UNIVERSIDAD NACIONAL AUTÓNOMA DE NICARAGUA
UNAN-LEÓN

FACULTAD DE CIENCIAS
DEPARTAMENTO DE COMPUTACIÓN

TESIS PARA OPTAR AL TITULO DE
INGENIERO EN SISTEMA DE INFORMACIÓN

TEMA:
"SISTEMA DE MONITOREO DE SUBPROYECTOS DE PEQUEÑAS AYUDAS DE INVESTIGACIÓN APROBADOS POR LA VIP DE LA UNAN-LEÓN"

PRESENTADO POR:

- **BR. MARLON ANTONIO MARTÍNEZ RUEDA.**
- **BR. HEBERTH JOSÉ QUINTERO COREA.**

TUTOR:

MSC. ALVARO ALTAMIRANO.

ENERO 2008
LEÓN,NICARAGUA

INDICE

I. INTRODUCCION.

La Investigación es una de las funciones sustantivas y uno de los procesos estratégicos de la UNAN-León, al facilitar la generación de conocimiento y la búsqueda de solución de los problemas locales, nacionales y regionales mediante el planeamiento y ejecución de actividades coherentes que tratan de responder a una pregunta central de investigación.

La Vicerrectoría de Investigación y Postgrado (VIP) de la UNAN-León es la dependencia encargada de coordinar los Subproyectos de Pequeñas Ayudas de Investigación y de darle seguimiento a las actividades realizadas por dichos subproyectos.

La situación de partida es que no existía ningún sistema informático, toda la gestión se realiza de forma artesanal, por lo cual resultaba tedioso darle seguimiento a los subproyectos de Investigación.

Por tal razón es necesario crear un software de calidad que cumpla con los nuevos estándares en el desarrollo del mismo y que cubra con la necesidad de automatización del proceso de monitoreo de las actividades realizadas por los subproyectos de Pequeñas Ayudas de Investigación aprobados por la Vicerrectoría de Investigación y Postgrado de la UNAN-LEON.

Hemos decidido desarrollar este sistema haciendo uso de la tecnología .NET utilizando el entorno de desarrollo de Visual Studio 2005 específicamente C#, atacando una base de datos SQL Server 2005.

II. ANTECEDENTES.

La Vicerrectoría de Investigación y Postgrado (VIP) coordina el Programa de Pequeñas Ayudas de Investigación (PAI) desde el año 2004 como una oportunidad de financiamiento de proyectos de investigación dirigido a los académicos con grado de M.Sc y Ph.D. de la UNAN – León.

El proyecto de investigación es el instrumento mediante el cual se realiza investigación en la UNAN-León. El mismo se caracteriza por poseer objetivos, costos y duración claramente definidos. Las actividades del proyecto se organizan mediante un plan de trabajo racional que incluye la utilización de recursos humanos, físicos y financieros.

El programa de pequeñas ayudas de investigación es un esfuerzo coordinado por la VIP con apoyo financiero de la **Agencia Sueca para el Desarrollo Internacional (ASDI)**, para impulsar la búsqueda de nuevo conocimiento, a través de investigaciones que puedan ser realizadas por los académicos con grado de maestría o doctorado.

En la actualidad la VIP solo cuenta con una pequeña base de datos en Access, la cual lleva un registro secuencial de los subproyectos recepcionados, el resto de la información se lleva en Excel y en papel motivo por el cual resulta tedioso darle seguimiento a las actividades realizadas por dichos subproyectos.

Tampoco se cuenta con un entorno de información organizada que facilite el acceso directo a la información (los diferentes subproyectos) basándose en atributos tales como coordinador, titulo, área de investigación o facultad.

III. JUSTIFICACION.

El motor que impulsa el desarrollo de este sistema es la creciente necesidad de automatización de la parte de monitoreo de los subproyectos de pequeñas ayudas de investigación aprobados en la Vicerrectoría de investigación y postgrado de la UNAN-LEON,

El propósito de nuestro proyecto es desarrollar una herramienta piloto con la cual se pretende explotar las características hacia la que esta orientada la estrategia de diseño de un sistema capaz de llevar un mejor seguimiento de las actividades realizadas por los subproyectos de investigación., además de realizar el soporte y gestión de la información que hasta ahora se realizaba de forma manual, en una hoja de Excel y el resto en papel.

Además el volumen cada vez mayor de subproyectos hace necesario un monitoreo de los mismos. Anexando a esto, también es completa la gestión que permitirá este sistema, ya que el control automatizado, acelera la respuesta que muchas veces necesitan los proyectos de investigación para lograr su ejecución.

Nuestro sistema será una herramienta estratégica la cual potenciara la eficiencia de las actividades realizadas por los subproyectos de investigación evitado de esta forma que estos lleguen a caer en subejecución de fondos.

Realizar esta aplicación nos permitirá obtener el titulo de Ingeniero en Sistema de Información, además de adquirir experiencia en el manejo de esta nueva herramienta como lo es Visual Studio.NET 2005

IV. OBJETIVOS

1. **Objetivos Generales**:

- Realizar un aporte a la Vicerrectoría de Investigación y postgrado de la UNAN-LEON, diseñando e implementando un software de fácil uso, que cubra con la necesidad de automatización del proceso de monitoreo de las actividades efectuadas por los subproyectos de Pequeñas Ayudas de Investigación.

2. Objetivos Específicos:

- Llevar un registro de todos los Subproyectos de Investigación recepcionados en la Vicerrectoría de Investigación y Postgrado de la UNAN-LEON.

- Diseñar un sistema capaz de llevar un mejor seguimiento a la parte de monitoreo de las actividades realizadas por los subproyectos de investigación y que además permita la generación de reportes actualizados.

- Crear un programa que gestione la información de la base de datos de los subproyectos de investigación.

V. MARCO TEORICO

Vicerrectoría de Investigación y Postgrado VIP
Programa de Pequeñas Ayudas de Investigación PAI

La Vicerrectoría de Investigación y Postgrado (VIP) de la UNAN-León es una dependencia de apoyo a la gestión del Rector de esta institución y tiene como misión el facilitar la generación del conocimiento y la búsqueda de solución de los problemas locales, nacionales y regionales mediante la investigación y la educación de postgrado para contribuir a la excelencia en la formación integral de las personas y a la transformación positiva de la sociedad nicaragüense.

Para cumplir con su misión, el quehacer de la VIP se enfoca a la total satisfacción de las necesidades de los usuarios de los servicios que presta y al mejoramiento continuo de sus procesos.

El proyecto de investigación es el instrumento mediante el cual se realiza investigación en la UNAN-León. El mismo se caracteriza por poseer objetivos, costos y duración claramente definidos. Las actividades del proyecto se organizan mediante un plan de trabajo racional que incluye la utilización de recursos humanos, físicos y financieros.

El programa de pequeñas ayudas de investigación es un esfuerzo coordinado por la VIP con apoyo financiero de la Agencia Sueca para el Desarrollo Internacional, para impulsar la búsqueda de nuevo conocimiento, a través de investigaciones que puedan ser realizadas por nuestros propios académicos con grado de maestría o doctorado.

Las oportunidades de financiamiento se centran en 7 áreas estratégicas que son:

1) Salud
2) Medio Ambiente
3) Educación

4) Democracia y Estado de Derecho
5) Producción y Economía
6) Tecnologías de la Información y de la Comunicación (TIC)
7) Energía

El planeamiento operacional de la investigación definirá con claridad qué se desea o qué se pretende lograr dentro de un proyecto específico de investigación, cómo y cuando se realizará esto y quién será el encargado. Esta etapa del planeamiento consta de dos partes: el proceso de planeamiento operacional y el plan operacional.

En el caso de la UNAN-León, el proceso de planeamiento consiste en la participación continua de las autoridades y todas aquellas personas clave para la estructuración de los planes y, lo más importante, para la obtención de resultados tangibles para la institución, como un todo, y para las Facultades y Departamentos que la conforman.

Se hace necesario hacer énfasis en la importancia que adquiere el trabajo de equipo durante este proceso. Trabajar como equipo es la base para la elaboración de un plan en el que los involucrados directos se sientan propietarios del mismo y de los resultados proyectados.
El proceso de Planeamiento Operacional de la Investigación tendrá un horizonte máximo de un año y el producto de este proceso será la obtención de los resultados deseados mediante la implementación de un Plan Operacional Anual.

El Plan Operacional Anual (POA) de investigación: es un documento que identifica los resultados específicos que se necesita lograr dentro del tiempo establecido (un año). Este documento incluirá las acciones y los recursos específicos que se necesitan para obtener estos resultados. El POA de investigación estará constituido por seis elementos:

1. Objetivos de Investigación
2. Resultados Requeridos
3. Indicadores Clave de Rendimiento
4. Objetivos Operacionales
5. Planes de Acción
6. Monitoreo y Evaluación del Plan

Los Objetivos de Investigación: son los objetivos prioritarios, contemplados en el diseño del proyecto, para los cuales se requiere lograr resultados durante el período proyectado de planeamiento. En dependencia de la duración del proyecto (que puede ser más de un año) y del orden cronológico establecido, algunos objetivos de investigación serán la guía para la toma de decisiones durante el primer año, otros lo serán para el segundo, etc.

Los Resultados Requeridos: representan aquellos logros que esperamos obtener y que constituyen el alcance de los objetivos de investigación. Estos resultados se escriben en forma de aseveraciones. Por ejemplo, “Se dispone de un prueba validada y estandarizada para determinar la presencia del fenómeno X”.

Los resultados de PRODUCTO son aquellos en los que se espera es producir un logro tangible. Por ejemplo, “bases de datos de los sujetos de estudio completas y actualizadas”; “catálogo actualizado de la flora del área de estudio”, etc.

Los resultados de EFECTO representan un cambio en la conducta de las personas. Por ejemplo, “los investigadores utilizan sin errores la técnica Z”; “el personal de laboratorio participa efectivamente en el programa de aseguramiento de la calidad”, etc.

Los resultados de IMPACTO representan un cambio profundo y, generalmente a mediano o largo plazo, en la institución y sus procesos o en la comunidad; pueden también representar logros trascendentales para el cumplimiento de la misión de la institución y/o facultad. Por ejemplo, “Diagnóstico moderno y eficaz de la enfermedad X”; “Gestión participativa de los recursos hídricos de la comunidad Y por parte de sus habitantes”, etc.

Los Indicadores Clave de Rendimiento (ICR): son factores, dentro de cada uno de los Resultados Requeridos, que permiten controlar el progreso logrado hacia el cumplimiento de los objetivos. Aunque algunos indicadores son cuantificables, otros no los son. En todo caso, los participantes en el proceso de planeamiento deben entender y estar de acuerdo con los indicadores seleccionados.

En otras palabras, los ICR deben ser los factores medibles que nos guían en el monitoreo del proceso. Son los ICR los que nos permiten identificar qué tanto nos acercamos o nos alejamos del cumplimiento de nuestros objetivos, o sea, que tanto progresa nuestro plan. No debe olvidarse que, por cada resultado requerido debe identificarse por lo menos un ICR.

Los Objetivos Operacionales: representan resultados específicos y mensurables que se alcanzarán dentro del período del plan. Estos caen dentro de uno o más Resultados Requeridos e incorporan algunos de los Indicadores Clave de Rendimiento como sus principales factores mensurables.

Es indispensable incluir por lo menos un ICR en el objetivo operacional. Eso ahorrará mucho trabajo en el futuro

Los Planes de Acción: representan las acciones específicas requeridas para lograr cada objetivo. Estos Planes pueden Manifestarse en cada una de las formas siguientes o en una combinación de las mismas:

Los Planes de Acción incluirán marcos de tiempo específicos, requisitos de recursos y responsabilidad por cada paso.

El proceso de Monitoreo y Evaluación del Plan: cierra el circuito en el proceso de planeamiento operacional de la investigación, asegurando que lo que se desea lograr se traduzca en la acción que conlleve a esos resultados.

Este proceso debe estar diseñado para ayudar a los gestores del plan a controlar el rendimiento contra los objetivos, de tal manera que se pueda tomar una acción correctiva o aplicar planes de contingencia cuando sea necesario.

Actividades de un Sistema de Información: Un sistema de información realiza cuatro actividades básicas: entrada, almacenamiento, procesamiento y salida de información.

Entrada de Información. Es el proceso mediante el cual el Sistema de Información toma los datos que requiere para procesar la información. Las entradas pueden ser manuales o automáticas. Las manuales son aquellas que se proporcionan en forma directa por el usuario, mientras que las automáticas son datos o información que provienen o son tomados de otros sistemas o módulos. Esto último se denomina interfaces automáticas.

Las unidades típicas de entrada de datos a las computadoras son las terminales, las cintas magnéticas, las unidades de disquete, los códigos de barras, los escáner, la voz, los monitores sensibles al tacto, el teclado y el Mouse, entre otras.

Almacenamiento de Información El almacenamiento es una de las actividades o capacidades más importantes que tiene una computadora, ya que a través de esta propiedad el sistema puede recordar la información guardada en la sección o proceso anterior. Esta información suele ser almacenada en estructuras de información denominadas archivos.

La unidad típica de almacenamiento son los discos magnéticos o discos duros, los discos flexibles o disquetes y los discos compactos (CD-ROM).

Procesamiento de Información

Es la capacidad del sistema de información para efectuar cálculos de acuerdo con una secuencia de operaciones preestablecida. Estos cálculos pueden efectuarse con datos introducidos recientemente en el sistema o bien con datos que están almacenados. Esta característica de los sistemas permite la transformación de datos fuente en información que puede ser utilizada para la toma de decisiones, lo que hace posible, entre otras cosas, que un tomador de decisiones genere una proyección financiera a partir de los datos que contiene un estado de resultados o un balance general de un año base.

Salida de Información

La salida es la capacidad de un Sistema de Información para sacar la información procesada o bien datos de entrada al exterior. Las unidades típicas de salida son las impresoras, terminales, disquetes, cintas magnéticas, la voz, los graficadores y los plotters, entre otros.

Es importante aclarar que la salida de un sistema de información puede constituir la entrada a otro Sistema de Información o módulo. En este caso, también existe una interfase automática de salida.

Los sistemas de información cumplen tres objetivos básicos dentro de las organizaciones

a. Automatización de procesos operativos.
b. Proporcionar información que sirva de apoyo al proceso de toma de decisiones.
c. Lograr ventajas competitivas a través de su implantación y uso.

Se puede decir que los sistemas de información son muy importantes dentro de una organización, porque aparte que facilita el trabajo de muchas personas, proporciona Información útil y precisa que es de gran ayuda para los gerentes de las organizaciones al momento de tomar decisiones. Tipos de Sistemas de Información.

Los sistemas de información son desarrollados de acuerdo a diferentes propósitos, las necesidades de la organización y de los niveles organizacionales de la misma. Cada uno de estos sistemas difieren en sus características y cada uno tiene un objetivo fundamental para lograr satisfacer las necesidades de un sistema dentro de una organización.

Bases de Datos

Una base de datos es un conjunto de datos que pertenecen al mismo contexto almacenados sistemáticamente para su uso posterior.

Podemos considerar que es un conjunto de datos de varios tipos, organizados e interrelacionados. Estos datos deben estar libres de redundancias innecesarias y ser independientes de los programas que los usan.

Se puede decir que son aplicaciones informáticas, que sirven para manejar información en forma de fichas, clientes, artículos etc.

La mayoría de las bases de datos actualmente permiten listados, consultas crear pantallas de visualización de datos, controlar el acceso de los usuarios etc.

En la actualidad y gracias al desarrollo tecnológico de campos como la informática y la electrónica la mayoría de las base de datos, tienen un formato electrónico que ofrece un alto rango de soluciones al problema de almacenar datos.

En informática existen los sistemas gestores de base de datos (SGBD), en inglés DBMS, DataBase Manager Systems, los cuales permiten almacenar y posteriormente acceder a los datos de forma rápida.

Las aplicaciones más usuales son para la gestión de empresas e instituciones públicas. También son ampliamente utilizadas en entornos científicos con el objeto de almacenar la información experimental.

Las bases de datos pueden clasificarse de varias maneras de acuerdo al criterio elegido para su clasificación.

Bases de datos estáticas: son bases de datos de solo lectura, utilizadas primordialmente para almacenar datos históricos que posteriormente se pueden utilizar para estudiar el comportamiento de un conjunto de datos.

Bases de datos dinámicas: son bases de datos donde la información almacenada se modifica con el tiempo permitiendo operaciones como inserción y actualización de datos además de las operaciones fundamentales de consulta.

***Microsoft SQL Server 2005*:** Es una plataforma de base de datos que se utiliza en el procesamiento de transacciones en línea (OLTP) a gran escala, el almacenamiento de datos y las aplicaciones de comercio electrónico; es también una plataforma de Business Intelligence para soluciones de integración, análisis y creación de informes de datos.

***BACKUP*:** se refiere a la copia de datos de tal forma que estas copias adicionales puedan restaurar un sistema después de una pérdida de información. Para restaurar un ordenador a un estado operacional después de un desastre (copias de seguridad del sistema)

Para restaurar un pequeño número de ficheros después de que hayan sido borrados o dañados accidentalmente (copias de seguridad de datos). En el mundo de la empresa, además es útil y obligatorio, para evitar ser sancionado por los órganos de control en materia de protección de datos.

Las copias de seguridad en un sistema informático tienen por objetivo el mantener cierta capacidad de recuperación de la información ante posibles pérdidas. Esta capacidad puede llegar a ser algo muy importante, incluso crítico, para las empresas. Se han dado casos de empresas que han llegado a desaparecer ante la imposibilidad de recuperar sus sistemas al estado anterior a que se produjese un incidente de seguridad grave.

