considerado parte del equipo de desarrollo.

- **La simplicidad:** Este valor se aplica en todos los aspectos de la programación extrema. Desde diseños muy sencillos donde lo más relevante es la funcionalidad necesaria que requiere el cliente, hasta la simplificación del código mediante la refactorización del mismo. La programación XP no utiliza sus recursos para la realización de actividades complejas, sólo se desarrolla lo que el cliente demanda, de la forma más sencilla.

[5] Weitzenfeld, Alfredo. Ingenieria del software orientada a objetos con UML, Java e internet. Editorial THOMSON. 2005. p. 55.

- **La retroalimentación:** Se presenta desde el comienzo del proyecto, ayuda a encaminarlo y darle forma. Ésta se presenta en los dos sentidos, por parte del equipo de trabajo hacia el cliente, con el fin de brindarle información sobre la evolución del sistema, y desde el cliente hacia el equipo en los aportes a la construcción del proyecto.

- **El coraje:** El equipo de desarrollo debe estar preparado para enfrentarse a los continuos cambios que se presentarán en el transcurso de la actividad. Cada integrante debe tener el valor de exponer los problemas o dudas que halle en la realización del proyecto. Aún con estas variaciones, las jornadas de trabajo deben proporcionar el máximo rendimiento.

8.1.1.5 Prácticas

A partir de los valores se plantea una serie de prácticas que sirven de guía para los desarrolladores en esta metodología. Una de los aspectos más importantes para XP son las doce reglas que se plantean, las cuales se caracterizan por su grado de simplicidad y por su enfoque en la practicidad, además de que cada regla se complementa con las demás. A continuación se realizará una breve descripción de cada una de ellas.

El desarrollo está dirigido por pruebas. Antes de realizar una unidad de código, es necesario contar con su respectiva unidad de pruebas. El programador realiza pruebas dirigidas al funcionamiento de nuevas adiciones o módulos al sistema.

- **El juego de la planificación:** Desde el comienzo del desarrollo se requiere que el grupo y el cliente tengan una visión general y clara del proyecto, es decir, deben entender y estar de acuerdo con lo que el "otro" plantee. "En el transcurso del proyecto se realizan diferentes reuniones, con el fin de organizar las tareas e ideas que surgen tanto por parte del cliente como por el equipo".[6]

- **Cliente in-situ:** El cliente, o un representante del mismo, deben estar en el sitio de desarrollo para solucionar las preguntas o dudas que se puedan presentar a medida que se realice el proyecto.

- **Programación en parejas:** XP propone que exista una pareja de programadores por monitor y teclado, como medida para aumentar la calidad del código. Esta práctica busca reducir los errores de codificación, mientras uno de los programadores busca una forma de dar funcionalidad a un módulo, el otro programador aprueba dicho código y busca la forma de simplificarlo.

[6] Newkirk, James. La pogramacion extrema en la práctica. Editorial Eddison Wesley. 2002. P. 18.

- **Entregas pequeñas:** En la programación extrema se realizan entregas constantes de módulos funcionales completos, de tal forma que en todo momento el cliente tiene una parte de aplicación funcionando. En XP no existe el desarrollo incompleto de una tarea, ésta se ejecuta en su totalidad o no se hace.

- **Refactorización sin piedad:** El código se revisa de forma permanente para depurarlo y simplificarlo, buscando la forma de mejorarlo. La refactorización se realiza durante todo el proceso de desarrollo.

- **Integración continúa del código:** El código de los módulos debe ser integrado a cortos plazos de tiempo, preferiblemente no mayores a un día. Esto facilita la búsqueda y la corrección de errores de codificación e integración que se presenten en el proceso.

- **Diseño simple:** Sólo se realiza lo necesario para que la aplicación cumpla con la funcionalidad requerida por el cliente.

- **Utilización de metáforas del sistema:** Para el mejor entendimiento de los elementos del sistema por parte del equipo de desarrollo se acude a la utilización de metáforas, como una forma de universalizar el lenguaje del sistema.

- **Propiedad colectiva del código:** El código no es conocido por una sola persona del grupo de trabajo, esto facilita implementar cambios al programa por parte de otros integrantes del equipo.

- **Convenciones de código:** La aplicación de estándares de programación al código fuente de la aplicación, permite que todas las personas que conforman el grupo de trabajo puedan entender y realizar modificaciones al código del sistema.

- **No trabajar horas extras:** Es preferible volver a estimar los tiempos de entrega. Con esta práctica se busca utilizar al máximo el rendimiento y energía del programador.

8.1.1.6 Alcance

La programación extrema es conveniente en ciertas situaciones, pero también es necesario saber que presenta controversia en otras. “Esta metodología es

aplicable con resultados positivos a proyectos de mediana y pequeña envergadura, donde los grupos de trabajo no superan 20 personas".[7]

Otro aspecto importante en la selección de esta metodología radica en el ambiente cambiante que se presenta en los requerimientos de la aplicación. La metodología XP está encaminada hacia los desarrollos que requieren de cambios continuos en el transcurso de un proyecto. La metodología es recomendada para proyectos en los cuales el costo de cambio no se incremente a medida que transcurre vida del mismo.

8.1.2 UML (Lenguaje Unificado de Modelado)

"Lenguaje gráfico para visualizar, especificar, construir y documentar un sistema. UML ofrece un estándar para describir un modelo del sistema, incluyendo aspectos conceptuales tales como procesos de negocio, funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y compuestos reciclados. Diseñado para trabajar con el paradigma orientado a objetos para el desarrollo de software".[8]

8.1.3 StarUML

Será la herramienta utiliza en las fases de análisis de requerimientos y diseño orientado a objetos comprendidos en el ciclo de vida del software, para definir y modelar el propio sistema, detallando los artefactos necesarios para su adaptabilidad; para documentar y construir especificando y describiendo los métodos o procesos a diseñar que posteriormente se desarrollaran.

8.1.4 Herramientas RAD

Para el diseño de la aplicación o prototipo se utilizaran Herramienta Rápida de Desarrollo (RAD), las cuales se basan en el proceso de desarrollo de software creado inicialmente por James Maslow en 1980, para el desarrollo rápido de interfaces gráficas de usuario o como entornos de desarrollo integrado (IDE). El método comprende el desarrollo interactivo, la construcción de prototipos y el uso de herramientas CASE. Tiende a englobar también la usabilidad, utilidad y la

[7] Newkirk, James. La pogramacion extrema en la práctica. Editorial Eddison Wesley. 2002. P. 18
[8] E. Lopez, Teniente, A. Olivé, Ramon A. Diseño de sistemas software en UML. Editorial POLITEXT. 2003. P. 39.

rapidez de ejecución proporcionando utilidades que automatizan tareas del desarrollo de la aplicación haciéndola transparente para el programador.

8.1.4.1 Visual Studio .Net Express Edition

Es una herramienta (RAD) ligera de carácter gratuito y es proporcionado por la compañía Microsoft Corporation, para la siguiente generación de Internet que son los Servicios Web XML. Es un entorno de desarrollo integrado que nos ayuda a diseñar, desarrollar, depurar e implementas con rapidez soluciones basadas en .NET Framework, en el cual se pueden crear aplicaciones Windows Forms y Web Forms que integran datos y lógica de negocio; todo esto con una gran facilidad, rapidez y bajo costo y para la cual ORACLE Corporation desarrollo un Kit de herramientas para la integración con esta arquitectura.

8.1.5 Oracle Database 11g Express Edition (Oracle Database XE)

Es una versión libre de la base de datos relacional más robusta del mundo. Oracle Database XE es fácil de instalar, fácil de administrar y fácil de desarrollar. Con Oracle Database XE, se utiliza una interfaz intuitiva basada en navegador para administrar la base de datos, crear tablas, vistas y otros objetos de base de datos, importación, exportación, y ver los datos de la tabla, ejecutar consultas y scripts SQL y generar informes. Es gratis para desarrollar, implementar y distribuir, fácil de descargar y fácil de administrar.

8.1.6 Oracle SQL Data Modeler

Es una herramienta gratuita para el modelado de datos que se apoya en el diseño lógico de diagramas de entidad relación y modelos relacionales, con capacidades de ingeniería directa e inversa entre los dos. Es compatible con el flujo multidimensional de datos, tipo de datos y los modelos físicos, y permite a los archivos que se importan de una variedad de fuentes y exportados a una variedad de destinos. Permite a los usuarios configurar las convenciones de nombres y verificar diseños utilizando un conjunto de normas de diseño predefinidas.

8.1.7 Oracle SQL Developer

Es un entorno de desarrollo integrado gratuito que simplifica el desarrollo y gestión de base de datos Oracle. SQL Developer ofrece un desarrollo completo de

extremo a extremo de las aplicaciones PL / SQL, una hoja de cálculo para ejecutar consultas y scripts, una consola de DBA para la gestión de la base de datos, una interfaz de informes, una solución completa de migración para mover el bases de datos.

8.1.8 Herramienta CASE

"Las herramientas CASE (Ingeniería del software asistida por computador) que se especializan en Ingeniería del software, es la aplicación de tecnología informática a las actividades, las técnicas y las metodologías propias de desarrollo, su objetivo es acelerar el proceso para el que han sido diseñadas, en el caso de CASE para automatizar o apoyar una o más fases del ciclo de vida del desarrollo de sistemas."[9]

Para la fase de ingeniería del software se utilizara la herramienta MyGeneration que es una potente aplicación informática destinada a aumentar la productividad en el desarrollo de software reduciendo el costo de las mismas en términos de tiempo y de dinero. Su principal ventaja para el proyecto es que está orientada a la tecnología .NET de Microsoft, ya que permite crear código en VB.NET.

8.2 MARCO CONCEPTUAL

Modelo de parámetros que se utilizaran para llevar a cabo el proyecto

8.2.1 Software

"Es el conjunto de los programas de cómputo, procedimientos, reglas, documentación y datos asociados, que forman parte de las operaciones de un sistema de computación, que comprende el conjunto de los componentes lógicos necesarios que hacen posible la realización de tareas específicas".[10] Si bien esta distinción es, en cierto modo, arbitraria, y a veces confusa, a los fines prácticos se puede clasificar al software en tres grandes tipos:

8.2.1.1 Software del Sistema

[9] Stair, Ralph M. Principios de sistemas de información: Enfoque administrativo. Cuarta edición. Editorial THOMPSON. 1999. P. 543.
[10] Wikipedia. Software. Disponible en es.wikipedia.org/wiki/software.

Es aquel que permite que el hardware funcione. Su objetivo es desvincular adecuadamente al programador de los detalles del computador en particular que se use, aislándolo especialmente del procesamiento referido a las características internas de: memoria, discos, puertos y dispositivos de comunicaciones. Incluye entre otros: Sistemas operativos, controladores de dispositivo, herramientas de diagnóstico, corrección y optimización, servidores y utilidades.

8.2.1.2 Entorno de Desarrollo Integrado (IDE)

Es el conjunto de herramientas que permiten al programador desarrollar programas informáticos, usando diferentes alternativas y lenguajes de programación, de una manera práctica. Incluye entre otros: Editores de texto, compiladores, intérpretes, enlazadores y depuradores. Agrupan las anteriores herramientas, usualmente en un entorno visual, de forma que el programador no necesite introducir múltiples comandos para compilar, interpretar, depurar.

8.2.1.3 Software de aplicación

"Aquel que permite a los usuarios llevar a cabo una o varias tareas específicas, en cualquier campo de actividad susceptible de ser automatizado o asistido, con especial énfasis en los negocios. Incluye entre otros: Aplicaciones de sistema de control y automatización industrial, aplicaciones ofimáticas, software educativo, software médico, video juegos". [11]

8.2.2 Ingeniería del software

"Es una disciplina, un área de la Informática o una ciencia de la computación, que ofrece métodos y técnicas para desarrollar y mantener software de calidad que satisface ciertas necesidades, donde su funcionamiento es eficiente sobre maquinas reales".[12]

[11] FALGERAS, Campderrich Benet. Ingenieria del software. Primera edición. Editorial UOC. 2003. P.17.

[12] Lbid., P.19

8.2.3 Lenguajes de programación

"Un lenguaje de programación es un lenguaje que puede ser utilizado para controlar el comportamiento de una máquina, particularmente una computadora. Éste a su vez consiste en un conjunto de símbolos y reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos y expresiones"[13].

8.2.4 Base de datos

"Una base de datos es una colección de datos lógicamente relacionados, junto con una descripción de estos datos, que están destinados para satisfacer las necesidades de información de una organización".[14]

Se aplicará la configuración a las extensiones, cada una con los permisos y servicios antes estipulados. Es necesario tener en cuenta en esta actividad que para la configuración de las líneas se deben conocer previamente los conceptos más importantes sobre el protocolo y registro de las líneas.

8.2.5 Programación Orientada a Objetos (POO)

Lo que se busca con una POO, es identificar claramente los problemas de la vida real, relacionándolo con un componente real, del cual se puede crear una abstracción más clara y precisa, como lo indica "A nivel organizacional el concepto del objeto nos acerca más a la manera de pensar de la gente al agregar un nivel de abstracción adicional donde internamente la estructura del programa se ajusta a la arquitectura de la máquina. Los datos globales desaparecen, asignando a cada objeto sus propios datos y funciones locales, resultando en un programa o aplicación definido exclusivamente en término de objetos y sus relaciones entre sí".[15]

Un software orientado a objetos (POO), cuenta con una serie de aspectos, lo que convierte el proyecto de mejor calidad.

Características Mínimas de los Lenguajes Orientados a Objetos:

---

[13] FALGERAS, Campderrich Benet. Ingenieria del software. Primera edición. Editorial UOC. 2003. P.17.
[14] Lbid., p. 23
[15] Lbid., p. 28.

- **Abstracción:** Denota las características esenciales de un objeto, donde se capturan sus comportamientos. Cada objeto en el sistema sirve como modelo de un "agente" abstracto que puede realizar trabajo, informar y cambiar su estado, y "comunicarse" con otros objetos en el sistema sin revelar cómo se implementan estas características.

- **Encapsulamiento:** Significa reunir todos los elementos que pueden considerarse pertenecientes a una misma entidad, al mismo nivel de abstracción. Esto permite aumentar la cohesión de los componentes del sistema. Algunos autores confunden este concepto con el principio de ocultación, principalmente porque se suelen emplear conjuntamente.

- **Modularidad:** Se denomina modularidad a la propiedad que permite subdividir una aplicación en partes más pequeñas (llamadas módulos), cada una de las cuales debe ser tan independiente como sea posible de la aplicación en sí y de las restantes partes. Estos módulos se pueden compilar por separado, pero tienen conexiones con otros módulos. Al igual que la encapsulación, los lenguajes soportan la modularidad de diversas formas.

- **Principio de ocultación:** Cada objeto está aislado del exterior, es un módulo natural, y cada tipo de objeto expone una interfaz a otros objetos que específica cómo pueden interactuar con los objetos de la clase. El aislamiento protege a las propiedades de un objeto contra su modificación por quien no tenga derecho a acceder a ellas; solamente los propios métodos internos del objeto pueden acceder a su estado.

- **Polimorfismo:** Comportamientos diferentes, asociados a objetos distintos, pueden compartir el mismo nombre; al llamarlos por ese nombre se utilizará el comportamiento correspondiente al objeto que se esté usando. O, dicho de otro modo, las referencias y las colecciones de objetos pueden contener objetos de diferentes tipos, y la invocación de un comportamiento en una referencia producirá el comportamiento correcto para el tipo real del objeto referenciado.

- **Herencia:** Las clases no están aisladas, sino que se relacionan entre sí, formando una jerarquía de clasificación. Los objetos heredan las propiedades y el comportamiento de todas las clases a las que pertenecen. La herencia organiza y facilita el polimorfismo y el encapsulamiento, permitiendo a los objetos ser definidos y creados como tipos especializados de objetos preexistentes. Estos pueden compartir (y extender) su comportamiento sin tener que volver a implementarlo.

8.2.6 RAD (Desarrollo Rápido de Aplicaciones):

Son herramientas para el desarrollo rápido de interfaces gráficas de usuario o entorno de desarrollo integrado completo. El desarrollo rápido de aplicaciones, es un proceso de desarrollo de software, desarrollado inicialmente por James Maslow en 1980. El método comprende el desarrollo interactivo, la construcción de prototipos y el uso de utilidades CASE (Computer Aided Software Engineering). Tradicionalmente, el desarrollo rápido de aplicaciones tiende a englobar también la usabilidad, utilidad y la rapidez de ejecución.

Herramientas de desarrollo de software que permite construir sistemas utilizables en poco tiempo, normalmente de 60 a 90 días, frecuentemente con algunas concesiones.

8.2.7 UML

"Es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad. Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema. UML ofrece un estándar para describir un modelo del sistema, incluyendo aspectos conceptuales tales como procesos de negocio, funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y compuestos reciclados".[16]

8.2.8 StarUML

Herramienta utilizada para el modelado basada en el estándar UML, es una alternativa gratuita de open source.

8.2.9 Herramientas CASE

Son diversas aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de software reduciendo el costo de las mismas en términos de tiempo y de dinero. Estas herramientas pueden ayudar en todos los aspectos del ciclo de vida de desarrollo del software.

[16] FALGERAS, Campderrich Benet. Ingenieria del software. Primera edición. Editorial UOC. 2003. P.32.

8.2.10 ERP

Son sistemas de información gerenciales que integran y manejan muchos de los negocios asociados con las operaciones de producción y de los aspectos de distribución de una compañía en la producción de bienes o servicios.

8.2.11 .NET

"Es un framework de Microsoft que hace un énfasis en la transparencia de redes, con independencia de plataforma de hardware y que permita un rápido desarrollo de aplicaciones".[17]

8.2.12 Visual Basic (VB.NET)

"Es un lenguaje de programación orientado a objetos que se puede considerar una evolución de Visual Basic implementada sobre el framework .NET".[18]

8.2.13 Oracle Database

Es un sistema de gestión de base de datos objeto-relacional, desarrollado por Oracle Corporation.

8.2.14 DBMS

"Sistemas de gestión de bases de datos (DBMS) son aplicaciones que interactúan con el usuario, otras aplicaciones y la propia base de datos para capturar y analizar datos especialmente diseñados".[19]

8.2.15 RDBMS

Sistema de gestión de base de datos objeto-relacional.

---

[17] BIRNIOS, Mariano. Microsoft Visual Basic.NET quia del programador. Edicion ilustrada. Editorial MP ediciones s.a. 2002. P. 34
[18] Lbid., p. 35
[19] Lbid., p. 123.

9 MARCO METODOLÓGICO

9.1 TIPO DE INVESTIGACIÓN

El tipo de investigación en la cual está enfocado el desarrollar del proyecto, es la investigación cuantitativa. Donde se pueden adquirir conocimientos fundamentales y la elección del modelo más adecuado que nos permita conocer la realidad del problema de una manera más imparcial y eficiente, ya que se recogen y analizan los datos a través de los conceptos y variables. Según su clasificación está relacionada con la *Investigación Aplicada*, que se caracteriza porque busca utilizar los conocimientos que se adquieren. La investigación aplicada se encuentra estrechamente vinculada con la investigación básica, pues depende de los resultados y avances de esta última; ya que toda investigación aplicada requiere de un marco teórico.

9.2 MÉTODO DE INVESTIGACIÓN

Todo proyecto de software desarrollado bajo la metodología de desarrollo XP inicia con una o varias reuniones con el cliente, en las cuales se da claridad a la necesidad puntual del mismo a través de las historias de usuario. Estas también sirven de base para crear una metáfora del sistema con el cual todo el equipo de trabajo tendrá una idea general de la aplicación a implementar. Con base en las historias de usuario se crean las pruebas de aceptación las cuales deben ser diseñadas antes de iniciarla codificación. Concluida esta etapa, se debe acordar un plan de entregas con el cliente del cual surge el número inicial de iteraciones y duración de las mismas. Esta reunión de entregas puede repetirse en el transcurso del proyecto, siempre que la velocidad del mismo cambie lo suficiente para tener que replantear el plan de entregas o que surjan nuevas historias de usuario que justifiquen la alteración de dicho plan.

A continuación se detalla cada una de las etapas de desarrollo comprendidas por la metodología XP (planeación, diseño, codificación y pruebas) describiendo cada uno de los aspectos que la componen, son las siguientes:

9.2.1 Planificación

En este punto se comienza a interactuar con el cliente y el resto del grupo de desarrollo para descubrir los requerimientos del sistema y de los usuarios a través de *User Stories*, se obtendrán los requerimientos a partir del documento generado

en la primera fase de la metodología planteada. Se definirán las iteraciones a realizar para el desarrollo de la herramienta de acuerdo al número de *User Stories* obtenidos, y se definirá un plan de pruebas inicial y pruebas unitarias, que se irán refinando en cada iteración. A continuación se enumeran los elementos de esta etapa:

- **Historias de usuario:** El sistema es desarrollado para el cliente, por lo tanto, el usuario es quien decide que tareas realizará la aplicación. Este planteamiento se desarrolla a lo largo del proyecto: el cliente es quien decide que hacer. Como primer paso, se debe proporcionar una idea clara de lo que será el proyecto en sí. Las historias de usuario son utilizadas como herramienta para dar a conocer los requerimientos del sistema al equipo de desarrollo. "Se puede considerar que las historias de usuario en XP juegan un papel similar a los casos de uso en otras metodologías. Las historias de usuario también son utilizadas para estimar el tiempo que el equipo de desarrollo tomará para realizar las entregas. En una entrega se puede desarrollar una o varias historias de usuario, esto depende del tiempo que demore la implementación de cada una de las mismas".[20]

- **Velocidad del proyecto:** Es una medida de la capacidad que tiene el equipo de desarrollo para evacuar las historias de usuario en una determinada iteración. Esta medida se calcula totalizando el número de historias de usuario realizadas en una iteración. Para la iteración siguiente se podrá (teóricamente) implementar el mismo número de historias de usuario que en la iteración anterior.

- **Iteraciones:** "En la metodología XP, la creación del sistema se divide en etapas para facilitar su realización. Por lo general, los proyectos constan de más de tres etapas, las cuales toman el nombre de iteraciones, de allí se obtiene el concepto de metodología iterativa. La duración ideal de una iteración es de una a tres semanas. Para cada iteración se define un módulo o conjunto de historias que se van a implementar".[21]

- **Entregas Pequeñas:** La duración de una iteración varía entre una y tres semanas, al final de la cual habrá una entrega de los avances del producto, los cuales deberán ser completamente funcionales. Estas entregas deben caracterizarse por ser frecuentes.

- **Reuniones:** El planeamiento es esencial para cualquier tipo de metodología, es por ello que XP requiere de una revisión continua del plan de trabajo. A pesar de ser una metodología que evita la documentación exagerada, es muy estricta en la organización del trabajo.

[20] Newkirk, James. La pogramacion extrema en la práctica. Editorial Eddison Wesley. 2002. P. 18
[21] Lbid., pag 25.

- **Plan de entregas:** Al comenzar el proyecto se realiza una reunión entre el equipo de trabajo y los clientes. En dicha reunión se define el marco temporal de la realización del sistema. El cliente expone las historias de usuario a los integrantes de grupo, quienes estimarán el grado de dificultad de la implementación de cada historia.

- **Inicial de Iteración:** Al comenzar una iteración se realiza una reunión de la misma, donde se organizan las actividades de programación a realizar. Las historias de usuario son traducidas a tareas y asignadas a los desarrolladores. Los desarrolladores estiman los tiempos para la realización de las tareas.

- **Diarias o "stand-up meeting":** Estas reuniones se realizan al comenzar la jornada laboral. Todo el equipo de desarrollo se reúne para exponer los problemas e ideas que se estén presentando, esto con el fin que el equipo en conjunto construya una mejor solución.

- **Roles XP:** En esta metodología se utiliza el concepto de roles para organizar quienes se encargaran de cada una de las actividades que deben realizarse en el transcurso del proyecto. "Cada uno de estos papeles son desempeñados por uno o varios integrantes del grupo, sin descartar la posibilidad de rotar los roles entre el equipo durante la realización del sistema"[22]. Los roles dentro de la metodología son:

- **El jefe de proyecto:** tiene como responsabilidad la dirección y organización de las reuniones que se realizan durante el proyecto.

- **El usuario o cliente:** determina qué se va a construir en el sistema, además de decidir el orden en que se entregarán cada segmento del proyecto.

- **Los programadores, diseñadores y analistas:** Los programadores son quienes construyen el sistema y realizan las pruebas correspondientes a cada módulo o unidad de código.

- **El entrenador (coach):** Es el responsable de que el proceso se realice de forma correcta. Se asegura de que los conceptos de la metodología se apliquen al proyecto, además de brindar ayuda continua a los demás integrantes del equipo.

- **El tester:** o quien realiza las pruebas, colabora en la realización de las pruebas de aceptación y es quien muestra los resultados de las mismas. En este proceso, ayuda al cliente a diseñar tales pruebas y a verificar que las pruebas sean aprobadas.

[22] Newkirk, James. La pogramacion extrema en la práctica. Editorial Eddison Wesley. 2002. P. 28.

- **El rastreador (tracker):** tiene como tarea observar la realización del sistema.

- **Traslado de personal:** Al mover el personal se evitan problemas relacionados con la pérdida de conocimiento y cuellos de botella. Todos los miembros del grupo deben tener suficiente conocimiento de la estructura del código de modo tal que se eviten las islas de conocimiento las cuales son susceptibles de generar pérdidas de información importante.

- **Ajustar XP:** Todos los proyectos tienen características específicas por lo cual XP puede ser modificado para ajustarse bien al proyecto en cuestión. Al iniciar el proyecto se debe aplicar XP tal como es, sin embargo no se debe dudar en modificar aquellos aspectos en que no funcione. Eso no quiere decir que los desarrolladores pueden hacer lo que se les antoje. Antes de implementarse un cambio, este debe ser discutido y aprobado por el grupo.

9.2.2 Diseño

Etapa donde se definirán los términos que se emplearán en el grupo de trabajo, y se establecerán las partes y funcionalidades de la herramienta a través de diagramas. En nuestro caso utilizaremos UML (*Unified Modelling Language*).

Los aspectos que se tratarán a continuación son las fases comprendida en esta etapa: simplicidad en el diseño, metáfora del sistema, tarjetas CRC, spike solution, no solucionar antes de tiempo y refactoring.

- **Simplicidad en el diseño:** Una de las partes más importantes de la filosofía XP es la simplicidad en todos los aspectos. Se considera que un diseño sencillo se logra más rápido y se implementa en menos tiempo, por lo cual esto es lo que se busca. En XP se prefiere tener una descripción del sistema o parte de él, en lugar de una serie de complejos diagramas que probablemente tomen más tiempo y sean menos instructivos:

- **Metáfora del sistema:** “Se trata de plasmar la arquitectura de sistema en una “historia” con la cual se le dé al grupo de desarrollo una misma visión sobre el proyecto además de brindarles un primer vistazo muy completo a los nuevos integrantes del grupo para hacer su adaptación más rápida. Es muy importante dentro del desarrollo de la metáfora darle nombres adecuados a todos los elementos del sistema constantemente, y que estos correspondan a un sistema de nombres consistente”. [23]

[23] Newkirk, James. La pogramacion extrema en la práctica. Editorial Eddison Wesley. 2002. P. 19.

- **Tarjetas de clase, responsabilidad, colaboración (CRC cards):** La principal funcionalidad que tienen estas, es ayudar a dejar el pensamiento procedimental para incorporarse al enfoque orientado a objetos. Cada tarjeta representa una clase con su nombre en la parte superior, en la sección inferior izquierda están descritas las responsabilidades y a la derecha las clases que le sirven de soporte.

- **Soluciones puntuales (Spike Solution):** "En muchas ocasiones los equipos de desarrollo se enfrentan a requerimientos de los clientes (en este caso historias de usuario) los cuales generan problemas desde el punto de vista del diseño o la implementación"[24]. Spike Solution, es una herramienta de XP para abordar este inconveniente. Se trata de una pequeña aplicación completamente desconectada del proyecto con la cual se intenta explorar el problema y propone una solución potencial.

- **No solucionar antes de tiempo:** Los desarrolladores tienden a predecir las necesidades futuras e implementarlas antes. Según mediciones, esta es una práctica ineficiente, concluyendo que tan solo el 10% de las soluciones para el futuro son utilizadas, desperdiciando tiempo de desarrollo y complicando el diseño innecesariamente. En XP sólo se analiza lo que se desarrollará en la iteración actual, olvidando por completo cualquier necesidad que se pueda presentar en el futuro, lo que supone uno de los preceptos más radicales de la programación extrema.

- **Refactorización (Refactoring):** Como se trató al principio de este apartado, el diseño es una tarea permanente durante toda la vida del proyecto y la refactorización concreta este concepto. Como en cualquier metodología tradicional en XP se inicia el proceso de desarrollo con un diseño inicial.

9.2.3 Codificación

En esta etapa se desarrollará la herramienta, definiendo estándares y parámetros de codificación. El desarrollo se realiza por parejas, implementando una funcionalidad a la vez, y en un orden específico, de acuerdo a la criticidad de los requerimientos. Se usará el paradigma de programación Orientada a Objetos.

"La codificación es un proceso que se realiza en forma paralela con el diseño y la cual está sujeta a varias observaciones por parte de XP consideradas

[24] Newkirk, James. La pogramacion extrema en la práctica. Editorial Eddison Wesley. 2002. P. 21.

controversiales por algunos expertos tales como la rotación de los programadores o la programación en parejas".[25]

- **Cliente siempre presente:** Uno de los requerimientos de XP es que el cliente esté siempre disponible. No solamente para solucionar las dudas del grupo de desarrollo, debería ser parte de éste.

- **Codificar primero la prueba:** Cuando se crea primero una prueba, se ahorra mucho tiempo elaborando el código que la haga pasar, siendo menor el tiempo de hacer ambos procesos que crear el código solamente. Una de las ventajas de crear una prueba antes que el código es que permite identificar los requerimientos de dicho código. En otras palabras, al escribir primero las pruebas se encuentran de una forma más sencilla y con mayor claridad todos los casos especiales que debe considerar el código a implementar.

- **Programación en parejas:** Todo el código debe ser creado por parejas de programadores sentados ambos frente a un único computador lo que en principio representa una reducción de un 50% en productividad, sin embargo, según XP no es tal la pérdida. Se entiende que no hay mucha diferencia, en lo que a la cantidad se refiere. Cuando se trabaja en parejas se obtiene un diseño de mejor calidad y un código más organizado y con menores errores que si se trabajase solo, además de la ventaja que representa contar con un compañero que ayude a solucionar inconvenientes en tiempo de codificación, los cuales se presentan con mucha frecuencia.

- **Integración secuencial:** Uno de los mayores inconvenientes presentados en proyectos de software tiene que ver con la integración, sobre todo si todos los programadores son dueños de todo el código. Para saldar este problema han surgido muchos mecanismos, como darle propiedad de determinadas clases a algunos desarrolladores, los cuales son los responsables de mantenerlas actualizadas y consistentes.

- **Integraciones frecuentes:** Se deben hacer integraciones cada pocas horas y siempre que sea posible no debe transcurrir más un día entre una integración y otra. De esta forma se garantiza surjan problemas como que un programador trabaje sobre versiones obsoletas de alguna clase.

- **Estándares y propiedad colectiva del código:** Así como se recomienda que la programación se haga siempre en parejas ubicadas en un único computador, también se aconseja que estas se vayan rotando no solo de compañero sino de partes del proyecto a implementar, con el fin de que se logre tener una propiedad colectiva del código. Uno de los principales motivos

[25] Newkirk, James. La pogramacion extrema en la práctica. Editorial Eddison Wesley. 2002. P. 24.

por los que se promueve esta práctica dentro de la programación extrema es la posibilidad que brinda de evitar los cuellos de botella.

9.2.4 Pruebas

Se ejecutan las pruebas unitarias, determinando desde los resultados obtenidos si se puede continuar iterando o no. Posteriormente, se ejecuta el plan de pruebas final. XP enfatiza mucho los aspectos relacionados con las pruebas, clasificándolas en diferentes tipos y funcionalidades específicas, indicando quién, cuándo y cómo deben ser implementadas y ejecutadas. Del buen uso de las pruebas depende el éxito de otras prácticas, tales como la propiedad colectiva del código y la refactorización.

- **Pruebas unitarias:** Estas pruebas se aplican a todos los métodos no triviales de todas las clases del proyecto con la condición que no se liberará ninguna clase que no tenga asociada su correspondiente paquete de pruebas. Uno de los elementos más importantes en estas es que idealmente deben ser construidas antes que los métodos mismos, permitiéndole al programador tener máxima claridad sobre lo que va a programar antes de hacerlo, así como conocer cada uno de los casos de prueba que deberá pasar, lo que optimizará su trabajo y su código será de mejor calidad.