RESTORE: Permite copiar el contenido de un paquete de discos que fueron usados para respaldo a su origen o en el destino del disco donde se pretende copiar dicho contenido.

Microsoft.NET es el conjunto de nuevas tecnologías en las que Microsoft ha estado trabajando durante los últimos años con el objetivo de obtener una plataforma sencilla y potente para distribuir el software en forma de servicios que puedan ser suministrados remotamente y que puedan comunicarse y combinarse unos con otros de manera totalmente independiente de la plataforma, lenguaje de programación y modelo de componentes con los que hayan sido desarrollados. Ésta es la llamada **plataforma .NET**.

La Infraestructura .NET la cual comprende la plataforma .NET, Microsoft Studio .NET, los Servidores Corporativos .NET y Microsoft Windows .NET

La Infraestructura .NET se refiere a todas las tecnologías que constituyen el nuevo ambiente para crear y ejecutar aplicaciones robustas, escalables y distribuidas. La parte de .NET que permite desarrollar estas aplicaciones es la plataforma .NET.

La Plataforma .NET consiste de un Lenguaje Común en Tiempo de Ejecución (CLR) y la Biblioteca de Clases de la Plataforma .NET algunas veces llamada la Biblioteca de Clases Base (CBL).

El CLR es como una máquina virtual (el código que corre dentro del CLR en ejecución en un ambiente encapsulado y controlado, separados de otros procesos en la máquina) en la cual funcionan las aplicaciones .NET, todos los lenguajes .NET tienen la biblioteca de clases de la Plataforma .NET a su disposición.

C# es un lenguaje de Programación Orientada a Objetos desarrollado por Microsoft. Es el último lenguaje en llegar y todo hace indicar que será el más empleado por los desarrolladores de software. Se emplea como base de su plataforma .Net. Al igual que Java, se compila a un código intermedio, llamado MILD, que posteriormente se ejecuta en cualquier plataforma, mediante la máquina virtual JIT.

C# o C Sharp es un lenguaje de programación que está incluido en la Plataforma .NET y corre en el Lenguaje Común en Tiempo de Ejecución (CLR, Common Language Runtime).

El primer lenguaje en importancia para el CLR es C#, mucho de lo que soporta la Plataforma .NET está escrito en C#.

C# intenta ser el lenguaje base para escribir aplicaciones .NET

C# deriva de C y C++, es moderno, simple y enteramente orientado a objetos, simplifica y moderniza a C++ en las áreas de clases, namespaces, sobrecarga de métodos y manejo de excepciones. Se elimino la complejidad de C++ para hacerlo más fácil de utilizar y menos propenso a errores.

Características del CLR.

- Integración de lenguajes a través del *Common Language Specification*
- Administración automatica de memoria, a través del recolector de basura.
- Manejo de excepciones de lenguajes.
- Seguridad *type safety*.
- Soporte de versiones.
- Modelo Simplificado para la interacción de componentes.

Un concepto importante relacionado a la CLR es el código administrado, el código administrado es justo el código que esta ejecutándose bajo el auspicio de la CLR y por consiguiente comienza a ser controlado por el CLR.

El CLS se refiere a la interoperabilidad entre lenguajes, por lo que es necesario seguir los tipos y características del CLS, para ello es necesario conocer los tipos primitivos, arreglos, tipos, miembros tipo, métodos, campos, propiedades, enumeraciones, excepciones, interfaces, eventos, atributos personalizables, delegados, identificadores, etc. que la propia especificación define.

Virtual Execution System - VES

El Sistema Virtual de Ejecución implementa la VOS y se crea implementando un motor de ejecución (*Execution Engine* EE). Los componentes de la VES son:

- **Lenguaje Intermedio** (*Intermediate Language* - IL), diseñado para ser fácilmente traducido a una amplia gama de lenguajes, por lo que el compilador C# es capaz de generar el lenguaje intermedio.

- **Carga del Código Administrado** (Loading Managed Code), resuelve nombres, obtiene clases de la memoria, crea stubs que son necesarios para la compilación JIT. La *class loader* forza la seguridad.

- **Conversión de IL a Código Nativo vía JIT**, el código del lenguaje intermedio no esta diseñado como un interprete tradicional bytecode o árbol de código, la conversión del lenguaje intermedio es realmente una compilación.

- ***Carga de Metadatos***, se encarga de checar la seguridad de tipos y la integridad de los métodos.

- **Recolector de Basura y Manejo de Excepciones** (Garbage Collection), el código administrado permite rastrear el apilado en el runtime, para que el runtime entienda el apilado individual de frames un código administrado tiene que ser proporcionado por el JITer o por el compilador.

- **Servicios de debugging**, estos servicios dependerán de la información producida por el compilador del lenguaje fuente y se emiten dos mapas, un mapa del lenguaje fuente de la construcción de direcciones en el flujo de instrucciones y un mapa de las direcciones de las localidades en el apilado de frames.

ADO.NET es una evolución del modelo de acceso a datos de ADO que controla directamente los requisitos del usuario para programar aplicaciones escalables. Se diseñó específicamente para el Web, teniendo en cuenta la escalabilidad, la independencia y el estándar XML. ADO.NET utiliza algunos objetos ADO, como **Connection** y **Command**, y también agrega objetos nuevos. Algunos de los nuevos objetos clave de ADO.NET son **DataSet**, **DataReader** y **DataAdapter**.

La diferencia más importante entre esta fase evolucionada de ADO.NET y las arquitecturas de datos anteriores es que existe un objeto, **DataSet**, que es independiente y diferente de los almacenes de datos. Por ello, **DataSet** funciona como una entidad independiente. Se puede considerar el objeto DataSet como un conjunto de registros que siempre está desconectado y que no sabe nada sobre el origen y el destino de los datos que contiene. Dentro de un objeto **DataSet**, de la misma manera que dentro de una base de datos, hay tablas, columnas, relaciones, restricciones, vistas, etc.

El objeto ***DataAdapter*** es el objeto que se conecta a la base de datos para llenar el objeto **DataSet**. A continuación, se vuelve a conectar a la base de datos para actualizar los datos de dicha base de datos a partir de las operaciones realizadas en los datos contenidos en el objeto **DataSet**. En el pasado, el procesamiento de datos se basaba principalmente en la conexión. Ahora, con el fin de proporcionar a las aplicaciones multinivel mayor eficacia, se está adoptando para el procesamiento de datos un enfoque basado en mensajes que manipulan fragmentos de información.

En el centro de este enfoque se sitúa el objeto **DataAdapter**, que proporciona un puente entre un objeto **DataSet** y un almacén de datos de origen para recuperar y guardar datos. Para ello, envía solicitudes a los comandos SQL apropiados que se ejecutan en el almacén de datos.

El objeto ***DataSet*** basado en XML proporciona un modelo de programación coherente que funciona con todos los modelos de almacenamiento de datos: sin formato, relacional o jerárquico. Funciona sin tener 'conocimiento' del origen de los datos y representa a los datos que contiene como colecciones y tipos de datos. Independientemente del origen de los datos del objeto **DataSet**, éstos se manipulan mediante el mismo conjunto de API estándar expuestos a través del objeto **DataSet** y sus objetos subordinados.

Aunque el objeto **DataSet** no tiene conocimiento del origen de sus datos, el proveedor administrado tiene información detallada y específica.

La función del proveedor administrado es conectar, llenar y almacenar el objeto **DataSet** desde almacenes de datos (o viceversa). Los proveedores de datos OLE DB y SQL Server de .NET (System.Data.OleDb y System.Data.SqlClient) que forman parte de .Net Framework proporcionan cuatro objetos básicos: **Command**, **Connection**, **DataReader** y **DataAdapter**.

- Objetos **Connection**. Para conectar con una base de datos y administrar las transacciones en una base de datos.

- Objetos **Command**. Para emitir comandos SQL a una base de datos.

- Objetos **DataReader**. Proporcionan una forma de leer una secuencia de registros de datos sólo hacia delante desde un origen de datos SQL Server.

- Objetos **DataSet**. Para almacenar datos sin formato, datos XML y datos relacionales, así como para configurar el acceso remoto y programar sobre datos de este tipo.

- Objetos DataAdapter. Para insertar datos en un objeto DataSet y reconciliar datos de la base de datos.

Conexiones

Para establecer la comunicación con bases de datos, se utilizan las conexiones y se representan mediante clases específicas de proveedor, como **SQLConnection**. Los comandos viajan por las conexiones y devuelven conjuntos de resultados en forma de secuencias que puede leer un objeto **DataReader** o que se pueden insertar en un objeto

DataSet.

Las conexiones se pueden abrir explícitamente mediante llamadas al método **Open** de la conexión; también se pueden abrir implícitamente al utilizar un objeto **DataAdapter**.

Objetos DataReader

El objeto **DataReader** es, en cierto modo, sinónimo de un cursor de sólo lectura y sólo hacia delante para datos. La API de **DataReader** es compatible con datos sin formato y con datos jerárquicos. Cuando se ejecuta un comando en la base de datos, se devuelve un objeto **DataReader**. El formato del objeto **DataReader** devuelto es distinto de un conjunto de registros. Por ejemplo, podría utilizarse el objeto **DataReader** para mostrar los resultados de una lista de búsqueda en una página Web.

Objetos DataSet: Es similar al objeto **Recordset** de ADO, pero más eficaz y con una diferencia importante: **DataSet** siempre está desconectado. El objeto **DataSet** representa a una memoria caché de datos, con estructuras análogas a las de una base de datos, como tablas, columnas, relaciones y restricciones. Sin embargo, aunque se puede utilizar un objeto **DataSet** como una base de datos (y su comportamiento es muy similar), es importante recordar que los objetos **DataSet** no interactúan directamente con bases de datos ni con otros datos de origen.

Esto permite al programador trabajar con un modelo de programación que siempre es coherente, independientemente de dónde resida el origen de datos.

En los objetos **DataSet** se pueden colocar datos provenientes de una base de datos, un archivo XML, código o información escrita por el usuario. A continuación, a medida que se realizan cambios en el objeto **DataSet**, se puede hacer un seguimiento y una comprobación de los cambios antes de actualizar los datos de origen.

El método **GetChanges** del objeto **DataSet** crea en realidad otro objeto **DataSet** que sólo contiene los cambios realizados en los datos. Posteriormente, un objeto **DataAdapter** u otros objetos, utilizan este objeto **DataSet** para actualizar el origen de datos original.

El objeto DataSet tiene muchas características de XML, incluida la capacidad de producir y consumir datos XML y esquemas XML. Los esquemas XML se pueden utilizar para describir esquemas intercambiables a través de servicios Web. De hecho, un objeto **DataSet** con un esquema puede compilarse con seguridad de tipos y finalización automática de instrucciones.

Objetos DataAdapter (SQL)

El objeto **DataAdapter** funciona como un puente entre el objeto **DataSet** y los datos de origen. El uso del objeto **SqlDataAdapter** específico del proveedor (junto con los objetos **SqlCommand** y **SqlConnection** asociados) permite aumentar el rendimiento global al trabajar con bases de datos de Microsoft SQL Server. El objeto **DataAdapter** utiliza comandos para actualizar el origen de datos después de hacer modificaciones en el objeto **DataSet**. Si se utiliza el método **Fill** del objeto **DataAdapter**, se llama al comando SELECT; si se utiliza el método **Update** se llama al comando INSERT, UPDATE o DELETE para cada fila modificada.

Es posible establecer explícitamente estos comandos con el fin de controlar las instrucciones que se utilizan en tiempo de ejecución para resolver cambios, incluido el uso de procedimientos almacenados. Un objeto **CommandBuilder** puede generarlos en tiempo de ejecución a partir de una instrucción de selección. Sin embargo, para generar en tiempo de ejecución hay que hacer un viaje de ida y vuelta adicional al servidor con el fin de recopilar los metadatos necesarios; por tanto, si se proporcionan explícitamente los comandos INSERT, UPDATE y DELETE en tiempo de diseño, el rendimiento en tiempo de ejecución mejorará.

- La tecnología ADO.NET, integrada en .Net Framework, es el siguiente estado de evolución de ADO.

- Se diseñó teniendo en cuenta los modelos multinivel, la independencia y el estándar XML. Para estos escenarios se proporcionan dos objetos nuevos, DataSet y DataAdapter.

- Se puede utilizar ADO.NET para obtener datos de una secuencia o para almacenar datos en una memoria caché a fin de realizar actualizaciones.

- La documentación contiene mucha más información acerca de ADO.NET.

- Hay que tener en cuenta que se puede ejecutar un comando directamente en la base de datos para realizar inserciones, actualizaciones y eliminaciones. Para insertar, actualizar o eliminar datos no hay que colocarlos primero en un objeto DataSet.

- Además, se puede utilizar un objeto DataSet para enlazar con los datos, examinarlos y explorar sus relaciones.

VI. MATERIAL Y METODO

1. Material:

Hardware: Tiene las Siguientes Características.

- Microprocessador Intel Pentium 4 de 3.2 GHz de velocidad.
- 80 GB de Disco Duro
- 1 GB de RAM

Software: el software donde se llevara acabo la aplicación será.

- Microsoft Windows XP Profesional son SP 2 como sistema operativo.
- Microsoft SQL Server 2005 como gestor de Base de Datos.
- Microsoft Visual C# como lenguaje de programación.
- Microsoft Office XP para la elaboración de documentos y diapositivas.

2. Diseño Metodológico

Para la debida realización de este sistema utilizamos el método "ciclo de vida clásico" el cual brinda las condiciones necesarias para el debido almacenamiento y tratamiento de la información, las fases a seguir para la elaboración de este sistema son las siguientes:

Planificación del Sistema: En esta fase se realizó la debida entrevista al usuario o cliente para recopilación de información y tener en cuenta todas necesidades del sistema

Análisis: El principal objetivo de esta fase es llegar a comprender realmente lo que el usuario desea y representarlos en diferentes formas para que éste no fracase.
Esta etapa puede parecer una tarea relativamente sencilla, pero las apariencias engañan. Abundan los casos en que se puede llegar a malas interpretaciones o falta de información, esta etapa nos proporciona las siguientes herramientas:

- Diagrama de Flujo de Datos (DFD).
- Diccionario de Datos (DD).
- Especificación de Requisitos Software (ERS).

Diseño: Es la fase de identificación de las diferentes funciones pero realmente es un proceso multiplazo que se enfoca sobre cuatro atributos distintos del programa: la estructura de los datos, la arquitectura del Software, el detalle procedimental y la caracterización de la interfaz.

El proceso de diseño traduce los requisitos en una representación del Software que pueda ser establecida de forma que obtenga la calidad requerida antes de que comience la codificación. Al igual que los requisitos, el diseño se documenta y forma parte de la configuración del Software.

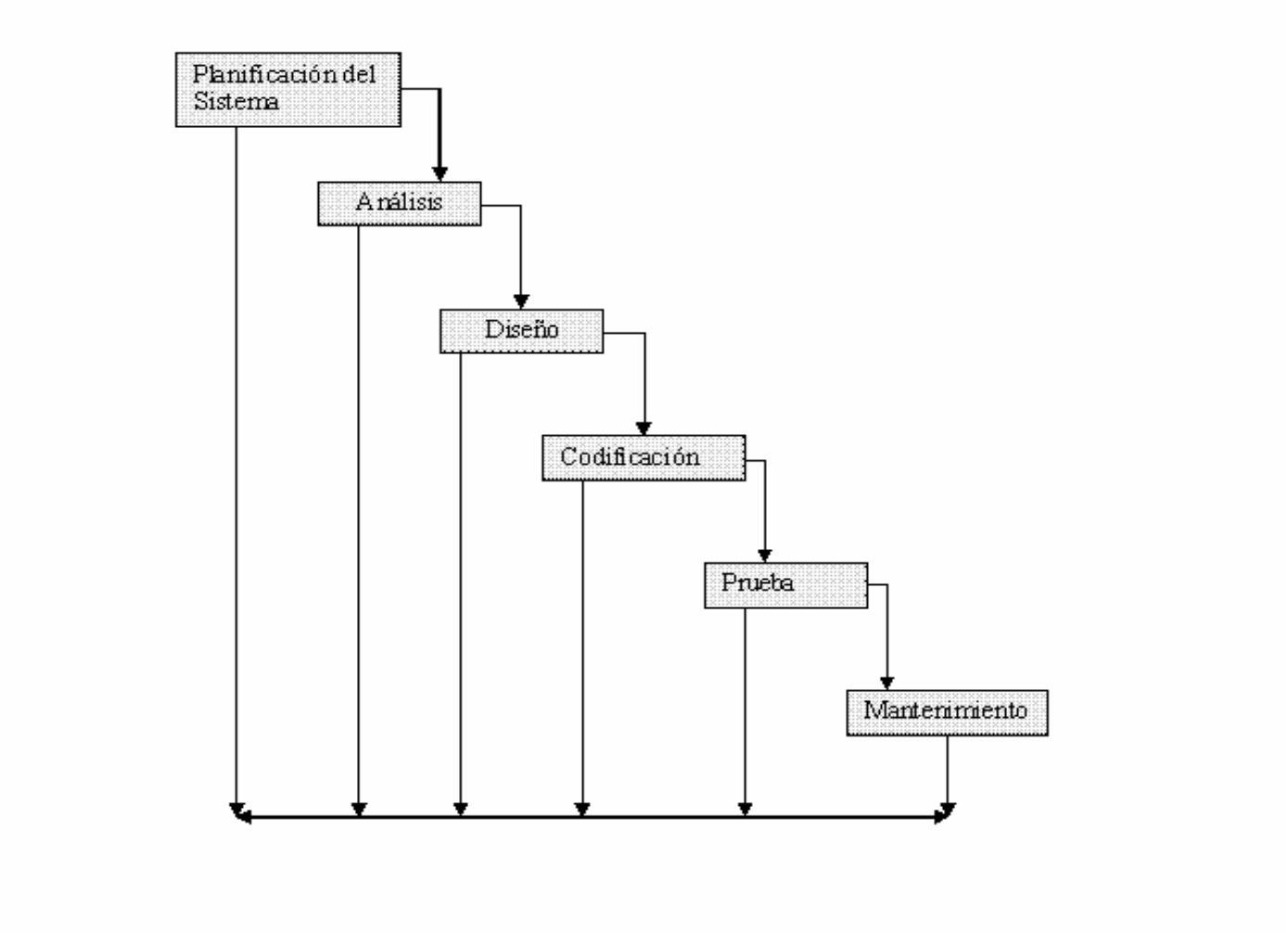
Codificación: El diseño debe traducirse en una forma legible para la máquina. Si el diseño se realiza de una manera detallada, la codificación puede realizarse mecánicamente, para debida realización de este sistema usamos el entorno Microsoft Visual Studio .Net (el cual es un entorno de desarrollo integrado que contiene múltiple funciones y todas las herramientas necesarias para construir programas para Microsoft Windows, es por ello que consideramos que esta plataforma se adapta al desarrollo de nuestra aplicación, además hicimos uso del gestor de base de datos SQL Server 2005 para la elaboración de la diferentes tablas, relaciones, consultas y la herramienta Reportviewer de Visual Studio.Net para la elaboración de los informes.