  Pruebas de aceptación: Las pruebas de aceptación, también llamadas pruebas funcionales son supervisadas por el cliente basándose en los requerimientos tomados de las historias de usuario. En todas las iteraciones, cada una de las historias de usuario seleccionadas por el cliente deberá tener una o más pruebas de aceptación, de las cuales deberán determinar los casos de prueba e identificar los errores que serán corregidos.

  Las pruebas de aceptación son pruebas de caja negra, que representan un resultado esperado de determinada transacción con el sistema. Para que una historia de usuario se considere aprobada, deberá pasar todas las pruebas de aceptación elaboradas para dicha historia.

- **Cuando se encuentra un error:** Al momento de encontrar un error debe escribirse una prueba antes de intentar corregirlo. De esta forma tanto el cliente logrará tener completamente claro cuál fue y dónde se encontraba el mismo como el equipo de desarrollo podrá enfocar mejor sus esfuerzos para solucionarlo. Por otro lado se logrará evitar volver a cometerlo.

9.3 MARCO LEGAL Y NORMATIVO

La entidad Universidad Libre Seccional Pereira, se rige bajo el decreto 2649 de 1993 norma que regula la profesión contable:

Decreto 2649 de 1993: “Fue expedido el 29 de diciembre de 1993, por medio del cual se reglamentan las normas del Código de Comercio en materia de contabilidad y se fijan los principios y normas contables generalmente aceptadas en Colombia”[26].

Conjunto de postulados, conceptos y limitaciones que fundamentan y circunscriben la información contable.

Código de Comercio: “Delimita la materia mercantil en función de los actos calificador legalmente como actor de comercio. Se ocupa de regular las relaciones entre los fundamentos de los agentes del comercio sus relaciones y los objetos del comercio. Además cubre temas importantes como el trabajo en el área contable, jurídica, de patrimonio, etc.”[27]

[26] LEGIS. Plan Único de Cuenta. 8Ed Bogotá D.C. 2011.
[27] Código de Comercio Bogotá D.C. Ediciones Lito imprerio 1998.

10 LEVANTAMIENTO DE REQUERIMIENTOS

Con la finalidad de comprender todo el proceso y descubrir las necesidades que enmarca la gestión de créditos educativos directos y cartera, otorgados por la universidad a los estudiantes; y recolectar toda la información necesaria para el desarrollo del proyecto, se realizó una entrevista inicial con la coordinadora del área contable junto con el empleado responsable de la gestión de dicha cartera. Para poder establecer los requisitos funcionales y no funcionales del prototipo.

Descripción del servicio: El solicitante se debe acercar a la oficina de tesorería y cartera y solicitar la información de las condiciones para acceder al crédito directo.

- El solicitante debe diligenciar el paquete de crédito directo (Carta de Instrucción ST-GF-03-P-01-F07, Pagaré ST-GF-03-P-01-F06, Datos del Solicitante e información del codeudor ST-GF-03-P-01-F05) y documentos que soportan el crédito (según los requisitos establecidos por la Universidad).

- El solicitante del crédito directo debe estar a paz y salvo y presentar un buen comportamiento en créditos anteriores si los hubiera tenido.

- Como el crédito es inferior al 80%, el estudiante deberá pagar el excedente con un nuevo recibo, que será expedido en cartera.

- Una vez cancelado el excedente por el estudiante, esté debe dirigirse a cartera para terminar el proceso de legalización de la matrícula.

10.1 REQUERIMIENTOS FUNCIONALES

Los requerimientos funcionales del sistema se enfocarán principalmente en lo que el sistema debe hacer.

Los requerimientos funcionales pueden clasificarse por medio de tres categorías:

- **Evidentes:** Debe realizarse y el usuario debe estar consciente de que se va a realizar.

- **Ocultas:** Debe realizarse pero debe ser transparente para el usuario.

- **Opcional:** Añadirlas no implica un incremento de tiempo elevado o afectar otros requerimientos funcionales.

Tabla 1. Requerimientos funcionales.

FORMATO DE TRABAJO DE CAMPO		
CODIGO	**TIPO**	**FUENTE**
RF	Observación Directa	Dora Patricia Ospina Parra
ANALISTA	**FECHA**	**TIEMPO**
Mauricio Ramírez Giraldo	15 Junio 2013	3 Horas
Versión	1.0	

N°	**Requerimiento Funcional**	**Categoría**
1	Se debe contar con un control de acceso a la aplicación y dos perfiles usuario, uno digitador con acceso a solo consultas y entrada de datos y con restricción a ejecución de procesos críticos. Y un usuario administrador con acceso a gestionar parámetros y procesos sensibles para la funcionalidad del sistema.	Evidente
2	La aplicación debe poder acceder a una base de datos externa.	Ocultas
3	Todos los usuarios pueden consultar la información relevante de los créditos educativos y los respectivos datos del estudiante o deudor. Pero no pueden modificar ningún dato.	Evidente
4	Con la información de la fecha de aprobación del crédito y la fecha actual se debe calcular los intereses generados entre estas dos fechas. Y realizar la respectiva liquidación de la cartera. Calculando el saldo neto con la operación: valor del préstamo menos la sumatoria total de los pagos (recibos de caja) o abonos. Y calcular el saldo total con la operación: saldo neto más los intereses causados a la fecha.	Evidente
5	La aplicación debe calcular automáticamente los días de diferencia entre la fecha de aprobación del crédito y una fecha de liquidación de la cartera o fecha de corte.	Ocultas

6	La fecha de liquidación de la cartera o fecha de corte se debe poder seleccionarse variablemente.	Evidente
7	Los intereses generados según los días de diferencia, se deben calcular automáticamente.	Ocultas
8	Con la información de los pagos (Recibos de caja) o abonos al crédito, se debe poder liquidar los intereses del crédito y el saldo a la fecha. Y mostrar la operación resultante en una tabla tipo amortización.	Evidente
9	La aplicación debe permitir crear y almacenar los datos personales de un tercero como codeudor y poder asociarlo a un crédito y estudiante/deudor. Y posteriormente poderlos consultar.	Evidente
10	La aplicación debe permitir adicionar, eliminar y modificar los datos personales del codeudor.	Evidente
11	El usuario administrador puede administrar el perfil de los usuarios del sistema. Adicionar, modificar, eliminar e inhabilitar.	Evidente
12	La aplicación debe poder visualizar en pantalla y generar la salida por impresora la respectiva liquidación de la cartera, Liquidación de intereses.	Evidente
13	En la aplicación se debe poder parametrizar algunos valores para que no sean constantes: Porcentaje de intereses, entre otros. Donde solo se puedan modificar los valores de dichos parámetros y no se pueda eliminar.	Evidentes
14	El sistema debe permitir salir adecuadamente del sistema	Ocultas

Fuente: El autor.

10.2 Requerimientos no-funcionales

Describen aspectos del sistema que son visibles por el usuario que no incluyen una relación directa con el comportamiento funcional del sistema. Los requerimientos no funcionales incluyen restricciones. Algunos criterios para clasificar a los requerimientos no-funcionales es por medio del modelo FURPS+ (por sus siglas en inglés): Funcionalidad, Facilidad de Uso, Confiabilidad, Rendimiento, Soporte y el símbolo de + que indica requerimientos adicionales

como: Restricciones de diseño, Requisitos de Implementación, Requisitos de Interface y Requisitos Físicos.

Tabla 2. Requerimiento no-funcional. Usabilidad

Requerimiento no-funcional	Usabilidad
Versión	1.0
Sugerido por	Desarrolladores y usuarios del sistema
Descripción	El software debe ser intuitivo, entendible y de fácil manejo para cualquier usuario del sistema sin conocimientos avanzados de contabilidad o sistemas.
	El software debe tener una interface del sistema amigable y de colores estándar tipo profesional.

Fuente: El autor.

Tabla 3. Requerimiento no-funcional. Portabilidad

Requerimiento no-funcional	Portabilidad
Versión	1.0
Sugerido por	Desarrolladores
Descripción	El software debe garantizar que pueda ser ejecutado en sistemas operativos Windows en arquitecturas a 32 y 64bits.

Fuente: El autor.

Tabla 4. Requerimiento no-funcional. Seguridad

Requerimiento no-funcional	Confiabilidad y Seguridad
Versión	1.0

Sugerido por	Desarrolladores y usuarios del sistema
Descripción	Tanto la aplicación como la base de datos, deben tener los estándares mínimos de seguridad e integridad para evitar que los datos puedan ser alterado o destruidos por personas no autorizadas.
	Tanto la aplicación como la base de datos, deben ser confiables para garantizar el acceso oportuno a la información.

Fuente: El autor.

Tabla 5. Requerimientos no-funcionales. Rendimiento

Requerimiento no-funcional	Rendimiento
Versión	1.0
Sugerido por	Desarrolladores y usuarios del sistema
Descripción	La aplicación debe responder rápidamente a las solicitudes y peticiones de los usuarios, disminuyendo los tiempos que se empleaban en los procesos antes de ser sistematizados.

Fuente: El autor.

Tabla 6. Requerimientos no-funcionales. Implementación

Requerimiento no-funcional	Implementación
Versión	1.0
Sugerido por	Desarrolladores
Descripción	Por restricciones de portabilidad de la plataforma utilizada para el desarrollo de la aplicación .NET y ORACLE XE, se debe implementar únicamente

	en sistemas operativos Windows.

Fuente: El autor.

Tabla 7. Requerimiento no-funcional. Soporte

Requerimiento no-funcional	Soporte
Versión	1.0
Sugerido por	Desarrolladores
Descripción	Documentar con base en estándares cada una de las partes que comprenden el software: código fuente y base de datos, con el fin de Facilitar labores de mantenimiento, capacitación o mejoras futuras.

Fuente: El autor.

11 ANALISIS DE REQUERIMIENTOS

11.1 DIAGRAMAS DE CASOS DE USO

Los diagramas de casos de uso nos sirven para especificar la comunicación y el comportamiento del sistema, mediante el análisis de su interacción con los usuarios y/u otros sistemas.

11.1.1 Generalización de los actores.

Como una primera aproximación identificamos a los actores que interactúan con el sistema. Representando de forma generalizada la forma en como un Cliente (Actor) opera con el sistema en desarrollo.

Ilustración 1. Diagrama de generalización de los actores.

Fuente: El autor.

11.1.2 Diagrama Contextual.

Un diagrama contextual está representado por los actores y las acciones por las cuales el programa o prototipo tendrá que satisfacer las necesidades específicas del proyectó o cliente.

Ilustración 2. Diagrama de casos de uso contextual.

Fuente: El autor.

11.2 ANALISIS DE CASOS DE USO

En la descripción de los Casos de uso se detalla una secuencia de interacciones que se desarrollarán entre un sistema y sus actores en respuesta a un evento que inicia un actor principal sobre el mismo sistema

11.2.1 Caso de uso Ingresar al Sistema

11.2.1.1 Descripción del caso de uso

Tabla 8. Descripción caso de uso ingresar al sistema.

CASO DE USO EXTENDIDO			
Caso de uso	Ingresar al Sistema		
Actores	Usuario y Administrador		
Propósito	Dar un nivel de seguridad a la aplicación para evitar ingresos no autorizados		
Descripción	Al ejecutar la aplicación el sistema solicita el ingreso de las credenciales de acceso o autorización, como nombre usuario único y su respectiva contraseña de acceso, permitiendo determinar el rol del actor en el sistema y cargar el perfil.		
Precondiciones	Los usuario deben estar creados en la base de datos		
Tipo	Esencial		
Referencias Cruzadas	RF-1		
Flujo normal de los eventos			
Paso	**Acción del actor**	**Paso**	**Respuesta del sistema**
1	El caso de uso comienza cuando el usuario ejecuta o abre la aplicación.		
2	Digita el nombre de usuario asignado por el administrador e		

	ingresa su respectiva contraseña de autenticación.		
3	Selecciona la opción “VALIDAR”.	4	Confronta y valida los datos digitados por el usuario, con los que están almacenados en los perfiles de usuarios.
		5	Cierra la pantalla de autenticación y deshabilita la pantalla y los controles principales de la aplicación.
Flujo alterno a los eventos			
Paso	**Acción del actor**	**Paso**	**Respuesta del sistema**
4.1	Intento de ingreso en más de tres ocasiones.	4.2	Cierra la pantalla de autenticación y cierra la aplicación.
Excepciones			
Paso 4	Al confrontar los datos con los almacenados en la base de datos y no encuentra una coincidencia; el sistema muestra un mensaje “Los datos de acceso Usuario y/o Contraseña no tienen acceso al sistema. Verifique e inténtelo de nuevo.”		
Paso 4.1	Al detectar el sistema un intento fallido mayor a tres, muestra un mensaje de alerta. “Ha excesito el límite de intentos de acceso. Por seguridad la aplicación se cerrará”		

Fuente: El autor.

11.2.1.2 Diagrama de secuencia ingresar al sistema

Ilustración 3. Diagrama de secuencia ingresar al sistema

Fuente: El autor.

11.2.1.3 Diagrama de actividades Ingresar al Sistema

Ilustración 4. Diagrama de actividades ingresar al sistema.

Fuente: El autor.

11.2.2 Caso de uso Liquidar Cartera

11.2.2.1 Descripción del caso de uso

Tabla 9. Descripción caso de uso liquidar cartera.

CASO DE USO EXTENDIDO			
Caso de uso	Liquidación de Cartera		
Actores	Usuario		
Propósito	Conocer el saldo total a pagar con los intereses incurridos en por el no pago del crédito educativo. E imprimir el resultado.		
Descripción	Se hace una búsqueda por número de documento de identificación y número de crédito, para conocer el estado del mismo, el total de abonos realizados y el saldo actual en cartera. Calculando de forma automática el número de días incurridos en mora, proporcionando la opción de seleccionar una fecha de corte para la liquidar la cartera. Posteriormente se da la opción de imprimir el reporte.		
Precondiciones	La información del deudor y del crédito debe estar almacenada en la base de datos.		
Tipo	Esencial		
Referencias Cruzadas	RF-3, RF-4, RF5, RF-6, RF-7, RF-12		
Flujo normal de los eventos			
Paso	**Acción del actor**	**Paso**	**Respuesta del sistema**
1	El caso de uso comienza cuando el usuario le da clic a la opción “LIQUIDAR CARTERA” .		
2	En el campo correspondiente digita el número de identificación del estudiante/deudor.		

3	Selecciona la opción “CONSULTAR”.	4	Buscar créditos relacionados con el número de documento digitado y muestra en una lista desplegable.
5	En la opción “CREDITO” lista desplegable, selecciona el número de crédito a liquidar.	6	Busca todos los datos referentes al crédito y los muestra en pantalla.
7	Selecciona la opción de “FECHA DE LIQUIDACIÓN”.		
8	Selecciona la opción de “LIQUIDAR”	9	Efectúa los cálculos: días liquidados, interés diario, total de abonos, saldo neto, el total de intereses y el saldo total a la de la deuda.
10	Selecciona la opción de “IMPRIMIR”	11	Abre el reporte en pantalla y da la opción de enviar a impresora.

Flujo alterno a los eventos			
Paso	**Acción del actor**	**Paso**	**Respuesta del sistema**
11.1	Selecciona la opción de exportar a Excel, PDF entre otros formatos.	11.2	Exporta el reporte a la opción selecciona y lo visualiza en pantalla.

Excepciones	
Paso 4	Al confrontar los datos con los almacenados en la base de datos y no encuentra una coincidencia; Se muestra un mensaje informativo: “El dato de búsqueda no arrojo ningún resultado”.
Paso 4.1	Si el con el número de identificación no encuentra ningún crédito asociado, se muestra un mensaje “El estudiante/deudor no tiene asignado ningún crédito educativo”
Paso 7	Si la fecha de liquidación es menor a la fecha de aprobación del crédito, se muestra un mensaje de alerta, en pantalla: “Fecha de liquidación es menor a la fecha de aprobación del crédito”

Fuente: El autor.

11.2.2.2 Diagrama de secuencia Liquidar Cartera

Ilustración 5. Diagrama de secuencia liquidar cartera.

Fuente. El autor.

11.2.2.3 Diagrama de actividades Liquidar Cartera

Ilustración 6. Diagrama de actividades liquidar cartera

Fuente: El autor.

11.2.3 Caso de uso liquidar intereses

11.2.3.1 Descripción del caso de uso

Tabla 10. Descripción caso de uso liquidar intereses.

CASO DE USO EXTENDIDO			
Caso de uso	Liquidar Intereses		
Actores	Usuario		
Propósito	Con la información de los pagos (Recibos de caja) o abonos al crédito, se debe poder liquidar los intereses del crédito y el saldo a la fecha. Y mostrar la operación resultante en una tabla tipo amortización.		
Descripción	Se hace una búsqueda por número de documento de identificación y número de crédito, para conocer el estado del mismo. Con los dato del número de crédito se consulta todos los pagos o abonos realizados a dicho crédito para diseñar una tabla tipo amortización con el detalle del recibo de pago. Con estos datos realizar un cálculo automático de cuanto se generó por concepto de intereses y cuanto se descuenta al capital. Mostrando el saldo final de la operación representado en saldo actual del crédito.		
Precondiciones	La información del deudor y del crédito debe estar almacenada en la base de datos.		
Tipo	Esencial		
Referencias Cruzadas	RF-8, , RF-12		
Flujo normal de los eventos			
Paso	**Acción del actor**	**Paso**	**Respuesta del sistema**
1	El caso de uso comienza cuando el usuario le da clic a la opción “LIQUIDAR INTERESES” del menú principal de la aplicación.		
2	En el campo correspondiente digita el número de identificación del		

	estudiante/deudor.		
3	Selecciona la opción “CONSULTAR”.	4	Buscar todos los créditos relacionados con el número de documento digitado y muestra en una lista desplegable.
5	En la opción “CREDITO” lista desplegable selecciona el número de crédito a liquidar intereses.	6	Busca todos los datos referentes al crédito y los muestra en pantalla.
		7	Busca datos todos los recibos de pago efectuados al crédito seleccionado y realiza el cálculo de intereses a la fecha de pago contra la fecha aprobación, y realizar la respectiva operación de abono a capital, intereses y saldo final. Muestra la información en una tabla tipo amortización.
8	Selecciona la opción de “IMPRIMIR”		Abre el reporte en pantalla y da la opción de enviar a impresora.

Flujo alterno a los eventos			
Paso	**Acción del actor**	**Paso**	**Respuesta del sistema**
9.1	Selecciona la opción de exportar a Excel, PDF entre otros formatos.	9.2	Exporta el reporte a la opción selecciona y lo visualiza en pantalla.

Excepciones	
Paso 4	Al confrontar los datos con los almacenados en la base de datos y no encuentra una coincidencia; Se muestra un mensaje informativo: “El dato de búsqueda no arrojo ningún resultado”.
Paso 4.1	Si el con el número de identificación no encuentra ningún crédito asociado, se muestra un mensaje “El estudiante/deudor no tiene asignado ningún crédito educativo”
Paso 7	Si no encuentra ningún recibo de caja relacionado al número del crédito muestra un mensaje informativo: “El crédito actual no tiene ningún pago o abono realizado”

Fuente: El autor.

11.2.3.2 Diagrama de secuencia liquidar Intereses

Ilustración 7. Diagrama de secuencia liquidar intereses.

Fuente: El autor.

11.2.3.3 Diagrama de actividades liquidar intereses

Ilustración 8. Diagrama de actividades liquidar intereses.

Fuente: El autor.

11.2.4 Caso de uso Administrar Codeudores

11.2.4.1 Descripción del caso de uso

Tabla 11. Descripción del caso de uso administrar codeudores.

CASO DE USO EXTENDIDO			
Caso de uso	Administración de codeudores		
Actores	Usuario		
Propósito	Tener una utilidad para gestiona y/o administrar los datos personales de los codeudores de cada crédito educativo asignado a un estudiante/deudor.		
Descripción	Al abrir el modulo se hace la búsqueda por documento de identidad y número del crédito. Se solicita al usuario seleccionar el crédito al cual va asociar el codeudor y posteriormente se la da la opción de ingresar los diferentes datos personales del codeudor, modificar datos ya existentes o eliminar un codeudor de un crédito.		
Precondiciones	Debe de estar almacenado los datos del estudiante/deudor y el respectivo crédito.		
	Para modificar o eliminar un codeudor debe estar previamente almacenado en la base de datos.		
Tipo	Esencial		
Referencias Cruzadas	RF-9, RF-10, RF-12		
Flujo normal de los eventos			
Paso	**Acción del actor**	**Paso**	**Respuesta del sistema**
1	El caso de uso comienza cuando el usuario le da clic a la opción “ADMINISTRAR CODEDUDORES” del menú principal de la aplicación.		
2	Selecciona la opción de búsqueda “CREDITO” o “ESTUDIANTE/DEUDOR”		

3	En el campo correspondiente digita el número de identificación del estudiante/deudor o en su efecto el número del crédito.		
4	Selecciona la opción “CONSULTAR”.	5	Valida la opción de búsqueda si es por número de crédito o número de identificación del estudiante/deudor.
		6	Busca todos los datos referentes al crédito y estudiante/deudor y los muestra en pantalla de solo lectura.
7	En las opciones correspondiente digita los datos personales del cliente		
8	Selecciona la opción ”INGRESAR”	9	Inserta todos los datos digitados a la base de datos y muestra un mensaje con el resultado de la inserción.
Flujo alterno a los eventos			
Paso	**Acción del actor**	**Paso**	**Respuesta del sistema**
5.1	Selecciona la opción de búsqueda ”CRÉDITO”	5.1.2	Muestra en pantalla toda la información del crédito, estudiante/deudor y codeudor asociado al crédito consultado.
5.2	Selecciona la opción de búsqueda ”ESTUDIANTE/CODEUDOR”	5.2.1	Busca créditos relacionados con el dato de búsqueda y muestra y los muestra en pantalla una lista desplegable.
5.2.2	En la opción “CREDITO” lista desplegable selecciona el número de crédito al cual desea asignarle un codeudor		
9.1	Selecciona la opción ”MODIFICAR”	9.1.1	Se habilita los campos con los datos del codeudor, para dejarlos editar.

9.1.2	Selecciona la opción “GUARDAR”	9.1.3	Hace una actualización de los datos del codeudor.
9.2	Selecciona la opción “CANCELAR”	9.2.1	Se deshabilita los campos con los datos del codeudor, para no dejarlos editar.
9.3	Selecciona la opción ”ELIMINAR”	9.3.1	Pide mensaje de confirmación de eliminación del codeudor.
Excepciones			
Paso 7	Al confrontar los datos de búsqueda con los almacenados en la base de datos y no encuentra una coincidencia; Se muestra un mensaje informativo: “El dato de búsqueda no arrojo ningún resultado”.		
Paso 5.2	Si el con el número de identificación no encuentra ningún crédito asociado, se muestra un mensaje “El estudiante/deudor no tiene asignado ningún crédito educativo”		
Paso 9	Si todos los campos que contienen los datos personales no están diligenciados, muestra un mensaje de alerta “Faltan campos sin diligenciar. Todos los datos son obligatorios. Revise e inténtelo de nuevo”. No hace la inserción.		

Fuente: El autor.

11.2.4.2 Diagrama de secuencia administrar codeudores

Ilustración 9. Diagrama de secuencia administrar codeudores.

Fuente: El autor.

11.2.4.3 Diagrama de actividades Administrar Codeudores

Ilustración 10. Diagrama de actividades administrar codeudores.

Fuente: El autor.

11.2.5 Caso de uso administrar usuario

11.2.5.1 Descripción del caso de uso

Tabla 12. Descripción caso de uso administrar usuarios.

CASO DE USO EXTENDIDO			
Caso de uso	Administrar usuarios		
Actores	Administrador		
Propósito	Tener una utilidad para gestiona y/o administrar los usuarios y si nivel o perfil de acceso al sistema.		
Descripción	Al abrir el modulo se listan todos los usuarios registrador en el sistema y sus diferentes datos. Brinda la opción de modificar, adicionar e inhabilitar un usuario.		
Precondiciones	Tener acceso al sistema como administrador.		
Tipo	Esencial		
Referencias Cruzadas	RF-11		
Flujo normal de los eventos			
Paso	**Acción del actor**	**Paso**	**Respuesta del sistema**
1	El caso de uso comienza cuando el usuario le da clic a la opción “ADMINISTRAR USUARIOS” del menú principal de la aplicación.		
2	Selecciona la opción “CREAR”	3	Inserta el nuevo usuario a la base de datos.
4	Selecciona la opción “MODIFICAR”	5	Habilita los campos con los datos del usuario para su edición.
5	Selecciona la opción “GUARDAR”	6	Hace la modificación en el registro específico del usuario del sistema.
7	Selecciona la opción “INABILITAR”	8	Habilita el usuario para poderse

	o “HABILITAR”		autenticar en la aplicación.
Flujo alterno a los eventos			
Paso	**Acción del actor**	**Paso**	**Respuesta del sistema**
		3.1	Valida que el usuario no exista en la base de datos.
Excepciones			
Paso 3.1	Si el usuario existe muestra un mensaje de error: “El nombre de usuario ya existe en la base de datos. Digite otro nombre diferente e inténtelo de nuevo”		
Paso 2 y 6	Si el usuario no ha ingresado todos los campos requeridos muestra un mensaje de advertencia: “Faltan campos por diligenciar correctamente. Revise e inténtelo de nuevo.”		

Fuente: El autor.

11.2.5.2 Diagrama de secuencia Administrar Usuarios

Ilustración 11. Diagrama de secuencia administrar usuarios.

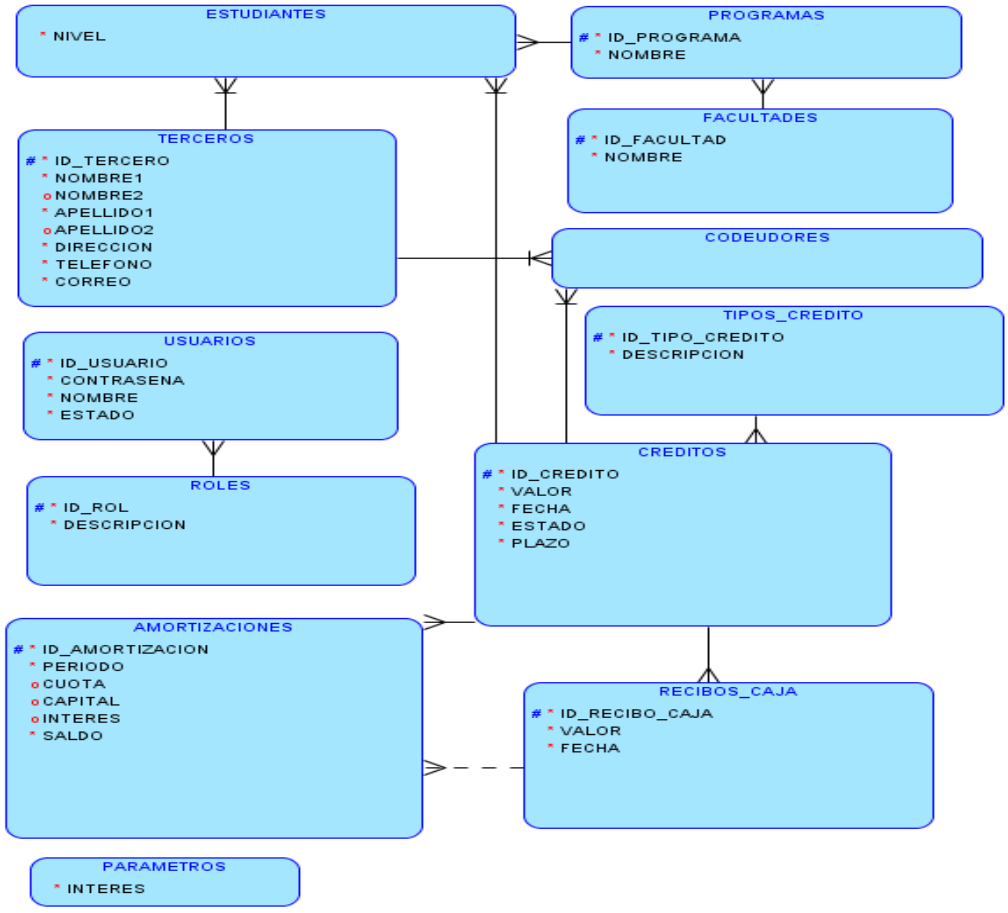
Fuente: El autor.

12 DISEÑO DEL PROTOTIPO DE LA APLICACIÓN

Para el diseño del prototipo de la aplicación a desarrollar se realiza un modelo de base de datos relacional que tiene como finalidad poder almacenar información relevante para cubrir en su totalidad los requerimientos funcionales de la aplicación: Administración de codeudores, control de usuarios del sistema, tabla tipo amortización para liquidación de intereses y la liquidación de cartera; y los parámetros generales de la aplicación. Simulación de tablas maestras: créditos, recibos de caja, terceros, programas y facultades.

12.1 Modelo lógico

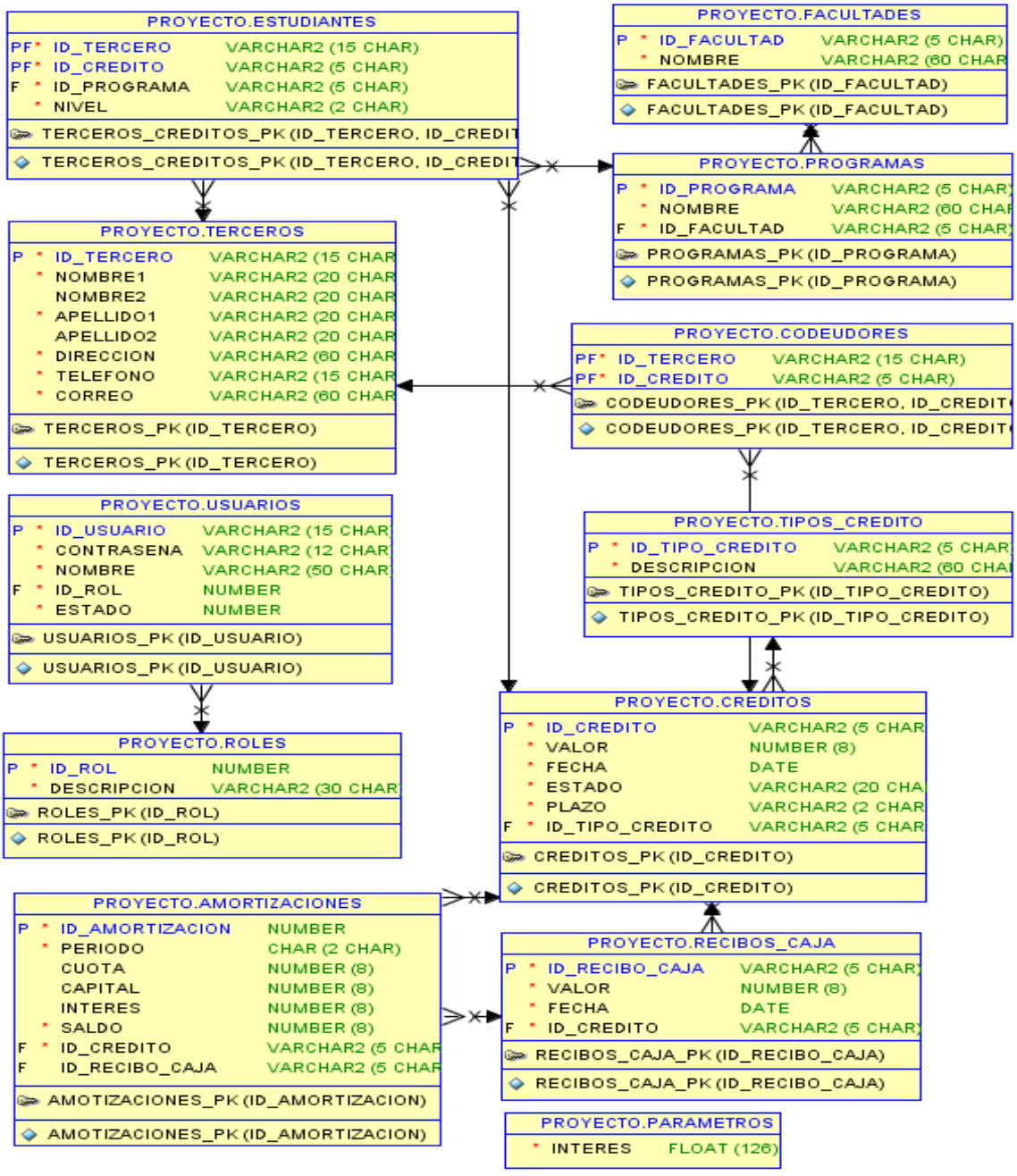
Ilustración 12. Modelo lógico de la base de datos.



Fuente: El autor.

12.2 Modelo Relacional E/R

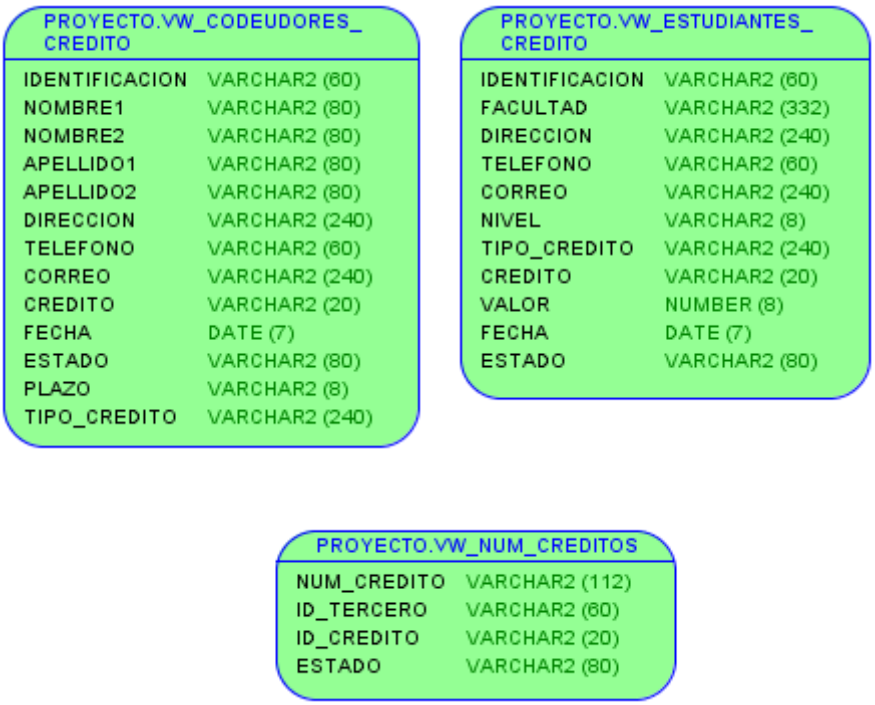
Ilustración 13. Modelo relacional E/R.



Fuente: El autor.

12.3 Modelo de vistas

Ilustración 14. Modelo de vistas de sistema.



Fuente: El autor.

12.4 MODELO FÍSICO (Diccionario de datos)

A continuación se describe el modelo físico con el diccionario de la base de datos y la descripción detallada de la estructura de cada tabla diseñas para almacenar la diferente información relevante y necesaria para la aplicación; se utilizaran las siguientes siglas en los parámetros de la tabla:

- Campo NULO
  Con los valores N=NO, S=SI.
- Campo CLAVE
  Con los valores PK=Primary Key (de las siglas del inglés) llave primaria, FK=Foreing Key (de las siglas del inglés) Llave Foranea.

12.4.1 **Tabla Terceros.** Tabla de la base de datos para almacenar los datos personales de todas las personas involucradas en el proceso, bien sea como deudor o codeudor.

Tabla 13. Diccionario de datos tabla Terceros.

TERCEROS				
CAMPO	TIPO(TAMAÑO)	NULO	CLAVE	DESCRIPCION
ID_TERCERO	Varchar2(15)	N	PK	Número del documento de identificación
NOMBRE1	Varchar2(20)	N		Primer nombre
NOMBRE2	Varchar2(20)	S		Segundo nombre
APELLIDO1	Varchar2(20)	N		Primer apellido
APELLIDO2	Varchar2(20)	S		Segundo apellido
DIRECCION	Varchar2(60)	N		Dirección de la residencia
TELEFONO	Varchar2(15)	N		Teléfono de contacto
CORREO	Varchar2(60)	N		Dirección de correo electrónico

Fuente: El autor.

12.4.1.1 **Código de script SQL.** CREATE TABLE TERCEROS (ID_TERCERO VARCHAR2 (15 CHAR) NOT NULL, NOMBRE1 VARCHAR2 (20 CHAR) NOT NULL, NOMBRE2 VARCHAR2 (20 CHAR), APELLIDO1 VARCHAR2 (20 CHAR) NOT NULL, APELLIDO2 VARCHAR2 (20 CHAR), DIRECCION VARCHAR2 (60 CHAR) NOT NULL, TELEFONO VARCHAR2 (15 CHAR) NOT NULL, CORREO VARCHAR2 (60 CHAR) NOT NULL, CONSTRAINT TERCEROS_PK PRIMARY KEY (ID_TERCERO) ENABLE);

12.4.2 **Tabla Estudiantes.** Tabla de la base de datos para almacenar el dato que identifica a un tercero en el rol de estudiante y su respectivo número de crédito.

Tabla 14. Diccionario de datos. Tabla Estudiantes.

ESTUDIANTES				
CAMPO	TIPO(TAMAÑO)	NULO	CLAVE	DESCRIPCION
ID_TERCERO	Varchar2(15)	N	PK	Número del documento de identificación
ID_CREDITO	Varchar2(5)	N	PK	Identificador único del crédito
ID_PROGRAMA	Varchar2(5)	N	FK	Identificador único del programa académico
NIVEL	Varchar2(2)	N		Nivel académico. En años o meses según programa

Fuente: El autor.

12.4.2.1 **Código de script SQL.** CREATE TABLE ESTUDIANTES (ID_TERCERO VARCHAR2 (15 CHAR) NOT NULL, ID_CREDITO VARCHAR2 (5 CHAR) NOT NULL, ID_PROGRAMA VARCHAR2 (5 CHAR) NOT NULL, NIVEL VARCHAR2 (2 CHAR) NOT NULL, CONSTRAINT TERCEROS_CREDITOS_PK PRIMARY KEY (ID_TERCERO, ID_CREDITO) ENABLE). ALTER TABLE ESTUDIANTES ADD CONSTRAINT ESTUDIANTES_PROGRAMA FOREIGN KEY (ID_PROGRAMA) REFERENCES PROGRAMAS (ID_PROGRAMA) ON DELETE CASCADE ENABLE; ALTER TABLE ESTUDIANTES ADD CONSTRAINT TERCEROS_CREDITOS_CREDITOS_FK FOREIGN KEY (ID_CREDITO) REFERENCES CREDITOS (ID_CREDITO) ON DELETE CASCADE ENABLE; ALTER TABLE ESTUDIANTES ADD CONSTRAINT TERCEROS_CREDITOS_TERCERO_FK FOREIGN KEY (ID_TERCERO) REFERENCES TERCEROS (ID_TERCERO) ON DELETE CASCADE ENABLE;

12.4.3 **Tabla Codeudores**. Tabla de la base de datos para almacenar el dato que identifica a un tercero el en rol de codeudor y su respectivo crédito.

Tabla 15. Diccionario de datos. Tabla Codeudores.

CODEUDORES				
CAMPO	TIPO(TAMAÑO)	NULO	CLAVE	DESCRIPCION
ID_TERCERO	Varchar2(15)	N	PK	Número del documento de identificación
ID_CREDITO	Varchar2(5)	N	PK	Identificador único del crédito

Fuente: El autor.

12.4.3.1 **Código de script SQL.** CREATE TABLE CODEUDORES (ID_TERCERO VARCHAR2 (15 CHAR) NOT NULL, ID_CREDITO VARCHAR2 (5 CHAR) NOT NULL, CONSTRAINT CODEUDORES_PK PRIMARY KEY ID_TERCERO, ID_CREDITO) ENABLE) ALTER TABLE CODEUDORES ADD CONSTRAINT CODEUDORES_CREDITOS_FK FOREIGN KEY (ID_CREDITO) REFERENCES CREDITOS (ID_CREDITO) ON DELETE CASCADE ENABLE; ALTER TABLE CODEUDORES ADD CONSTRAINT CODEUDORES_TERCEROS_FK FOREIGN KEY (ID_TERCERO) REFERENCES TERCEROS (ID_TERCERO) ON DELETE CASCADE ENABLE;

12.4.4 **Tabla Créditos.** Tabla de la base de datos para almacenar los datos del crédito otorgado al estudiante.

Tabla 16. Diccionario de datos. Tabla Créditos.

CREDITOS				
CAMPO	TIPO(TAMAÑO)	NULO	CLAVE	DESCRIPCION
ID_CREDITO	Varchar2(5)	N	PK	Identificador único del crédito
VALOR	Number(8,0)	N		Valor en pesos del pagare o préstamo

FECHA	Date	N		Fecha de aprobación del crédito
ESTADO	Varchar2(20)	N		Estado actual del crédito
PLAZO	Varchar2(2)	N		Plazo en meses para pagar el pagare
ID_TIPO_CREDITO	Varchar2(5)	N	FK	Identificador único del tipo de crédito

Fuente: El autor.

12.4.4.1 **Código de script SQL.** CREATE TABLE CREDITOS (ID_CREDITO VARCHAR2 (5 CHAR) NOT NULL, VALOR NUMBER (8, 0) NOT NULL, FECHA DATE NOT NULL, ESTADO VARCHAR2 (20 CHAR) NOT NULL, PLAZO VARCHAR2 (2 CHAR) NOT NULL, ID_TIPO_CREDITO VARCHAR2 (5 CHAR) NOT NULL, CONSTRAINT CREDITOS_PK PRIMARY KEY (ID_CREDITO) ENABLE) ALTER TABLE CREDITOS ADD CONSTRAINT CREDITOS_TIPOS_CREDITO_FK FOREIGN KEY (ID_TIPO_CREDITO) REFERENCES TIPOS_CREDITO (ID_TIPO_CREDITO) ON DELETE CASCADE ENABLE;

12.4.5 **Tabla Tipos de Crédito.** Tabla de la base de datos para almacenar los diferentes tipos de crédito o líneas de crédito autorizadas por la universidad.

Tabla 17. Diccionario de datos. Tabla Tipos Crédito.

TIPOS_CREDITO				
CAMPO	TIPO(TAMAÑO)	NULO	CLAVE	DESCRIPCION
ID_TIPO_CREDITO	Varchar2(5)	N	PK	Identificador único del tipo de crédito
DESCRIPCION	Varchar2(60)	N		Descripción o nombre del tipo de crédito

Fuente: El autor.

12.4.5.1 **Código de script SQL.** CREATE TABLE TIPOS_CREDITO (ID_TIPO_CREDITO VARCHAR2 (5 CHAR) NOT NULL, DESCRIPCION VARCHAR2 (60 CHAR) NOT NULL, CONSTRAINT TIPOS_CREDITO_PK PRIMARY KEY (ID_TIPO_CREDITO) ENABLE).

12.4.6 **Tabla Recibos de Caja.** Tabla de la base de datos para almacenar los recibos de caja, por concepto de pagos o abonos al saldo del crédito educativo vigente.

Tabla 18. Diccionario de datos, tabla Recibos de Caja

RECIBOS_CAJA				
CAMPO	TIPO(TAMAÑO)	NULO	CLAVE	DESCRIPCION
ID_RECIBO_CAJA	Varchar2(5)	N	PK	Identificador único del recibo de caja
VALOR	Number(8,0)	N		Valor en pesos del recibo de caja
FECHA	Date	N		Fecha del recibo de caja
ID_CREDITO	Varchar2(5)	N	FK	identificador único del crédito que le realizó el abono

Fuente: El autor.

12.4.6.1 **Código sentencia SQL.** CREATE TABLE RECIBOS_CAJA (ID_RECIBO_CAJA VARCHAR2 (5 CHAR) NOT NULL, VALOR NUMBER (8, 0) NOT NULL, FECHA DATE NOT NULL, ID_CREDITO VARCHAR2 (5 CHAR) NOT NULL CONSTRAINT RECIBOS_CAJA_PK PRIMARY KEY (ID_RECIBO_CAJA) ENABLE) ALTER TABLE RECIBOS_CAJA ADD CONSTRAINT RECIBOS_CAJA_CREDITOS_FK FOREIGN KEY (ID_CREDITO) REFERENCES CREDITOS (ID_CREDITO) ON DELETE CASCADE ENABLE;

12.4.7 **Tabla Amortizaciones.** Tabla de la base de datos para almacenar la liquidación de los intereses en formato de tabla tipo amortización.

Tabla 19. Diccionario de datos tabla Amortizaciones.

AMORTIZACIONES				
COLUMNA	TIPO(TAMAÑO)	NULO	CLAVE	DESCRIPCIÓN
ID_AMORTIZACION	Number(22,0)	N	PK	Identificador único auto-incremental
PERIODO	Char(2)	N		Periodo o mes de amortización
CUOTA	Number(8,0)	S		Valor de la cuota o valor del recibo de caja
CAPITAL	Number(8,0)	S		Valor de la cuota que se va a capital
INTERES	Number(8,0)	S		Valor de la cuota que se va a intereses
SALDO	Number(8,0)	N		Saldo total de la deuda al periodo amortizado
ID_CREDITO	Varchar2(5)	N	FK	Identificador único del crédito
ID_RECIBO_CAJA	Varchar2(5)	S	FK	Identificador único del recibo de caja

Fuente: El autor.

12.4.7.1 **Código de script SQL.** CREATE TABLE AMORTIZACIONES (ID_AMORTIZACION NUMBER (*, 0) NOT NULL, PERIODO CHAR (2 CHAR) NOT NULL, CUOTA NUMBER (8, 0), CAPITAL NUMBER (8, 0), INTERES NUMBER (8, 0), SALDO NUMBER (8, 0) NOT NULL, ID_CREDITO VARCHAR2 (5 CHAR) NOT NULL, ID_RECIBO_CAJA VARCHAR2 (5 CHAR) CONSTRAINT AMOTIZACIONES_PK PRIMARY KEY (ID_AMORTIZACION) ENABLE) ALTER TABLE AMORTIZACIONESADD CONSTRAINT AMOTIZACIONES_CREDITOS_FK FOREIGN KEY (ID_CREDITO) REFERENCES CREDITOS (ID_CREDITO) ON DELETE CASCADE ENABLE; ALTER TABLE AMORTIZACIONES ADD CONSTRAINT AMOTIZACIONES_RECIBOS_CAJA_FK FOREIGN KEY

(ID_RECIBO_CAJA) REFERENCES RECIBOS_CAJA (ID_RECIBO_CAJA) ON DELETE CASCADE ENABLE;

12.4.8 **Tabla Facultades.** Tabla de la base de datos para almacenar las diferentes facultades que tiene la universidad.

Tabla 20. Diccionario de datos, tabla Facultades.

FACULTADES				
CAMPO	TIPO(TAMAÑO)	NULO	CLAVE	DESCRIPCION
ID_FACULTAD	Varchar2(5)	N	PK	Identificador único que identifica la facultad
NOMBRE	Varchar2(60)	N		Nombre de la facultad

Fuente: El autor.

12.4.8.1 **Código de script SQL.** CREATE TABLE FACULTADES (ID_FACULTAD VARCHAR2 (5 CHAR) NOT NULL, NOMBRE VARCHAR2 (60 CHAR) NOT NULL, CONSTRAINT FACULTADES_PK PRIMARY KEY (ID_FACULTAD) ENABLE)

12.4.9 **Tabla Programas.** Tabla de la base de datos para almacenar los diferentes programas académicos ofertados por la universidad y adscrito a una facultad.

Tabla 21. Diccionario de datos, tabla programas.

PROGRAMAS				
CAMPO	TIPO(TAMAÑO)	NULO	CLAVE	DESCRIPCION
ID_PROGRAMA	Varchar2(5)	N	PK	Identificador único de cada programa académico
NOMBRE	Varchar2(60)	N		Nombre del programa académico
ID_FACULTAD	Varchar2(5)	N	FK	Identificador único de la

				facultad

Fuente: El autor.

12.4.9.1 **Código de script SQL.** CREATE TABLE PROGRAMAS (ID_PROGRAMA VARCHAR2 (5 CHAR) NOT NULL, NOMBRE VARCHAR2 (60 CHAR) NOT NULL, ID_FACULTAD VARCHAR2 (5 CHAR) NOT NULL, CONSTRAINT PROGRAMAS_PK PRIMARY KEY (ID_PROGRAMA) ENABLE) ALTER TABLE PROGRAMAS ADD CONSTRAINT PROGRAMAS_FACULTADES_FK FOREIGN KEY (ID_FACULTAD) REFERENCES FACULTADES (ID_FACULTAD) ON DELETE CASCADE ENABLE;

12.4.10 **Tabla Usuarios.** Tabla para almacenar los datos de acceso de los usuarios del sistema.

Tabla 22. Diccionario de datos, Tabla Usuarios.

USUARIOS				
CAMPO	TIPO(TAMAÑO)	NULO	CLAVE	DESCRIPCION
ID_USUARIO	Varchar2(15)	N	PK	Identificador único del nombre de usuario del sistema
CONTRASENA	Varchar2(12)	N		Contraseña de acceso al sistema
NOMBRE	Varchar2(50)	N		Nombre completo del usuario del sistema
ESTADO	Number(22,0)	N		Estado de actividad del usuario del sistema
ID_ROL	Number(22,0)	N	FK	Identificador único del rol o grupo en el sistema

Fuente: El autor.

12.4.10.1 **Código de script SQL.** CREATE TABLE USUARIOS (ID_USUARIO VARCHAR2 (15 CHAR) NOT NULL, CONTRASENA VARCHAR2 (12 CHAR) NOT NULL, NOMBRE VARCHAR2 (50 CHAR) NOT NULL, ID_ROL NUMBER (*, 0) NOT NULL, ESTADO NUMBER (*, 0)

NOT NULL, CONSTRAINT USUARIOS_PK PRIMARY KEY (ID_USUARIO) ENABLE) ALTER TABLE USUARIOSADD CONSTRAINT USUARIOS_ROLES_FK1 FOREIGN KEY (ID_ROL) REFERENCES ROLES (ID_ROL) ON DELETE CASCADE ENABLE;

12.4.11 **Tabla Roles.**Tabla para almacenar los diferentes roles, grupos o perfiles de acceso al sistema.

Tabla 23. Diccionario de datos, tabla Roles.

ROLES				
CAMPO	TIPO(TAMAÑO)	NULO	CLAVE	DESCRIPCION
ID_ROL	Number(22,0)	N	PK	Identificador único del rol o grupo en el sistema
DESCRIPCION	Varchar2(30)	N		Nombre que describe el rol o grupo en el sistema

Fuente: El autor.

12.4.11.1 **Códido de script SQL.** CREATE TABLE ROLES (ID_ROL NUMBER (*, 0) NOT NULL, DESCRIPCION VARCHAR2 (30 CHAR) NOT NULL, CONSTRAINT ROLES_PK PRIMARY KEY (ID_ROL) ENABLE)

12.4.12 **Tabla Parámetros.** Tabla libre de la base de datos para almacenar los parámetros globales de toda la aplicación.

Tabla 24. Diccionario de datos, tabla Parámetros.

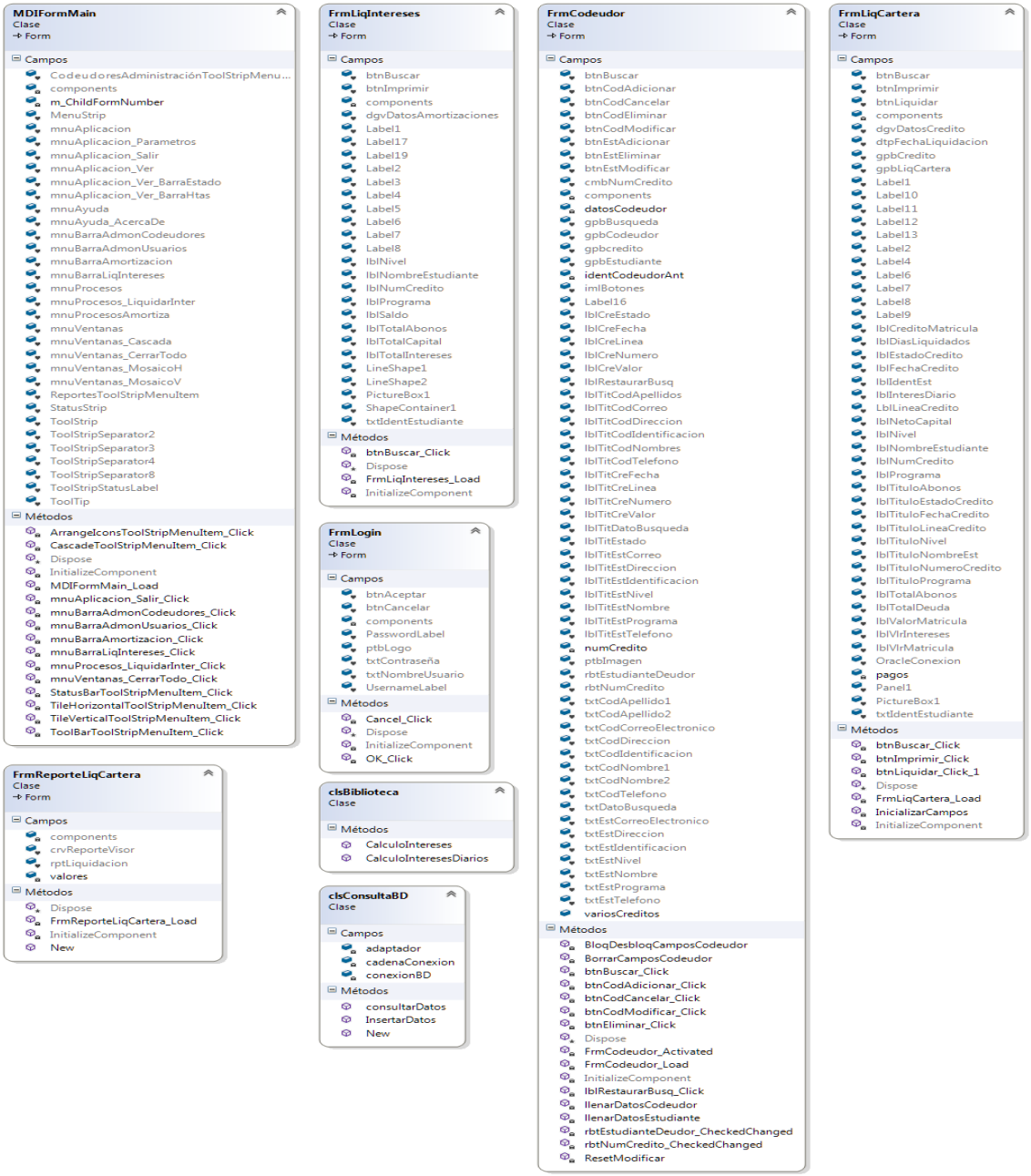
PARAMETROS				
CAMPO	TIPO(TAMAÑO)	NULO	CLAVE	DESCRIPCION
INTERES	Float	N		Campo para almacenar el valor del porcentaje interés.

Fuente: El autor.

12.4.12.1 **Código de script SQL.** CREATE TABLE PARAMETROS (INTERES FLOAT (126) NOT NULL);

12.5 DIAGRAMA DE CLASES

Ilustración 15. Diagramas de clases del sistema.

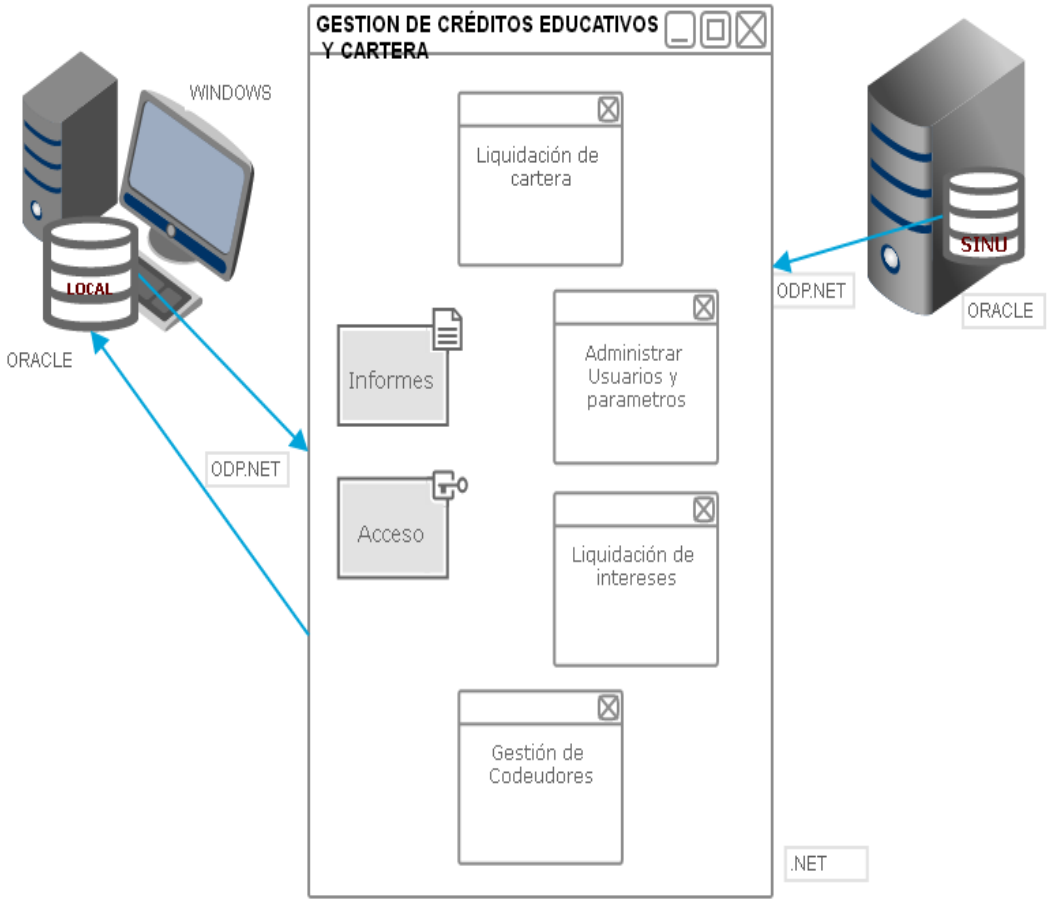


Fuente: El autor.

12.6 DISEÑO ARQUITECTURA DEL SISTEMA

A continuación se presenta el modelo con la arquitectura del sistema o prototipo desarrollado, diseñada y definida como aplicación de escritorio y una base de datos local para almacenar información y una base de datos externa con acceso de solo lectura.

Ilustración 16. Diseño arquitectónico del sistema.



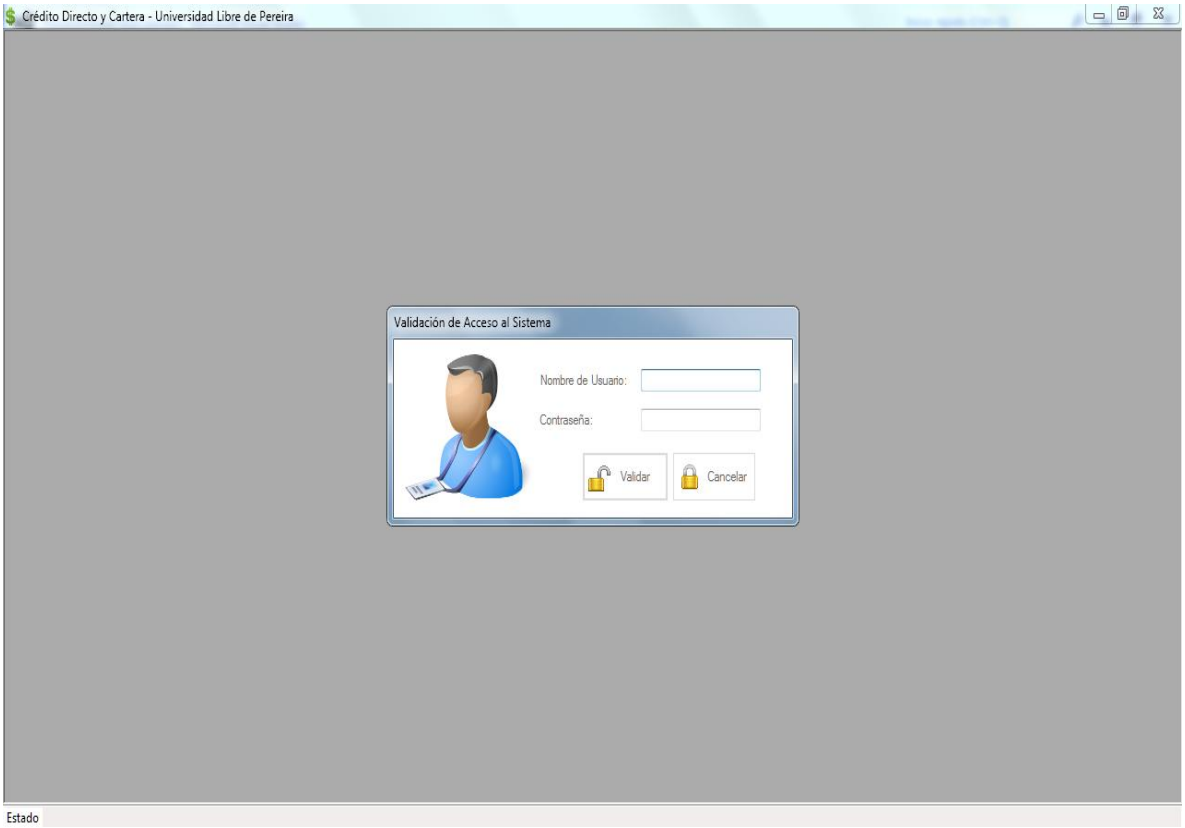
Fuente: El autor.

13 IMPLEMENTACIÓN DEL PROTOTIPO

13.1 Pantalla inicial de acceso

Pantalla inicial al abrir la aplicación donde solicita las credenciales de acceso a la aplicación y valida el rol o perfile del usuario para cargar o mostrar las opciones de acceso.

Ilustración 17. Pantalla inicial de la aplicación.



Fuente: El autor.

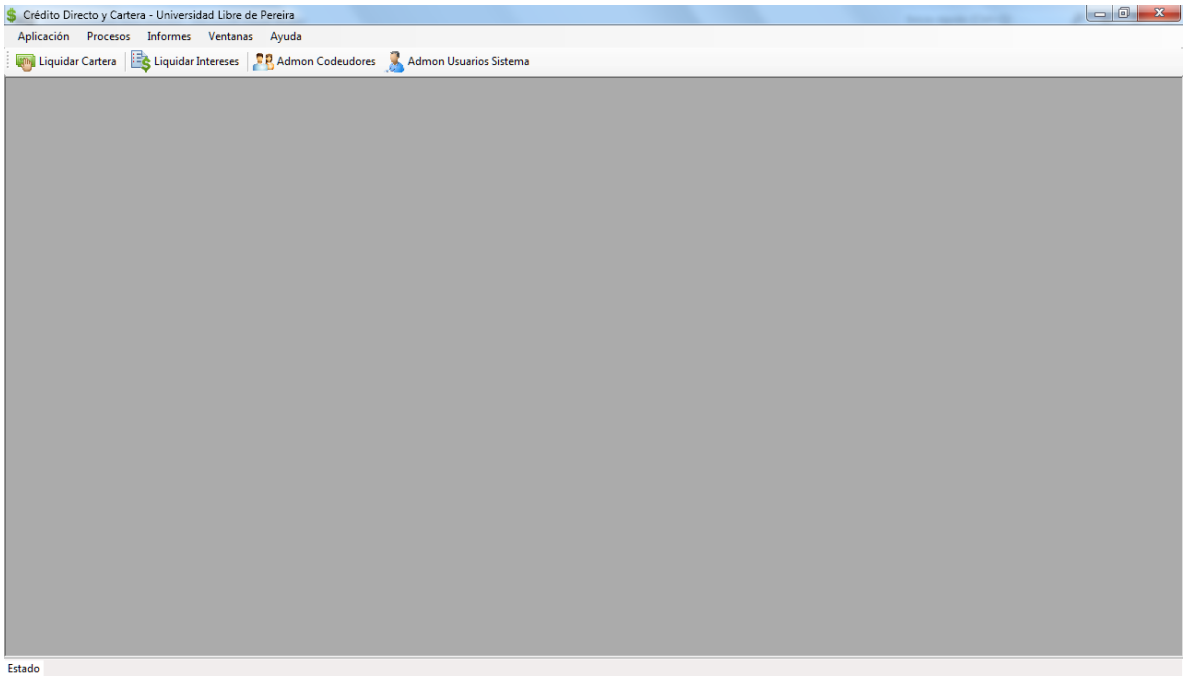
13.1.1 Código fuente o lógica del negocio

```
'CLASE: FrmLogin. 'NOMBRE: Validación de Acceso al Sistema. DISEÑO & CODIGO:  Mauricio Ramírez Giraldo, Diego Alejandro Madrid Roman. 'FECHA: 15 Septiembre de 2013. 'DESCRIPCIÓN: Valida y autentica los datos de acceso: nombre, contraseña; existen en la base de datos y que no estén inactivos si pasa la validación, se habilitan los controles del formulario contenedor principal. Si tiene más de tres Intentos fallidos se cierra toda la aplicación. 'VERSION: 1.0. 'MODIFICACIONES: Public Class FrmLogin Public intentosAutentica As Short 'Acumular los intentos de acceso fallidos Private Sub OK_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnAceptar.Click Dim consultaBD As New clsAccesoBD 'Instancia a clase de acceso a datos Dim datos As New DataTable 'Recuperar los datos devueltos por la consulta SQL If intentosAutentica < 4 Then 'Se valida que los intentos no pasen de 3 datos = consultaBD.consultarDatos("SELECT * FROM USUARIOS WHERE ID_USUARIO='" & txtNombreUsuario.Text & "' AND CONTRASENA='" & txtContraseña.Text & "'") 'Se asigna el origen de datos al data grid con la tabla devuelta por la consult If datos.Rows.Count > 0 Then 'Se valida que la consulta halla devuelto algún resultado If datos.Rows(0).Item("ESTADO") = 1 Then 'Validar que el usuario no este inabilitado MDIFormMain.BrrProcesos.Visible = True MDIFormMain.mnuBarraHtas.Visible = False 'Valida si el usuario tiene rol de superUsuario o Administrador If datos.Rows(0).Item("ID_ROL") = 1 Or datos.Rows(0).Item("ID_ROL") = 2 Then MDIFormMain.mnuBarraAdmonUsuarios.Enabled = True 'Muestra el botón End If Me.Close() Me.Dispose() 'Se liberan recursos Else MsgBox("El usuario digitado se encuentra actualmente inactivo en el sistema." & vbCrLf & _ "Consulte con el Administrador del sistema si cree que es un error.", MsgBoxStyle.Exclamation, "Validación de Acceso al Sistema") txtContraseña.Text = "" txtNombreUsuario.Text = "" txtNombreUsuario.Focus() End If Else MsgBox("Los datos de acceso Usuario y/o Contraseña no tienen acceso al sistema." & vbCrLf & "Verifique e intentelo de nuevo!", MsgBoxStyle.Critical, "Validación de Acceso al Sistema") txtContraseña.Text = "" txtNombreUsuario.Text = "" txtNombreUsuario.Focus() intentosAutentica += 1 'Acumular intentos fallidos End If Else MsgBox("Ha excedito el límite de intentos de acceso." & vbCrLf & "Por seguridad la aplicación se cerrará.", MsgBoxStyle.Exclamation, "Validación de Acceso al Sistema") Application.Exit()End If End Sub 'Evento o método que se lanza cuando se da clic en cancelar Private Sub Cancel_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnCancelar.Click Application.Exit() End Sub End Class.
```

13.2 Pantalla contenedora o principal

Pantalla principal y contenedora de los demás formularios, donde se brinda acceso a todas las opciones y procesos del sistema de forma directa con una barra de menú y una barra de herramientas.