Prueba: Una vez que se ha generado el código, comienza la prueba del programa. La prueba se centra en la lógica interna del Software, asegurando que todas las sentencias se han probado, y en las funciones externas, realizando pruebas que aseguren que la entrada definida produce los resultados que realmente se requieren. Los métodos de prueba del

Software tienen el objetivo de diseñar pruebas que descubran diferentes tipos de errores con menor tiempo y esfuerzo.

Mantenimiento: Es indudable que el Software una vez entregado al cliente sufrirá cambios posible excepción es el Software Empotrado). Los cambios ocurrirán debido a que se hayan encontrado errores o que el Software deba adaptarse a posibles cambios, además de contar con un respaldo y restauración del mismo

Estructura del Modelo de Vida Clásico



VII. ANÁLISIS

Especificación de Requisitos del Software(ERS)

1. Introducción

1.1 Propósito

Esta aplicación tiene por objeto llevar el monitoreo de las diferentes actividades realizadas en el Plan Operacional Anual de los Programas de Pequeñas Ayudas de Investigación en la Vicerrectoría de Investigación y Postgrado de la UNAN-LEON.

Este Documento se dirige al Oficial Financiero de la VIP de la UNAN-LEÓN y a los usuarios finales tienen que estudiarlo para poder comprender su funcionamiento y hacer uso correcto y eficiente de la aplicación.

1.2 Alcance

El nombre con el que se conocerá esta aplicación será: **Sistema de Monitoreo de los Subproyectos de Investigación (SiMSI).**

La meta general de este software es realizar de forma eficiente el monitoreo de las actividades de los distintos subproyectos de investigación, mas concretamente lo que se desea es:

- Llevar un registro eficiente de los Subproyectos de investigación recepcionados en la Vicerrectoría de investigación y postgrado de la UNAN-LEON.
- Llevar el control de las actividades efectuadas por los Subproyectos en el periodo previamente establecido en el plan operacional anual de los mismos (POA).

- Obtener reportes actualizados de los subproyectos de pequeñas ayuda de investigación que están en ejecución, en espera, concluidos y de aquellos que no hallan cumplido satisfactoriamente los requisitos para su aprobación o puesta en ejecución por motivos distintos o limitantes en falta de financiamiento.

Definiciones, acrónimos y abreviaturas:

1. **Usuario o Administrador:** Son Todos aquellos que podrán hacer uso de la aplicación.

2. **POA:** Plan Operacional Anual es un documento que identifica los resultados específicos que se necesita lograr dentro del tiempo establecido (un año).

3. **PAI:** Programa de Pequeñas Ayudas de Investigación El proyecto de investigación es el instrumento mediante el cual se realiza investigación en la UNAN-León.

4. **Subproyectos:** son todos aquellos proyectos de investigación.

5. **Reporte:** acción de representar un registro de una entidad.

6. **Extrafinanciamiento:** es un financiamiento extra que se le asigna a un Subproyecto.

7. **Presupuesto:** es el monto en dólares asignado a cada subproyecto para realizar sus actividades.

8. **Coordinador:** es una entidad que hace referencia al responsable de un subproyecto.

9. **Código Contable:** es el código Sorrolla asignado a cada Subproyecto aprobado.

2. Descripción General.

2.1 Relaciones del Producto

La Aplicación se desarrollará e implementará en un equipo con las siguientes características:

- Microprocessador Intel Pentium 4 de 3.2 GHz de velocidad.
- 80 GB de Disco Duro
- 1 GB de RAM

2.2 Funciones del Producto.

1. Capturar los datos de un nuevo Subproyecto.
2. Verificar Usuario.
3. Capturar los datos de las actividades de los Subproyectos a través del Plan Operacional Anual (POA).
4. Capturar los gastos realizados por los subproyectos.
5. Capturar los Extrafinanciamientos concedidos a los subproyectos.
6. Capturar los Datos de un nuevo Coordinador Facultativo.
7. Capturar los datos de un nuevo Coordinador de Subproyecto.
8. Actualizar los datos de los diferentes Coordinadores Facultativo.
9. Actualizar los datos de los Subproyecto.
10. Actualizar los datos de las actividades de los subproyectos hechas en el (POA).
11. Obtener un listado de todos los Subproyectos Recepcionados para ser presentados.
12. Obtener un listado de todos los Subproyectos Presentados para Aprobarse.
13. Obtener un listado de los proyectos que vayan a ejecutar una actividad con 5 días de anticipación y mostrar en que consiste dicha actividad, además de las actividades que están en subejecución.
14. Obtener un listado de todos los subproyectos Concluidos que no hayan entregado informe final.

15. Generación e Impresión de Reportes
- Por Área de conocimiento.
- De Coordinadores.
- Por Facultades.
- Por Fondos.
- De Gastos.
- De Propuestas.
- De Financiamientos.

16. Cambiar la Contraseña del Administrador.
17. Respaldar y Restaurar la información almacena en la Base Datos.
18. Obtener un Contendido de ayuda sobre toda la aplicación.

2.3 Características del Usuario:

El usuario Final de la aplicación debe tener conocimientos informáticos acerca del manejo básico de un sistema computarizado.

2.4 Restricciones Generales:

La División Administrativa usuario de nuestra aplicación deberá tener como requisitos mínimos:

- Computadores con: 80 GB de Disco Duro, 512 MB de memoria RAM, Procesadores con velocidad de 2.3 de GHz.
- Sistema Operativo Windows XP
- Visual Studio 2005.
- Microsoft SQL Server 2005.

3. Requisitos Específicos

3.1 Requisitos Funcionales

3.1.1. Capturar los datos de un nuevo subproyecto.

3.1.1.1. Especificación.

3.1.1.1.1 Introducción

Este proceso deberá capturar los datos correspondientes a cada subproyecto.

3.1.1.1.2 Entradas

Por pantalla: el usuario deberá proporcionar los siguientes datos.

2. Año de recepción del Subproyecto
3. Titulo del subproyecto.
4. Presupuesto Solicitado.
5. Nombre del Coordinador del Proyecto

Datos Proporcionados por el sistema: Referentes al Subproyecto**.**

- Código del Subproyecto
- Fecha de Recepción
- Tipo de Subproyecto
- Nombre del Coordinador Facultativo
- Facultad
- Área.
- Nombre del Coordinador del Proyecto si este ya existe en la base de datos.

3.1.1.1.3 Proceso

Se mostrara un formulario de introducción de datos al usuario.

El código del nuevo Subproyecto es asignado automáticamente, el coordinador y la Facultad los agregara por medio de un botón buscar donde se desplegara la lista de los coordinadores facultativos y la facultad a la que estos pertenecen, luego se desplegara otra lista con el nombre de los coordinadores de proyectos que pertenecen a una determinada facultad si el Coordinador del proyecto ya existe solo lo seleccionamos de lo contrario se desplegara un formulario de introducción de datos de un nuevo Coordinador de Proyectos.

Los datos tales como Titulo, Año, Fecha de Recepción, Presupuesto Solicitado, Area, Tipo de Subproyecto, deben ser ingresados de forma obligatoria.

El Nombre del Coordinador Facultativo, Nombre de la Facultad, Código y el coordinador de Proyecto en el caso que este ya este en la base de datos del sistema, serán datos brindados por el sistema, a partir de estos datos se registrara un nuevo Subproyecto y se actualizara la base de datos.

3.1.1.1.4 Salida

Aparecerá una ventana con un mensaje “El proyecto fue ingresado correctamente” un botón con la opción aceptar.

3.1.2. VERIFICAR USUARIO

3.1.2.1. Especificación

3.1.2.1.1. Introducción.

Esta función permitirá a los usuarios ingresar al sistema con la cuenta correspondiente para realizar las funciones necesarias sobre los Subproyectos.

3.1.2.1.2 Entradas.

Por Pantalla:

- Usuario: este campo representa el nombre del usuario.
- Clave: este campo representa la clave para entrar a la cuenta del usuario.

3.1.2.1.3. Proceso.

Se mostrará al usuario la interfaz a través de la cual podrá ingresar los campos usuario y Clave.

3.1.2.1.4. Salidas.

Por pantalla formulario de presentación del sistema.

3.1.3. Capturar los datos de las actividades de los subproyectos mediante el Plan Operacional Anual (POA).

3.1.3.1 Especificación

3.1.3.1.1. Introducción.

Este proceso se realizara solamente si el proyecto esta en ejecución, que haya sido presentado y aprobado por el comité.

3.1.3.1.2. Entradas.

Por pantalla: Datos para codificar un nuevo POA.

- Pasos de la acción
- Actividad.
- Responsable Primario
- Responsable otros.
- Tiempo.

- Mecanismo de retroalimentación.
- Rubro.
- Monto.
- Mes de ejecución
- Observaciones.

Datos Proporcionados por el sistema: Nuevo POA

- Código del proyecto.
- Titulo del Proyecto
- Numero de la acción.
- Fecha de Inicio
- Fecha de Fin

3.1.3.1.3. Proceso.

Se muestra un formulario de introducción de datos al usuario, botón Buscar automáticamente al darle clic al botón se desplegara un listado con los proyectos que están en ejecución. Posteriormente se actualizara la base de datos con el nuevo registro ingresado.

Datos necesarios a introducir:

Los datos necesarios son: Pasos de la acción, Actividad, Fecha inicio, Fecha fin, estos datos deben ser ingresados de manera obligatoria al sistema.

3.1.3.1.4. Salida.

Se mostrara por pantalla un mensaje “Los datos han sido ingresando correctamente”,

3.1.4. Capturar los gastos realizados por los subproyectos.

3.1.4.1 Especificación.

3.1.4.1.1. Introducción.

Este proceso realiza la captura de los diferentes gastos hechos por cada un de los subproyecto que están en ejecución.

3.1.4.1.2. Entradas:

Por pantalla: Datos para codificar un nuevo Gasto.

- Monto del gasto

Datos Proporcionados por el sistema: Nuevo Gasto

- Código del subproyecto.
- Titulo.
- Fecha del Gasto
- Numero _ actividad.
- Monto.
- Nombre de la actividad.
- Actividad en ejecución o subejecución.
- Numero de Gastos.
- Saldo Anterior
- Saldo Actual

3.1.4.1.3. Proceso.

Se muestra un formulario de introducción de datos del usuario, el cual muestra la actividad que requiere un gasto, el monto actual del subproyecto y si esta en ejecución o subejecución.

Luego la base de datos se actualizara automáticamente cuando el usuario de clic sobre el botón Guardar.

Datos necesarios a introducir:

Los datos como Fecha del Gastos, Monto del Gasto son los únicos datos obligatorios para realizar este proceso.

3.1.4.1.4. Salida.

Se mostrara un mensaje si el mensaje "El Gasto fue Realizado" y un botón con al opción aceptar

3.1.5. Capturar los Extrafinanciamientos concedidos a los subproyectos.

3.1.5.1. Especificación

3.1.5.1.1. Introducción.

Este proceso deberá capturar los datos referentes a un Extrafinanciamiento.

3.1.5.1.2. Entradas.

Por pantalla: el usuario deberá proporcionar los siguientes datos.

- Extrafinanciamiento
- Nota

Datos proporcionados por el sistema: Referentes al Extrafinanciamiento.

- Código del Subproyecto.
- Titulo
- Fecha del Extrafinanciamiento
- Número de Financiamiento.
- Nota Anterior.
- Presupuesto Aprobado o anterior.

3.1.5.1.3. Proceso.

Se muestra un formulario de introducción de datos al usuario. Luego la base de datos se actualizara automáticamente cuando el usuario de clic sobre el botón Guardar.

Los datos como Fecha del Extrafinanciamiento, Monto del Extrafinanciamiento son los únicos datos obligatorios para realizar este proceso.

3.1.5.1.4. Salida.

Se mostrara un mensaje “El Extrafinanciamiento fue realizado” y un botón con la opción aceptar.

3.1.6. Capturar los Datos de un nuevo Coordinador Facultativo.

3.1.6.1. Especificación

3.1.6.1.1. Introducción.

Este proceso deberá mostrar automáticamente un formulario para capturar los datos correspondientes al Nuevo Coordinador Facultativo.

3.1.6.1.2. Entradas

Por pantalla: el usuario deberá proporcionar los siguientes datos.

- Nombre del Coordinador Facultativo
- Facultad
- Teléfono Particular
- Teléfono Oficina
- Email.
- Extensión

3.1.6.1.3. Proceso.

Se muestra un formulario de introducción de datos del usuario. El Nombre del Coordinador facultativo, Facultad, teléfono oficina son datos meramente obligatorios para el ingreso del nuevo **Coordinador Facultativo**. Luego la base de datos se actualizara automáticamente cuando el usuario de clic sobre el botón **Guardar**

3.1.6.1.4. Salida.

Se mostrara por pantalla un mensaje "El Coordinador fue ingresado" y un botón aceptar.

3.1.7. Capturar los Datos de un nuevo Coordinador de Subproyecto.

3.1.7.1 Especificación

3.1.7.1.1. Introducción.

Esta función le permitirá al usuario hacer el ingreso de un nuevo Coordinador de Subproyecto.

3.1.7.1.2. Entradas.

Por pantalla: el usuario deberá proporcionar los siguientes datos.

- Nombre del Coordinador del Subproyecto
- Teléfono Particular
- Teléfono Oficina
- Email.
- Extensión

3.1.7.1.3. Proceso.

Se muestra un formulario de introducción de datos del usuario. El Nombre del Coordinador facultativo, teléfono oficina son datos meramente obligatorios para el ingreso del nuevo

Coordinador Facultativo. Luego la base de datos se actualizara automáticamente cuando el usuario de clic sobre el botón **Guardar**

3.1.7.1.4. Salida.

Se mostrara por pantalla un mensaje “El Coordinador fue ingresado” y un botón aceptar

3.1.8. Actualizar los datos de los diferentes Coordinadores Facultativo

3.1.8.1. Especificación

3.1.8.1.1. Introducción.

Este proceso deberá permitir la actualización de los datos referentes al Coordinador Facultativo.

3.1.8.1.2. Entradas.

Por pantalla: el usuario podrá actualizar los datos como

- Nombre del Coordinador Facultativo.
- Teléfono Particular.
- Teléfono de Oficina.
- Email
- Facultad
- Extensión

3.1.8.1.3. Proceso.

Se muestra un formulario de introducción de datos del usuario. El Nombre del Coordinador facultativo, Facultad, teléfono oficina son datos meramente obligatorios para actualización del

Coordinador Facultativo. Luego la base de datos se actualizara automáticamente cuando el usuario de clic sobre el botón **Actualizar.**

3.1.8.1.4. Salida.

Se mostrara por pantalla un mensaje “El Coordinador fue Actualizado” y botón aceptar.

3.1.9. Actualizar los Datos del Subproyectos

3.1.9.1. Especificación

3.1.9.1.1. Introducción

Este proceso deberá permitir la actualización de los datos referentes a cualquier Subproyecto ya ingresado.

3.1.9.1.2. Entradas

Por Pantalla: el usuario deberá actualizar datos como.

- Código Contable
- Año
- Titulo del Subproyecto
- Presupuesto Solicitado
- Presupuesto Aprobado
- Tipo de subproyecto
- Area
- Facultad
- Fecha recepción
- Fecha presentación
- Fecha aprobación
- El estado del Proyecto

Datos Proporcionados por el Sistema:

Referente al Subproyecto

- Código

3.1.9.1.3. Proceso

Se mostrara un formulario de introducción de datos al usuario. El Código Contable, Año, Titulo, Presupuesto Solicitado, Presupuesto aprobado, Tipo de subproyecto, Facultad Fecha recepción, Fecha presentación, Fecha aprobación son datos meramente obligatorios para la actualización del **Subproyecto.** Luego la base se actualizara cuando el usuario del clic sobre el botón **Actualizar.**

3.1.9.1.4. Salida

Se mostrara un mensaje "El Subproyecto fue Actualizado", y botón aceptar.