Ilustración 18. Pantalla o formulario principal, contenedora.



Fuente: El autor.

13.2.1 Código fuente o lógica del negocio

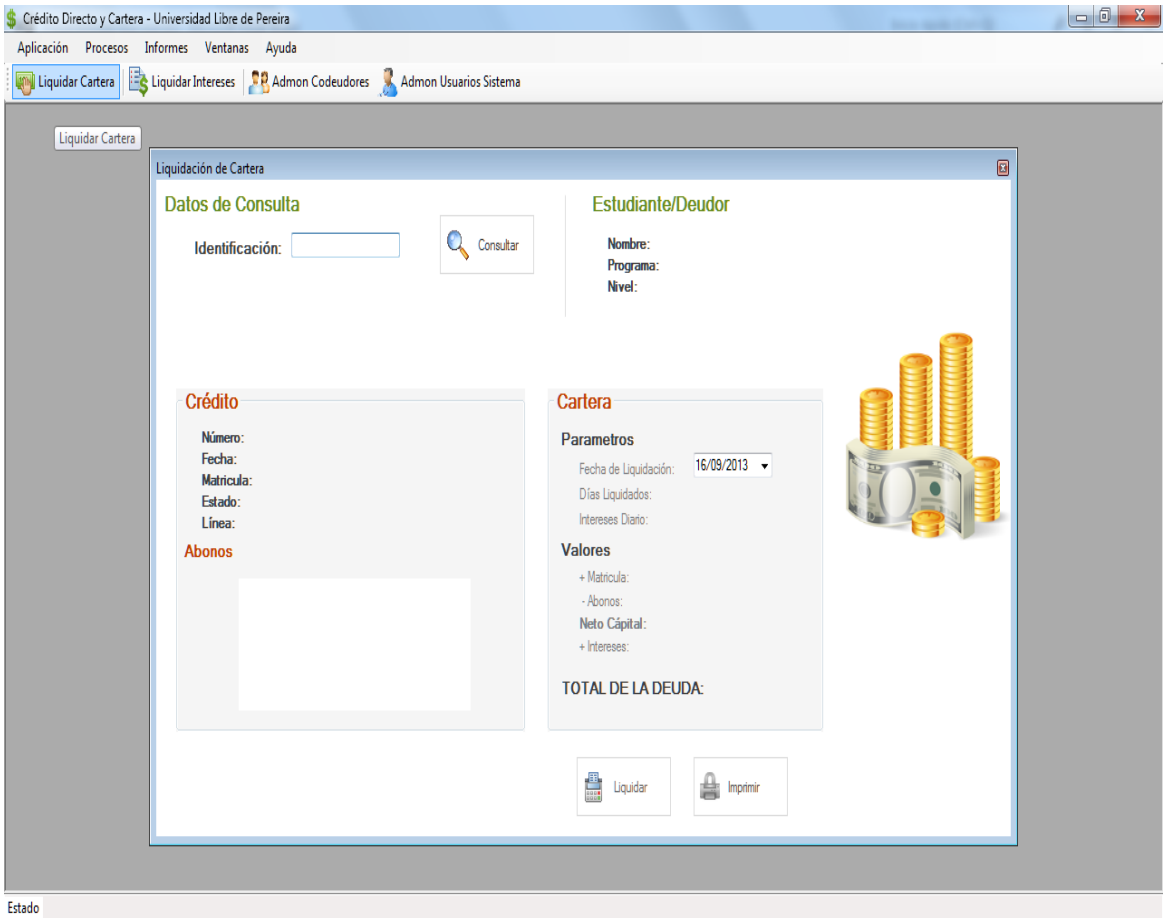
'CLASE: MDIFrmMain. NOMBRE: Formulario contenedor principal. 'DISEÑO & CODIGO: Mauricio Ramírez Giraldo Diego Alejandro Madrid Roman. 'FECHA: 15 Septiembre de 2013. 'DESCRIPCIÓN: Contiene la barra de menú y la barra de herramientas que provee acceso a todos los procesos de la aplicación. Formulario base para contener los demás formularios. 'VERSION: 1.0. 'MODIFICACIONES:Imports System.Windows.Forms Public Class MDIFormMain Private m_ChildFormNumber As Integer 'Eventos o método lanzado cuando se carga el formulario Private Sub MDIFormMain_Load(sender As Object, e As EventArgs) Handles MyBase.Load FrmLogin.ShowDialog() 'Abre el formulario de login End Sub 'Eventos o métodos lanzados cuando se selecciona alguna opción de la barra de herramientas Private Sub mnuBarraLiqIntereses_Click(sender As

```
Object, e As EventArgs) Handles mnuBarraLiqIntereses.Click Dim formCartera As New FrmLiqCartera(intereses) formCartera.MdiParent = Me 'Se convierte en un elemento secundario de este formulario MDI antes de mostrarlo. formCartera.Show() End Sub Private Sub mnuBarraAmortizacion_Click(sender As Object, e As EventArgs) Handles mnuBarraAmortizacion.Click Dim formIntereses As New FrmLiqIntereses(intereses) formIntereses.MdiParent = Me 'Se convierte en un elemento secundario de este formulario MDI antes de mostrarlo. formIntereses.Show() End Sub Private Sub mnuBarraAdmonCodeudores_Click(sender As Object, e As EventArgs) Handles mnuBarraAdmonCodeudores.Click FrmCodeudor.MdiParent = Me FrmCodeudor.Show() End Sub Private Sub ToolBarToolStripMenuItem_Click(ByVal sender As Object, ByVal e As EventArgs) Me.BrrProcesos.Visible = Me.mnuAplicacion_Ver_BarraHtas.Checked End Sub Private Sub StatusBarToolStripMenuItem_Click(ByVal sender As Object, ByVal e As EventArgs) Me.StatusStrip.Visible = Me.mnuAplicacion_Ver_BarraEstado.Checked End Sub Private Sub CascadeToolStripMenuItem_Click(ByVal sender As Object, ByVal e As EventArgs) Handles mnuVentanas_Cascada.Click Me.LayoutMdi(MdiLayout.Cascade) End Sub Private Sub TileVerticalToolStripMenuItem_Click(ByVal sender As Object, ByVal e As EventArgs) Handles mnuVentanas_MosaicoV.Click Me.LayoutMdi(MdiLayout.TileVertical) End Sub Private Sub TileHorizontalToolStripMenuItem_Click(ByVal sender As Object, ByVal e As EventArgs) Handles mnuVentanas_MosaicoH.Click Me.LayoutMdi(MdiLayout.TileHorizontal) End Sub Private Sub ArrangeIconsToolStripMenuItem_Click(ByVal sender As Object, ByVal e As EventArgs) Me.LayoutMdi(MdiLayout.ArrangeIcons)End Sub Private Sub mnuVentanas_CerrarTodo_Click(sender As Object, e As EventArgs) Handles mnuVentanas_CerrarTodo.Click ' Cierre todos los formularios secundarios del principal. For Each ChildForm As Form In Me.MdiChildren ChildForm.Close() Next End Sub Private Sub mnuProcesos_LiquidarInter_Click(sender As Object, e As EventArgs) Handles mnuProcesos_LiquidarInter.Click Dim formCartera As New FrmLiqCartera(intereses) formCartera.MdiParent = Me 'Se convierte en un elemento secundario de este formulario MDI antes de mostrarlo. formCartera.Show() End Sub Private Sub mnuAplicacion_Salir_Click(sender As Object, e As EventArgs) Handles mnuAplicacion_Salir.Click Me.Close() End Sub End Class.
```

13.3 Pantalla Liquidar Cartera

Pantalla que se abre cuando se selecciona la opción de liquidar cartera. Brinda la opción de hacer la consulta por el número de documento del estudiante/deudor y muestra toda la información concerniente al deudor y el crédito; y donde puede realizar la respectiva liquidación de cartera.

Ilustración 19. Pantalla o formulario Liquidar Cartera.



Fuente: El autor.

13.3.1 Código fuente o lógica del negocio

```
'CLASE: FrmLiqCartera 'NOMBRE: Liquidación de intereses 'DISEÑO & CODIGO: Mauricio Ramírez Giraldo Diego Alejandro Madrid Roman 'FECHA: 15 Septiembre de 2013 'DESCRIPCIÓN: Consulta por número de identificación de un estudiante/deudor todos los créditos otorgados.' Según el número de crédito seleccionado lista toda su información.' Calcula los días trascurridos entre fecha de aprobación y liquidación para calcular los 'Intereses, el capital neto. Calcula la sumatoria del total de abonos y el saldo de la deuda. 'VERSION: 1.0 'MODIFICACIONES: Imports Oracle.DataAccess.Client 'Importar librerías de acceso a datos de ORACLE Public Class FrmLiqCartera 'Propiedades de la clase formulario Private pagos As Long Private identEstudiante As String Private VlrInteres As Single 'Constructor para recibir los parámetros globales Public Sub New(ByVal intereses As Single) InitializeComponent() VlrInteres = intereses
```

```
End Sub 'Evento o procedimiento que se lanza cuando se da clic en el botón "CONSULTAR" Private Sub btnBuscar_Click(sender As Object, e As EventArgs) Handles btnBuscar.Click Dim BD As New clsAccesoBD 'Instancia de la clase que accede a los datos Dim creditos As New DataTable 'Data table para cargar los registros devueltos por la consulta Try identEstudiante = txtIdentEstudiante.Text If identEstudiante <> "" Then 'Validar que el campo de busqueda contenga algún valor creditos = BD.consultarDatos("SELECT * FROM VW_NUM_CREDITOS WHERE ID_TERCERO='" & txtIdentEstudiante.Text & "'") If creditos.Rows.Count > 0 Then 'Controlar que la consulta devuelva algún registro Me.cmbNumCredito.Visible = True lblTitCreditos.Visible = True cmbNumCredito.DataSource = creditos 'Asignar como datos de origen los registros del dataTable cmbNumCredito.DisplayMember = "NUM_CREDITO" cmbNumCredito.ValueMember = "ID_CREDITO" InicializarCampos() 'Procedimiento que incializa todos los campos de salida del formulario Else MsgBox("El dato de búsqueda no arrojo ningún resultado o el cliente no se encuentra en la base de datos." & vbCrLf & "Revise los datos e intentelo de nuevo.", MsgBoxStyle.Exclamation, "Crédito y Cartera - Mensaje") End If Else MsgBox("No ha digitado ningún dato en el campo de búsqueda 'Identificación'." & vbCrLf & "Verifique e intentelo de nuevo.", MsgBoxStyle.Exclamation, "Crédito y Cartera - Mensaje") End If Catch ex As Exception MsgBox("Ocurrio un error inesperado al consutar el los datos del estudiante. " & vbCrLf & "Consulte al administrador del programa sobre este error." & vbCrLf & ex.Message, MsgBoxStyle.Critical, "Crédito y Cartera - Mensaje") End Try 'Liberar recursos utilizados por los objetos BD = Nothing End Sub 'Evento o precedimiento que se lanza cuando se da clic en el botón "LIQUIDAR" Private Sub btnLiquidar_Click(sender As Object, e As EventArgs) Handles btnLiquidar.Click Dim consultaBD As New clsAccesoBD Dim calc As clsBiblioteca Dim datos As New DataTable Try Me.lblVlrMatricula.Text = FormatCurrency(Me.lblCreditoMatricula.Text, 0) datos = consultaBD.consultarDatos("SELECT SUM(VALOR) as sumaAbonos FROM RECIBOS_CAJA WHERE ID_CREDITO='" & Me.lblNumCredito.Text & "'") 'Calcular el valor total de los abonos o recibos de caja, sumando todos los valores contenidos en la columna con indice="VALOR" If Not IsDBNull(datos.Rows(0).Item("sumaAbonos")) Then 'Validar que la columna a sumar tenga datos. Me.lblTotalAbonos.Text = FormatCurrency(datos.Rows(0).Item("sumaAbonos"),0) Else Me.lblTotalAbonos .Text=0 End If calc = New clsBiblioteca 'Instancia de la clase que contiene los métodos de calculo 'Se calcula la diferencia en días entre la fecha de aprobación y liquidación Me.lblDiasLiquidados.Text = DateDiff(DateInterval.Day, CDate(Me.lblFechaCredito.Text), Me.dtpFechaLiquidacion.Value.Date) 'Realiza el calculo de capital neto valor de la matricula - el total de abonos realizados Me.lblNetoCapital.Text = FormatCurrency(CLng(Me.lblVlrMatricula.Text) - CLng(Me.lblTotalAbonos.Text), 0) 'Se calcula el valor de los intereses por día
```

```
Me.lblInteresDiario.Text=FormatCurrency(calc.CalculoInteresesDiarios(CLng(Me.lb
lNetoCapital.Text),VlrInteres) 0) 'Se calcula el valor de los intereses con la fórmula
contenida en la clase clsBiblioteca Me.lblVlrIntereses.Text =
FormatCurrency(calc.CalculoIntereses(CLng(Me.lblNetoCapital.Text),VlrInteres,Int
(Me.lblDiasLiquidados.Text) / 2), 0) 'Se calcula el total de la deuda restando el
capital neto + los intereses causados Me.lblTotalDeuda.Text =
FormatCurrency(CLng(Me.lblNetoCapital.Text) + CLng(Me.lblVlrIntereses.Text))
btnImprimir.Enabled = True 'Liberar recurosos utilizados por los objetos
consultaBD = Nothing Catch ex As Exception MsgBox("Ocurrio un error
inesperado al consutar el los datos del estudiante. " & vbCrLf & "Consulte al
administrador del programa sobre este error." & vbCrLf & ex.Message,
MsgBoxStyle.Critical, "Crédito y Cartera - Mensaje") End Try End Sub 'Evento o
precedimiento que se lanza cuando se da clic en el botón "IMPRIMIR" Private Sub
btnImprimir_Click(sender As Object, e As EventArgs) Handles btnImprimir.Click
Dim listaValores As New Collection Dim formReporte As FrmReporteLiqCartera
Try 'Lista de parametros para enviar al informe
listaValores.Add(Me.lblNumCredito.Text)listaValores.Add(Me.txtIdentEstudiante.Te
xt)listaValores.Add(Me.dtpFechaLiquidacion.Text)listaValores.Add(Me.lblVlrInteres
es.Text)listaValores.Add(Me.lblTotalAbonos.Text)listaValores.Add(Me.lblInteresDia
rio.Text)listaValores.Add(Me.lblTotalDeuda.Text)listaValores.Add(Me.lblNetoCapit
al.Text)'Instancia del formulario reporte de liquidación de cartera formReporte =
New FrmReporteLiqCartera(listaValores) 'Se envia la colección de valores 'Se
abre el formulario contenedor del reporte formReporte.MdiParent = MDIFormMain
formReporte.ShowDialog() btnImprimir.Enabled = False 'Se inabilita el boton
"IMPRIMIR" Catch ex As Exception MsgBox("Ocurrio un error inesperado al
consutar el los datos del estudiante. " & vbCrLf & "Consulte al administrador del
programa sobre este error." & vbCrLf & ex.Message, MsgBoxStyle.Critical,
"Crédito y Cartera - mensaje") End Try End Sub ' ### Procedimientos y funciones
personales ### 'Procedimiento de que consulta los datos del crédito selecionado
y los recibos de caja realizados a dicho crédito. Private Sub BuscarDatos()
'Declaración de variables Dim consultaBD As New clsAccesoBD Dim
datosEstudiante As New DataTable Dim tipoNivelAcadem As String Dim
numCredito As String = "" Dim estado As String = "" Try 'Consulta a la vista de
la BD con el filtro de estudiante seleccionado datosEstudiante =
consultaBD.consultarDatos("SELECT * FROM VW_ESTUDIANTES_CREDITO
WHERE IDENTIFICACION='" & _txtIdentEstudiante.Text & "' AND CREDITO='" &
cmbNumCredito.SelectedValue & "'") If datosEstudiante.Rows.Count > 0 Then
'Se llenan los campos del formulario For Each registro In datosEstudiante.Rows()
'Recorrer los registros devueltos por la consulta
Me.lblNombreEstudiante.Text = registro(1).ToString Me.lblPrograma.Text=
registro(6).ToString & " - " & registro(7).ToString If registro(6).ToString <>
"DERECHO" Then 'Se valida el programa acádemico para mostrar el nivel en años
o semestres tipoNivelAcadem = " Semestre" Else tipoNivelAcadem = " Año" End If
Me.lblNivel.Text = registro(5).ToString & tipoNivelAcadem Me.lblFechaCredito.Text
```

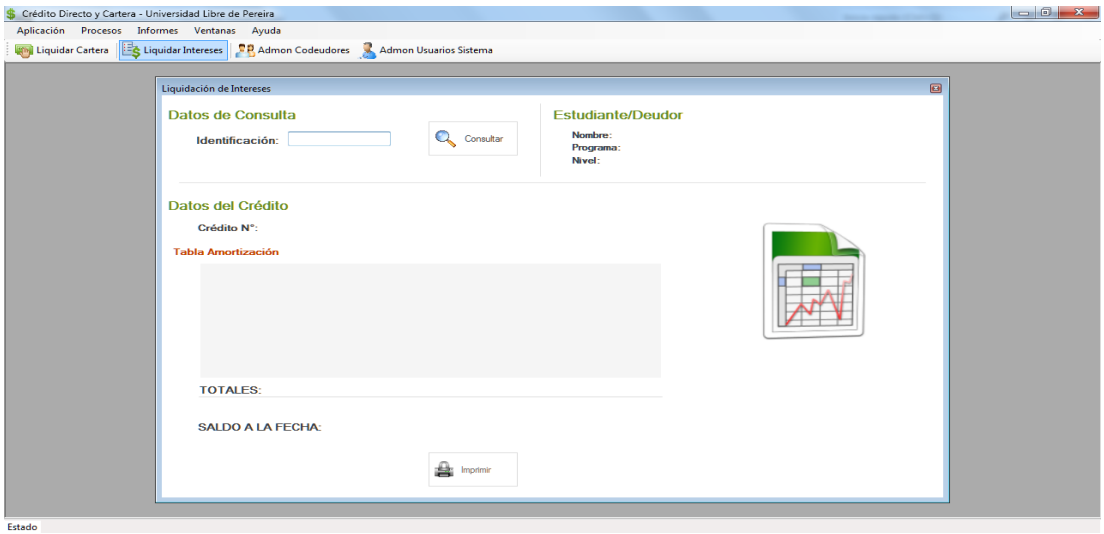
```
= registro(11).ToString Me.lblCreditoMatricula.Text=formatCurrency(registro(10)
.ToString, 0) numCredito = registro(9).ToString Me.lblNumCredito.Text =
numCreditoestado = registro(12).ToString Me.lblEstadoCredito.Text =
estadoMe.LblLineaCredito.Text = registro(8).ToString Next btnLiquidar .Enabled =
True 'Se habilita el botón de liquidar 'Llenamos el dataGrid con los datos devueltos
por la consulta a Recibos de Caja Me.dgvDatosCredito.DataSource =
consultaBD.consultarDatos("SELECT ID_RECIBO_CAJA, VALOR, FECHA FROM
RECIBOS_CAJA WHERE ID_CREDITO='" & Me.lblNumCredito.Text & "' ORDER
BY FECHA") 'Se asigna el origen de datos al data grid con la tabla devuelta por la
consulta Me.dgvDatosCredito.Columns(0).HeaderText "RECIBO CAJA"
Me.dgvDatosCredito. Columns(1).DefaultCellStyle.Format = "c0" 'Formato de
moneda a la columna(1) campo "VALOR" 'Asegurar el formato del
dataGrid letra color= Negra, tamaño=8; sin negrilla Dim formato As New
Font(FontFamily.GenericSansSerif, 8, FontStyle.Regular)
dgvDatosCredito.DefaultCellStyle.Font=formatodgvDatosCredito.DefaultCellStyle.F
oreColor=Color.BlackdgvDatosCredito.ColumnHeadersDefaultCellStyle.Font=form
atodgvDatosCredito.ColumnHeadersDefaultCellStyle.ForeColor=Color.BlackgvDat
osCredito.RowsDefaultCellStyle.Font=formatodgvDatosCredito.AlternatingRowsDe
faultCellStyle.Font=formatodgvDatosCredito.AlternatingRowsDefaultCellStyle.Fore
Color=Color.Blackformato.Dispose() Validamos  el estado del crédito If estado <>
"Aprobado" Then MsgBox("El crédito " & numCredito & ", consultado
del estudiante/Deudor " &
datosEstudiante.Rows(0).Item("IDENTIFICACION").ToString & " se encuentra " &
estado & vbCrLf & "La información suminstrada es de solo lectura.",
MsgBoxStyle.Information, "Crédito y Cartera - Mensaje")
btnLiquidar.Enabled = False 'Inabilitar el botón "LIQUIDAR" End If Else
MsgBox("No se pudo cargar la información del crédito" & vbCrLf & "Revise los
datos e intentelo de nuevo.", MsgBoxStyle.Exclamation, "Crédito y Cartera -
Mensaje") End If 'liberar recursos utilizados por los objetos
datosEstudiante.Clear() datosEstudiante.Dispose() consultaBD = Nothing Catch ex
As Exception MsgBox("Ocurrio un error inesperado al consutar en los datos del
estudiante. " & vbCrLf & "Consulte al administrador del programa sobre este error."
& vbCrLf & ex.Message, MsgBoxStyle.Critical, "Crédito y Cartera - Mensaje")
End Try End Sub 'Procedimientos para borrar asegurar que en cada consulta se
muestre información del crédito consultado Private Sub InicializarCampos()
cmbNumCredito.Text = "Seleccione número" cmbNumCredito.ForeColor =
Color.DarkGray 'Campos del estudiante/codeudor lblNombreEstudiante.Text =
"" lblPrograma.Text = "" lblNivel.Text = "" 'campos del crédito
lblNumCredito.Text = "" lblFechaCredito.Text = "" lblCreditoMatricula.Text = ""
lblEstadoCredito.Text = "" LblLineaCredito.Text = ""
dgvDatosCredito.DataSource = Nothing dgvDatosCredito.Refresh() 'Campos en el
grupo de cartera dtpFechaLiquidacion.Value = Today 'Asignamos la fecha actual
lblDiasLiquidados.Text = "" lblInteresDiario.Text = ""lblVlrMatricula.Text =
""lblTotalAbonos.Text = ""lblNetoCapital.Text = ""lblVlrIntereses.Text =
```

""lblTotalDeuda.Text = "" End Sub 'Evento o procedimientos que se lanza cuando se pone el foco en el objeto Private Sub cmbNumCredito_GotFocus(sender As Object, e As EventArgs) Handles cmbNumCredito.GotFocus cmbNumCredito.ForeColor = Color.Black End Sub 'Evento o procedimientos que se lanza cuando se selecciona un valor del número del crédito Private Sub cmbNumCredito_SelectionChangeCommitted(sender As Object, e As EventArgs) Handles cmbNumCredito.SelectionChangeCommitted 'Validar que antes de llamar al procedimiento que consulta los datos del crédito y recibos de caja 'Tenga algún valor If cmbNumCredito.Text <> "" And cmbNumCredito.Text = "Seleccione número" Then BuscarDatos() 'Llamada al procedimiento que realiza la consulta de la inf de los créditos y recibos de caja End If End Sub 'Evento o procedimientos que se lanza cuando se cierra el formulario Private Sub FrmLiqCartera_FormClosed(sender As Object, e As FormClosedEventArgs) Handles Me.FormClosed Me.Dispose() 'Aseguramos que se liberan recursos utilizados por el formulario End Sub Private Sub FrmLiqCartera_Load(sender As Object, e As EventArgs) Handles MyBase.Load End Sub End Class

13.4 Pantalla Liquidar Intereses

Pantalla que se abre al seleccionar la opción de liquidar intereses, permite la consulta por número de documento del deudor, y posteriormente muestra todos los créditos otorgados al estudiante/deudor; donde se puede seleccionar el crédito a calcular intereses, mostrando el resultado en una tabla tipo amortización.

Ilustración 20. Pantalla o formulario liquidar intereses.



Fuente: El autor.

13.4.1 Código fuente o lógica del negocio

```
'CLASE: FrmLiqIntereses 'NOMBRE: Liquidación de intereses 'DISEÑO & CODIGO: Mauricio Ramírez Giraldo Diego Alejandro Madrid Roman 'FECHA: 15 Septiembre de 2013 'DESCRIPCIÓN: Consulta por número de identificación de un estudiante/deudor todos los créditos otorgados.' Según el número de crédito seleccionado lista toda su información.' Calcula los días trasncurridos entre fecha de aprobacion y liquidación para calcular los 'Intereses, el capital neto. Calcula la sumatoria del total de abonos y el saldo de la deuda. 'VERSION: 1.0 'MODIFICACIONES: Imports Oracle.DataAccess.Client Public Class FrmLiqIntereses Private identEstudiante As String Private VlrInteres As Single 'Constructor para recibir los parametros globales Public Sub New(ByVal intereses As Single) InitializeComponent() VlrInteres = intereses End Sub Private Sub btnBuscar_Click(sender As Object, e As EventArgs) Handles btnBuscar.Click Dim BD As New clsAccesoBD 'Instancia de la clase que accede a los datos Dim creditos As New DataTable 'Data Table para cargar los registros devueltos por la consulta Try identEstudiante = txtIdentEstudiante.Text If identEstudiante <> "" Then 'Validar que el campo de busqueda contenga algún valor creditos = BD.consultarDatos("SELECT * FROM VW_NUM_CREDITOS WHERE ID_TERCERO='" & txtIdentEstudiante.Text & "'")If creditos.Rows.Count > 0 Then 'Controlar que la consulta devuelva algún registro Me.cmbNumCredito.Visible = True lblTitCreditos.Visible = True cmbNumCredito.DataSource = creditos 'Asignar como datos de origen los registros del dataTable cmbNumCredito.DisplayMember = "NUM_CREDITO" cmbNumCredito.ValueMember = "ID_CREDITO" InicializarCampos() 'Procedimiento que incializa todos los campos de salida del formulario Else MsgBox("El dato de búsqueda no arrojo ningún resultado o el cliente no se encuentra en la base de datos." & vbCrLf & "Revise los datos e intentelo de nuevo.", MsgBoxStyle.Exclamation, "Crédito y Cartera - Mensaje") End If Else MsgBox("No ha digitado ningún dato en el campo de búsqueda 'Identificación'." & vbCrLf & "Verifique e intentelo de nuevo.", MsgBoxStyle.Exclamation, "Crédito y Cartera - Mensaje") End If Catch ex As Exception MsgBox("Ocurrio un error inesperado al consutar el los datos del estudiante. " & vbCrLf & "Consulte al administrador del programa sobre este error." & vbCrLf & ex.Message, MsgBoxStyle.Critical, "Crédito y Cartera - Mensaje") End Try 'Liberar recursos utilizados por los objetos BD = Nothing End Sub Private Sub cmbNumCredito_SelectionChangeCommitted(sender As Object, e As EventArgs) Handles cmbNumCredito.SelectionChangeCommitted 'Validar que antes de llamar al procedimiento que consulta los datos del crédito y recibos de caja 'Tenga algún valor If cmbNumCredito.Text <> "" And cmbNumCredito.Text = "Seleccione número" Then cargarDatos() 'Llamada al procedimiento que realiza la consulta de la inf de los créditos y recibos de caja End If End Sub Private Sub cargarDatos() 'Variables de acceso a datos 'Dim consultaBD As New clsAccesoBD Dim datosEstudiante As New DataTable Dim datosRecibosCaja As DataTable Dim
```

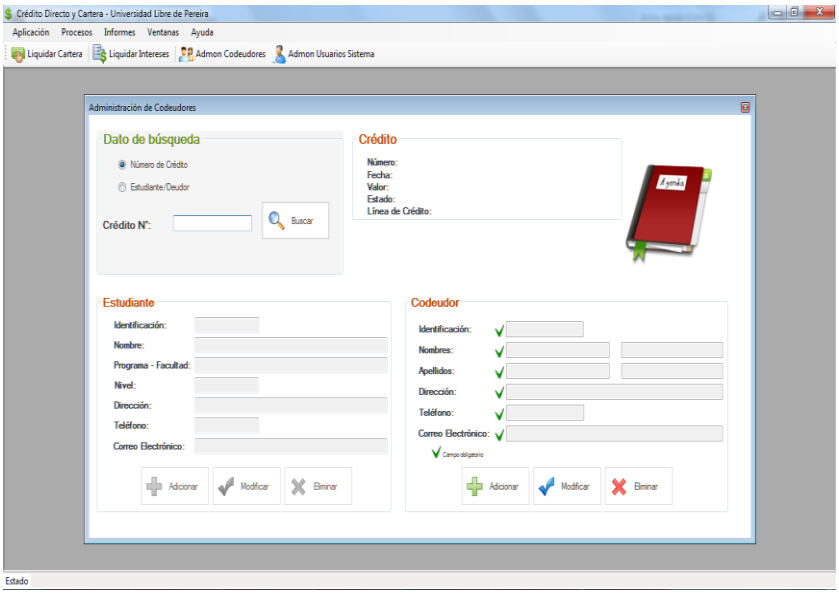
```
datosAmortizacion As DataTable Dim SQLUpdate As String 'Variables para almacenar los datos requeridos del crédito Dim BD = New clsAccesoBD Dim calc = New clsBiblioteca Dim tipoNivelAcadem As String Dim periodo As Short = 0 'Variable ir incrementando el periodo amortizado Dim fechaCredito As Date Dim valorPagare As Long Dim saldoCredito As Long Dim intereses As Long Dim capital As Long Dim sumaCapital, sumaIntereses, sumaAbonos As Long try datosEstudiante = BD.consultarDatos("SELECT * FROM VW_ESTUDIANTES_CREDITO WHERE IDENTIFICACION='" & _ txtIdentEstudiante.Text & "' AND CREDITO='" & cmbNumCredito.SelectedValue & "'") If datosEstudiante.Rows.Count > 0 Then 'Se llenan los campos del deudor y el crédito For Each registro In datosEstudiante.Rows 'Recorres los registros devueltos por la consulta Me.lblNombreEstudiante.Text = registro(1).ToString Me.lblPrograma.Text = registro(6).ToString & " - " & registro(7).ToString If registro(6).ToString <> "DERECHO" Then 'Se valida el programa académico para mostrar el nivel en años o semestres tipoNivelAcadem = " Semestre" Else tipoNivelAcadem = " Año" End If Me.lblNivel.Text = registro(5).ToString & " " & tipoNivelAcadem fechaCredito = registro(11) valorPagare = registro(10) Next Me.lblNumCredito.Text = cmbNumCredito.SelectedValue saldoCredito = valorPagare '### Consulta a los recibos de caja asociados al crédito seleccionado datosRecibosCaja=BD.consultarDatos("SELECT ID_RECIBO_CAJA, VALOR, FECHA FROM RECIBOS_CAJA WHERE ID_CREDITO='"&_ cmbNumCredito.SelectedValue & "' ORDER BY FECHA") 'Se recorre cada registro devuelto por la consulta de recibos de caja y se efectua el calculo: 'Días transcurridos entre fecha del crédito y primer pago se calcula el abono a intereses 'Con la restar de los intereses y valor pagado se obtiene el abono a capital 'Y con la resta del saldo anterior menos abono a capital se obtiene el saldo a esa fecha For Each registro In datosRecibosCaja.Rows datosAmortizacion = BD.consultarDatos("SELECT ID_CREDITO,ID_RECIBO_CAJA FROM AMORTIZACIONES WHERE ID_CREDITO='"&_ cmbNumCredito.SelectedValue & "' AND ID_RECIBO_CAJA='" & registro(0) & "'") If datosAmortizacion.Rows.Count = 0 Then 'Validar que la consulta devuelva datos If periodo = 0 Then 'Validar si es el primer registro a insertar, se envia otra consulta sin cálculos SQLUpdate = "INSERT INTO AMORTIZACIONES (ID_AMORTIZACION, PERIODO, SALDO, ID_CREDITO) VALUES(INCREMENTO_ID_AMORTIZACION. NEXTVAL,'" & _CStr(periodo) & "'," & saldoCredito & ",'" & cmbNumCredito.SelectedValue & "')" BD.InsertarDatos(SQLUpdate) 'Se envia la consulta al controlador BD periodo = periodo + 1 'Se incremente en uno el periodo End If 'Calculo de abono a intereses con los días obtenidos de los dos fechas intereses = calc.CalculoIntereses(saldoCredito, VlrInteres, DateDiff(DateInterval.Day, fechaCredito, registro(2))) 'Calculo de abono a capital capital = registro(1) – intereses 'Calculo del saldo a la fecha del capital saldoCredito = saldoCredito – capital SQLUpdate = "INSERT INTO AMORTIZACIONES VALUES(INCREMENTO_ID_AMORTIZACION.NEXTVAL,'" &
```

```
CStr(periodo) & "'," & registro(1) & _ "," & capital & "," & intereses & "," & saldoCredito & ",'" & cmbNumCredito.SelectedValue & "','" & registro(0) & "')" BD.InsertarDatos(SQLUpdate) 'Se envia la consulta al controlador BD periodo = periodo + 1 'Se incremente en uno el periodo End If Next '### Cargar los datos de la amortización calculada datosAmortizacion = BD.consultarDatos("SELECT * FROM AMORTIZACIONES WHERE ID_CREDITO = '" & cmbNumCredito.SelectedValue & "' ORDER BY PERIODO") If datosAmortizacion.Rows.Count > 0 Then With dgvDatosAmortizaciones .DataSource = datosAmortizacion 'Se asigna el origen de datos al data grid con la tabla devuelta por la consulta 'Formato de las celdas en el dataGrid .Columns(0).Visible = False .Columns(6).Visible = False .Columns(7).Name = "RECIBO CAJA" .Columns(2).DefaultCellStyle.Format = "$ ##,##" .Columns(3).DefaultCellStyle.Format = "$ ##,##" .Columns(4).DefaultCellStyle.Format = "$ ##,##" .Columns(5).DefaultCellStyle.Format = "$ ##,##" End With 'Sumatoria total de las columnas sumaAbonos = datosAmortizacion.Compute("SUM(CUOTA)", Nothing) sumaCapital = datosAmortizacion.Compute("SUM(CAPITAL)", Nothing) sumaIntereses = datosAmortizacion.Compute("SUM(INTERES)", Nothing) 'Se asignan las sumatorias a los campos respectivos Me.lblTotalAbonos.Text = FormatCurrency(sumaAbonos, 0) Me.lblTotalCapital.Text = FormatCurrency(sumaCapital, 0) Me.lblTotalIntereses.Text = FormatCurrency(sumaIntereses, 0) Me.lblTotalSaldo.Text = FormatCurrency(datosAmortizacion.Rows(datosAmortizacion.Rows.Count-).Item("SALDO").ToString, 0) Else MsgBox("No se pudo cargar la información de la amortización del crédito.", MsgBoxStyle.Exclamation, "Crédito y Cartera - Mensaje") End If Else MsgBox("No se pudo cargar la información del crédito" & vbCrLf & "Revise los datos e intentelo de nuevo.", MsgBoxStyle.Exclamation, "Crédito y Cartera - Mensaje") End If Catch ex As Exception MsgBox("Ocurrio un error inesperado al consutar el los datos del estudiante. " & vbCrLf & _ "Consulte al administrador del programa sobre este error." & vbCrLf & ex.Message, MsgBoxStyle.Critical, "Crédito y Cartera - Mensaje") End Try End Sub Private Sub InicializarCampos() cmbNumCredito.Text = "Seleccione número" cmbNumCredito.ForeColor = Color.DarkGraylblNombreEstudiante.Text = "" lblNivel.Text = "" lblNumCredito.Text = "" lblPrograma.Text = "" lblTotalAbonos.Text = "" lblTotalCapital.Text = ""lblTotalIntereses.Text = "" lblTotalSaldo.Text = "" dgvDatosAmortizaciones.DataSource = Nothing End Sub 'Evento o procedimientos que se lanza cuando se pone el foco en el objeto Private Sub cmbNumCredito_GotFocus(sender As Object, e As EventArgs) Handles cmbNumCredito.GotFocus cmbNumCredito.ForeColor = Color.Black End Sub Private Sub FrmLiqIntereses_Activated(sender As Object, e As EventArgs) Handles Me.Activated txtIdentEstudiante.Focus() End Sub End Class.
```

13.5 Pantalla Administrar Codeudores

Pantalla que se abre cuando se selecciona la opción de administrar codeudores, donde permite hacer la consulta por número de crédito o por número de identificación. Y el cual permite la captura o entrada de datos del codeudor, dando diferentes opciones de adición, modificar y eliminar.

Ilustración 21. Pantalla o formulario Administrar Codeudores.



Fuente: El autor.

13.5.1 Código Fuente o lógica del negocio

'CLASE: FrmCodeudor 'NOMBRE: Administrar Información de Codeudores 'DISEÑO & CODIGO: Mauricio Ramírez Giraldo Diego Alejandro Madrid Roman 'FECHA: 15 Septiembre de 2013'DESCRIPCIÓN: Buscar información de codeudor por número de crédito o número de documento de identidad. Extrae la información del estudiante/deudor, información del crédito y del codeudor. 'Tiene la opción de adicionar, modificar y eliminar un codeudor. 'VERSION: 1.0 MODIFICACIONES: Imports Oracle.DataAccess.Client Public Class FrmCodeudor Public variosCreditos As Boolean = False 'Validar si un estudiante o deudor tiene mas de dos créditos Dim numCredito As String 'Variable para almacenar temporalmente en el formulario el número del crédito Dim identCodeudorAnt As String 'Variable para almacenar el número de identificación del deudor, antes de modificar o eliminar Dim datosCodeudor As Collection 'Almacenar todos los datos del

```
codeudor antes de modificar 'Evento o método que se lanza antes de cargar el formulario Private Sub FrmCodeudor_Load(sender As Object, e As EventArgs) Handles MyBase.Load BloqDesbloqCamposCodeudor(False) End Sub 'Evento o método que se lanza cuando se da clic en buscar Private Sub btnBuscar_Click(sender As Object, e As EventArgs) Handles btnBuscar.Click 'Declaración de variable para acceso a datos Dim creditos As DataTable Dim BD As clsAccesoBD Try 'Consulta a la vista de los datos del estudiante o deudor y el crédito If Me.rbtNumCredito.Checked Then cargarDatos(txtDatoBusqueda.Text) ElseIf rbtEstudianteDeudor.Checked Then creditos = New DataTable 'Data Table para cargar los registros devueltos por la consulta BD = New clsAccesoBD If txtDatoBusqueda.Text <> "" Then 'Validar que el campo de busqueda contenga algún valor creditos = BD.consultarDatos("SELECT * FROM VW_NUM_CREDITOS WHERE ID_TERCERO='" & txtDatoBusqueda.Text & "'") If creditos.Rows.Count > 0 Then 'Controlar que la consulta devuelva algún registro Me.cmbNumCredito.Visible = True lblTitCreditos.Visible = True cmbNumCredito.DataSource = creditos 'Asignar como datos de origen los registros del dataTable cmbNumCredito.DisplayMember = "NUM_CREDITO" cmbNumCredito.ValueMember = "ID_CREDITO" cmbNumCredito.Text = "Seleccione número" cmbNumCredito.ForeColor = Color.DarkGray 'InicializarCampos() 'Procedimiento que incializa todos los campos de salida del formulario Else MsgBox("El dato de búsqueda no arrojo ningún resultado o el cliente no se encuentra en la base de datos." & vbCrLf & "Revise los datos e intentelo de nuevo.", MsgBoxStyle.Exclamation, "Administración de Codeudores - Mensaje") End If Else MsgBox("No ha digitado ningún dato en el campo de búsqueda 'Identificación'." & vbCrLf & "Verifique e intentelo de nuevo.", MsgBoxStyle.Exclamation, "Administración de Codeudores - Mensaje") End If End If Catch ex As Exception MsgBox("Ocurrio un error inesperado al realizar una consulta o modificación en la tabla codeudores. " & vbCrLf & "Consulte al administrador del programa sobre este error." & vbCrLf & ex.Message, MsgBoxStyle.Critical, "Administración de Coodeudores - Mensaje") End Try End Sub 'Evento o método que se lanza cuando se da clic en adicionar Private Sub btnCodAdicionar_Click(sender As Object, e As EventArgs) Handles btnCodAdicionar.Click Dim BD As New clsAccesoBD Dim resultado1, resultado2 As Short 'Capturar el valor devuelto por la consulta 1=ok 0=fallo Dim nombre As String Try 'Insertar el codeudor como tercero resultado1 = BD.InsertarDatos("INSERT INTO TERCEROS VALUES('" & txtCodIdentificacion.Text & "','" & txtCodNombre1.Text & "','" & txtCodNombre2.Text & "','" & _ txtCodApellido1.Text & "','" & txtCodApellido2.Text & "','" & txtCodDireccion.Text & "','" & txtCodTelefono.Text & "','" & txtCodCorreoElectronico.Text & "')") 'Insertar al tercero como codeudor resultado2 = BD.InsertarDatos("INSERT INTO CODEUDORES VALUES('" & txtCodIdentificacion.Text & "','" & numCredito & "')") If resultado1 = 1 And resultado2 = 1 Then 'Valida si las dos consultar no retornaron error nombre = txtCodNombre1.Text & " " & txtCodNombre2.Text & " " & txtCodApellido1.Text & " "
```

```
& txtCodApellido2.Text 'nombre completo MsgBox("El codeudor: " & nombre & vbCrLf & "Indentificacion: " & txtCodIdentificacion.Text & vbCrLf & "Se adiciono con éxito a la base de datos de codeudores.", MsgBoxStyle.Information, "Crédito y Cartera - Mensaje") 'Habilita botones para poder modificar y eliminar btnCodModificar.Enabled = True btnCodEliminar.Enabled = True BloqDesbloqCamposCodeudor(False) Else MsgBox("Ocurrio un error al insertar el codeudor." & vbCrLf & "Verifique los datos e intentelo de nuevo.", MsgBoxStyle.Exclamation, "Administración de Codeudores - Mensaje") End If Catch ex As Exception MsgBox("Ocurrio un error inesperado al realizar la insercción a la base de datos. " & vbCrLf & "Consulte al administrador del programa sobre este error." & vbCrLf & ex.Message, MsgBoxStyle.Critical, "Administración de Coodeudores - Mensaje") End Try End Sub 'Evento o método que se lanza cuando se da clic en modificar Private Sub btnCodModificar_Click(sender As Object, e As EventArgs) Handles btnCodModificar.Click Try If btnCodModificar.Text = " &Modificar" Then 'Valida que el nombre del botón sea modificar 'Habilita los campos para edición y cambia la acción del botón btnCodModificar.Text = " &Guardar" BloqDesbloqCamposCodeudor(True) identCodeudorAnt = txtCodIdentificacion.Text btnCodAdicionar.Visible = False btnCodEliminar.Visible = False btnCodModificar.ImageIndex = 3 btnCodCancelar.Visible = True 'Se saca una copia de los datos del codeudor. Para restaurar si se cancela la modificación datosCodeudor = New Collection datosCodeudor.Add(txtCodIdentificacion.Text,"id") datosCodeudor .Add(txtCodNombre1 .Text, "nom1") datosCodeudor. Add(txtCodNombre2.Text, "nom2") datosCodeudor.Add( txtCodApellido1.Text, "ape1")datosCodeudor.Add( txtCodApellido2.Text, "ape2")datosCodeudor.Add( txtCodDireccion.Text, "dir")datosCodeudor.Add( txtCodTelefono.Text, "tel") datosCodeudor.Add( txtCodCorreoElectronico.Text, "mail") Else Dim BD As New clsAccesoBD 'Instancia clase de acceso a datos Dim resultado1 As Short Dim mensaje As String 'SQL para actualizar los datos de codeudor resultado1 = BD.InsertarDatos("UPDATE TERCEROS SET ID_TERCERO='" & txtCodIdentificacion.Text & "', NOMBRE1='" & txtCodNombre1.Text & "', NOMBRE2='" & txtCodNombre2.Text & "', " & _ "APELLIDO1='" & txtCodApellido1.Text & "', APELLIDO2='" & txtCodApellido2.Text & "', DIRECCION='" & txtCodDireccion.Text & "', TELEFONO='" & txtCodTelefono.Text & "', CORREO='" & txtCodCorreoElectronico.Text & "' WHERE ID_TERCERO='" & identCodeudorAnt & "'") If resultado1 = 1 Then 'Valida que la actualización no retorne error If identCodeudorAnt <> txtCodIdentificacion.Text Then 'Valida si se modifico el dato principal el número de identificación. mensaje = "El codeudor con indentificación: " & identCodeudorAnt & vbCrLf & "Cambio por el codeudor con identificación: " & txtCodIdentificacion.Text Else mensaje = "El codeudor con indentificacion: " & txtCodIdentificacion.Text End If MsgBox(mensaje & vbCrLf & "Se modifico con éxito en la base de datos de codeudores.", MsgBoxStyle.Information, "Crédito y Cartera - Mensaje")
```

```
BloqDesbloqCamposCodeudor(False) Else MsgBox("Ocurrio un error al insertar el codeudor." & vbCrLf & "Intentelo de nuevo.", MsgBoxStyle.Exclamation, "Administración de Codeudores - Mensaje") End If ResetModificar() 'Resetear estado, nombre, valor de los objetos End If Catch ex As Exception MsgBox("Ocurrio un error inesperado al realizar la actualización a la base de datos. " & vbCrLf & "Consulte al administrador del programa sobre este error." & vbCrLf & ex.Message, MsgBoxStyle.Critical, "Administración de Coodeudores - Mensaje") ResetModificar() End Try End Sub 'Evento o método que se lanza cuando se da clic en eliminar Private Sub btnEliminar_Click(sender As Object, e As EventArgs) Handles btnCodEliminar.Click Dim confirmacion As DialogResult Dim nombre As String Dim BD As New clsAccesoBD 'Instancia de la clase de acceso a datos Dim resultado1, resultado2 As Short nombre = txtCodNombre1.Text & " " & txtCodNombre2.Text & " " & txtCodApellido1.Text & " " & txtCodApellido2.Text 'Nombre completo 'Mensaje que retorna la opción del tipo de eliminación 'Trese opciones: (yes) quita solo vinculo (no) elimina tercero y rol de codeudor (cancelar) no hace nada confirmacion = MsgBox("Desea quitar el vinculo del codeudor: " & nombre & " con número de identificación: " & txtCodIdentificacion.Text & _ " con el crédito número: " & numCredito & ". O desea eliminar el codeudor." & vbCrLf & "(Si) para quitar vinculo." & vbCrLf & _ "(No) para eliminar codeudor." & vbCrLf & "(Cancelar) no hacer nada.", MsgBoxStyle.Question + MsgBoxStyle.YesNoCancel, "Administración Codeudores - Confirmar Eliminación") If confirmacion = Windows.Forms.DialogResult.Yes Then 'Valida si la respuesta fue si 'Enviar la consulta de eliminación resultado1 = BD.InsertarDatos("DELETE FROM CODEUDORES WHERE ID_TERCERO='" & txtCodIdentificacion.Text & "' AND ID_CREDITO='" & numCredito & "'") If resultado1 = 1 Then 'Valida que la eliminación no halla retornado error MsgBox("Se quito el vinculo del codeudor con el crédito seleccionado.", MsgBoxStyle.Information, "Administración de Codeudores - Mensaje") BorrarCamposCodeudor("") 'borra los campos del codeudor Else MsgBox("No se pudo el vinculo del codeudor con el crédito seleccionado." & vbCrLf & "Intentelo de nuevo.", MsgBoxStyle.Critical, "Administración de Codeudores - Mensaje") End If ElseIf confirmacion = Windows.Forms.DialogResult.No Then 'Valida si la respuesta fue no 'Mensaje que retorna la opción de confirmación antes de eliminar confirmacion = MsgBox("Procedera a eliminar el codeudor: " & nombre & " con número de identificación: " & txtCodIdentificacion.Text & " definitivamente de la base de datos." & vbCrLf & _ "Esta acción no se puede revertir. ¿Está seguro que desea eliminar este codeudor?", MsgBoxStyle.Exclamation + MsgBoxStyle.YesNo, "Administración Codeudores - Confirmar Eliminación") If confirmacion = Windows.Forms.DialogResult.Yes Then 'SQL de eliminación de la tabla terceros. Eliminación en cascada tabla codeudores resultado2 = BD.InsertarDatos("DELETE FROM CODEUDORES WHERE ID_TERCERO='" & txtCodIdentificacion.Text & "' AND ID_CREDITO='" & numCredito & "') resultado1 = BD.InsertarDatos("DELETE FROM TERCEROS WHERE
```

```
ID_TERCERO='" & txtCodIdentificacion.Text & "'") If resultado1 = resultado2 Then MsgBox("Se elimino con éxito de la base de datos el codeudor seleccionado.", MsgBoxStyle.Information, "Administración de Codeudores - Mensaje") BorrarCamposCodeudor("") Else MsgBox("No se pudo eliminar de la base de datos el codeudor seleccionado" & vbCrLf & "Intentelo de nuevo.", MsgBoxStyle.Critical, "Administración de Codeudores - Mensaje") End If End If End Sub 'Método personal para cargar todos la informaciópn del dato de consulta Private Sub cargarDatos(ByVal datoBusqueda As String) 'Declaración de variable para acceso a datos Dim datosEstudiante As New DataTable Dim datosCodeudor As New DataTable Dim registro As DataRow 'Declaración de variables generales Dim BD As New clsAccesoBD Dim sql As String 'Dim cantRegistros As Integer Try If datoBusqueda <> "" Then 'Valida que se halla digitado algún dato sql = "SELECT * FROM VW_ESTUDIANTES_CREDITO WHERE CREDITO='" & datoBusqueda & "'" datosEstudiante = BD.consultarDatos(sql) If datosEstudiante.Rows.Count > 0 Then 'Valida que la consulta retorne algún resultado 'Se llena los datos del crédito registro = datosEstudiante.Rows(0) numCredito = registro("CREDITO").ToString Me.lblCreNumero.Text = numCredito Dim fecha As Date fecha = registro("FECHA") Me.lblCreFecha.Text = fecha.Date Me.lblCreValor.Text = FormatCurrency(registro("VALOR"),0)Me.lblCreEstado.Text=registro("ESTADO").ToString Me.lblCreLinea.Text=registro("TIPO_CREDITO"). toString 'Se llenan los datos del estudiante o deudor llenarDatosEstudiante(registro) 'Llamada al procedimiento que llena los campos del estudiante o deudor registro = Nothing 'Se borrar los valor en los campos del codeudor BorrarCamposCodeudor("") 'Consulta a la vista de la base de datos codeudor + crédito datosCodeudor = BD.consultarDatos("SELECT * FROM VW_CODEUDORES_CREDITO WHERE CREDITO='" & datoBusqueda & "'") If datosCodeudor.Rows.Count > 0 Then 'Validamos si la consulta obtuvo algún resultado registro = datosCodeudor.Rows(0) llenarDatosCodeudor(registro) 'Llamada al procedimiento que llena los campos del codeudor btnCodModificar.Enabled = True btnCodEliminar.Enabled = True BloqDesbloqCamposCodeudor(False) Else MsgBox("El crédito número: " & datoBusqueda & vbCrLf & "¡NO! tiene asociado un codeudor.", MsgBoxStyle.Information, "Administración de Codeudores - Mensaje") BloqDesbloqCamposCodeudor(True) btnCodModificar.Enabled = False btnCodEliminar.Enabled = False Me.txtCodIdentificacion.Focus() End If Else MsgBox("El crédito número ( " & datoBusqueda & " ) no se encuentra en la base de datos." vbCrLf & "Verifique e intentelo de nuevo.", MsgBoxStyle.Information, "Administración de codeudores - Mensaje") End If Else MsgBox("No ha digitado ningún dato en el campo de búsqueda" & vbCrLf & "Verifique e intentelo de nuevo.", MsgBoxStyle.Exclamation, "Administración de Codeudores - Mensaje") End If Catch ex As Exception MsgBox("Ocurrio un error inesperado al realizar una consulta o modificación en la tabla codeudores. " & vbCrLf & "Consulte al administrador del programa sobre este error." & vbCrLf & ex.Message, MsgBoxStyle.Critical, "Administración de Coodeudores -Mensaje") End Try End
```

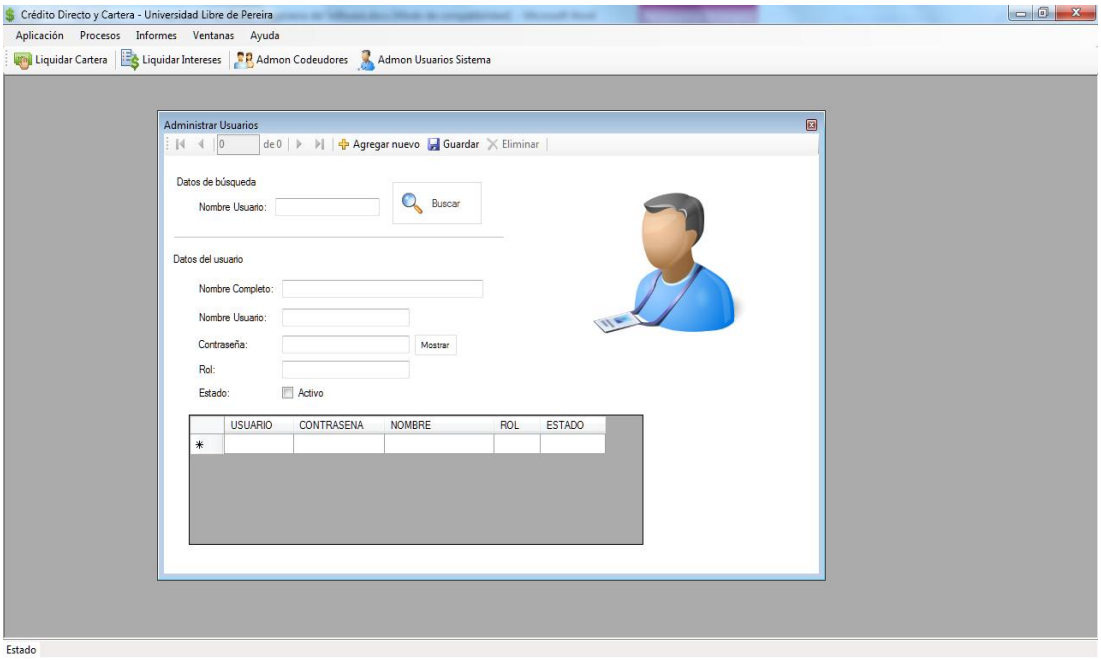
```
Sub 'Método personal para cargar datos del codeudor Private Sub llenarDatosCodeudor(ByVal datos As DataRow) Me.txtCodIdentificacion.Text = datos("IDENTIFICACION").ToString Me.txtCodNombre1.Text= datos("NOMBRE1").ToString Me.txtCodNombre2.Text = datos("NOMBRE2").ToString Me.txtCodApellido1.Text = datos("APELLIDO1").ToString Me.txtCodApellido2.Text = datos("APELLIDO2").ToString Me.txtCodDireccion.Text=datos("DIRECCION" ).ToStringMe.txtCodTelefono.Text=datos("TELEFONO").ToStringMe.txtcodCorreoElectronico .Text = datos("CORREO").ToString End Sub 'Método personal para cargar datos del estudiante/deudor Private Sub llenarDatosEstudiante(ByVal datos As DataRow) Dim tipoNivelAcadem As String txtEstIdentificacion.Text =datos("IDENTIFICACION").ToString Me.txtEstNombre.Text = datos("NOMBRE").ToString Me.txtEstPrograma.Text = datos("PROGRAMA").ToString & " - " & datos("FACULTAD").ToString If datos("PROGRAMA").ToString <> "DERECHO" Then 'Se valida el programa acádemico para mostrar el nivel en años o semestres tipoNivelAcadem = " Semestre" Else tipoNivelAcadem = " Año" End If Me.txtEstNivel.Text = datos("NIVEL").ToString & " " & tipoNivelAcadem 'concatena al nivel. el tipo de nivel Me.txtEstDireccion.Text = datos("DIRECCION").ToString Me.txtEstTelefono.Text = datos("TELEFONO").ToString Me.txtEstCorreoElectronico.Text = datos("CORREO").ToString End Sub 'Evento o método que se lanza primero cuando se activa el formulario Private Sub FrmCodeudor_Activated(sender As Object, e As EventArgs) Handles Me.Activated Me.txtDatoBusqueda.Focus() Me.rbtNumCredito.Checked = True End Sub 'Evento o método al hacer clic en radio botón crédito Private Sub rbtNumCredito_CheckedChanged(sender As Object, e As EventArgs) Handles rbtNumCredito.CheckedChanged Me.lblTitDatoBusqueda.Text = "Crédito N°:" Me.cmbNumCredito.Dispose() Me.cmbNumCredito.Visible = False Me.txtDatoBusqueda.Visible=TrueMe.txtDatoBusqueda.ResetText() Me.rbtNumCredito .Checked =True txtDatoBusqueda.Focus() End Sub 'Evento o método al hacer clic en radio botón identificación Private Sub rbtEstudianteDeudor_CheckedChanged(sender As Object, e As EventArgs) Handles rbtEstudianteDeudor.CheckedChanged Me.lblTitDatoBusqueda.Text="Identificación:" Me.cmbNumCredito.Dispose() Me.cmbNumCredito.Visible=FalseMe.txtDatoBusqueda.Visible=TrueMe.txtDatoBusqueda.ResetText() Me.rbtNumCredito.Checked=True txtDatoBusqueda .Focus() End Sub 'Evento o método al hacer clic en radio buton cancelar Private Sub btnCodCancelar_Click(sender As Object, e As EventArgs) Handles btnCodCancelar.Click txtCodIdentificacion.Text=datosCodeudor("id").ToString txtCodNombre1.Text= datosCodeudor("nom1").ToString txtCodNombre2.Text= datosCodeudor("nom2").ToStringtxtCodApellido1.Text=datosCodeudor("ape1").ToString txtCodApellido2.Text = datosCodeudor("ape2").ToString txtCodDireccion.Text = datosCodeudor("dir").ToString txtCodTelefono.Text = datosCodeudor("tel").ToString txtCodCorreoElectronico.Text =
```

```
datosCodeudor("mail").ToString ResetModificar() datosCodeudor.Clear()
datosCodeudor = Nothing End Sub 'Método personal para desbloquear o bloquear
campos de codeudor Private Sub BloqDesbloqCamposCodeudor(ByVal valor As
Boolean) txtCodIdentificacion.Enabled = valor txtCodNombre1.Enabled=valor
txtCodNombre2.Enabled=valor txtCodApellido1.Enabled = valor
txtCodApellido2.Enabled = valor txtCodDireccion.Enabled = valor
txtCodTelefono.Enabled = valor txtCodCorreoElectronico.Enabled = valor End
Sub 'Método personal para borrar o inicializar campos del codeudor Private Sub
BorrarCamposCodeudor(ByVal valor As String) txtCodIdentificacion.Text = valor
txtCodNombre1.Text = valor txtCodNombre2.Text = valor txtCodApellido1.Text
= valor txtCodApellido2.Text = valor txtCodDireccion.Text = valor
txtCodTelefono.Text = valor txtCodCorreoElectronico.Text = valor
txtCodIdentificacion.Focus() End Sub 'Método personal para resetear campos y
botones cuando se da clic en modificar Private Sub ResetModificar()
btnCodModificar.Text= "&Modificar" btnCodModificar.ImageIndex= 1
btnCodAdicionar.Visible=True btnCodEliminar.Visible=True BloqDesbloqCampos
Codeudor(False) btnCodCancelar.Visible=False txtDatoBusqueda .Focus() End
Sub 'Evento o método que se lanza cuando se cambia el valor del comboBox
número de crédito Private Sub cmbNumCredito_SelectedIndexChanged(sender
As Object, e As EventArgs) Handles cmbNumCredito.SelectedIndexChanged
cmbNumCredito.ForeColor = Color.Black End Sub 'Evento o método que se lanza
cuando se selecciona un indice del comboBox número de crédito Private Sub
cmbNumCredito_SelectionChangeCommitted(sender As Object, e As EventArgs)
Handles cmbNumCredito.SelectionChangeCommitted
cargarDatos(cmbNumCredito.SelectedValue) End Sub End Class.
```

13.6 Pantalla Administrar Usuarios

Pantalla que se abre cuando se selecciona la opción de administrar usuario del sistema, permite hacer la búsqueda por nombre de usuario en el sistema para modificar o inhabilitar usuarios; permitiendo igualmente adicionar nuevos usuarios para acceso al sistema.

Ilustración 22. Pantalla o formulario de Administración de Usuarios.



Fuente: El autor.