3.1.10. Actualizar los datos de las actividades de los subproyectos hechas en el (POA).

3.1.10.1. Especificación

3.1.10.1.1. Introducción

El proceso deberá permitir la actualización de los datos de las actividades hechas en el **Plan Operacional Anual (POA)**

3.1.10.1.2. Entradas.

Por Pantalla: el usuario deberá actualizar datos como:

- Numero de la acción
- Pasos de la Acción
- Actividad
- Responsable Primario

- Responsable Otro
- Fecha Inicio
- Fecha Fin
- Tiempo
- Mecanismo de Retroalimentación
- Monto
- Rubro
- Mes de Ejecución
- Observaciones

Datos Proporcionados por el Sistema: referente al Plan Operacional Anual (POA).

a. Código

3.1.10.1.3. Proceso.

Se mostrara un formulario de introducción de datos al usuario. El Número de la Acción, Pasos de la Acción, Actividad, Fecha Inicio, Fecha Fin, Tiempo, Mes de Ejecución son datos obligatorios para la actualización del **Plan Operacional Anual (POA).** Luego la base se actualizara cuando el usuario del clic sobre el botón **Guardar.**

3.1.10.1.4. Salida.

Se mostrara un mensaje “El POA fue Actualizado”, y botón aceptar.

3.1.11. Obtener un listado de todos los Subproyectos Recepcionados para ser presentados.

3.1.11.1. Especificación.

3.1.11.1.2. Introducción.

Este proceso muestra un listado completo de todos los Subproyecto recepcionados por año.

3.1.11.1.2.Entradas.

Por Pantalla: el usuario deberá proporcionar los siguientes datos.

- Año

Datos Proporcionados por el Sistema: referente al los Subproyecto Recepcionados.

- Código.
- Titulo del Subproyecto.
- Fecha de Presentación

3.1.11.1.3.Proceso

Se mostrara un formulario en cual se encontraran todos los subproyectos recepcionados por año. Si el usuario desea modificar (Presentar un subproyecto) la fecha de Presentación es el único dato obligatorio. Posteriormente la base de datos se actualizara.

3.1.11.1.4.Salida

Cuando el usuario haya seleccionado la fecha de presentación y de aceptar en el botón inmediatamente el subproyecto desaparecerá de la lista de los subproyectos recepcionados.

3.1.12. Obtener un listado de todos los Subproyectos Presentados para Aprobarse.

3.1.11.1. Especificación.

3.1.11.2. Introducción.

Este proceso mostrara un listado de todos los **Subproyectos presentados por año.**

3.1.12.1.2.Entradas.

Por Pantalla: el usuario deberá proporcionar los siguientes datos.

- Año
- Código Contable.
- Presupuesto Solicitado

Datos Proporcionados por el Sistema: referente al los Subproyectos presentados.

- Código.
- Titulo del Subproyecto.
- Fecha Aprobación.

3.1.12.1.3.Proceso.

Se mostrara un formulario en cual se encontraran todos los subproyectos presentados por año. Si el usuario desea modificar (Aprobar un subproyecto) la fecha de aprobación, Codigo Contable y el Presupuesto Aprobado son los datos obligatorios. Posteriormente la base de datos se actualizara.

3.1.12.1.4.Salida.

Cuando el usuario haya seleccionado la fecha de aprobación, Código Contable, Presupuesto Aprobado y de aceptar en el botón aceptar inmediatamente el subproyecto desaparecerá de la lista de los subproyectos recepcionados.

3.1.13. Obtener un listado de los proyectos que vayan a ejecutar una actividad con 5 días de anticipación y mostrar en que consiste dicha actividad, además de las actividades que están en subejecución.

3.1.13.1. Especificación

3.1.13.1.1 Introducción.

Este proceso deberá mostrar automáticamente al iniciar una nueva cesión un listado de los proyectos que vayan a ejecutar una determinada actividad con 5 días de anticipación.

3.1.13.1.2.Entradas.

Por pantalla: Listado con todos los proyectos que realizaran actividades.

Clic de selección del proyecto del cual empezara a ver su actividad.

Datos proporcionados por el sistema:

- Código.
- Código Contable
- Número de la acción.
- Titulo del subproyecto.

3.1.13.1.3.Proceso.

Se mostrar la pantalla de de selección al usuario, el cual deberá seleccionar uno de los proyectos

3.1.13.1.4.Salida.

Se mostrara por pantalla la información referente a la actividad que estará ejecutando un determinado proyecto durante un tiempo especificado, tal como el numero de la actividad, nombre, fecha de inicio y fin de la actividad, los responsables etc. Toda la información referente al POA de ese proyecto.

3.1.14. Obtener un listado de todos los subproyectos Concluidos que no hayan entregado informe.

3.1.14.1. Especificación

3.1.14.1.1. Introducción.

Este proceso mostrara una lista de todos los **subproyectos Concluidos**

3.1.14.1.2.Entradas.

Por pantalla: Se mostrara una listado de todos aquellos proyectos que están concluido pero están pendiente de informe.

- Código.
- Código Contable
- Titulo del Subproyecto

3.1.14.1.3.Procesos.

Esta formulario solo mostrara un historial de todos aquellos subproyectos que ya concluyeron todas sus actividades pero están pendientes de la entrega del informe

3.1.14.1.4.Salida.

No hay ninguna salida solo servirá para un control de todos eso subproyectos.

3.1.15. Generación de Reportes de Áreas actualizados.

Especificación.

3.1.15.1. Introducción.

Este proceso mostrara un reporte de Area Actualizado de todos los subproyectos.

3.1.15.2. Entradas.

Por Pantalla: el usuario nada más seleccionara en el submenú el r**eporte de Area Actualizado**

Datos Proporcionados por el Sistema: referente al Reporte de Area Actualizado

a. Cantidad de subproyectos por Area
b. Area

3.1.15.3. Proceso.

Se mostrara un formulario de Impresión con las respectivas configuraciones.

3.1.15.4. Salida.

Todos los reportes constaran con un encabezado referente a la UNAN-León y la Vicerrectoría Investigación y Postgrado. El periodo que evaluara será desde el 2004 hasta la fecha.

Los medios de salida de información que se implementara serán a través de pantalla impresos. Además podrá exportar ha Excel o a formato PDF.

3.1.16. Generación de Reportes de Coordinadores Actualizados.

3.1.16.1. Especificación

3.1.16.1.1. Introducción

Este proceso solo mostrara todos los Coordinadores Facultativo y de Subproyectos.

3.1.16.1.2. Entradas.

Por pantalla: el usuario nada más seleccionara en el submenú el reporte de Coordinadores.

Datos Proporcionados por el sistema: referente al Reporte de Coordinadores Facultativo Actualizado

- Nombre de Coordinador
- Teléfono Particular.
- Teléfono Oficina.
- Email.
- Facultad

3.1.16.1.3. Proceso.

Se mostrara un formulario de Impresión con las respectivas configuraciones.

3.1.16.1.4. Salida.

Todos los reportes constaran con un encabezado referente a la UNAN-León y la Vicerrectoría Investigación y Postgrado. El periodo que evaluara será desde el 2004 hasta la fecha.

Los medios de salida de información que se implementara serán a través de pantalla impresos. Además podrá exportar ha Excel o a formato PDF.

3.1.17. Generación de Reportes de Facultad actualizado

3.1.17.1. Especificación.

3.1.17.1.1. Introducción.

Este proceso mostrara un reporte de Facultad Actualizado.

3.1.17.1.2. Entradas.

Por pantalla:

Tipo de Reporte: Cantidad de subproyectos aprobados por facultad, El monto que se le ha asignado a cada facultad

Datos proporcionados por el sistema:

- Facultad
- Cantidad de Proyectos aprobados.
- Monto asignado para cada Facultad

3.1.17.1.3. Proceso.

Tipo de Reporte: Es el tipo de reporte que el usuario desea generar.

Cada reporte contara con los datos necesarios para la respectiva impresión.

3.1.17.1.4. Salida.

Todos los reportes constaran con un encabezado referente a la UNAN-León y la Vicerrectoría Investigación y Postgrado. El periodo que evaluara será desde el 2004 hasta la fecha.

Los medios de salida de información que se implementara serán a través de pantalla impresos. Además podrá exportar ha Excel o a formato PDF.

3.1.18. Generación de Reportes Fondos actualizados

3.1.18.1. Especificación.

3.1.18.1.1. Introducción.

Este proceso consiste en mostrar dos reportes relacionado con los fondos utilizados en el año que transcurre y un año anterior.

3.1.18.1.2.Entradas.

Por pantalla:

Tipo de reporte: Porcentaje de Ejecución de Fondos del año que Transcurre mas disponibilidad de fondos, Porcentaje de Ejecución de fondos del año anterior mas disponibilidad de fondos.

Datos proporcionados por el sistema:

- Porcentaje de Fondo Asignado (que año elije)
- Importe asignado
- Importe Gastado
- Saldo Actual
- Disponibilidad Real de Fondos.

3.1.18.1.3.Proceso.

Tipo de Reporte: Es el tipo de reporte que el usuario desea generar.

Cada reporte contara con los datos necesarios para la respectiva impresión.

3.1.18.1.4.Salida.

Todos los reportes constaran con un encabezado referente a la UNAN-León y la Vicerrectoría Investigación y Postgrado. El periodo que evaluara será desde el 2004 hasta la fecha.

Los medios de salida de información que se implementara serán a través de pantalla impresos. Además podrá exportar ha Excel o a formato PDF.

3.1.19. Generación de Reportes de Gastos.

3.1.19.1. Especificación.

3.1.19.1.1.Introducción.

Este proceso consiste mostrar un estado de cuenta de todos aquellos reportes que se encuentre en estado de ejecución.

3.1.19.1.2.Entradas.

Por pantalla:

Tipo de reporte: Último Gasto, Estado de Cuenta.

Datos proporcionados por el sistema:

- Código Contable
- Titulo del los Subproyectos aprobados.
- Importe asignado.
- Importe Gasto.
- Saldo Actual a la Fecha.
- Fecha del Ultimo Gasto

3.1.19.1.3.Proceso.

Tipo de Reporte: Es el tipo de reporte que el usuario desea generar.

Cada reporte contara con los datos necesarios para la respectiva impresión.

3.1.19.1.4.Salida.

Todos los reportes constaran con un encabezado referente a la UNAN-León y la Vicerrectoría Investigación y Postgrado. El periodo que evaluara será desde el 2004 hasta la fecha.

Los medios de salida de información que se implementara serán a través de pantalla impresos. Además podrá exportar ha Excel o a formato PDF.

3.1.20. Generación de Reportes de Propuestas.

3.1.20.1. Especificación.

3.1.20.1.1.Introducción.

Este proceso será la parte medular de la aplicación es donde se encontraran la mayoría de información relacionada con todos los subproyectos (Recepcionados, Financiados, Presentados, Gráfico de Presentados)

3.1.20.1.2.Entradas.

Por pantalla:

Tipo de Informe: Estado de todos los Proyectos Recepcionados, Financiados, Presentados, Gráfico de Presentados.

Datos proporcionados por el sistema:

- Año
- Fecha de Recepción
- Código

- Código Contable
- Propuestas por Año
- Cantidad de subproyectos
- Presupuesto Aprobado
- Titulo del subproyecto.
- Presupuesto Solicitado.
- Cantidad de subproyectos decepcionados
- Cantidad de subproyectos presentados
- Cantidad de subproyectos aprobados
- Cantidad de subproyectos pendientes de presentación.

3.1.20.1.3.Proceso.

Tipo de Reporte: Es el tipo de reporte que el usuario desea generar.

Cada reporte contara con los datos necesarios para la respectiva impresión.

3.1.20.1.4.Salida.

Todos los reportes constaran con un encabezado referente a la UNAN-León y la Vicerrectoría Investigación y Postgrado. El periodo que evaluara será desde el 2004 hasta la fecha.

Los medios de salida de información que se implementara serán a través de pantalla impresos. Además podrá exportar ha Excel o a formato PDF.

3.1.21. Generación de Reportes de Financiamiento.

3.1.21.1. Especificación

3.1.21.1.1. Introducción.

Esta Proceso consiste en mostrar un reporte de todos los aquellos subproyecto que recibieron un Extrafinanciamiento.

3.1.21.1.2. Entradas.

Tipo de Reporte:

Datos proporcionados por el sistema:

- Año
- Código Contable.
- Titulo del Subproyecto.
- Cantidad de Financiamientos.
- Monto de cada Financiamiento.

3.1.21.1.3. Proceso.

El reporte contara con los datos necesarios para la respectiva impresión.

3.1.21.1.4. Salida.

Todos los reportes constaran con un encabezado referente a la UNAN-León y la Vicerrectoría Investigación y Postgrado. El periodo que evaluara será desde el 2004 hasta la fecha.

Los medios de salida de información que se implementara serán a través de pantalla impresos. Además podrá exportar ha Excel o a formato PDF.

3.1.22. Cambiar la Contraseña.

3.1.22.1 Especificación

3.1.22.1.1. Introducción.

En la actualidad la mayoría de sistemas cuenta con un sistema de seguridad creando clave de acceso para ingresar al sistema. Evitando la manipulación de información por personas ajenas.

3.1.22.1.2.Entradas.

Por pantalla:

- Contraseña Actual.
- Nueva Contraseña.

3.1.22.1.3.Proceso.

Este proceso consiste en validar la cuenta del administrador, esta fue anteriormente creada por el administrador.

Al ingresar la contraseña actual este dato será comparado con una que posee el sistema, si el dato es erróneo se enviara un mensaje “Contraseña Incorrecta” con botón aceptar.

3.1.22.1.4.Salida.

Con la nueva contraseña ingresada el administrador, podrá acceder automáticamente al sistema cuando inicie una nueva sesión.

3.1.23. Respaldar y Restaurar toda información almacena en la Base Datos.

3.1.23.1. Especificación

3.1.23.1.1. Introducción.

Una las principales características de los sistema información es la de guardar una copia de seguridad de toda la información de la base de datos del sistema para evitar la perdida definitiva de la información, además de poder restaura de su respectiva restauración del cual medio de almacenamiento secundario.

3.1.23.1.2.Entradas.

Por pantalla:

- Nombre del Archivo.
- Ruta de ubicación del Archivo.

•Funcionalidad(Respaldo o Restauración)

El usuario debe aplicar los cambios para que la operación se lleve acabo.

3.1.23.1.3.Proceso.

Se mostrara un por pantalla la introducción y selección de los datos requeridos.

Nombre del archivo: es un dato obligatorio tipo de texto.

Ruta: es un dato requerido, su contendido describe un la ruta o path de acceso a la carpeta o la dirección origen donde se realizara la función

Funcionalidad: es un dato requerido, se debe seleccionar la operación a realizar si respaldo (guardar) y restauración (abrir).

3.1.23.1.4.Salida.

Funcionalidad Respaldo: Se creara un archivo con el nombre que le asigne le administrador y es donde almacenara toda información de la base de datos.

Funcionalidad Restauración: se hará una copia de los archivos, provenientes de la fuente de restauración o medio secundario.

3.1.24. Ayuda.

3.1.24.1. Especificación.

3.1.24.1.1. Introducción.

Este proceso solo Mostrara todo el contenido acerca del diseño y elaboración del sistema.

3.1.24.1.2. Entradas.

Por pantalla:

•Índice

•Contenido

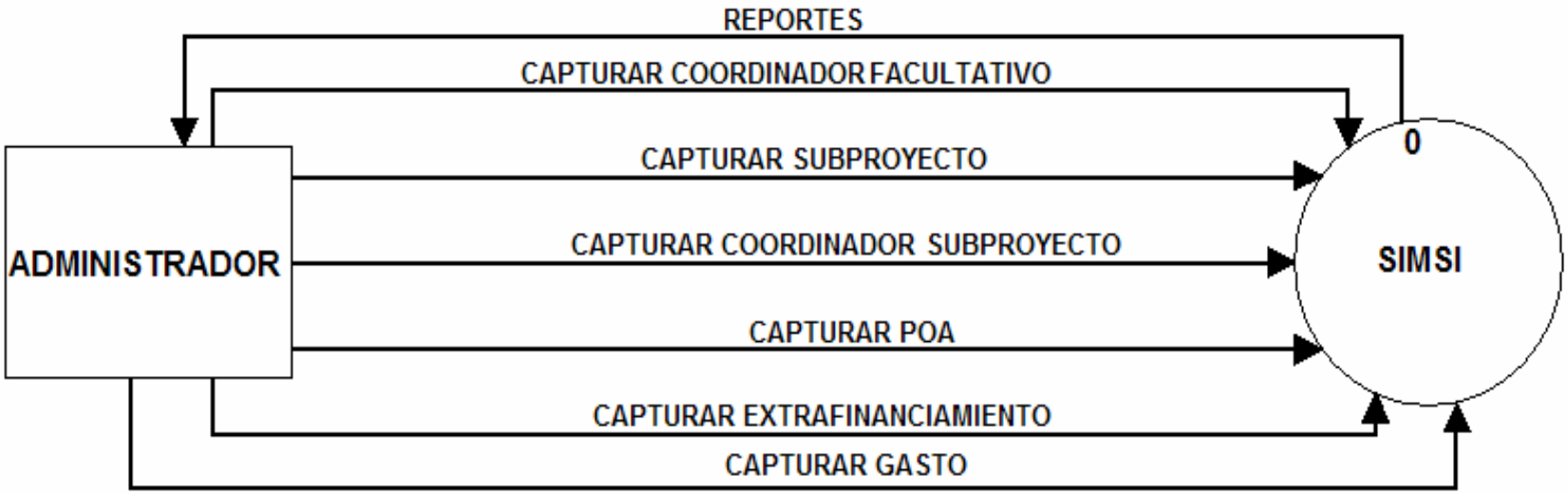
3.1.24.1.3.Proceso.

El usuario tendrá la opción de visualizar todo la información acerca del diseño del sistema

3.1.24.1.4.Salida.