13.6.1 Código fuente o lógica del negocio

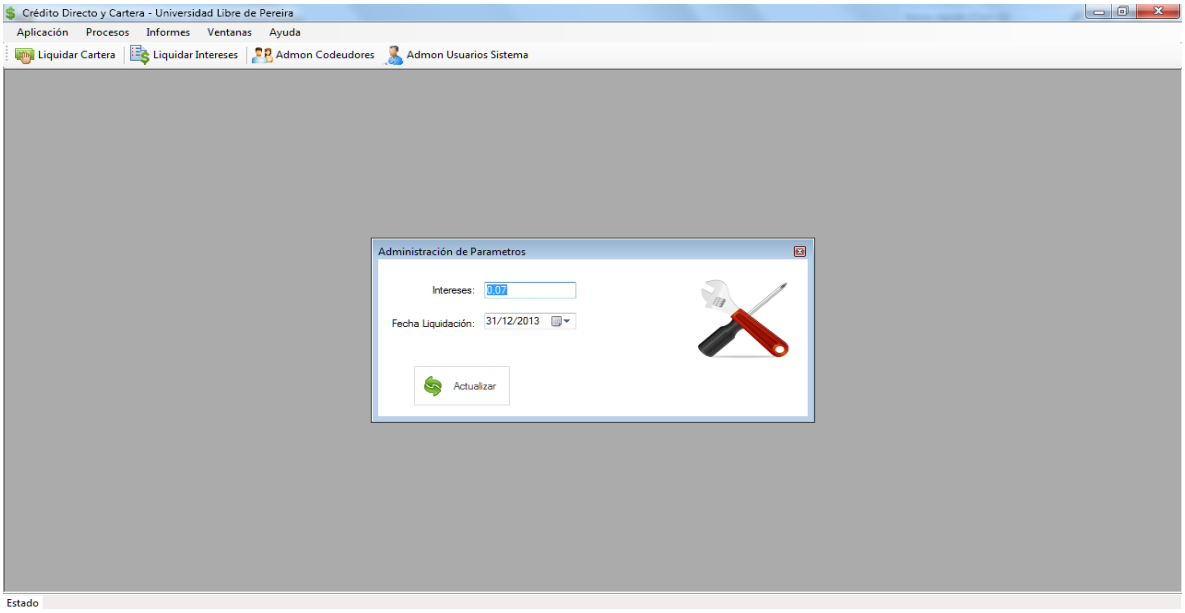
```
'CLASE: FrmAdmonUsuario 'NOMBRE: Administrar Usuarios del Sistema
'DISEÑO & CODIGO:  Mauricio Ramírez Giraldo Diego Alejandro Madrid Roman
'FECHA: 15 Septiembre de 2013 'DESCRIPCIÓN: Muestra toda la información de
todos los usuarios. Tiene la funcionalidad de buscar, eliminar y modificar.
'VERSION: 1.0 'MODIFICACIONES: Imports Oracle.DataAccess.Client Public
Class FrmAdmonUsuarios Dim datos As New DataTable Private adaptadorOracle
As OracleDataAdapter Private Sub FrmAdmonUsuarios_Load(sender As Object, e
As EventArgs) Handles MyBase.Load enlazarDatos() End Sub Private Sub
enlazarDatos() Dim claves(0) As DataColumn Dim conexion As OracleConnection
Dim cadenaConexion As String Try cadenaConexion = "Data
Source=localhost;Persist Security Info=True; conexion = New
OracleConnection(cadenaConexion) adaptadorOracle= New
OracleDataAdapter("SELECT ID_USUARIO AS USUARIO,CONTRASENA
,NOMBRE,ID_ROL AS ROL,ESTADO FROM USUARIOS", conexion)
adaptadorOracle.Fill(datos) 'Llenar el dataTable 'Definimos la clave primaria
en el bindingSuorce claves(0) = New DataColumn claves(0) =
datos.Columns("USUARIO") datos.PrimaryKey = claves 'Asignamos clave
bnsDatos.DataSource = datos 'Asignamos al origen del bindingSource los datos
devueltos por la consulta bngRegistros.BindingSource = bnsDatos 'Asignamos los
```

```
registros dgvDatos.DataSource = bnsDatos 'Origen de datos del datGrid los datos del bindingSourcedgvDatos.AutoResizeColumns() 'Ajustamos todas las columnas 'Vinculamos todos los campos de texto con el binding txtNombreCompleto.DataBindings.Add(New Binding("Text", bnsDatos, "NOMBRE", True)) txtContraUsuario.DataBindings.Add(New Binding("Text", bnsDatos, "CONTRASENA", True)) txtNomUsuario.DataBindings.Add(New Binding("Text", bnsDatos, "USUARIO", True)) txtRol.DataBindings.Add(New Binding("Text", bnsDatos, "ROL", True)) chbEstado.DataBindings.Add(New Binding("CheckState", bnsDatos, "ESTADO", True)) Catch ex As Exception MessageBox.Show("Ocurrio un error inesperado al cargar los datos. Error: "& ex.Message, Me.Text, MessageBoxButtons.OK, MessageBoxIcon.Error) End Try End Sub 'Método para quitar formato de passwords al campo de contraseña Private Sub btnVerTexto_Click(sender As Object, e As EventArgs) Handles btnVerTexto.Click If txtContraUsuario.UseSystemPasswordChar Then txtContraUsuario.UseSystemPasswordChar = False btnVerTexto.Text = "Ocultar" Else txtContraUsuario.UseSystemPasswordChar = True btnVerTexto.Text = "Mostrar" End If End Sub 'Evento o método que se lanza cuando se da clic en buscar Private Sub btnBuscar_Click(sender As Object, e As EventArgs) Handles btnBuscar.Click Dim datoFila As Short datoFila = bnsDatos.Find("USUARIO", txtDatoBusqueda.Text) 'Obtenemos la fila donde se encontro el dato de búsqueda If datoFila <> -1 Then 'Si retorna -1 no encontro nada bnsDatos.Position = datoFila 'Ponemos la posición del cursor en la fila encontrada Me.dgvDatos.FirstDisplayedScrollingRowIndex = datoFila 'Seleccionamos la fila de dataGridView en la fila encontrada Else MessageBox.Show("El dato de búsqueda no arrojo ningún resultado o no existe el usuario.", "Administrar Usuario - Mensaje", MessageBoxButtons.OK, MessageBoxIcon.Question) txtDatoBusqueda.Focus() End If End Sub Private Sub GuardarToolStripButton_Click(sender As Object, e As EventArgs) Handles GuardarToolStripButton.Click 'Si se han producido cambios en el DataTable y la collecion GetChanges esta vacia se actualiza el conjunto de datos. If Not datos.GetChanges() Is Nothing Then Try adaptadorOracle.Update(datos) 'Actualizamos el dataTable con el origen de datos datos.AcceptChanges() 'Acepatamos cambios Catch ex As Exception MessageBox.Show("Ocurrio un error inesperado. No se puedo guardar los datos. Error: " & ex.Message, Me.Text, MessageBoxButtons.OK, MessageBoxIcon.Error) datos.RejectChanges() End Try End If End Sub Private Sub BindingNavigatorDeleteItem_Click(sender As Object, e As EventArgs) Handles BindingNavigatorDeleteItem.Click 'Si vamos a elimiar un registro, debemos pedirle siempre confirmacion al usuario. if MsgBox("Seguro que desea elimina?", MsgBoxStyle.YesNo)=MsgBoxResult.No Then Me.bnsDatos.CancelEdit() datos.RejectChanges() Exit Sub End If End Sub End Class
```

13.7 Pantalla Administrar parámetros

Pantalla que se abre cuando se selecciona la opción parámetros de la barra de menú, permite modificar y visualizar los diferentes parámetros globales que se manejan en el sistema.

Ilustración 23. Pantalla o formulario de Administración de Parámetros.



Fuente: El autor.

13.7.1 Código fuente o lógica del negocio

```
'CLASE: FrmParametros 'NOMBRE: Administración de parámetros 'DISEÑO & CODIGO: Mauricio Ramírez Giraldo Diego Alejandro Madrid Roman 'FECHA: 15 Septiembre de 2013 'DESCRIPCIÓN: Al abrir el formulario se pueden visualizar los parametros del sistema. Se pueden modificar y actualizar en la base de datos. VERSION: 1.0 'MODIFICACIONES: Public Class FrmParametros Dim BD As clsAccesoBD 'Evento o método que se lanza cuando se da clic en el botón actualizar Private Sub btnActualizar_Click(sender As Object, e As EventArgs) Handles btnActualizar.Click Dim resultado As Integer BD = NewclsAccesoBD If txtIntereses.Text <> "" Then resultado = BD.EjecutarComando("UPDATE PARAMETROS SET INTERES='" & txtIntereses.Text "', FECHA_LIQ='" & dtpFechaLiq.Value & "'") If resultado = 1 Then 'Se actualizar el valor  de los intereses en formulario principal, para el envió actualizado 'de del valor a los
```

formularios que dependen de este valor MDIFormMain.interes=CDec(Me.txtIntereses .Text) MessageBox.Show("Parametros actualizados con éxito.", "Administrar Parametros - Mensaje", MessageBoxButtons.OK, MessageBoxIcon.Information) Me.Close() End If Else MessageBox.Show("El parametro de 'INTERES' no puede estar nulo. No ha digitado ningún valor", "Administrar Parametros - Mensaje", MessageBoxButtons.OK, MessageBoxIcon.Error) txtIntereses.Focus() End If End Sub 'Evento o método que se lanza cuando se carga el formulario Private Sub FrmParametros_Load(sender As Object, e As EventArgs) Handles Me.Load Dim datos As DataTable BD = New clsAccesoBD Try datos = BD.consultarDatos("SELECT * FROM PARAMETROS") 'Consulta a la tabla para extraer todos los parámetros If datos.Rows.Count > 0 Then 'Se valida que la consulta evuelva algún resultado txtIntereses.Text = CDec(datos.Rows(0).Item("INTERES")) dtpFechaLiq.Value = datos.Rows(0).Item("FECHA_LIQ") End If 'Libererar recursos utilizados por los objetos datos.Dispose() BD = Nothing Catch ex As Exception MessageBox.Show("Ocurrio un error inesperado al cargar los parametros. Error: " & ex.Message, "Administrar Parametros - Mensaje", MessageBoxButtons.OK, MessageBoxIcon.Error) End Try End Sub End Class.

13.8 Otras Clases

13.8.1 Clase de Acceso a Datos

'CLASE: clsAccesoBD 'NOMBRE: Clase controladora del acceso a datos 'CODIGO: Mauricio Ramírez Giraldo Diego Alejandro Madrid Roman 'FECHA: 15 Septiembre de 2013 'DESCRIPCIÓN: Clase que contiene varios métodos para consulta información e insertar datos en la Base de datos 'VERSION: 1.0 'MODIFICACIONES: Imports Oracle.DataAccess.ClientPublic Class clsAccesoBD Public adaptador As OracleDataAdapter 'Adaptador para acceso a oracle Private conexionBD As OracleConnection 'Conexión a oracle Private cadenaConexion As String 'Cadena de conexión 'Constructor Public Sub New() 'Se inicializa la cadena de conexión cadenaConexion = "Data Source=localhost;Persist Security Info=True; " End Sub 'Método para consultar y extraer datos de una tabla especifica. Retorna un dataTable Public Function consultarDatos(cadenaSQL As String) As DataTable Dim datos As New DataTable Try 'Generar consulta a la Base de datos Using conexionBD = New OracleConnection(cadenaConexion), adaptador = New OracleDataAdapter(cadenaSQL, conexionBD.ConnectionString) adaptador.Fill(datos) 'Llenar el DataTable End Using Catch ex As Exception MessageBox.Show("Ocurrio un error al acceder a la base de datos." & vbCrLf & _ "Consulte al administrador sobre este error." & vbCrLf & "Error: " & ex.Message, "Crédito y cartera - Mensaje", MessageBoxButtons.OK, MessageBoxIcon.Error)

```
End Try Return datos End Function 'Método para enviar comandos a la base de datos. Update, Delete e insert Public Function EjecutarComando(ByVal cadenaSQL As String) As Short Dim resultadoComando As Short Dim comandoSQL As OracleCommand Try Using conexionBD = New OracleConnection(cadenaConexion) comandoSQL = New OracleCommand(cadenaSQL, conexionBD) conexionBD.Open() 'Se abre la conexión para acceso en modo conectado resultadoComando = comandoSQL.ExecuteNonQuery 'Ejecutamos el comando End Using Catch ex As OracleException 'Manejo de diferentes mensajes de excepción. Select Case ex.Number Case 1: MessageBox.Show("El campo principal del registro que desea adicionar ya existe en la base de datos." & vbCrLf & _ "No se permiten valores duplicados en esta tabla." & vbCrLf & "Error: " & ex.Message, "Crédito y cartera - Mensaje", MessageBoxButtons.OK, MessageBoxIcon.Error) Case 1400 : MessageBox.Show("Hay campos obligatorios que no tienen información. Verifique en inténtelo de nuevo" & vbCrLf & _"Error: " & ex.Message, "Crédito y cartera - Mensaje", MessageBoxButtons.OK,MessageBoxIcon.Error) Case Else : MessageBox.Show("Ocurrio un error inesperado en la inserción de los datos." & vbCrLf & _"Error: " & ex.Message, "Crédito y cartera - Mensaje", MessageBoxButtons.OK, MessageBoxIcon.Error) End Select End Try Return resultadoComando End Function End Class.
```

13.8.2 Clase varios métodos

```
'CLASE: clsBiblioteca 'NOMBRE: Clase contenedora de métodos 'CODIGO: Mauricio Ramírez Giraldo 'FECHA: 15 Septiembre de 2013 'DESCRIPCIÓN: Clase que contiene varios métodos que efectúa cálculos de intereses y otras funcionalidades' Utilizadas en toda la aplicación 'VERSION: 2.0 'MODIFICACIONES: Public Class clsBiblioteca Public interes As Decimal 'Metodo para calcular intereses según formula estipulada. Recibe tres parametros Public Function CalculoIntereses(ByVal valor As Long, ByVal porcentajeInteres As Decimal, ByVal diasALiquidar As Integer) As Long Return Math.Round((valor * porcentajeInteres) / 30 * diasALiquidar) End Function 'Método para calcular el valor de los intereses diarios. Recibe dos parámetros Public Function CalculoInteresesDiarios(ByVal valor As Long, ByVal porcentajeInteres As Decimal) As Long Return (valor * porcentajeInteres) / 30 End Function 'Método que instancia la otra clase de acceso a datos para consultar una vista y obtener el listado 'De todos los créditos según el número de identificación pasado como parámetros Public Function buscarCreditos(ByVal identificacion As String) Dim BD As New clsAccesoBD Dim numCreditos As New DataTable Return BD.consultarDatos("SELECT * FROM VW_NUM_CREDITOS WHERE ID_TERCERO='" & identificacion & "'") End Function 'Método que consulta el parametro intereses de la base de datos y lo retorna como valor Public Function intereses() As Decimal Dim datos As DataTable Dim BD As New clsAccesoBD Try
```

```
datos=BD.consultarDatos("SELECT INTERES FROM PARAMETROS") If datos.Rows.Count > 0 Then 'Validar que la consulta devuelva algún resultado interes = CDec(datos.Rows(0).Item("INTERES")) 'Cargar el valor de intereses End If  'Liberar recursos de los objetos datos.Dispose() BD = Nothing datos.Dispose()  Catch ex As Exception MessageBox.Show("Ocurrio un error inesperado al cargar los parametros. Error: " & ex.Message, "Parametros - Mensaje", MessageBoxButtons.OK, MessageBoxIcon.Error) End Try Return interes 'Retornamos el valor End Function End Class
```

14 PRUEBAS FINALES DE ACEPTACIÓN

14.1 PLAN DE PRUEBAS

Tabla 25. Ficha de la aplicación.

Nombre	Gestión de Créditos Directos y Cartera
Caso de prueba	Créditos directos y cartera de la Universidad Libre de Pereira
Versión	1.0
Estado	Finalizado
Desarrolladores	Mauricio Ramírez Giraldo
Grupo de pruebas	Carlos Alberto Huertas – Diego Alejandro Madrid Roman – Mauricio Ramirez Giraldo

Fuente: El autor.

14.2 Metodología de las pruebas

Con el fin de verificar la funcionalidad de la aplicación, se realiza el diseño de las pruebas teniendo como base la evaluación de diferentes criterios entre los que se encuentran:

Las entradas, la salida esperada y la salida obtenida de cada uno de los módulos que se tengan en estudio. Cada caso de prueba estará conformado de la siguiente manera:

Tabla 26. Metodología de pruebas.

Código	Es un identificador único que permite referenciar cada uno de los casos de prueba. Está compuesto por las siglas CP (Caso prueba).
Modulo	Nombre del módulo al que se le están aplicando las pruebas.

Acción	Respectivas acciones que se desean evaluar en el funcionamiento del módulo respectivo.
Datos de entrada	Información requerida para la realización de la acción.
Salida esperada	Descripción que el analista espera obtener por parte de la aplicación cuando le ha ingresado los datos requeridos para la realización de determinadas acciones.
Salida obtenida	Respuesta real de la aplicación cuando se le pide realizar una acción bajo determinadas condiciones. Se establece según el caso en **Correcto** ó **Incorrecto.**
Error	En caso que la respuesta obtenida del sistema no sea la esperada por el analista, se describe el defecto encontrado.
Estado	De acuerdo a la salida obtenida del sistema: • Sí ésta es igual a la salida esperada, el estado del caso de prueba es concluido. • Sí ésta es diferente a la salida esperada, el estado del caso de prueba es Erróneo. • Sí el caso de prueba ya ha sido realizado y su estado determinado como erróneo, pero ya se han corregido las inconsistencias presentadas, el estado del caso de prueba es Corregido y concluido.

Fuente: El autor.

14.3 ALCANCE DE LAS PRUEBAS

La metodología antes mencionada para la realización de las pruebas será aplicada a los siguientes módulos principales y vitales para garantizar el correcto funcionamiento del producto y para lo cual fue diseñado y que a su vez cumplan los requerimientos funcionales que dieron origen a la aplicación; acompañados de las respectivas acciones a las que estos pueden ser expuestos.

Tabla 27. Casos de pruebas finales.

Nombre del módulo	Acciones
Inicio o autenticación	Ingresar al sistema con: • Datos validos • Datos no validos
Liquidar cartera	• Consultar crédito • Liquidar
Liquidar Intereses	• Consultar crédito • Liquidar
Administrar codeudores	• Consultar por crédito • Consultar por deudor • Adicionar • Modificar • Eliminar

Fuente: El autor.

14.4 EVALUACIÓN DE LOS CASOS DE PRUEBA

14.4.1 Caso de prueba Ingresar al sistema

14.4.1.1 Acción ingresar al sistema con datos validos

Tabla 28. Prueba Inicio o autenticación.

Código	CP1
Modulo	Inicio o autenticación
Acción	Ingresar al sistema con datos validos
Datos de entrada	Nombre usuario: root Contraseña: valida
Salida esperada	Mostar el menú principal
Salida obtenida	Correcta
Error	Ninguno

Estado	Concluido

14.4.1.2 Acción ingresar al sistema con datos no validos

Código	CP2
Modulo	Inicio o login
Acción	Ingresar al sistema con datos no validos
Datos de entrada	Nombre usuario: anabel Contraseña: nada
Salida esperada	Mensaje de error usuario no autenticado. Usuario y/o contraseña no validos
Salida obtenida	Correcta
Error	Ninguno
Estado	Concluido

Fuente: El autor.

14.4.2 Caso de prueba liquidar cartera

14.4.2.1 Acción consultar un crédito

Tabla 29. Prueba consultar crédito.

Código	CP3
Modulo	Liquidar Cartera
Acción	Consultar un crédito
Datos de entrada	Identificación: 18520197
Salida esperada	Mostrar en un listado los créditos adjudicados al estudiantes identificado con la cedula respectiva

Salida obtenida	Correcta
Error	Ninguno
Estado	Concluido

Fuente: El autor.

14.4.2.2 Acción liquidar

Tabla 30. Prueba liquidar cartera

Código	CP4
Modulo	Liquidar Cartera
Acción	Liquidar
Datos de entrada	Fecha: liquidación
Salida esperada	Visualizar en pantalla los cálculos de: • Número de días transcurridos a la fecha de corte desde la fecha de aprobación. • Total de abonos al crédito. • Total de intereses generados. • Total neto. • Saldo total de la deuda a la fecha.
Salida obtenida	Correcta
Error	Ninguno
Estado	Concluido

Fuente: El autor.

14.4.3 Caso de prueba liquidar intereses

14.4.3.1 Acción consultar crédito

Tabla 31. Prueba consultar crédito.

Código	CP5
Modulo	Liquidar Intereses
Acción	Consultar un crédito
Datos de entrada	Identificación: 10258963
Salida esperada	Mostrar en un listado los créditos adjudicados al estudiantes identificado con la cedula respectiva
Salida obtenida	Correcta
Error	Ninguno
Estado	Concluido

Fuente: El autor.

14.4.3.2 Acción liquidar

Tabla 32. Prueba liquidar intereres

Código	CP6
Modulo	Liquidar Intereses
Acción	Liquidar
Datos de entrada	Número de crédito: 1145
Salida esperada	Visualizar en pantalla en una tabla tipo amortización los cálculos de cada pago realizado al crédito: • Valor que se abono a capital • Valor que se abono a intereses • Saldo actual
Salida obtenida	Correcta
Error	Ninguno
Estado	Concluido

Fuente: El autor.

14.4.4 Caso de prueba administrar codeudores

14.4.4.1 Acción consultar por crédito

Tabla 33. Prueba consultar por crédito

Código	CP7
Modulo	Administrar codeudores
Acción	Consultar por crédito
Datos de entrada	Número del crédito:1589
Salida esperada	Listar en pantalla todos los datos del crédito, datos del deudor y codeudor. (Si no existe el codeudor, mostrar mensaje de información)
Salida obtenida	Correcta
Error	Ninguno
Estado	Concluido

Fuente: El autor.

14.4.4.2 Acción consultar por estudiante/deudor

Tabla 34. Prueba consultar por crédito

Código	CP8
Modulo	Administrar codeudores
Acción	Consultar por estudiante/deudor
Datos de entrada	Identificación: 108852369
Salida esperada	• Listar los créditos adjudicados • Listar en pantalla todos los datos del crédito, datos del deudor y codeudor. (Si no existe el codeudor, mostrar mensaje de información)

Salida obtenida	Correcta
Error	Ninguno
Estado	Concluido

Fuente: El autor.

14.4.4.3 Acción adicionar

Tabla 35. Prueba adicionar codeudor.

Código	CP9
Modulo	Administrar codeudores
Acción	Adicionar
Datos de entrada	Identificación: 1088525698 Nombre1: Alejandro Nombre2: null Apellido1: Carmona Apellido2: null Dirección: Cra 45 # 89-85 Teléfono: 2154789632 Correo: alejocarmona13@gmail.com
Salida esperada	Adicionar al codeudor a la base de datos y mostrar mensaje de información que se adiciono sin errores.
Salida obtenida	Correcta
Error	Ninguno
Estado	Concluido

Fuente: El autor.

14.4.4.4 Accion modificar

Tabla 36. Prueba modificar codeudor.

Código	CP10
Modulo	Administrar codeudores
Acción	Modificar
Datos de entrada	Nombre1: Maria Nombre2: Yolanda Apellido1: Carmona Apellido2: null Dirección: Calle 96 # 23-85 Teléfono: 315289745 Correo: mariayoca@gmail.com
Salida esperada	Modificar los datos del codeudor en la base de datos y mostrar mensaje de información que se modifico sin errores.
Salida obtenida	Correcta
Error	Ninguno
Estado	Concluido

Fuente: El autor.

14.4.4.5 Acción eliminar

Tabla 37. Prueba eliminar codeudor.

Código	CP11
Modulo	Administrar codeudores

Acción	Eliminar
Datos de entrada	Ninguno
Salida esperada	Eliminar de la base de datos codeudor visualizado en pantalla y mostrar mensaje de advertencia antes de eliminar.
Salida obtenida	Correcta
Error	Ninguno
Estado	Concluido

Fuente: El autor.

15 MANUAL DEL USUARIO

15.1 INTRODUCCIÓN

15.1.1 Propósito y descripción de la aplicación

Es una aplicación o prototipo a nivel de software y con arquitectura para escritorio que tiene como finalidad apoyar el proceso de gestión de la cartera de créditos educativos directos que otorga la universidad libre de Pereira a sus estudiantes.

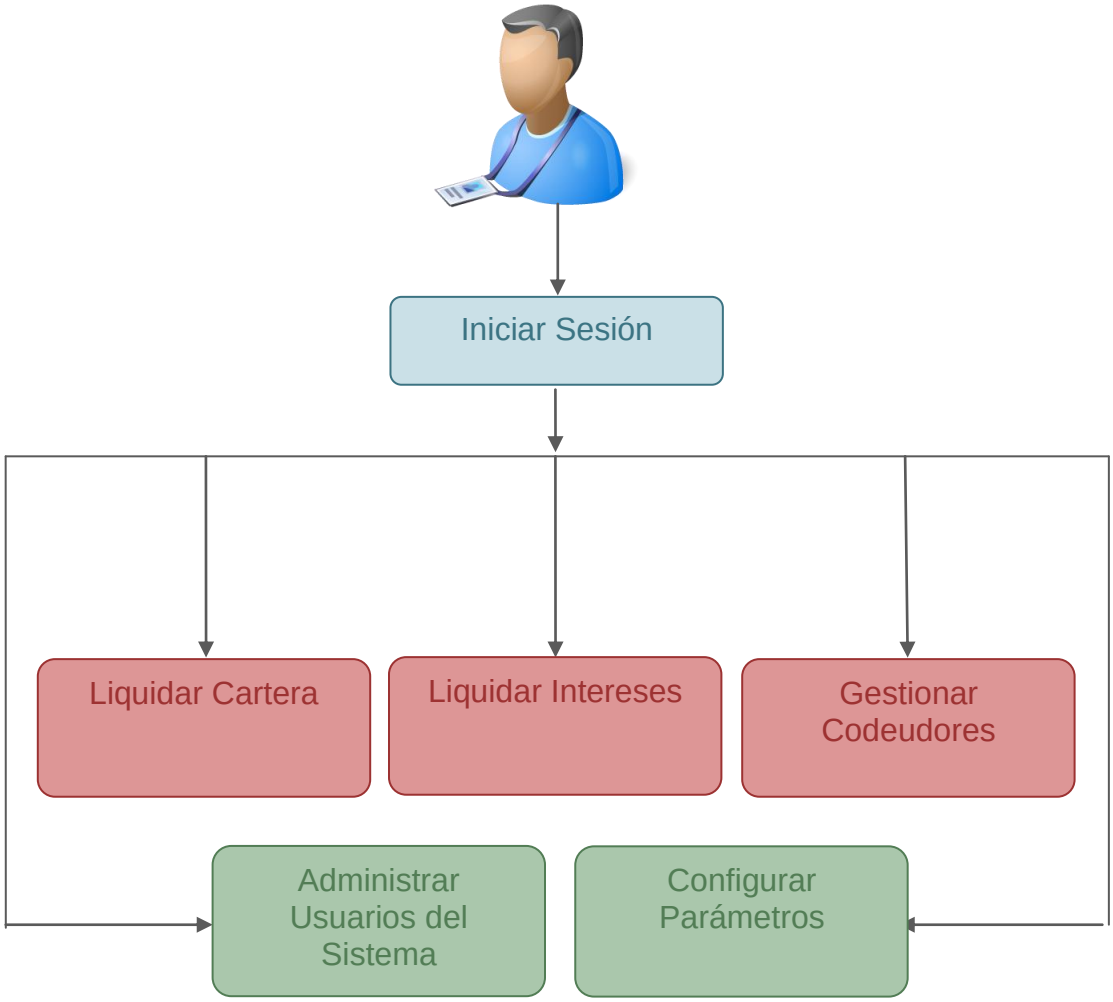
Es una utilidad que agiliza y sistematiza el proceso de liquidación de créditos de forma más eficiente para el usuario. Remplaza la necesidad de crear archivos en Excel por cada deudor y/o crédito; archivos en los cuales se realizaban modificaciones manuales a las formulas para liquidar la cartera y/o sus respectivos intereses, para poder obtener un resultado final de liquidación. Garantizando con esta nueva aplicación un resultado más confiable, seguro y sin re-procesos. Contiene un modulo donde se pueden adicionar, modificar y eliminar codeudores y asociarlos a su respectivo crédito. Como también una utilidad de restringir el acceso a personas no autorizadas. Y a su vez suple algunas carencias comprobadas que posee el software ERP de la universidad (SINU) para administrar dicha cartera de manera útil y parametrizable.

15.1.2 Propósito del documento

Este manual pretende acercar y/o enseñar al usuario la manera de cómo funciona la aplicación, y de cómo obtener los mejores resultados, aprovechando de manera más eficiente todo su potencial. E igualmente garantizar que después de la lectura de este manual, se tenga una correcta comprensión de su funcionalidad y brindar una guía de cómo realizar los diferentes procesos y de cómo es la interacción con los mismos.

15.1.3 Esquema de funcionalidad

Ilustración 24. Esquema de funcionalidad del sistema.



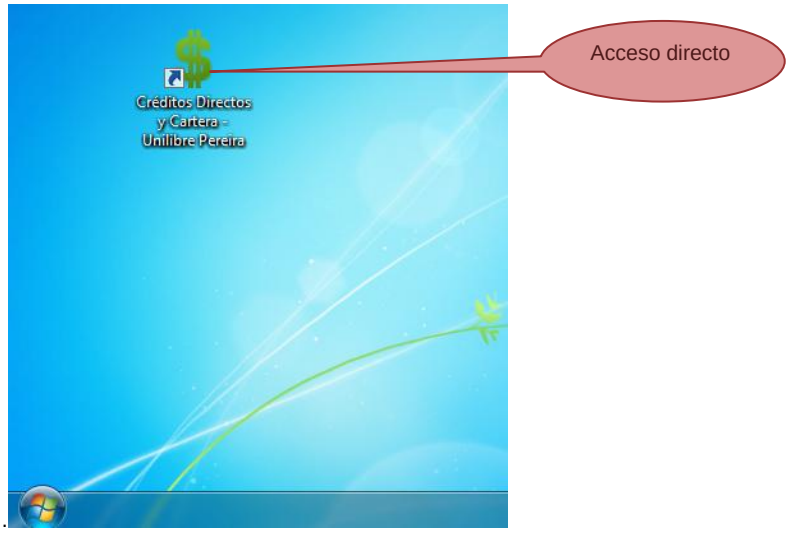
Fuente: El autor.

15.2 CONCEPTOS IMPORTANTES

15.2.1 ¿Cómo abrir la aplicación?

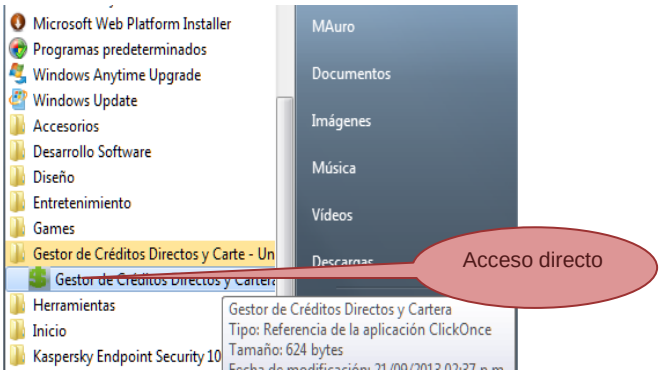
Para acceder a la aplicación se debe abrir el acceso directo que se encuentra en el escritorio de Windows. En la siguiente imagen se resalta el icono del acceso directo en un escritorio de Windows de ejemplo:

Ilustración 25: Ubicación del acceso directo a la aplicación



Fuente: El autor.

Ilustración 26. Acceso directo menú de inicio.



Fuente: El autor.

Para tener en cuenta: Para ingresar al sistema como administrador por primera vez, y crear los diferentes usuarios del sistema, se ingresa de la misma forma como se describe a continuación en el siguiente numeral. Con previa socialización de los datos de acceso como administrador por defecto. Por razones de seguridad no descritos en este manual del usuario.

15.2.2 ¿Cómo Acceder al Sistema?

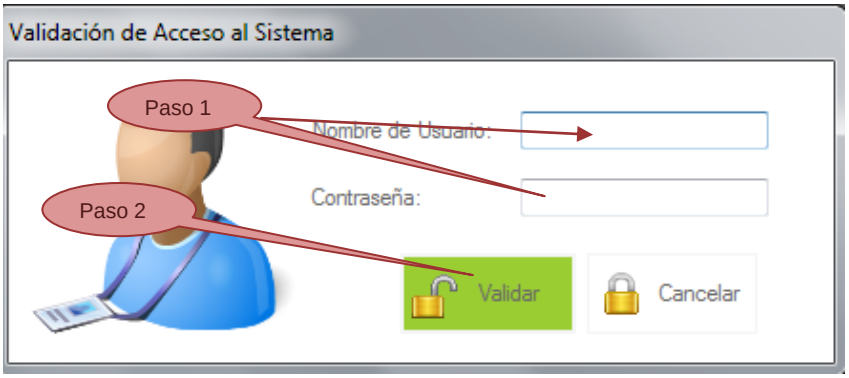
Al abrir la aplicación se muestra el pantallazo principal o inicial donde el sistema solicita los datos de autenticación en el sistema para empezar a utilizar la aplicación.

A continuación se describen los pasos y el formulario de autenticación:

Paso 1: Digitar los datos de autenticación en el sistema en cada campo correspondiente de “Nombre de Usuario:” y “Contraseña:” previamente creados por el administrador de la aplicación.

Paso 2: Dar clic en al botón resaltado en verde “Validar”.

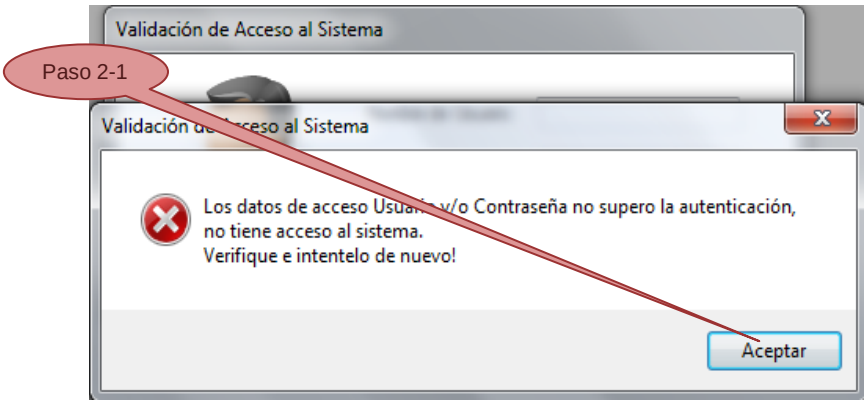
Ilustración 27: Formulario de acceso al sistema



Fuente: El autor.

Paso 2-1: Si los datos no son correctos el sistema muestra un mensaje de error al cual se le debe dar clic en “Aceptar” para cerrarlo. Y el cual se describe en detalle en la siguiente imagen:

Ilustración 28: Mensaje de acceso denegado.



Fuente: El autor.

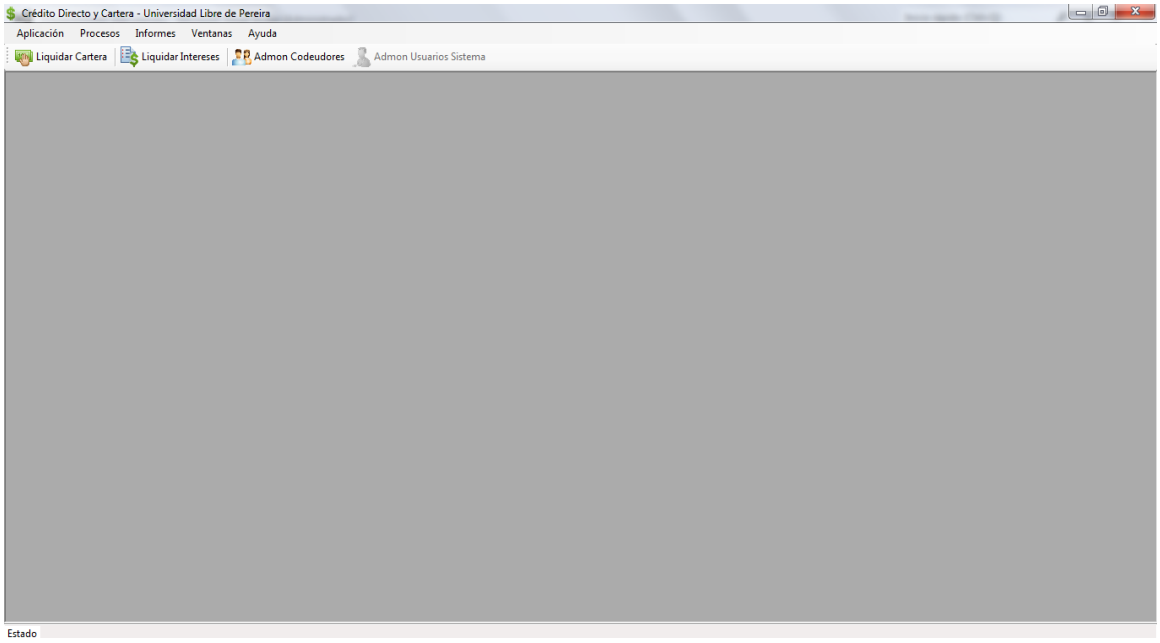
Paso 2-2: Si los datos digitados pasan la validación se cierra el formulario anterior y se muestra al usuario una barra de menú y una de herramientas donde se encuentras los ítems e iconos de acceso a los diferentes procesos. En la imagen del siguiente numeral (3) se describe en detalle estos menús.

15.3 DESCRIPCION DE LA APLICACIÓN

A continuación se describen todos los componentes del sistema como la barra de menú, sus diferentes ítems y opciones de acceso, como también la barra de herramientas y la descripción de cada botón y su funcionalidad.

15.3.1 Pantalla Inicial o Principal

Ilustración 29: Pantalla principal de la aplicación.

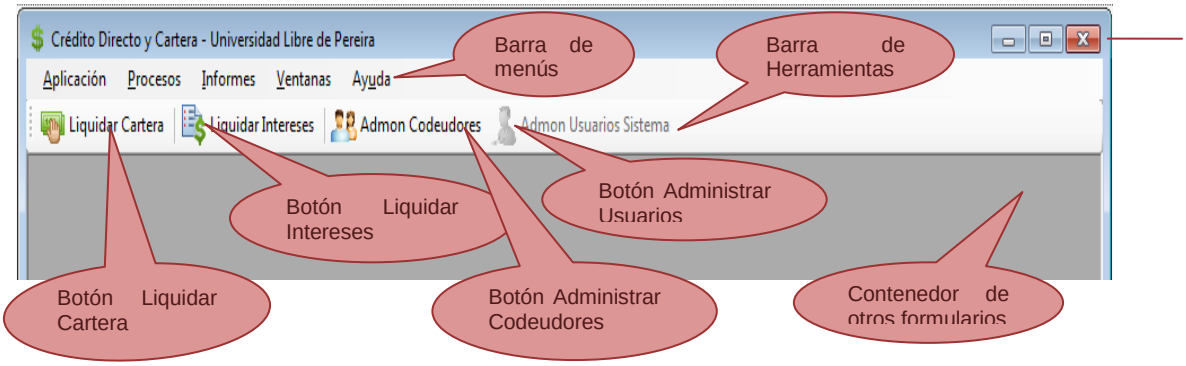


Fuente: El autor.

15.3.2 Descripción de componentes

En la siguiente imagen se describe los diferentes menús que comprenden la aplicación y una breve descripción de su funcionalidad. El detalle de cada funcionalidad se describe en el numeral (4) en la sesión “Guía de Uso”.

Ilustración 30: Descripción de componentes principales de la aplicación.

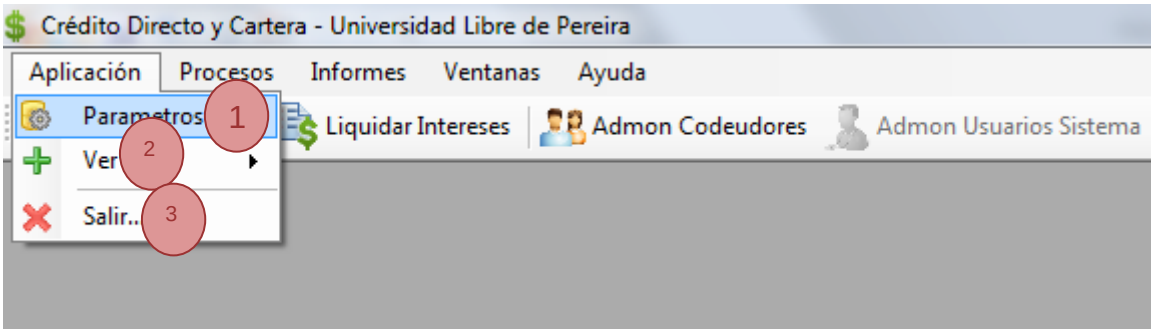


Fuente: El autor.

15.3.3 Barra de Menús

En las siguientes imágenes se describirán los ítems de cada menú y la funcionalidad que realizan dentro de la aplicación:

Ilustración 31: Descripción del menú "Aplicaciones"



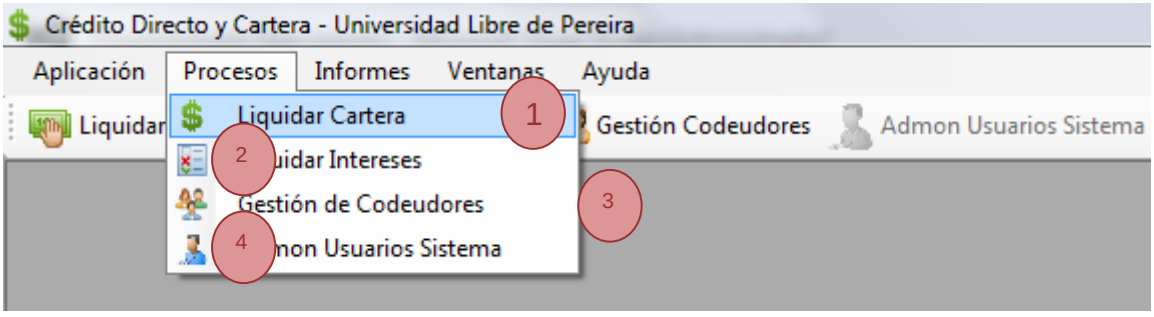
Fuente: El autor.

1 **Parámetros**: Opción que abre el formulario donde nos da la opción de modificar los diferentes parámetros globales que maneja la aplicación.

2 **Ver**: nos permite ocultar o visualizar las dos barras de Menú o la barra de herramientas.

3 **Salir**: Cierra la aplicación y todos los formularios abiertos.

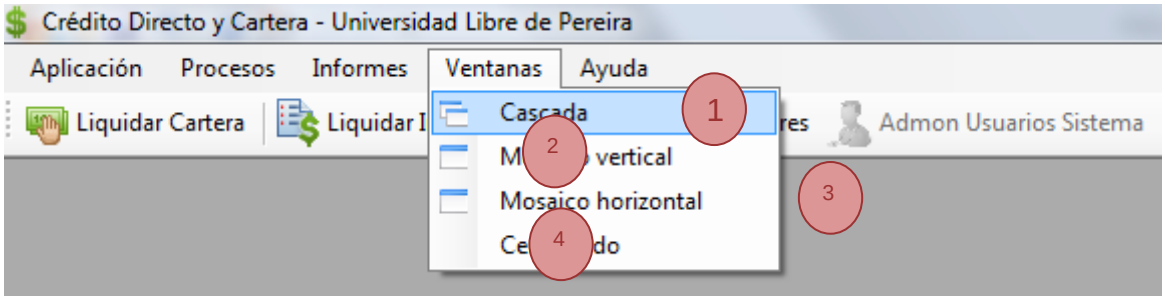
Ilustración 32: Descripción del menú "Procesos".



Fuente: El autor.

1. **Liquidar Cartera:** Abre el formulario que permite liquidar la cartera por estudiante.
2. **Liquidar Intereses:** Abre el formulario que permite liquidar los intereses por estudiante y por cada crédito.
3. **Gestión de Codeudores:** Abre el formulario de gestión de codeudores, donde permite adicionar, modificar y eliminar.
4. **Admon Usuarios Sistema:** Abre formulario que permite administrar los usuarios que pueden ingresar al sistema. Permite adicionar, modificar e inhabilitar.

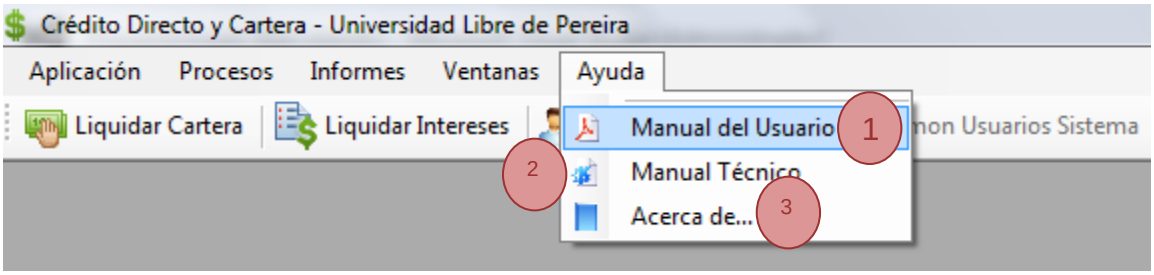
Ilustración 33: Descripción del menú "Ventanas".



Fuente: El autor.

1. **Cascada**: Distribuye todas las ventanas abiertas en una abanico en cascada.
2. **Mosaico Vertical:** Distribuye todas las ventanas en un orden vertical en seguidilla.
3. **Mosaico Horizontal:** Distribuye todas las ventanas en un orden horizontal en seguidilla.
4. **Cerrar Todo:** cierra todas las ventas abiertas, menos la ventana principal o inicial.

Ilustración 34: Descripción del menú "Ayuda".



Fuente: El autor.

1 **Manual del usuario:** Acceso directo desde la aplicación para abrir este documento que se está visualizando.

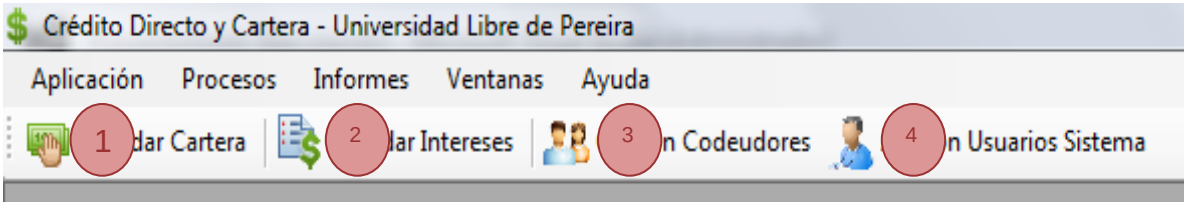
2 **Manual Técnico:** Acceso directo desde la aplicación para abrir el manual técnico de la aplicación.

3 **Acerca de:** Pantalla de visualización de créditos de la aplicación.

15.3.4 Barra de Herramientas

En las siguientes imágenes se describirán los botones de la barra de herramienta, y la funcionalidad que realizan dentro de la aplicación:

Ilustración 35: Descripción de la barra de herramientas.



Fuente: El autor.

1 **Liquidar Cartera:** Abre una ventana donde se pude consultar por estudiante y sus respectivos créditos, donde se puede realizar la respectiva liquidación de cartera.

2 **Liquidar Intereses:** Abre una ventana donde se pude consultar por estudiante y sus respectivos créditos, donde se puede realizar la respectiva liquidación de sus intereses.

3 **Gestión Codeudores:** Abre una ventana donde se puede consulta por estudiante o por crédito, visualizar su respectivo codeudor. Donde permite adicionar en caso que no tenga asociado un codeudor, modificar o eliminar.

4 **Admon Usuarios Sistema:** Abre una ventana donde se puede visualizar todos los usuarios del sistema. Se puede buscar por nombre de usuario y permite la opción de adicionar nuevos usuarios, modificar datos o en su efecto deshabilitar, no se permite eliminar usuarios del sistema.

Para tener en cuenta: Solo los usuarios con el rol de administrador puede utilizar este módulo, los demás usuario ven esta opción deshabilitada por defecto.

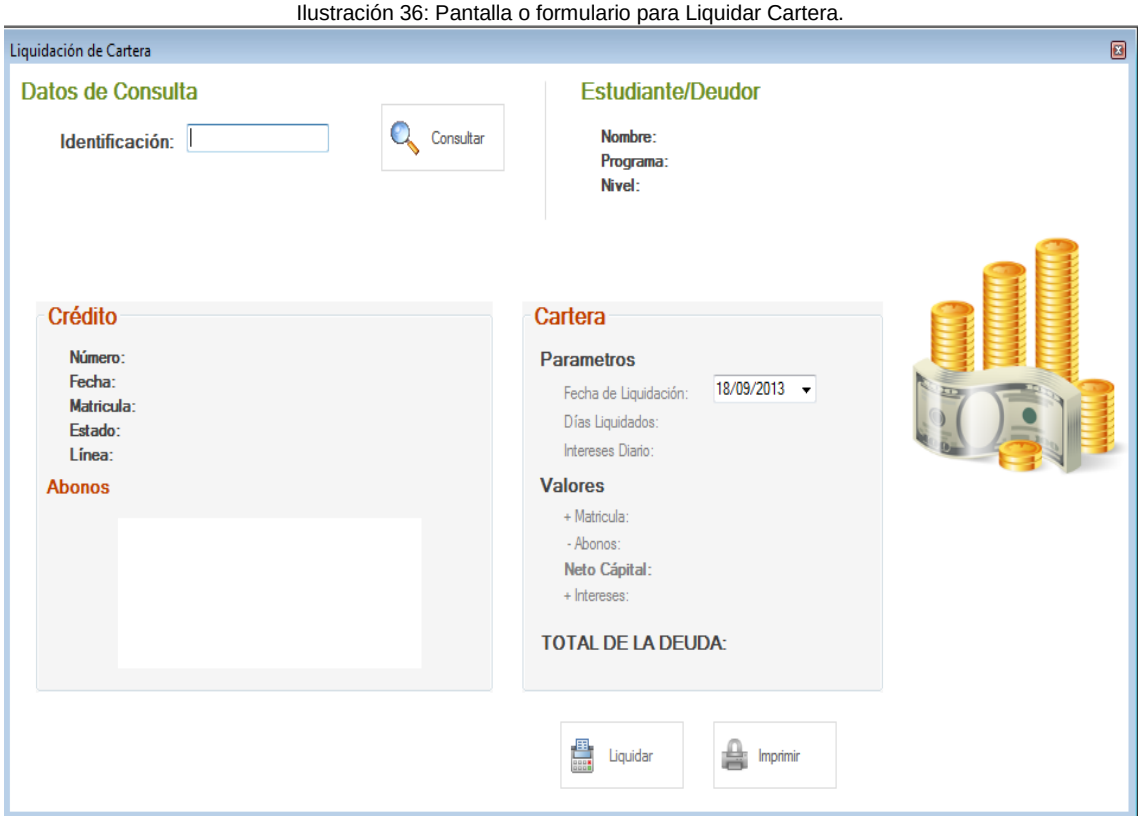
15.4 GUIA DE USO Y FUNCIONAMIENTO

15.4.1 Liquidar Cartera

Es el proceso por el cual se efectúa la correspondiente liquidación del crédito, donde el sistema calcula automáticamente el número de días transcurrido entre la fecha de aprobación del crédito y la fecha de liquidación seleccionada por el usuario; con esta información se puede proceder a liquidar el crédito, calculando el valor incurrido por intereses, el valor total de abonos realizados al crédito y el saldo neto del mismo, teniendo con todos estos valores el total adeudado a la fecha.

A continuación se muestra la pantalla con la interface inicial del formulario.

Ilustración 36: Pantalla o formulario para Liquidar Cartera.



Fuente: El autor.

A continuación se detalla cada paso que se debe seguir para lograr realizar la correcta liquidación de la cartera:

Paso 1: Para poder llegar a utilizar o abrir este módulo se debe seleccionar la opción “Liquidar Cartera”. Que se encuentra en la barra de menú o de herramientas.

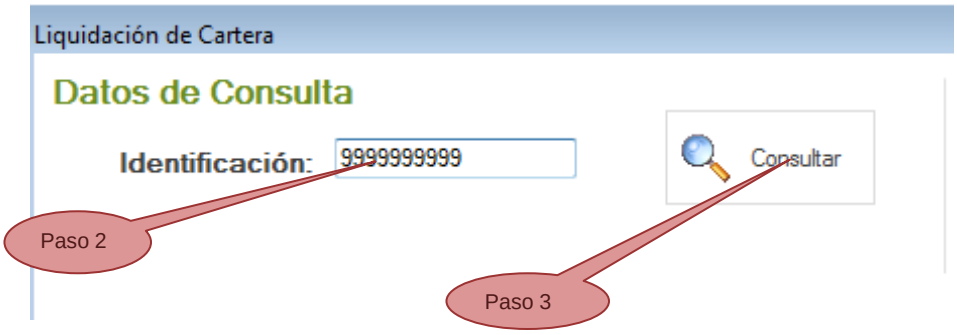
Paso 2: Se digita el número de documento de identificación del usuario que es beneficiario del crédito a liquidar.

Paso 3: Se da clic en la opción o botón “Consultar”

Paso 4: Seleccionar de lista desplegable el número del crédito asociado al estudiante que se desea liquidar.

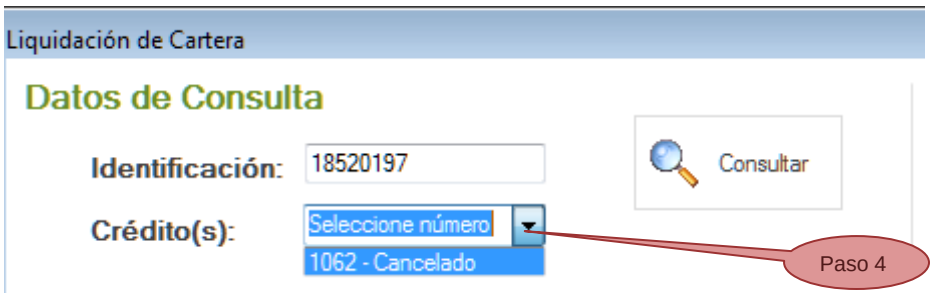
A continuación se muestra la grafica que ilustra los pasos 2, 3 y 4.

Ilustración 37: Descripción del proceso de búsqueda de un crédito



Fuente: El autor.

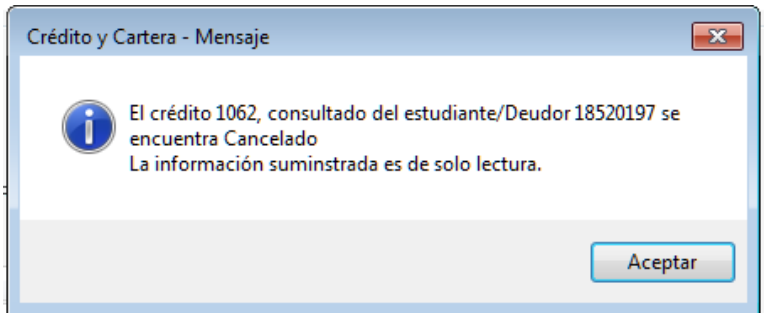
Ilustración 38: Descripción del proceso de selección del crédito a consultar.



Fuente: El autor.

Si selecciona un crédito que ya está cancelado, el sistema muestra un mensaje informativo que se ilustra en la siguiente imagen:

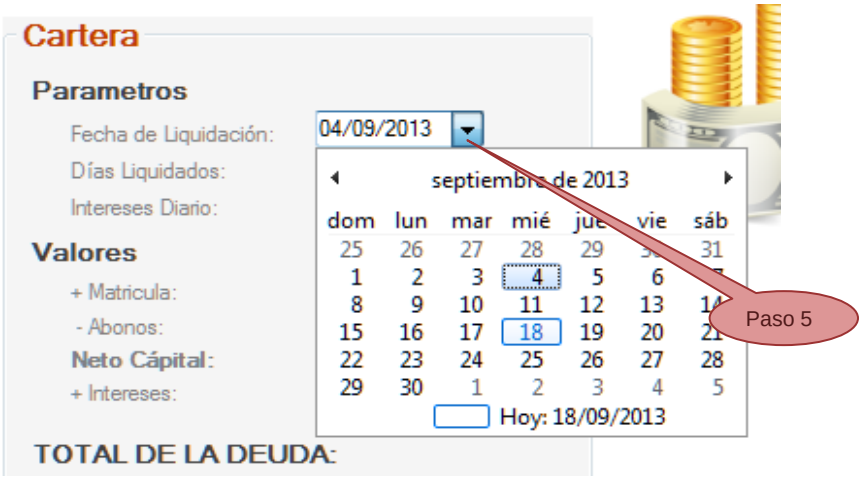
Ilustración 39: Mensaje de información del estado del crédito.



Fuente: El autor.

Paso 5: Seleccionar la “fecha de liquidación” fecha en la cual desea hacer el corte de la cartera. En la siguiente imagen se ilustra en más detalle este paso.

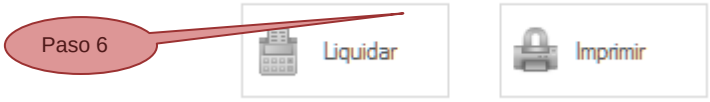
Ilustración 40: Descripción de la selección de "fecha de liquidación"



Fuente: El autor.

Paso 6: Para liquidar se da clic en “Liquidar”. En la siguiente imagen se ilustra este paso.

Ilustración 41: Descripción del botón a seleccionar para realizar la liquidación.



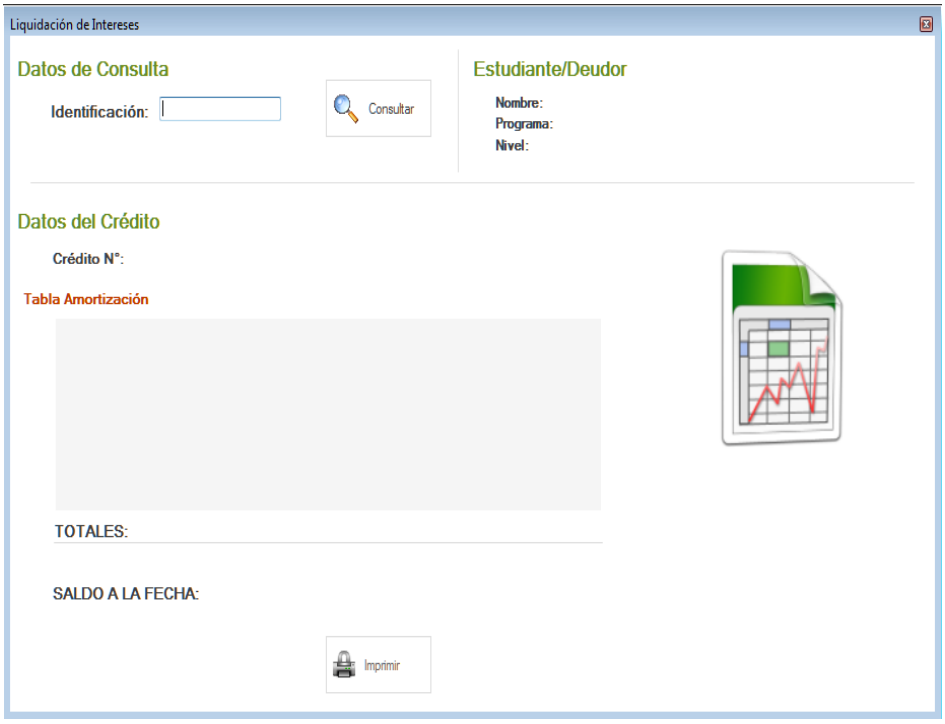
Fuente: El autor.

15.4.2 Liquidar Intereses

En este proceso es por el cual se realiza la correspondiente liquidación los intereses, el sistema consulta cada recibo de caja realizado al crédito y calcula los intereses pagados y el saldo que se va para capital, y da como resultado el saldo final que tenía a la fecha, y así sucesivamente por cada recibo de pago encontrado.

A continuación se muestra la pantalla con la interface completa de este módulo o formulario:

Ilustración 42: Pantalla o formulario donde se Liquidan Intereses.



Fuente: El autor.

A continuación se detalla cada paso que se debe seguir para lograr la correcta liquidación de intereses:

Paso 1: Para poder llegar a utilizar o abrir este módulo se debe seleccionar la opción “Liquidar Intereses”. Que se encuentra en la barra de menú o de herramientas.

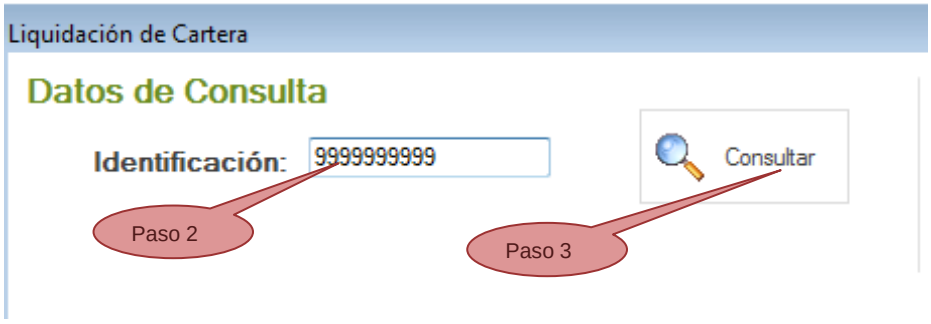
Paso 2: Se digita el número de documento de identificación del usuario que es beneficiario del crédito a liquidar.

Paso 3: Se da clic en la opción o botón “Consultar”

Paso 4: Seleccionar de lista desplegable el número del crédito asociado al estudiante que se desea liquidar.

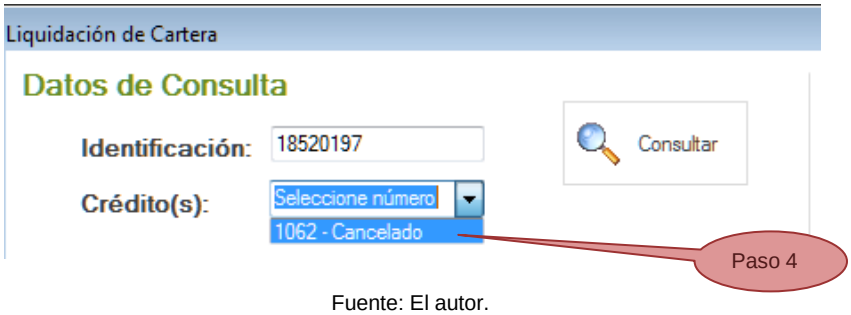
A continuación se muestran las imágenes que ilustran los pasos 2, 3 y 4.

Ilustración 43: Descripción del proceso de búsqueda de un crédito.



Fuente: El autor.

Ilustración 44: Descripción del proceso de selección del crédito.



Fuente: El autor.

Después que se selecciona el número del crédito el sistema realiza el cálculo respectivo de liquidación de intereses y muestra en una tabla tipo amortización el resultado. En la ilustración siguiente se muestra un ejemplo de este resultado de liquidación de intereses, después de seleccionar algún crédito:

Ilustración 45: Pantallazo con un ejemplo resultado de liquidación.

Fuente: El autor.

Paso 5: Este paso es opcional, si desea sacar el resultado por impresora debe seleccionar el botón “Imprimir” como se muestra en la siguiente ilustración:

Ilustración 46: Descripción del paso para imprimir la liquidación.

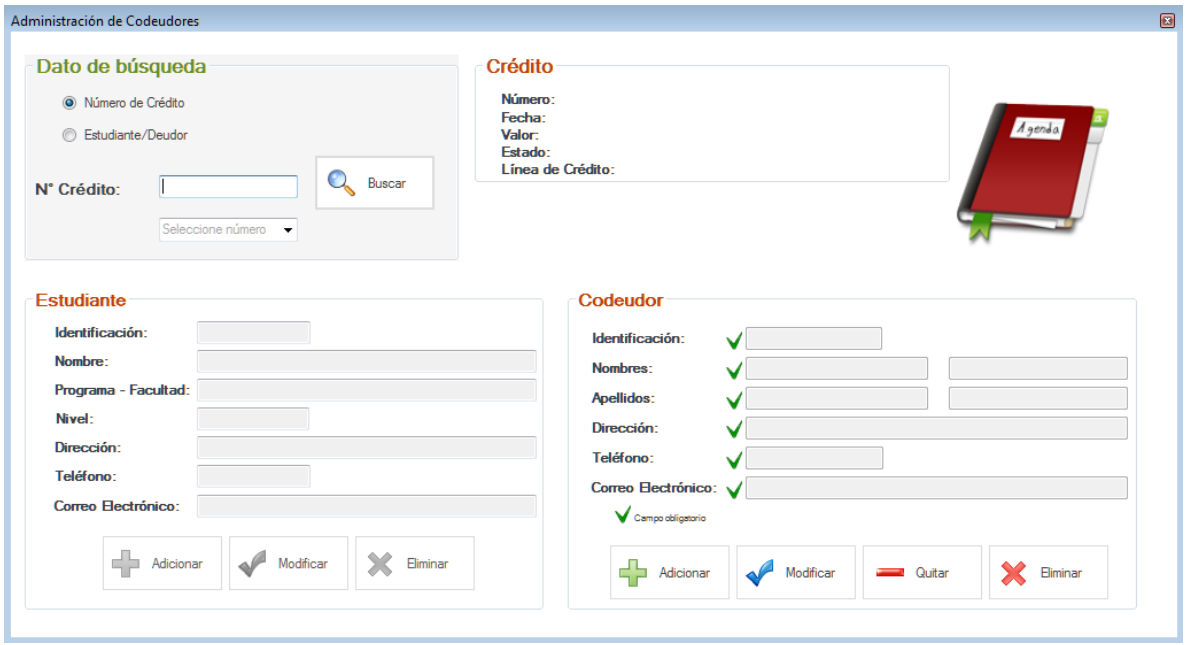
Fuente: El autor.

15.4.3 Administración de Codeudores

Proceso por el cual se hace la gestión o administración de los codeudores asociados a un crédito y un estudiante. En este módulo se puede adicionar un deudor a un crédito en el caso que un crédito no tenga uno, se puede modificar datos de uno ya existente, se puede desasociar o eliminar un codeudor de un crédito.

En la siguiente ilustración se muestra la pantalla con la interface completa de este módulo o formulario:

Ilustración 47: Pantalla o formulario del procedo de administrar codeudores.



Fuente: El autor.

Para poder llegar a utilizar o abrir este módulo se debe seleccionar la opción “Administrar Codeudores”. Que se encuentra en la barra de menú o de herramientas.

A continuación se detalla por sub-capítulos las posibles opciones que se pueden realizar en dicho módulo como consultar, adicionar, modificar, quitar y eliminar:

Consultar información del crédito, estudiante/deudor y codeudor

A continuación se detallan los respectivos pasos para hacer la búsqueda de dicha información:

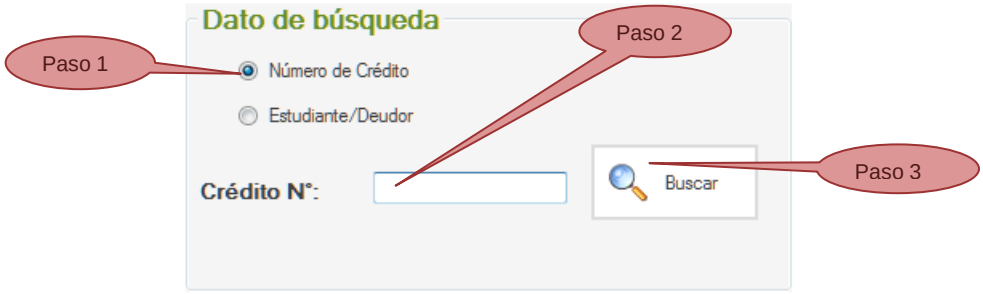
Paso 1: Se debe seleccionar la opción de búsqueda del crédito, si lo desea hacer por número de crédito o por número de identificación del estudiante.