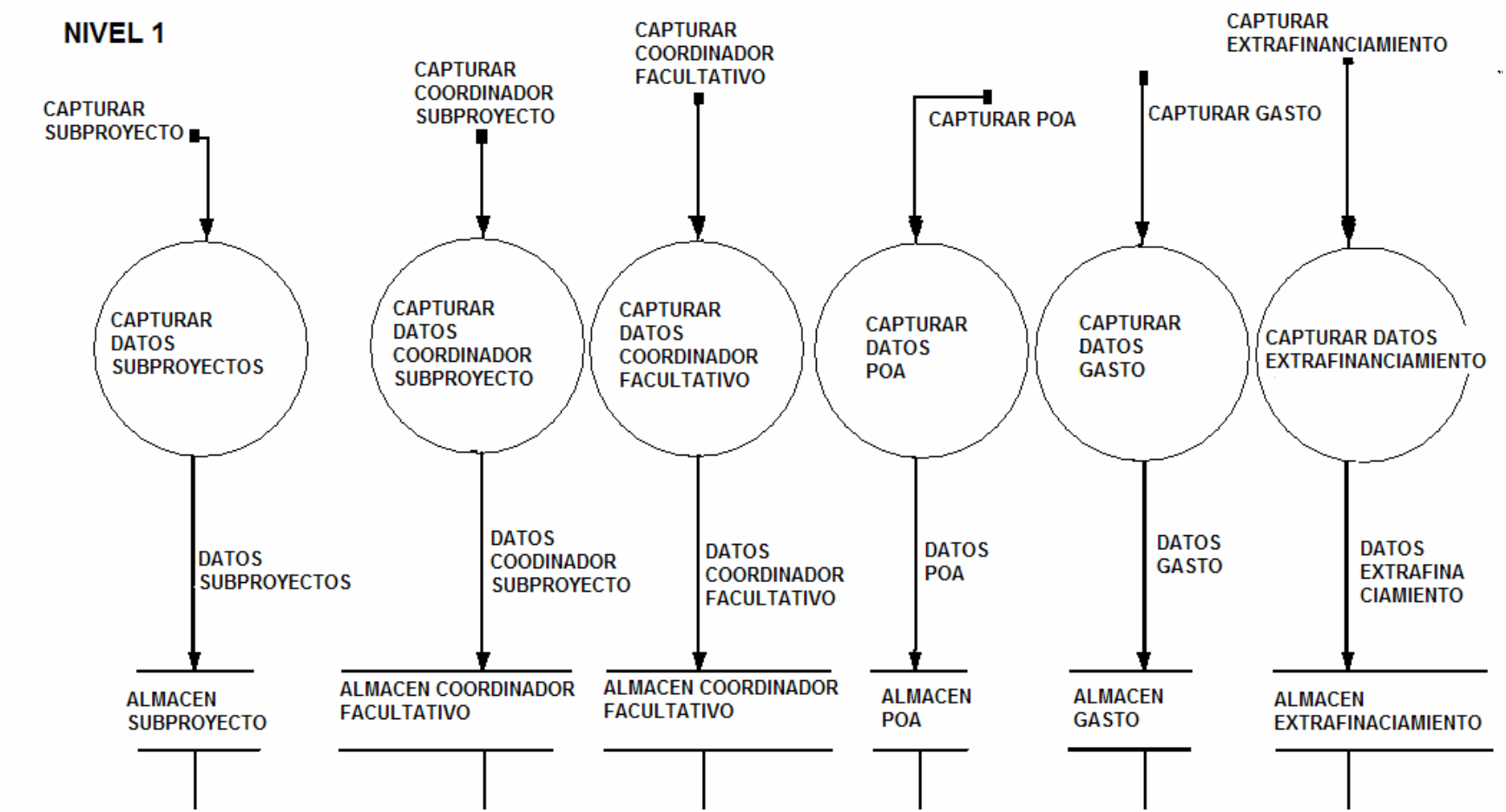
Los medios de salida de información que se implementara serán a través de pantalla impresos.

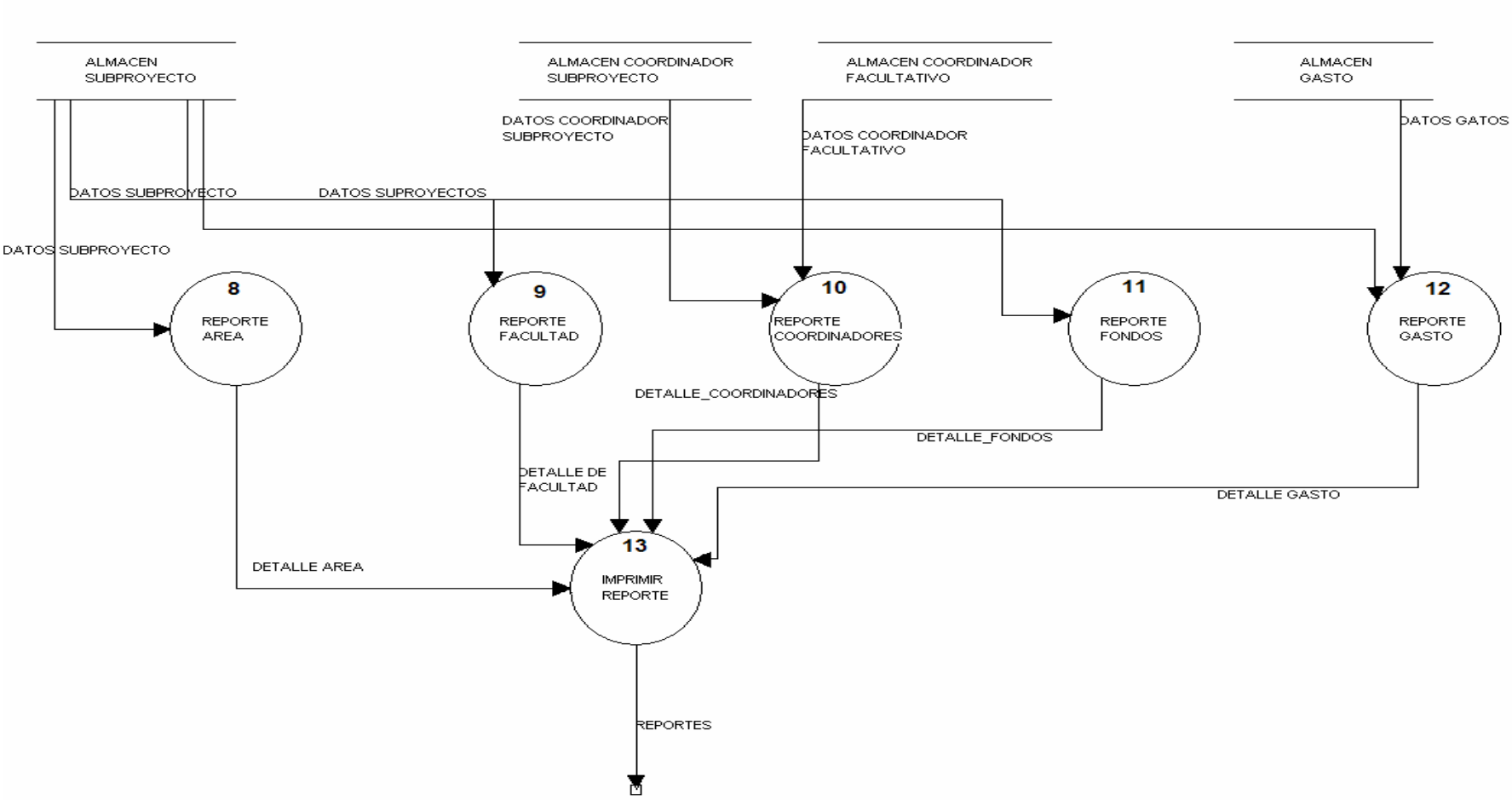
Diagrama de Flujo de Datos (DFD)



NIVEL 0

Diagrama de Flujo de Datos (DFD)


NIVEL 1
CAPTURAR SUBPROYECTO
CAPTURAR COORDINADOR SUBPROYECTO
CAPTURAR COORDINADOR FACULTATIVO
CAPTURAR POA
CAPTURAR GASTO
CAPTURAR EXTRAFINANCIAMIENTO
CAPTURAR DATOS SUBPROYECTOS
CAPTURAR DATOS COORDINADOR SUBPROYECTO
CAPTURAR DATOS COORDINADOR FACULTATIVO
CAPTURAR DATOS POA
CAPTURAR DATOS GASTO
CAPTURAR DATOS EXTRAFINANCIAMIENTO
DATOS SUBPROYECTOS
DATOS COODINADOR SUBPROYECTO
DATOS COORDINADOR FACULTATIVO
DATOS POA
DATOS GASTO
DATOS EXTRAFINA CIAMIENTO
ALMACEN SUBPROYECTO
ALMACEN COORDINADOR FACULTATIVO
ALMACEN COORDINADOR FACULTATIVO
ALMACEN POA
ALMACEN GASTO
ALMACEN EXTRAFINACIAMIENTO


ALMACEN SUBPROYECTO
ALMACEN COORDINADOR SUBPROYECTO
ALMACEN COORDINADOR FACULTATIVO
ALMACEN GASTO
DATOS COORDINADOR SUBPROYECTO
DATOS COORDINADOR FACULTATIVO
DATOS GATOS
DATOS SUBPROYECTO
DATOS SUPROYECTOS
DATOS SUBPROYECTO
8
REPORTE AREA
9
REPORTE FACULTAD
10
REPORTE COORDINADORES
11
REPORTE FONDOS
12
REPORTE GASTO
DETALLE_COORDINADORES
DETALLE_FONDOS
DETALLE DE FACULTAD
DETALLE GASTO
13
IMPRIMIR REPORTE
DETALLE AREA
REPORTES

El Diccionario de Datos: Es el listado donde se detallan todas funciones de los elementos del sistema (Flujos de datos, procesos y almacenes).

- **Capturar los datos de un nuevo Subproyecto.**

Año + Fecha de Recepción + Titulo del subproyecto + Tipo de subproyecto + Presupuesto Solicitado + Area + Nombre del Coordinador Primario

- **Verificar Usuario.**

Usuario + Contraseña

- **Capturar los datos de las actividades de los subproyectos a través del Plan Operacional Anual (POA).**

Pasos de la Acción + Actividad + Fecha Inicio + Fecha Fin + Tiempo + Mes Ejecución + Responsable Otro + Mecanismo de Retroalimentación + Rubro + Monto + Observaciones.

- **Capturar los gastos realizados por los subproyectos.**

Fecha del Gasto + Monto del Gasto.

- **Capturar los Extrafinanciamientos concedidos a los subproyectos**

Extrafinanciamiento + Fecha del Extrafinanciamiento + Presupuesto aprobado + Nota

- **Capturar los datos de un nuevo Coordinadores Facultativo.**

Nombre del Coordinador + Facultad + Teléfono Particular + Teléfono de Oficina + Email

- **Capturar los datos de un nuevo Coordinador de Subproyecto.**

Nombre del Coordinador + Teléfono Particular + Teléfono de Oficina + Extensión + Correo

- **Actualizar los datos de los diferentes Coordinadores Facultativos.**

Nombre del Coordinador + Teléfono Particular + Teléfono de Oficina + Email

- **Actualizar los datos de los Subproyecto.**

Año + Fecha de Recepción + Titulo del subproyecto + Tipo de subproyecto + Presupuesto Solicitado + Area + Presupuesto Aprobado + Fecha de Presentación + Fecha de Aprobación + Estado del Subproyecto

- **Actualizar los datos de las actividades de los subproyectos hechas en el POA.**

Pasos de la Acción + Actividad + Fecha Inicio + Fecha Fin + Tiempo + Mes Ejecución + Responsable primario + Responsable Otro + Mecanismo de Retroalimentación + Rubro + Monto + Observaciones.

- **Obtener un listado todos los Subproyectos Recepcionados para ser presentados.**

Año + Código + Titulo + Fecha de Recepción

- **Obtener un listado de todos los Subproyectos Presentados para Aprobarse.**

Año + Código +Titulo + Fecha de Aprobación + Código Contable + Presupuesto Aprobado + Nota.

- **Obtener un listado de los subproyectos que vayan a ejecutar una actividad con 5 días de anticipación y mostrar en que consiste dicha actividad, además de las actividades que están en subejecución.**

Código + Código Contable + Numero de Acción + Titulo

- **Obtener un listado de todos los subproyectos Concluidos que no hayan entregado informe**

Código + Código Contable + Titulo + Fecha de Entrega de Informe

- **Generación de Reportes de Áreas actualizados.**

Este proceso solo muestra un reporte de la Cantidad de proyectos PAI aprobados por Área de Conocimiento Período 2004 hasta la fecha.

- **Generación de Reportes de Coordinadores actualizados.**

Este proceso muestra un reporte de todos los Coordinadores con sus respectivos datos Actualizados.

- **Generación de Reportes de Facultades actualizados.**

Este proceso muestro la cantidad de subproyectos aprobados por facultades desde el 2004 a la fecha.

- **Generación de Reportes de Fondos actualizados**

Este proceso muestra la Absorción de Fondos y Disponibilidad real de Fondos

- **Reportes de Gastos**

Este proceso muestra un estado de cuenta todos los subproyectos que están en ejecución, además muestra los Últimos gastos realizados por los distintos subproyectos

- **Cantidad Propuestas de Subproyectos Recepcionados.**

Este proceso solo mostrara un reporte con los siguiente datos (Propuestas por Año + Cantidad + Presupuesto Solicitado).

- **Generación de Reportes de Financiamientos**

Este proceso mostrara un reporte con los siguientes datos (Año + Código Contable + Titulo + numero de Financiamiento + monto del Financiamiento.)

- **Cambiar la Contraseña del Administrador.**

Contraseña Actual + Nueva Contraseña

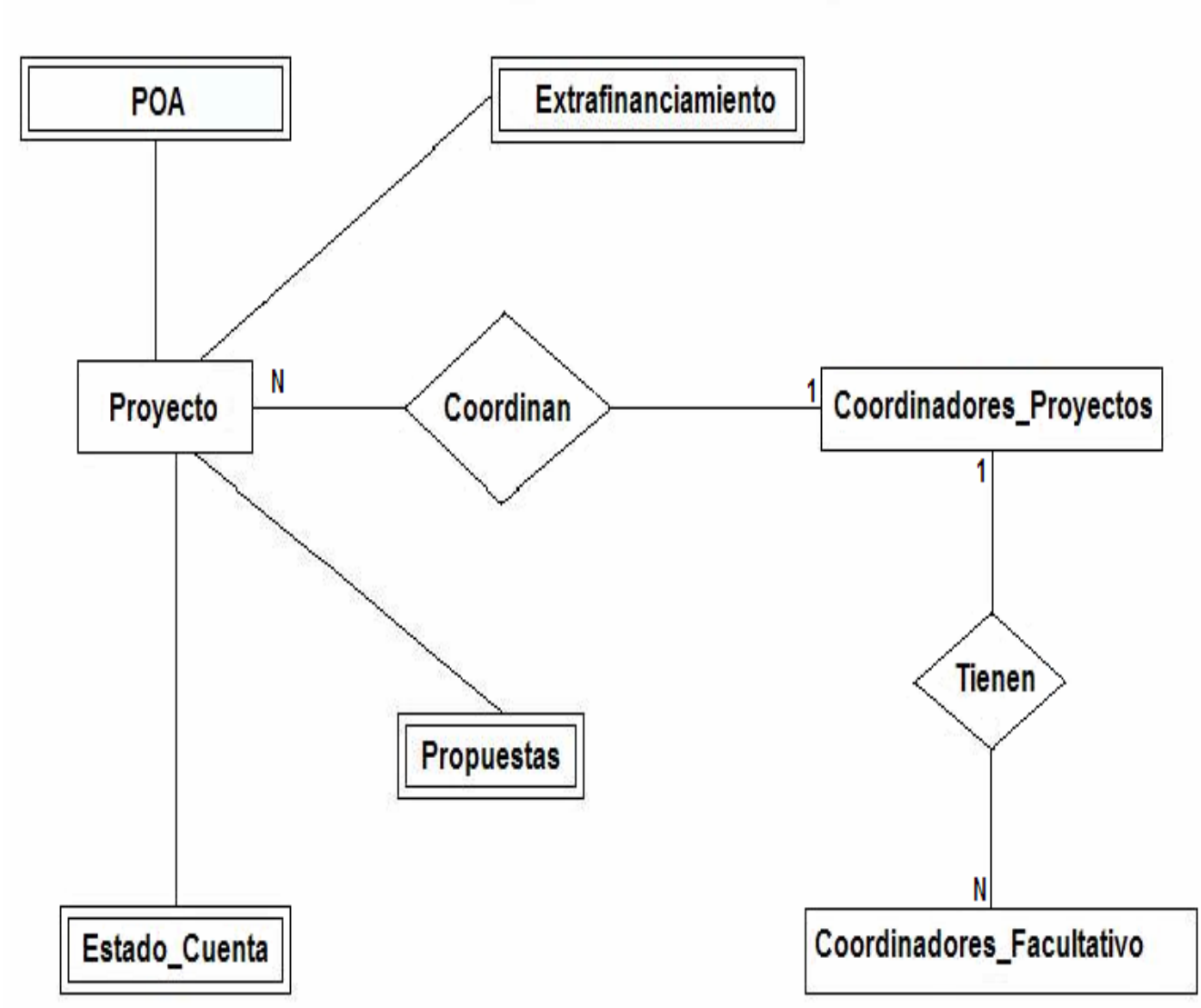
- **Respaldar y Restaurar de los toda información almacena en la Base Datos.**

Nombre de Archivo + Dirección de Almacenamiento + Funcionalidad (Respaldo o Restauración).

- **Obtener un Contenido de ayuda sobre toda la aplicación.**

Este proceso consiste en mostrar toda la información necesaria acerca del Software.