Paso 2: Se debe digitar el dato de búsqueda según se la opción de búsqueda crédito o estudiante.

Paso 3: Dar clic en la opción de "Consultar" para buscar los datos del crédito, estudiante/deudor y su respectivo codeudor.

A continuación se muestra la gráfica que ilustra los pasos 1, 2 y 3.

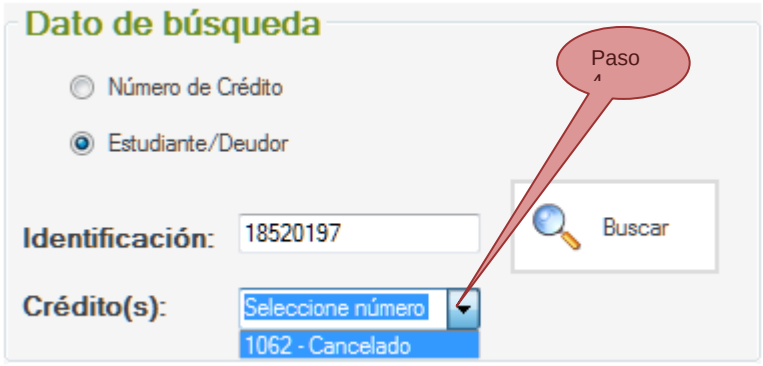
Ilustración 48: Descripción de los pasos y las opciones de búsqueda.



Fuente: El autor.

Paso 4: Si selecciona buscar por "estudiante/deudor" debe seleccionar un número de crédito del cual desea gestionar el codeudor, como se muestra en la ilustración siguiente:

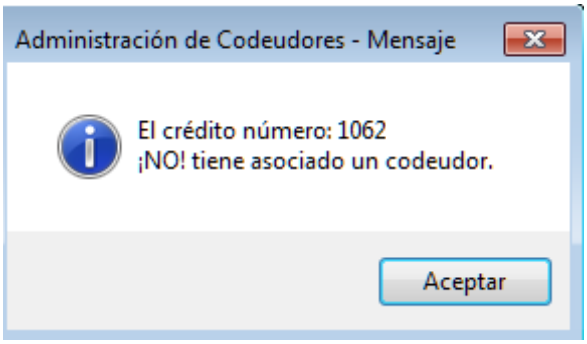
Ilustración 49: Descripción del paso para seleccionar un crédito a consultar.



Fuente: El autor.

Si al consultar los datos crédito, estudiante/deudor no se encuentra un codeudor que este asociado a dicho crédito, el sistema mostrar un mensaje informativo descrito en la siguiente ilustración:

Ilustración 50: Mensaje de crédito consultado no tiene un codeudor.



Fuente: El autor.

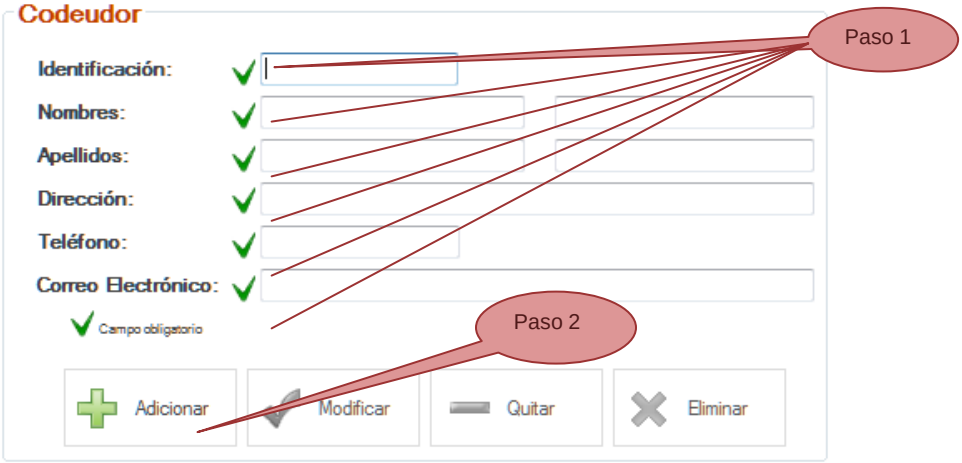
15.4.3.1 Adicionar un Codeudor

Paso 1: Previamente se debe haber consultado un crédito. Para adicionar un nuevo codeudor al crédito seleccionado o consultado, se debe digitar todos los datos correspondientes del codeudor, los campos marcados con una marca verde son los campos obligatorios, los únicos que no son obligatorios es el segundo nombre y el segundo apellido.

Paso2: Dar clic en la opción o botón de “Adicionar” para almacenar el codeudor en la base de datos y asociarlo al crédito seleccionado o consultado.

En la siguiente ilustración se detalla los 1 y 2 para adicionar un codeudor:

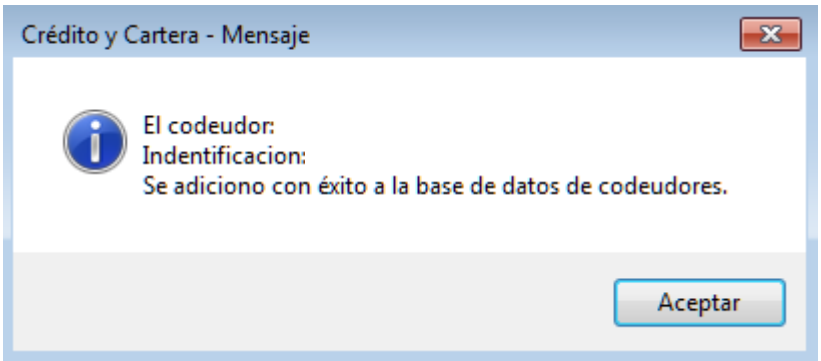
Ilustración 51: Descripción del proceso para adicionar un nuevo codeudor.



Fuente: El autor.

Si el deudor es adicionado correctamente a la base de datos, el sistema muestra un mensaje de información descrito en la siguiente imagen:

Ilustración 52: Mensaje de información de éxito en la adición del nuevo codeudor.



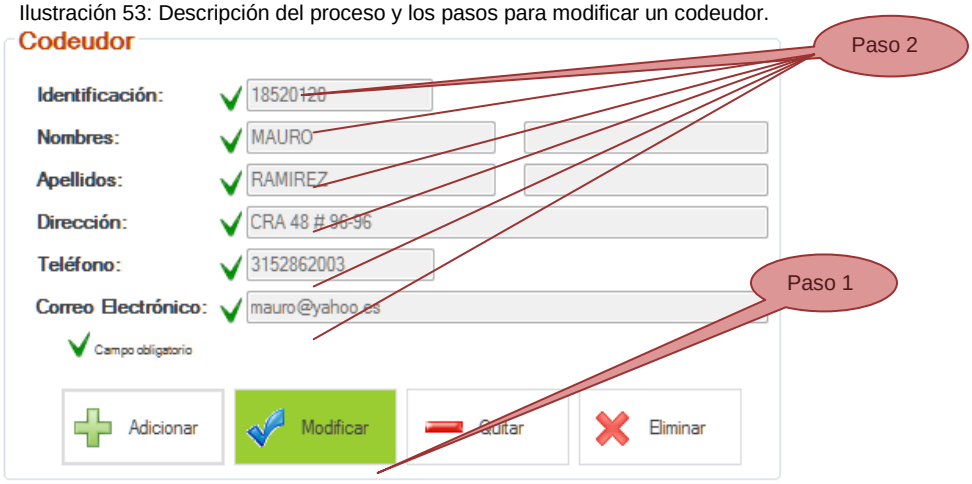
Fuente: El autor.

15.4.3.2 Modificar datos de un Codeudor

Paso 1: Previamente se debe haber hecho la consulta de un crédito. Para modificar los datos de un codeudor debe dar clic en la opción o botón de "modificar".

Paso 2: Para editar los datos de un codeudor se debe posicionar con el mouse y dar un clic sobre el campo o los campos que desea cambiar, y digitar la nueva información. En la siguiente ilustración se describe los campos:

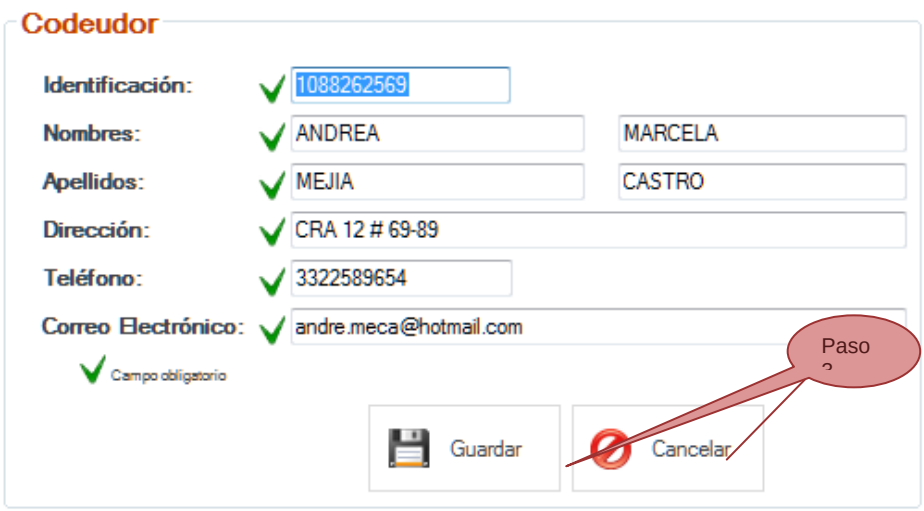
Ilustración 53: Descripción del proceso y los pasos para modificar un codeudor.



Fuente: El autor.

Paso 3: Después de cambiar los valores en los campos que desea se da clic o se selecciona la opción o botón "Guardar" o si no desea guardar cambio en el botón "Cancelar".

Ilustración 54: Descripción del proceso para guardar o cancelar la modificación.



Fuente: El autor.

Para tener en cuenta: Al dar clic en modificar los botones se cambian por otros que se acomodan a las acciones pertinentes para modificar.

15.4.3.3 Quitar un codeudor

Paso 1: Previamente se debe haber consultado un crédito. Para quitar un codeudor se debe seleccionar o dar clic en la opción o botón “Quitar”.

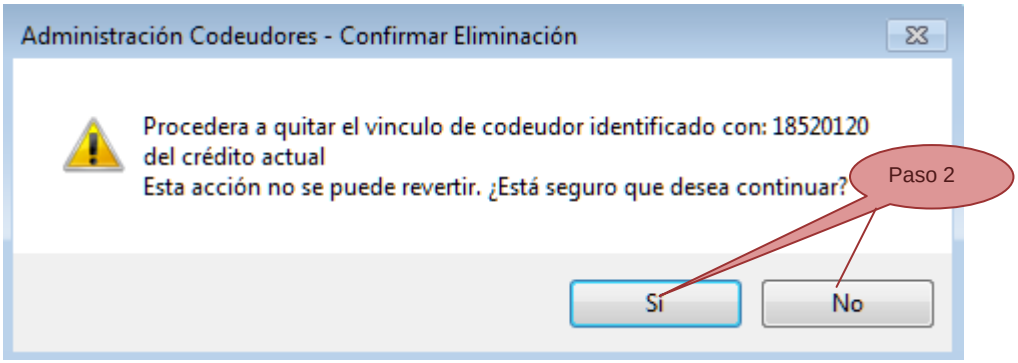
Para tener en cuenta: esta opción solo quita el vínculo que tiene el codeudor del crédito actual. Pero no lo elimina de la base de datos. En la ilustración siguiente se muestra este botón u opción:

Ilustración 55: Descripción del botón a seleccionar para quitar un codeudor.

Fuente: El autor.

Paso 2: Confirmar al sistema que realmente desea quitar el codeudor del vínculo que tiene actualmente con el crédito, Esta pantalla de confirmación se ilustra a continuación:

Ilustración 56: Mensaje de advertencia y confirmación antes de quitar un codeudor.

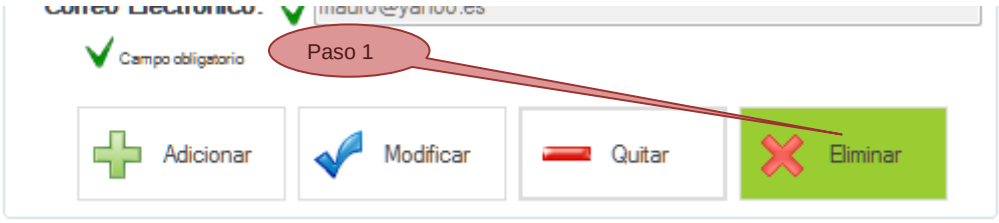


Fuente: El autor.

15.4.3.4 Eliminar un Codeudor

Paso 1: Previamente se debe haber consultado un crédito. Para eliminar un codeudor se debe seleccionar o dar clic en la opción o botón “Eliminar”.

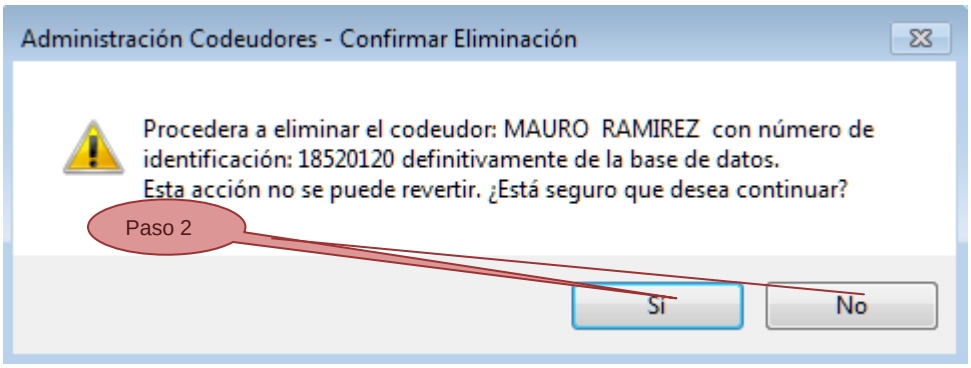
Ilustración 57: Descripción del botón a seleccionar para eliminar un codeudor.



Fuente: El autor.

Paso 2: Confirmar al sistema que realmente desea eliminar definitivamente el codeudor de la base de datos, Esta pantalla de confirmación se ilustra a continuación:

Ilustración 58: Mensaje de advertencia o confirmación antes de eliminar un codeudor.



Fuente: El autor.

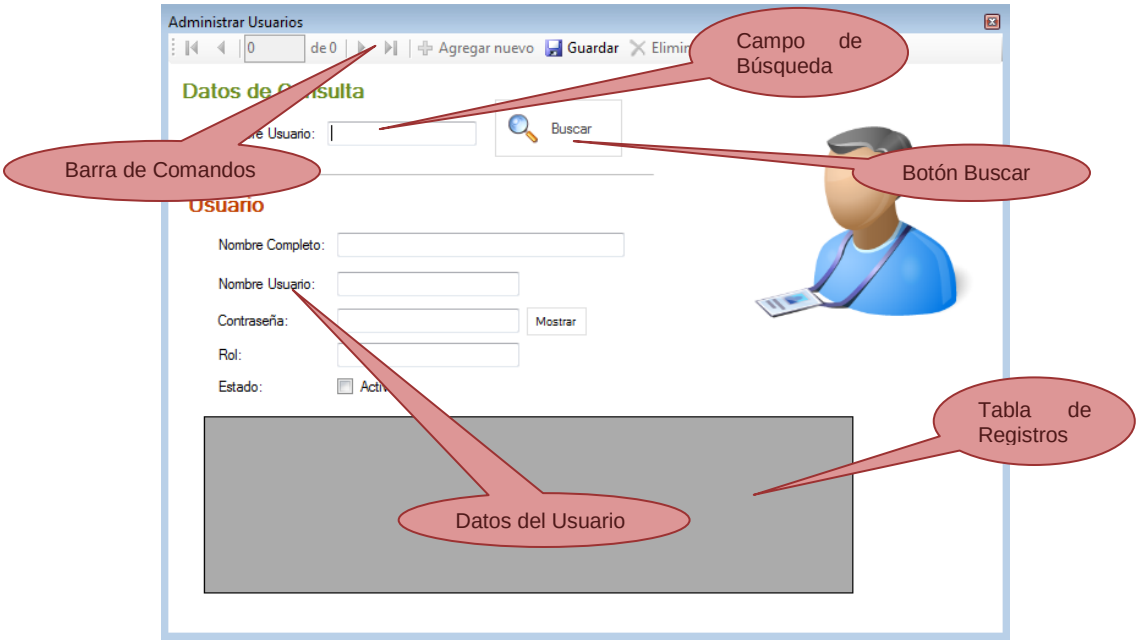
15.4.4 Administrar Usuarios del Sistema

En éste módulo se pueden administrar los diferentes usuarios que tendrán acceso al sistema, se pueden clasificar o agrupar por roles que desempeñaran en el sistema, Usuario estándar, digitador, administrador. Inicialmente el sistema carga o lista todos los usuario registrados en el sistema y donde se puede visualizar los diferentes registros en una tabla, dando la opción al usuario de hacer una

búsqueda por nombre de usuario y donde se puede adicionar uno nuevo, modificar datos, y desplazarse por los diferentes registros.

En la siguiente ilustración o imagen se describe la pantalla o formulario completo, detallando cada componente de la misma:

Ilustración 59: Descripción de los diferentes controles del formulario Admon Usuarios.



Fuente: El autor.

15.4.4.1 Buscar un Usuario

Paso 1: Para hacer la respectiva búsqueda de un usuario ya registrado se debe ingresar el nombre de usuario en el campo de la sesión de consulta con el que se creó inicialmente.

Paso 2: Seleccionar o dar clic en el botón “Consultar” para hacer la respectiva búsqueda por nombre de usuario.

Para tener en cuenta: El nombre de usuario es diferente al nombre completo del usuario, el nombre de usuario es un identificador único de la persona.

Ilustración 60: Descripción de los pasos para hacer la búsqueda de un usuario.

Fuente: El autor.

15.4.4.2 Agregar Usuario

Paso 1: Para agregar un nuevo usuario debe dar clic en la opción o botón de comandos “Agregar nuevo”.

Paso 2: Digitar los datos del usuario como nombre completo, nombre de usuario, contraseña, rol y estado.

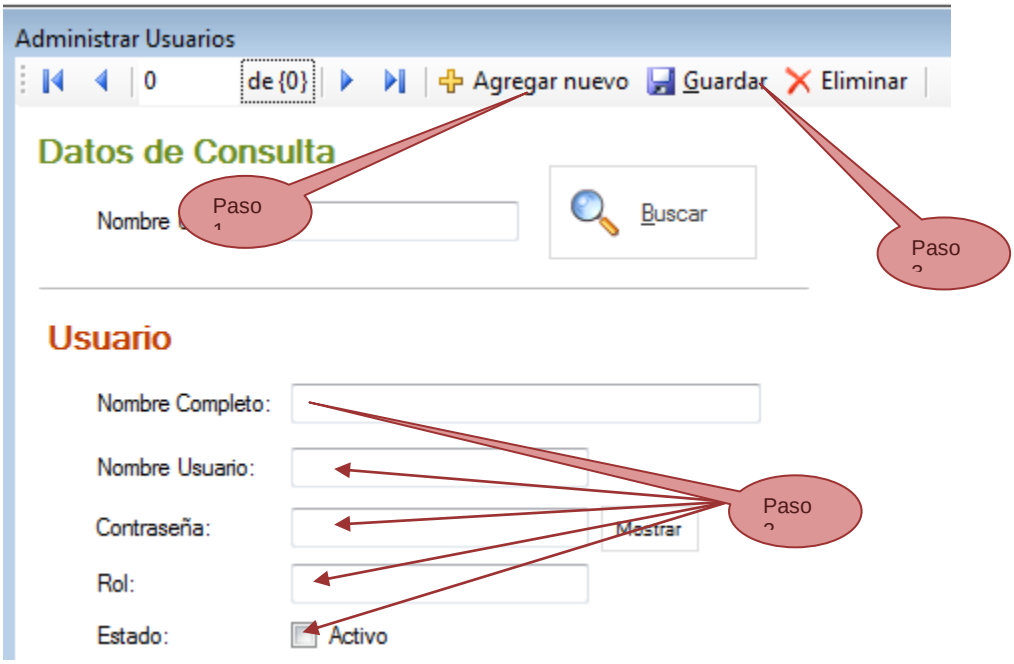
Paso 3: Agregar el usuario en la base de datos seleccionando o dando clic en el botón “Guardar” el sistema mostrar un mensaje informativo que muestra el resultado de agregar un nuevo usuario.

Para tener en cuenta: Los roles son el perfil o las acciones que van a estar autorizadas en el sistema para ese usuario:

- Usuario estándar: tiene acceso a todo el sistema menos las opciones que requieran cambios en la aplicación.
- Administrador: tendrá acceso a todas las opciones del sistema sin restricción.
- Digitador: solo puede ingresar datos y generar liquidaciones y no podrá visualizar en pantalla reportes ni impresiones.

La siguiente figura ilustra los pasos 1, 2 y 3 en más detalle:

Ilustración 61: Descripción del proceso para agregar un nuevo usuario.

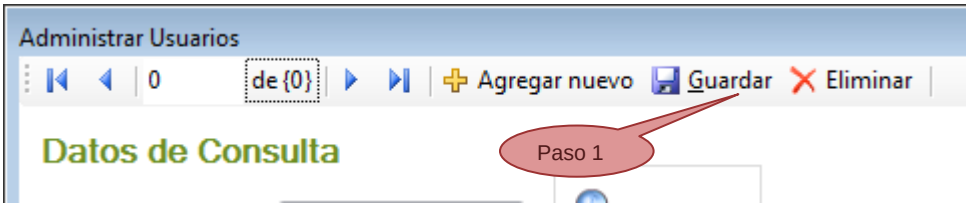


Fuente: El autor.

15.4.4.3 Eliminar un Usuario

Paso 1: Seleccionar la opción o botón de comando “Eliminar”

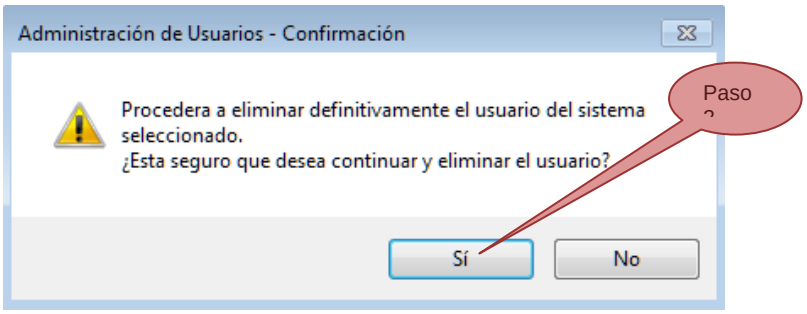
Ilustración 62: Descripción del proceso para eliminar un usuario.



Fuente: El autor.

Paso 2: Tras el paso anterior el sistema solicita una confirmación para la eliminación del usuario el cual debe dar “Aceptar” o “Cancelar”, en un cuadro de mensaje en pantalla que se ilustra en la imagen siguiente.

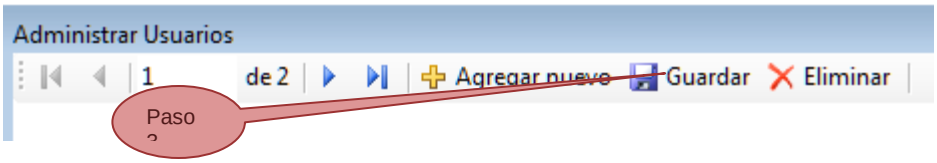
Ilustración 63: Mensaje de advertencia y confirmación antes de eliminar un codeudor.



Fuente: El autor.

Paso 3: Seleccionar la opción o botón de comando “Guardar” que se encuentra en la barra de herramientas o comandos, para guardar los cambios efectuados a la tabla de usuarios.

Ilustración 64: Descripción del botón a seleccionar para guardas cambios.

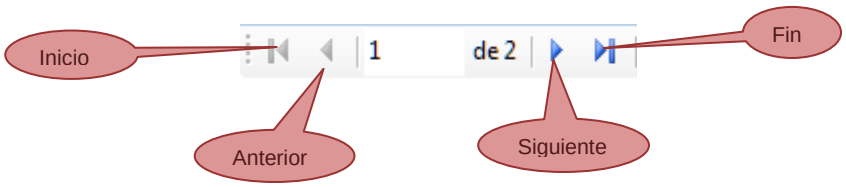


Fuente: El autor.

Para tener en cuenta: Desde este modulo se puede editar y modificar los datos de los usuarios directamente en la tabla donde se lista toda la información de los usuario y después “Guardar” para aceptar los cambios y modificarlos en la base de datos.

Se puede ir desplazando por los diferentes registros de usuarios seleccionando los botones de “Siguiente”, “Anterior”, “final” e “Inicial”; o digitar el número de registro. Estos botones se ilustran en la siguiente imagen.

Ilustración 65: Descripción de los botones para desplazarse por los registros.



Fuente: El autor.

Ilustración 66: Descripción de los botones de comandos.

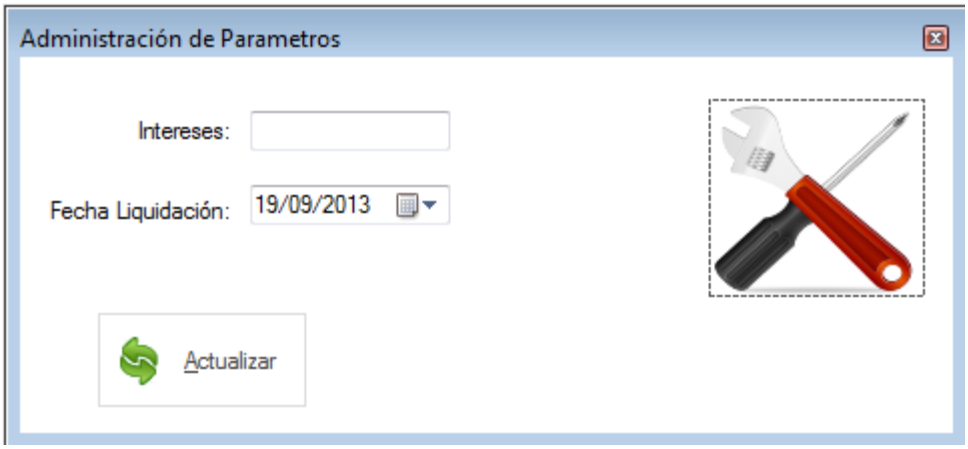
Fuente: El autor.

15.4.5 Administrar Parámetros

En éste módulo se pueden administrar los diferentes parámetros globales que se utilizan en toda la aplicación, como son: El parámetro del porcentaje de intereses con el que se deben liquidar los intereses. La fecha por defecto de liquidación, entre otros.

En la ilustración siguiente se muestra la interface de todo el formulario:

Ilustración 67: Pantalla o formulario para administrar los parámetros del sistema



Fuente: El autor.

15.4.5.1 Actualizar parámetros

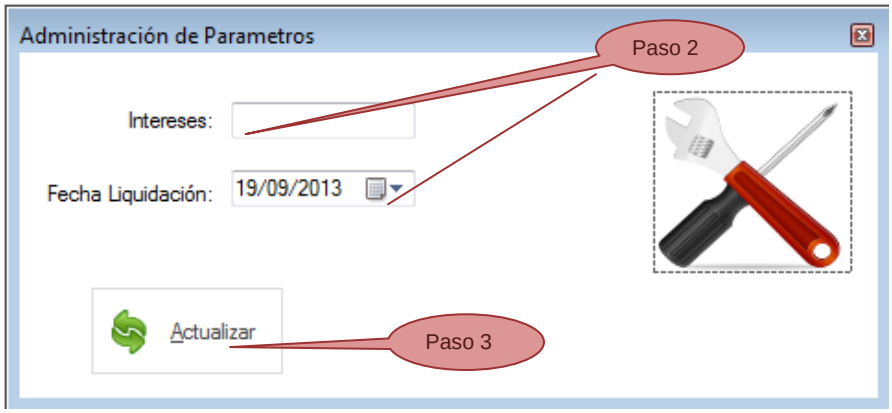
Paso 1: Se debe abrir el módulo por la opción de “Aplicación” “Parámetros” de la barra de menú.

Paso2: Al cargar el formulario se muestran los valores de los parámetros que están actualmente configurados en el sistema. Se debe posicionar sobre el campo que dese editar y digitar o seleccionar su nuevo valor.

Paso 3: Se selecciona la opción o el botón “Actualizar”. El sistema muestra un mensaje del resultado de la actualización y cierra el formulario.

Para tener en cuenta: Es este formulario no se puede eliminar parámetros o dejarlos en blanco. Ya que es información primordial para los procesos de liquidar cartera y liquidar intereses, los cuales utilizan el parámetro de valor de intereses para poder hacer la respectiva liquidación de intereses.

Ilustración 68: Descripción de los pasos para cambiar un parámetro.



Fuente: El autor.

16 CONCLUSIONES

- Con el software desarrollado se logro satisfacer todos los requerimientos funcionales y los objetivos planteados. Utilizando herramientas de desarrollo de última tecnología que llevaron a culminar con un software funcional y de interface amigable.

- Al culminar el proyecto fue enormemente satisfactorio el haber podido aplicar muchos de los conocimientos adquiridos durante la carrera y en todo el transcurso de nuestra la experiencia profesional, ya que nunca se había tenido la posibilidad de desarrollar un software desde sus inicios, partiendo desde la ingeniería del software hasta terminar con el producto final “El software”.

ANEXOS

- Manual Técnico (formato PDF).

BIBLIOGRAFIA

ICONTEC INTERNATIONAL. EL COMPENDIO DE TESIS Y OTROS TRABAJOS DE GRADO. {En línea}. {Consultado mayo 2013}. Disponible en: http://aprendeenlinea.udea.edu.co/lms/moodle/file.php/452/Instructivo_ICONTEC_2008.doc

UNIVERSIDAD NACIONAL DE COLOMBIA. INDICACIONES PARA ELABORAR LA PROPUESTA PARA TRABAJO DE GRADO. {En línea}. {Consultado mayo 2013}. Disponible en: http://www.docentes.unal.edu.co/flozanoo/docs/INDICACIONES%20PARA%20ELABORAR%20LA%20PROPUESTA.ppt

UNIVERSIDAD LIBRE. GUIA PARA LA PRESENTACIÓN DE PROPUESTAS Y ANTEPROYECTOS PARA PROYECTOS DE GRADO E INVENSTIGACION. {En línea}. {Consultado mayo 2013}. Disponible en: www.unilibrepereira.edu.co

OBJECT MANAGEMENT GROUP UML. OMG® SPECIFICATIONS. {En línea}. {Consultado mayo 2013}. Disponible en: http://www.omg.org/spec/BMM/1.1/PDF

MICROSOFT. MSDN LIBRARY DESARROLLO .NET. {En línea}. {Consultado Junio 2013}. Disponible en: http://msdn.microsoft.com/es-es/library/aa139615.aspx

ORACLE. ORACLE DATABASE DOCUMENTATION. {En línea}. {Consultado Junio 2013}. Disponible en: http://www.oracle.com/pls/db112/homepage

EXTREME PROGRAMMING: A GENTLE INTRODUCTION. {En línea}. {Consultado Junio 2013}. Disponible en: http://www.extremeprogramming.org

WIKIPEDIA: CASE – UML - RAD {En línea}. {Consultado Julio 2013}. Disponible en: http://es.wikipedia.org

PAVÓN MESTRAS, Juan. Programación orientada a objetos. Universidad Complutense de Madrid. 2009.

PRESSMAN, Roger S. Ingeniería de software un enfoque práctico. Séptima edición. McGraw Hill. 2010.

Newkirk, James. La pogramacion extrema en la práctica. Editorial Eddison Wesley. 2002. P. 18.

Weitzenfeld, Alfredo. Ingenieria del software orientada a objetos con UML, Java e internet. Editorial THOMSON. 2011. P. 55.

E. Lopez, Teniente, A. Olivé, Ramon A. Diseño de sistemas software en UML. Editorial POLITEXT. 2003. P. 39.

BIRNIOS, Mariano. Microsoft Visual Basic.NET quia del programador. Edicion ilustrada. Editorial MP ediciones s.a. 2002. P. 34