ESQUEMA RELACIONAL



VIII. Diseño

Diseño de Datos

Tabla: Proyecto		
Nombre del Campo	**Tipo de dato y longitud**	**Descripción**
Codigo	Int not null	Código para cada subproyecto, llave primaria
Codigo_C	Varchar (30)	Código contable para los subproyectos aprobados
Anyo	Int	Año de recepción del subproyecto
Área	Varchar (150)	Área de Conocimiento a la que pertenece el subproyecto
Estado_ejecucion	BIT	Estado en ejecución de los subproyectos aprobados
Estado_concluido	BIT	Estado que indica si el proyecto ya fue concluido
Presupuesto_Solic	Float	Presupuesto que solicitado por cada subproyecto
Presupuesto_Aprobado	Float	Presupuesto aprobado por el comité para cada subproyecto
Informe	BIT	Informe presentado por cada subproyecto al final de su conclusión
Tipo_Proyecto	Varchar (150)	Tipo de subproyecto puede ser: Becas, pequeñas ayudas o Otros
Id_coodinador	Int not null	Llave externa que identifica al coordinador del subproyecto

Tabla: POA		
Nombre del Campo	**Tipo de dato y longitud**	**Descripción**
Codigo	Int not null	Código para cada subproyecto, forma parte de la llave primaria
Numero_Accion	Int not null	Numero de acción para cada actividad en cada uno de los POA, llave primaria compuesta con el codigo.
Pasos_Accion	Varchar (500)	Pasos de la acción para cada POA
Responsable_Pri	Varchar (250)	Responsable Primario en cada subproyecto
Responsable_Otr	Varchar (250)	Responsables secundarios del subproyecto
Actividad	Varchar (500)	Actividades realizadas en cada POA
Fecha_Inicio	Date	Fecha de inicio de cada actividad
Fecha_fin	Date	Fecha de finalización de cada actividad
Tiempo	Varchar (50)	Tiempo en las horas que se realizara la actividad
Mecanismo_Retroa	Varchar (250)	Mecanismo de Retroalimentación hecha por la actividad
Monto	Float	Monto asignado para cada actividad
Rubro	Varchar (250)	
Mes_ejecucion	Varchar (50)	Mes de realización de la actividad
Observaciones	Varchar (500)	Observaciones hechas en la actividad
Revision	BIT	Revisión de cada actividad
Actividad_ejecucion	BIT	Actividad si esta en ejecución
Actividad_subjecucion	BIT	Indica si una actividad esta en Subejecución

Tabla: Coodinador_Proyecto		
Nombre del Campo	**Tipo de dato y longitud**	**Descripción**
Id_Coordinador	Int not null	Código para cada coordinador de un subproyecto, llave primaria
Nombre_Coordinador	Varchar (250)	Nombres y apellidos de coordinador del subproyecto
Telefono_particular	Nchar (10)	Teléfono celular o particular del coordinador
Telefono_oficina	Nchar (10)	Teléfono de oficina del coordinador
Ext	Nchar (10)	Extensión de teléfono de oficina del coordinador
Email	Varchar (250)	Correo Electrónico del Coordinador
Id_facultad	Int not null	Identificador de facultad, llave externa

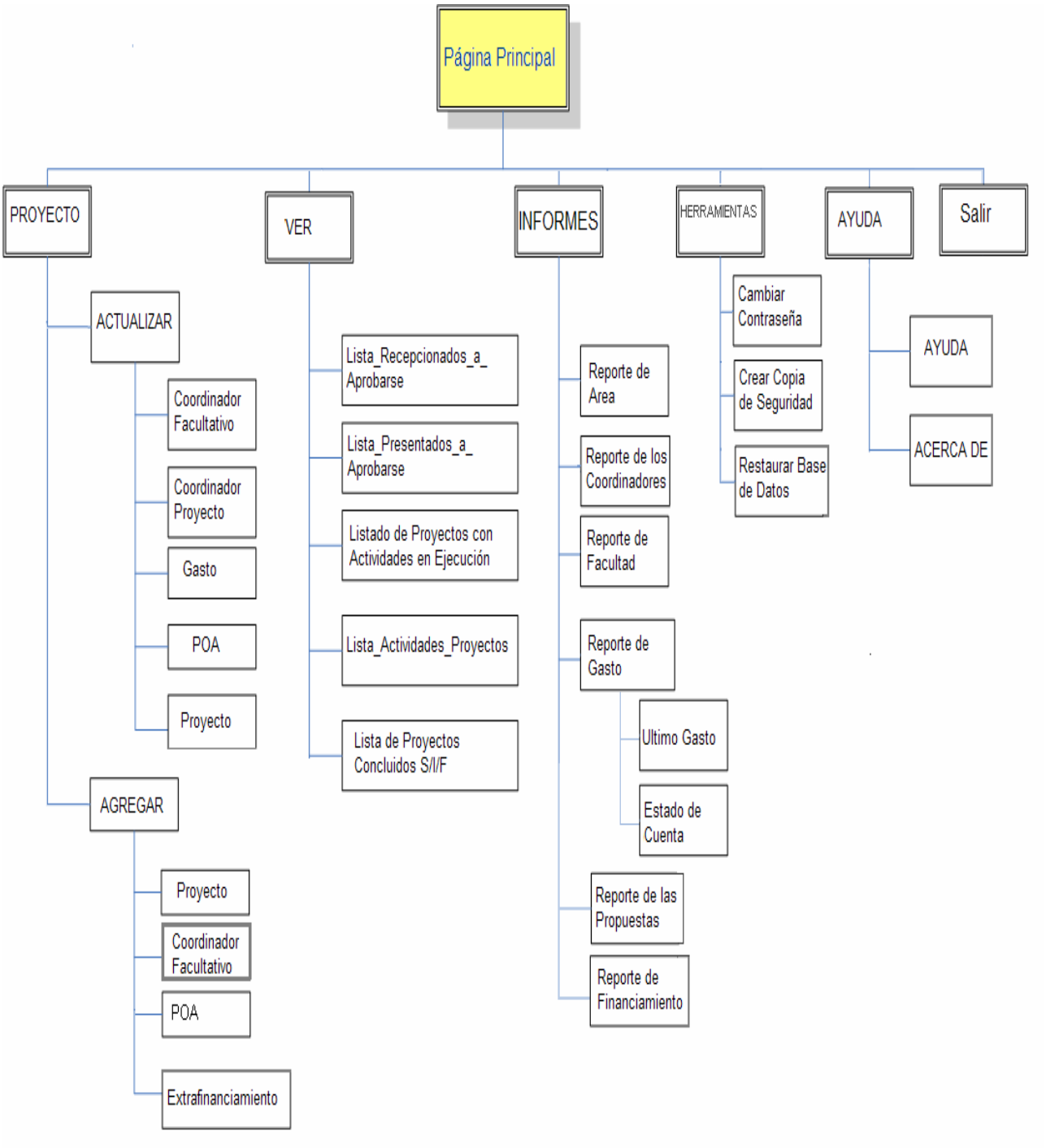
Tabla: Coodinador_Facultativo		
Nombre del Campo	**Tipo de dato y longitud**	**Descripción**
Id_Coordinador	Int not null	Código para cada coordinador facultativo, llave primaria.
Nombre_Coordinador	Varchar (250)	Nombres y apellidos de coordinador facultativo
Telefono_particular	Nchar (10)	Teléfono celular o particular del coordinador facultativo
Telefono_oficina	Nchar (10)	Teléfono de oficina del coordinador facultativo
Ext	Nchar (10)	Extensión de teléfono de oficina del coordinador facultativo
Email	Varchar (250)	Correo Electrónico del Coordinador facultativo

Tabla: Estado_Cuenta		
Nombre del Campo	**Tipo de dato y longitud**	**Descripción**
Codigo	Int not null	Código para cada subproyecto, forma parte de la llave primaria.
Numero_Gasto	Int not null	Numero de gastos realizados por cada subproyecto, llave primaria compuesta
Monto_Gasto	Float	Monto del gasto hecho algún subproyecto
Fecha	Date	Fecha de realización del gasto
Saldo_Actual	Float	Saldo actual para cada subproyecto

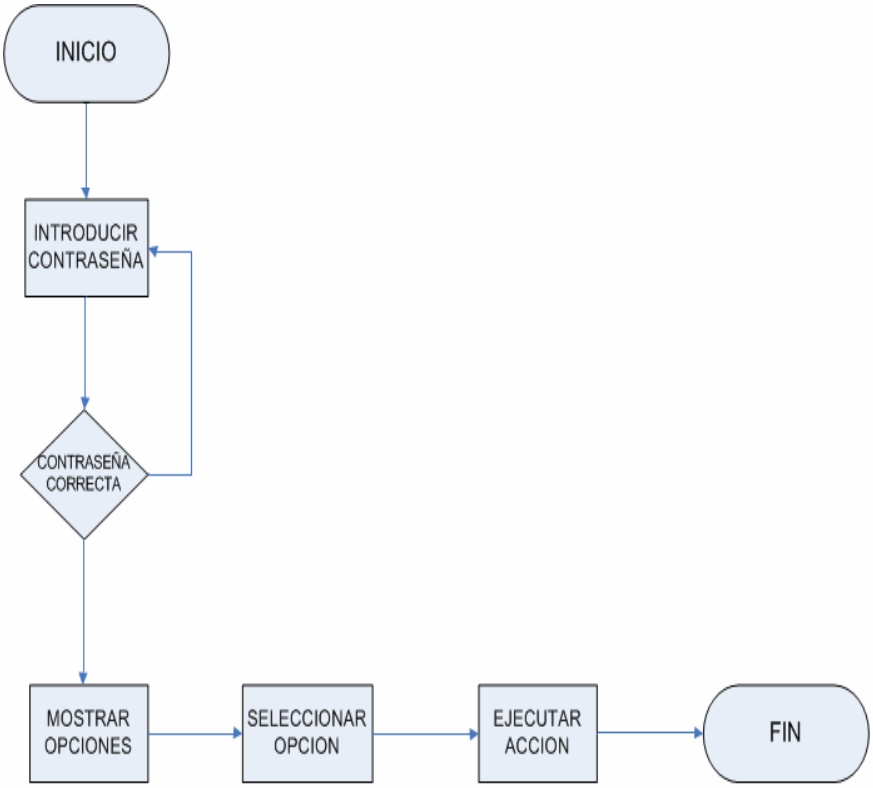
Tabla: Propuestas		
Nombre del Campo	**Tipo de dato y longitud**	**Descripción**
Codigo	Int not null	Código que identifica a cada Proyecto, este campo es una llave externa que hace referencia a la llave primaria de la tabla Proyecto, además es la llave primaria, no permite almacenar valores nulos.
Recepcionado	BIT	Es un campo de tipo Bolean que indica si un proyecto fue Recepcionado.
Presentado	BIT	Es un campo de tipo Bolean que indica si un proyecto fue Presentado.
Lista_Espera	BIT	Es un campo de tipo Bolean que indica si un proyecto se encuentra en Lista de Espera.
No_Aprobado	BIT	Es un campo de tipo Bolean que indica si un proyecto no fue Aprobado.
Fecha_Recepcion	Date	Fecha de Recepción de cada proyecto, campo de tipo fecha.
Fecha_Presentacion	Date	Fecha de Presentación de cada proyecto, campo de tipo fecha.
Fecha_Aprobacion	Date	Fecha que fue aprobado el proyecto, este campo de tipo fecha.
Nota	Varchar (250)	Nota sobre alguna observación hecha algún subproyecto
Abandonado	BIT	Solo indica si algún proyecto fue abandonado

Tabla: Extrafinanciamiento		
Nombre del Campo	**Tipo de dato y longitud**	**Descripción**
Codigo	Int not null	Código que identifica a cada Proyecto, este campo es una llave externa que hace referencia a la llave primaria de la tabla Proyecto, no permite almacenar valores nulos.
No_Financiamiento	Int not null	Numero de financiamiento hecho por un proyecto aprobado, campo de tipo entero, no permite valores nulos.
Fecha	Date	Fecha que se realizo el extrafinanciamiento, es un campo de tipo fecha.
Extrafinanciamiento	Float	Monto del extrafinanciamiento, es un campo de tipo flota.
Nota	Nchar (10)	Alguna nota que tenga que agregar al proyecto indicando por que fue hecha la extrafinanciación.

DISEÑO ARQUITECTÓNICO

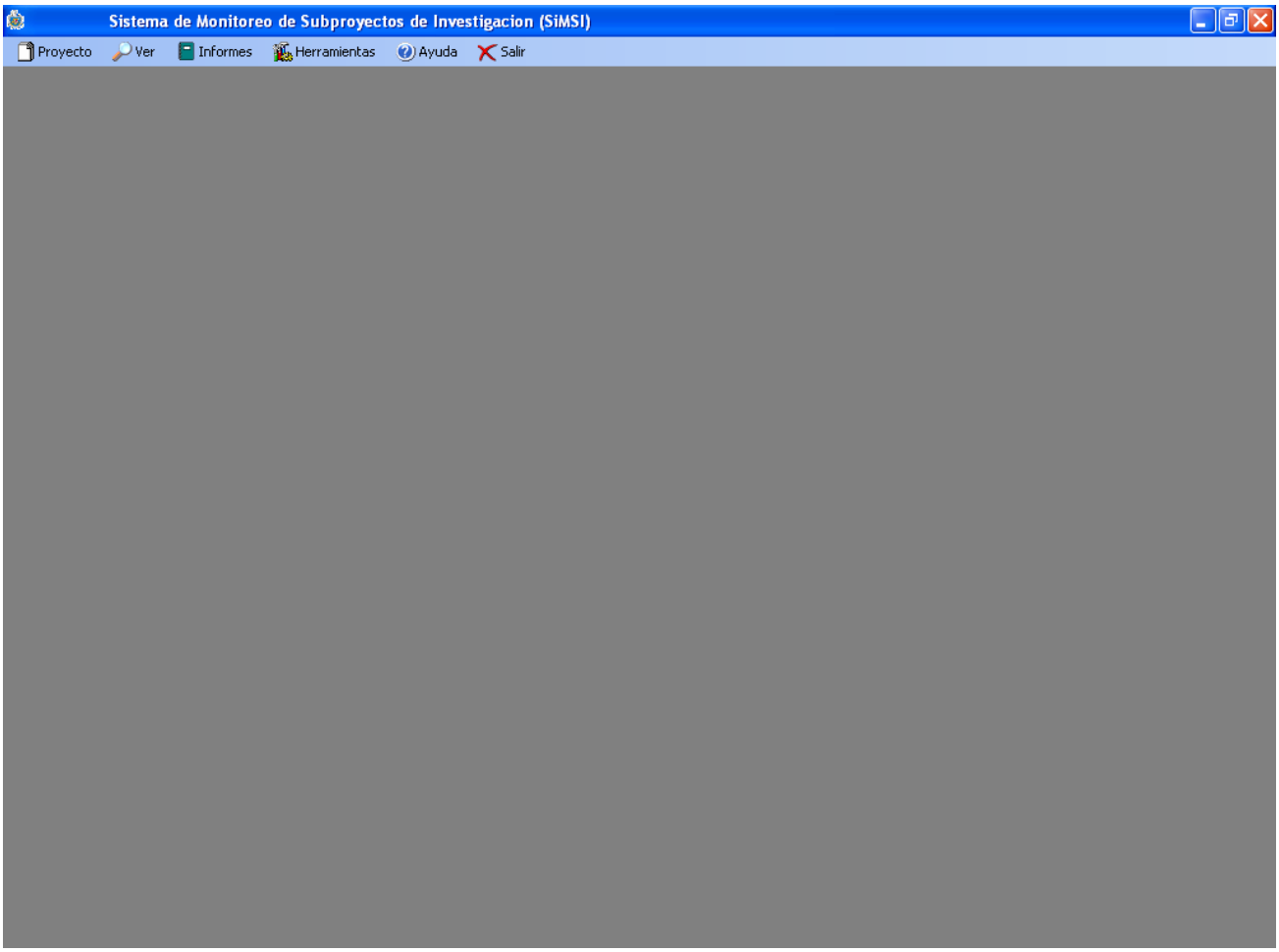


DISEÑO PROCEDIMENTAL

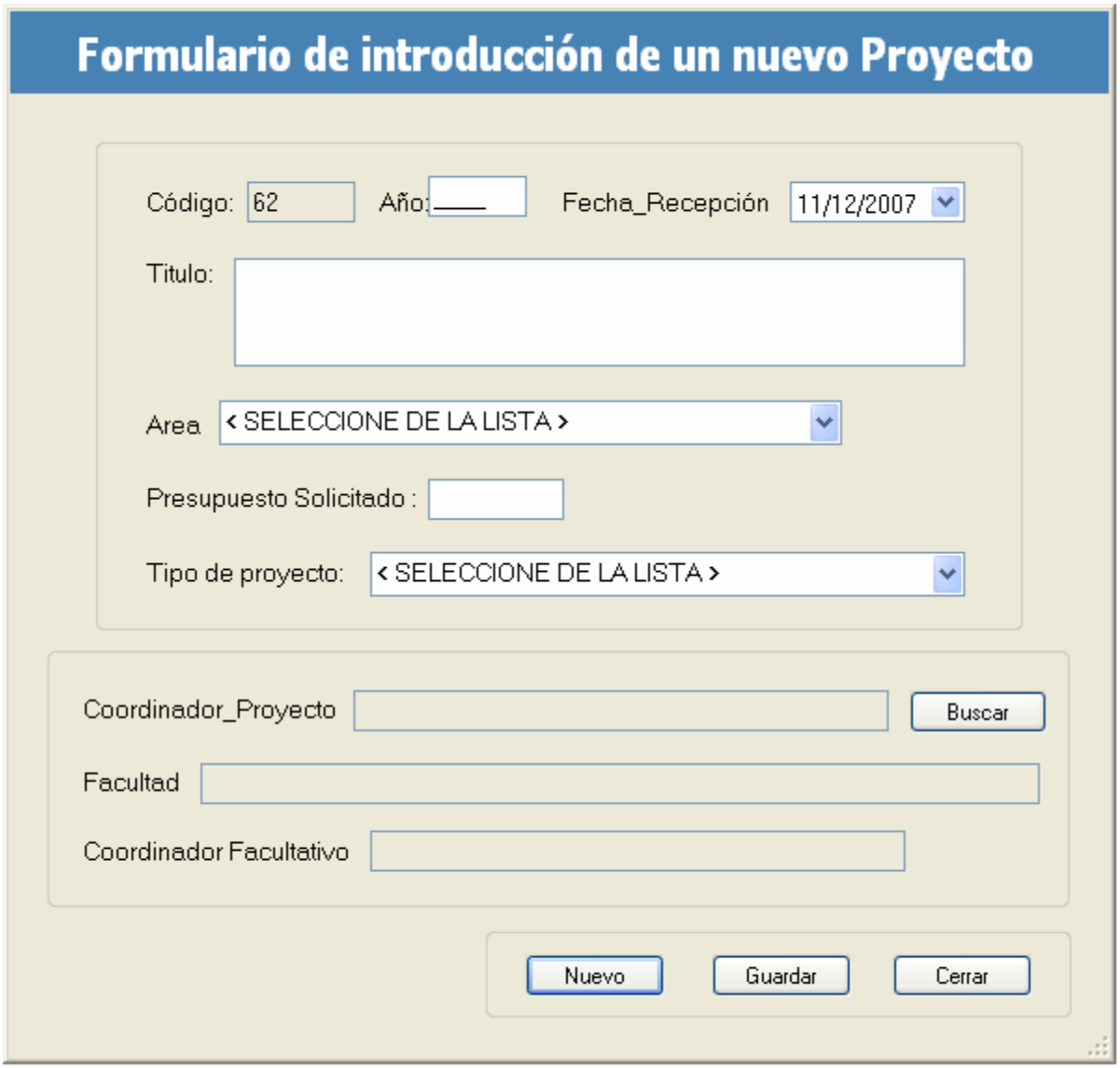


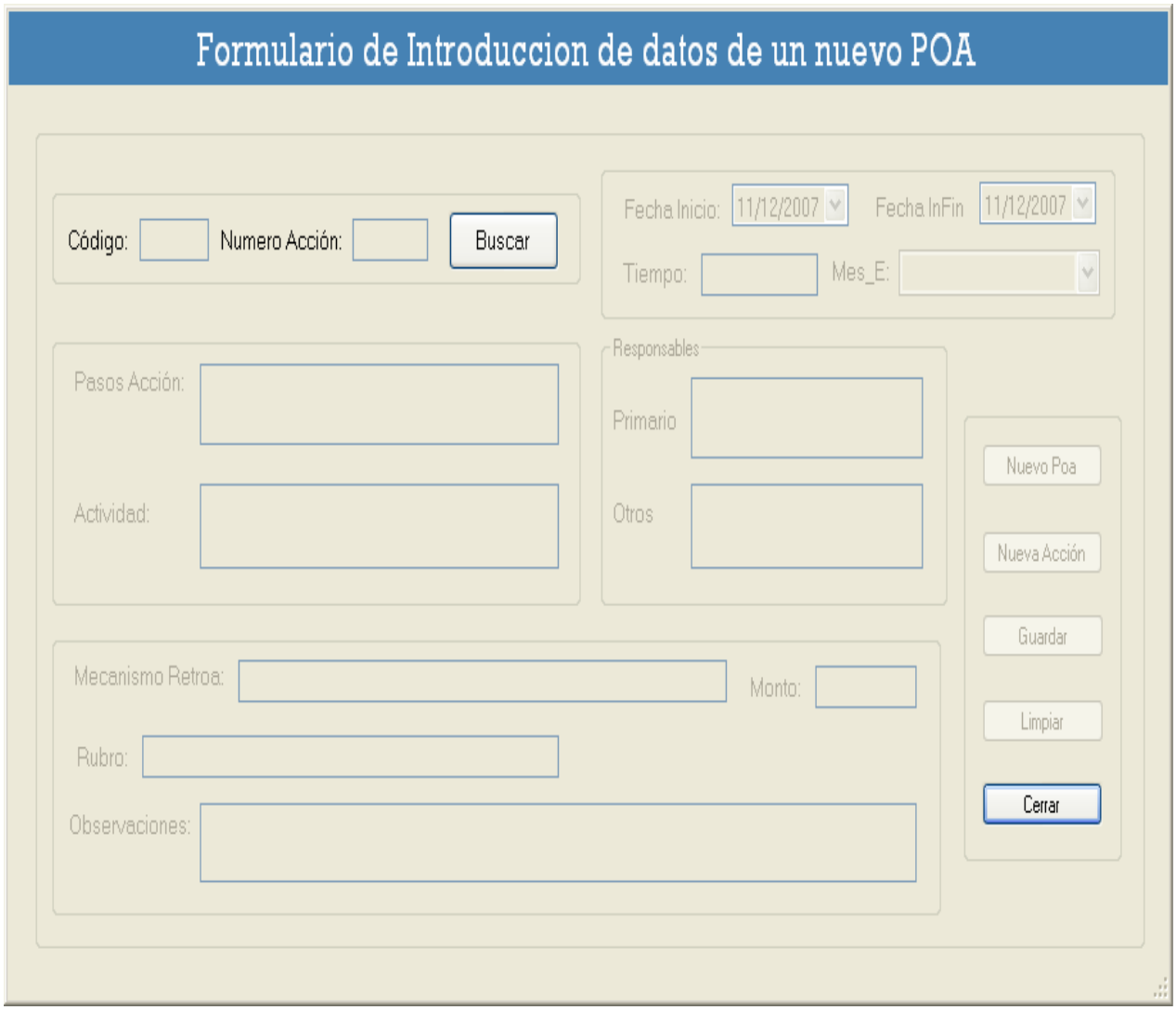
DISEÑO DE LA INTERFAZ

Formulario Principal

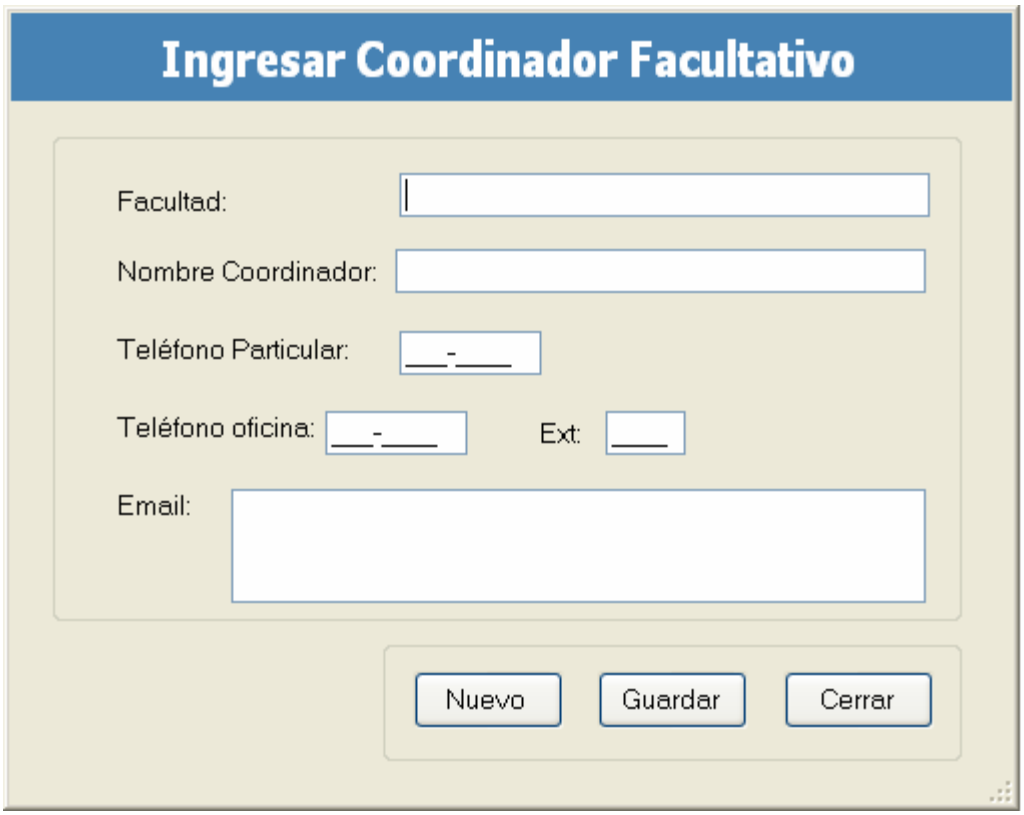


Formulario Agregar un Nuevo Proyecto

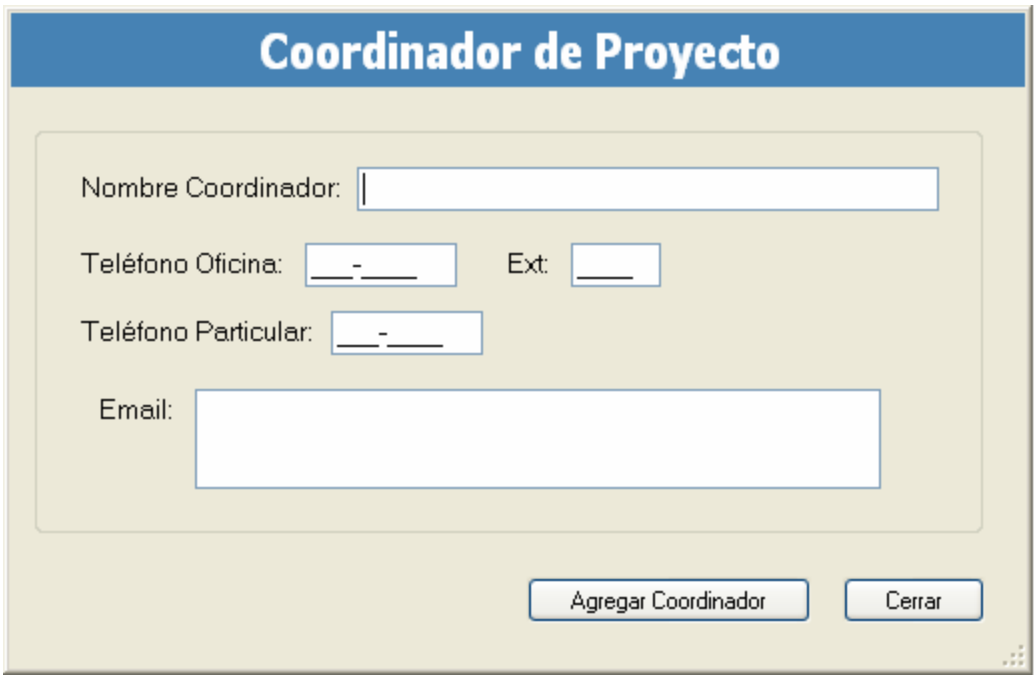




Formulario Agregar un POA

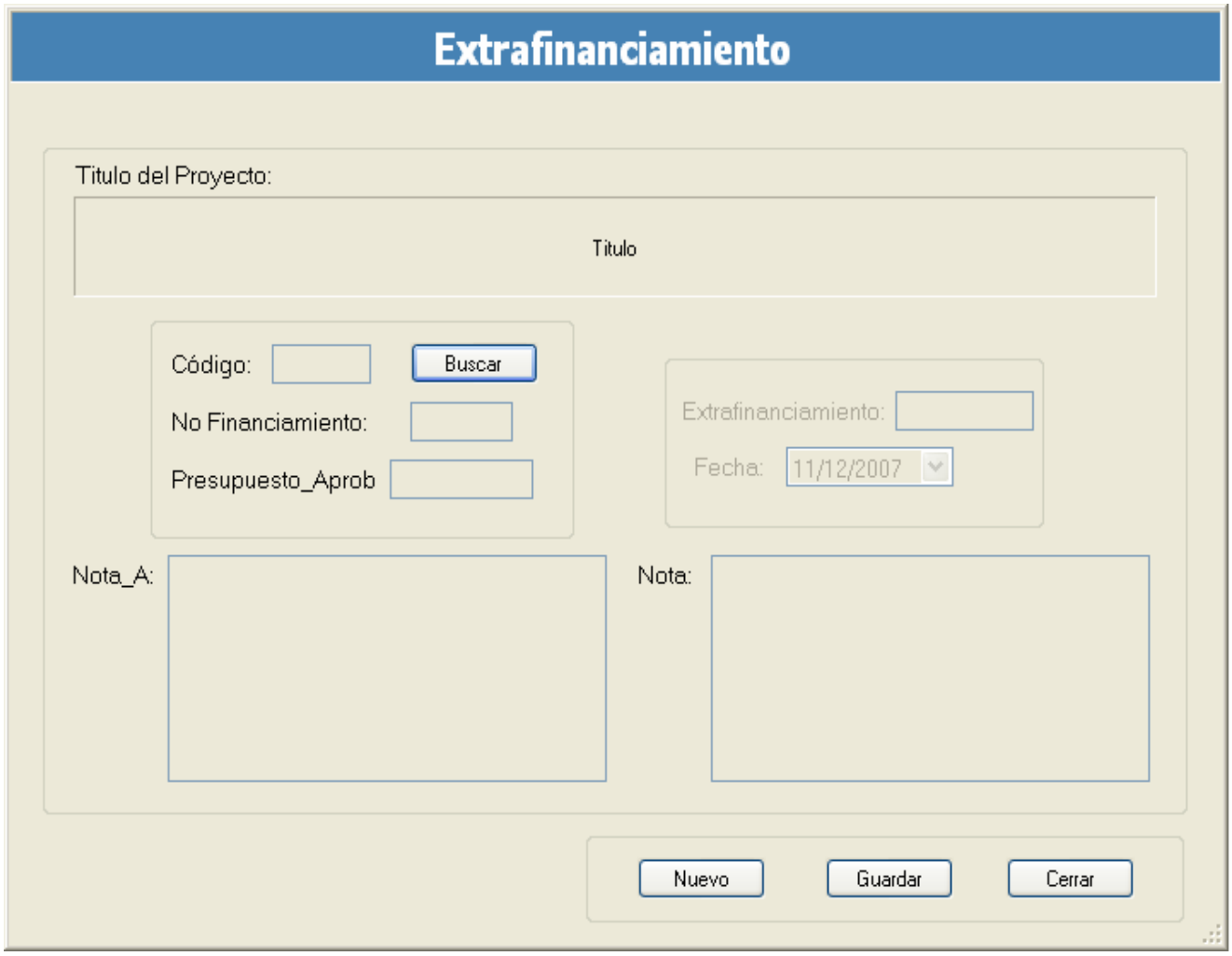


Formulario Agregar un nuevo Coordinador Facultativo

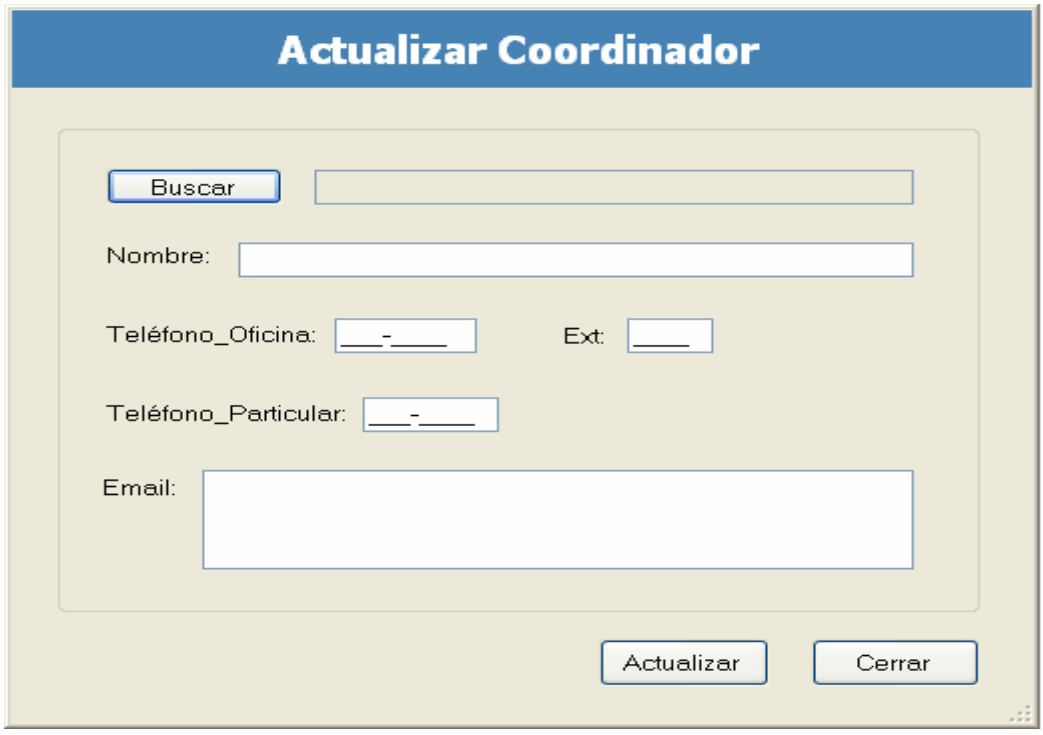


Formulario Agregar un nuevo Coordinador de Proyecto

Formulario Ingresar Clave



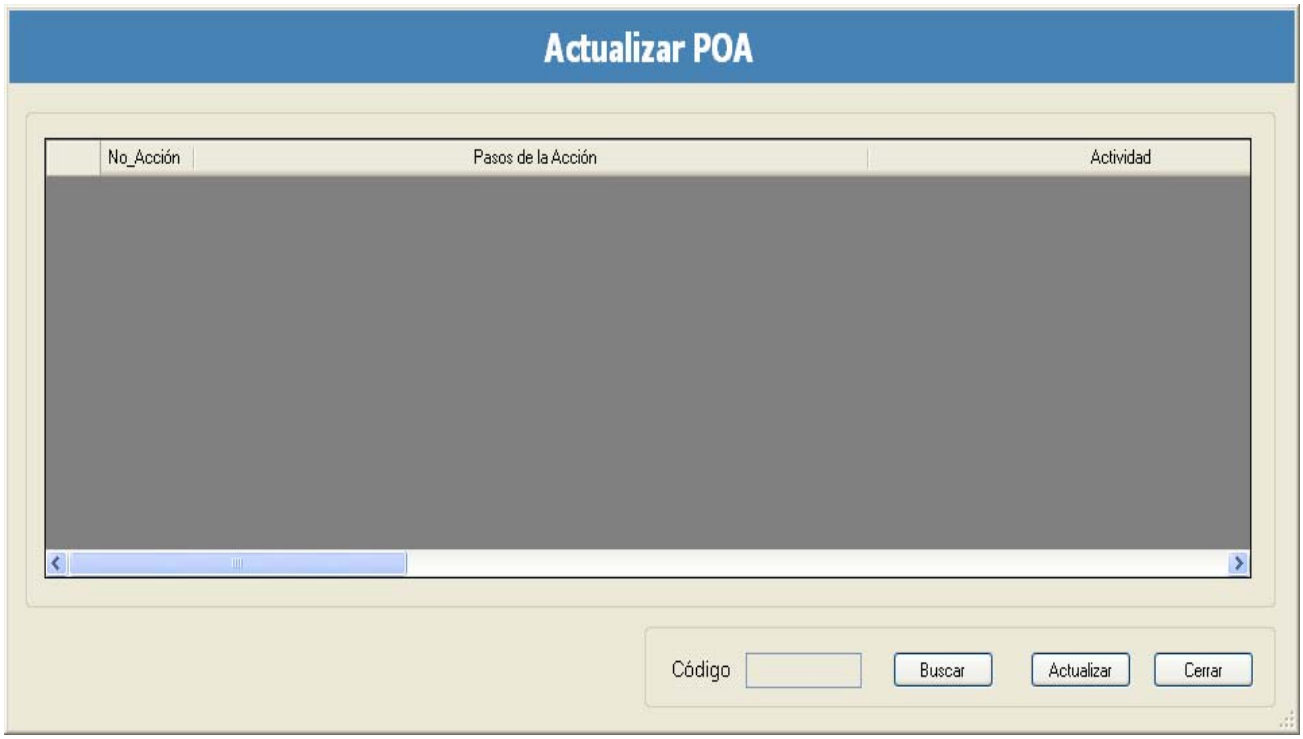
Formulario Agregar un Extrafinanciamiento



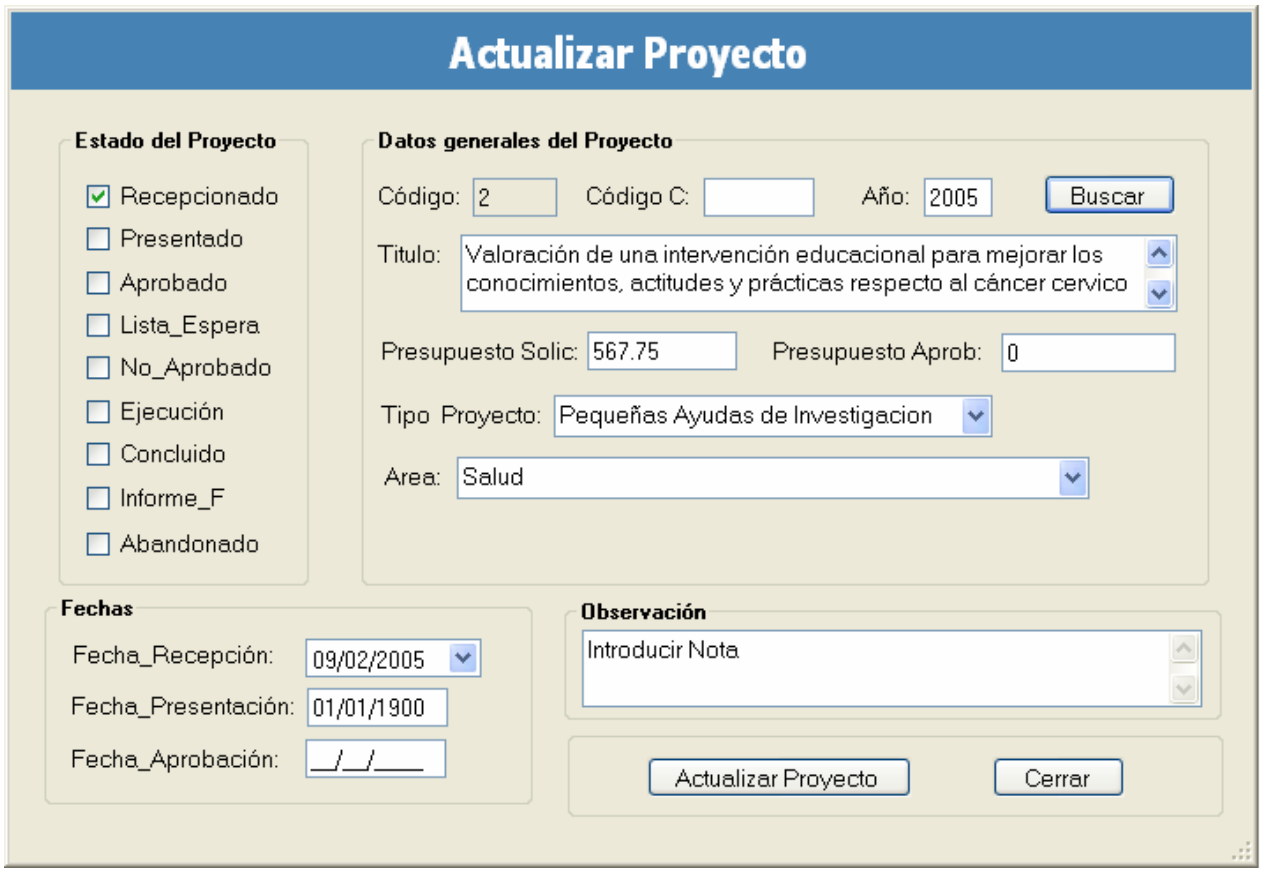
Formulario Actualizar Coordinador Facultativo

Formulario Actualizar Coordinador de Proyecto

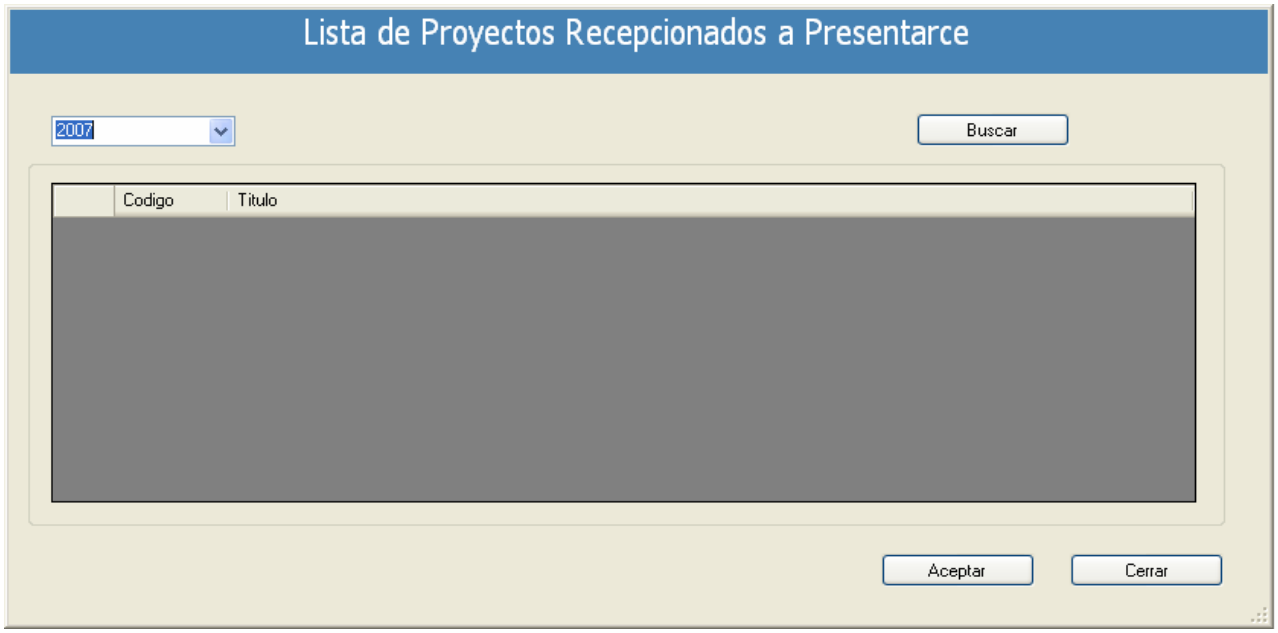
Formulario Actualizar Gasto



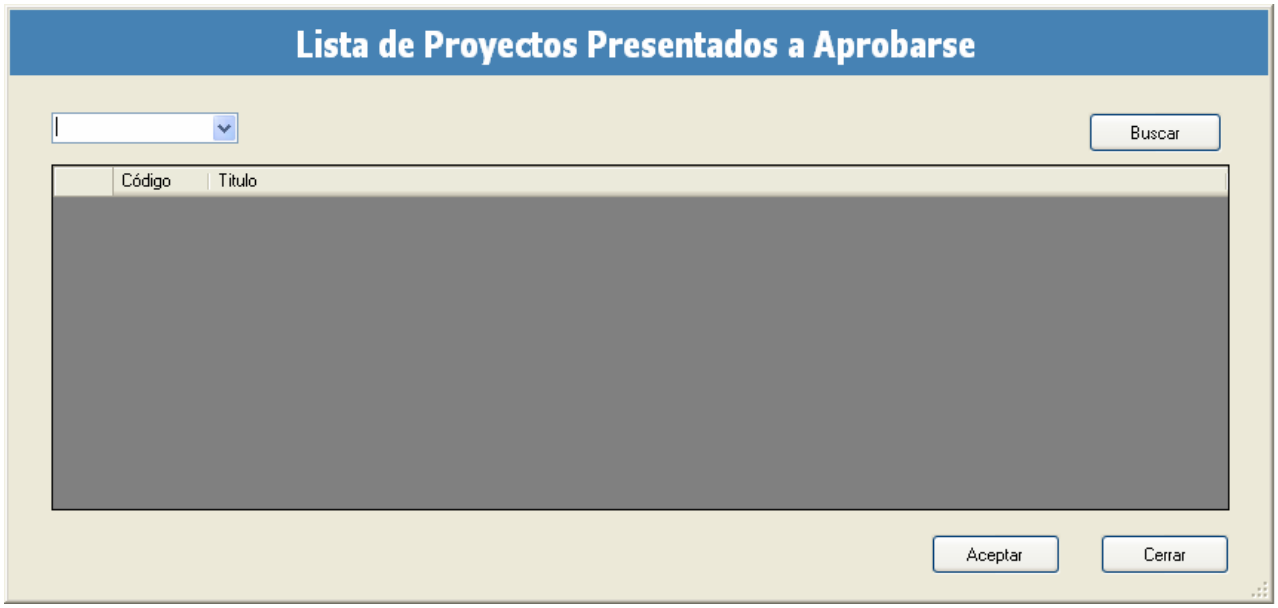
Formulario Actualizar POA



Formulario Actualizar Proyecto



Formulario Lista Recepcionados a Presentarse



Formulario Lista de Presentados a Aprobarse

CONCLUSIONES

Al concluir con nuestro trabajo-monográfico consideramos que hemos cumplido con los objetivos planteados, al proporcionar a la Vicerrectoría de Investigación y Postgrado (VIP) una herramienta de fácil uso la cual facilitará el seguimiento de las actividades realizadas por los subproyectos de investigación.

Finalmente se espera que esta herramienta sea el punto de partida para llevar un mejor control de los proyectos de investigación ejecutados en todas las facultades y dependencias de la UNAN-LEON.

Al llevar a cabo el desarrollo de nuestro Sistema y haber utilizado **Windows XP** (Sistema Operativo), **Visual C#** (Lenguaje de Programación), **SQL Server 2005** (Sistema Gestor de Base de Datos), concluimos que son herramientas potentes las cuales nos permitieron desarrollar nuestro sistema de forma rápida y eficiente.

Gracias a la utilización de estas herramientas (Visual C#, SQL Server 2005) se concluyó con el desarrollo de la aplicación para el Monitoreo de los Subproyectos de investigación aprobados por la VIP de la UNAN-LEÓN.

RECOMENDACIONES

Al concluir nuestro trabajo monográfico recomendamos:

Hacer una ampliación de esta herramienta de monitoreo para en un futuro utilizarla en todas las demás dependencias de la UNAN-LEON.

BIBLIOGRAFÍA

- Libro Enciclopedia de Visual C# Segunda Edición Francisco Javier Ceballos.
- http://www.elquintero.net/Manuales.aspx?Cat=2&SubCat=8&jscript=truehttp://es.tldp.org/Postgresql-es/web/navegable/tutorial/x56.html
- www.canalvisualbasic.net/temarios/manual_c_sharp.asp www.postgresql.org
- www.elguille.info
- www.microsoft.com/spain/sql/productinfo/features/top30features.mspx
- http://en.wikipedia.org/wiki/Microsoft_Visual_C_Sharp

ANEXOS

CODIFICACIÓN

```
//Clase CBaseDatos
class CBaseDatos
   {
      private SqlConnection con;

      public CBaseDatos()
      {
         try
         {
            con = new SqlConnection(@"integrated security=true;initial catalog = Proyectos;
data source=localhost\SQLEXPRESS");
            con.Open();
         }
         catch (Exception ex)
         {
            MessageBox.Show("Error de Conecion:  " + ex.Message);
         }
      }
// Constructor de la Clase CBaseDatos
      public CBaseDatos(string cadena)
      {
         try
         {
            con = new SqlConnection(cadena);
            con.Open();
         }
         catch (Exception ex)
         {
            MessageBox.Show("Error de Conecion:  " + ex.Message);
         }
      }
```

```
public void InsertarDatos(string consulta)
      {
         try
         {
            SqlCommand comando=new SqlCommand(consulta,con);
            SqlDataReader lector;


            lector = comando.ExecuteReader();
            lector.Close();


         }
         catch (Exception ex)
         {
            MessageBox.Show("Error al Insertar los Datos:  " + ex.Message);
         }

      }
//Función CargarDatosDataset
      public void CargarDatosDataset(DataSet dato, string consulta,string tabla)
      {
         SqlDataAdapter adaptador;
         try
         {
            adaptador = new SqlDataAdapter(consulta, con);
            adaptador.Fill(dato, tabla);
         }
         catch (Exception ex)
         {
            MessageBox.Show("Error cargar los Datos en el Dataset:  " + ex.Message);
         }
      }
//Función Obtener Datos
      public SqlDataReader ObtenerDatos(string consulta)
      {
         SqlCommand comando;
         SqlDataReader lector;
```

```
try
        {
            comando = new SqlCommand(consulta, con);
            return lector = comando.ExecuteReader();
        }
        catch (Exception ex)
        {
            MessageBox.Show("Error al Cargar los Datos:  " + ex.Message);
            return null;
        }

    }
//Función Ultima Posición
    public int UltimaPosicion(string consulta, string tabla)
    {
        int posicion;


        DataSet datos = new DataSet();
        CargarDatosDataset(datos, consulta, tabla);
        BindingSource proyectob = new BindingSource();
        proyectob.Add(datos);
        proyectob.DataSource = datos.Tables[tabla];

        if ((proyectob.Count) < 0)
        {
            posicion = 1;
            return posicion;
        }
        else
        {
            posicion = proyectob.Count;
            posicion += 1;
            return posicion;
        }
    }
//Función LlenarTabla
    public void LLenarTabla(DataTable tabla,string consulta)
    {
        SqlDataAdapter adaptador;
```

```
try
        {
            adaptador = new SqlDataAdapter(consulta, con);
            adaptador.Fill(tabla);
        }
        catch (Exception ex)
        {
            MessageBox.Show("Error cargar los Datos en la Tabla:  " + ex.Message);
        }
    }
//Función CargarAdactador
    public void CargarAdactador(string consulta, DataTable dato)
    {
        SqlDataAdapter adactador = new SqlDataAdapter();
        SqlCommand SqlSelect = new SqlCommand();

        adactador.SelectCommand = SqlSelect;
        SqlSelect.Connection = con;
        SqlSelect.CommandText = consulta;
        adactador.Fill(dato);
    }


//Función Cerrar Coneción
    public void CerrarConeccion()
    {
        con.Close();
    }
  }

////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
//Clase CClave
class CClave
  {
    private FileStream stream = null;
    private StreamReader reader = null;
    private StreamWriter writer = null;
//Función LeerDatos
    public string[] LeerDatos()
    {
        string file = "Archivo.txt";
        stream = new FileStream(file, FileMode.OpenOrCreate, FileAccess.ReadWrite);
```

```
            reader = new StreamReader(stream);
            string[] s = new string[2] ;
                s[0]=reader.ReadLine();
                reader.Peek();
                s[1] = reader.ReadLine();
            //Cerrar el Archivo
            reader.Close();
            stream.Close();
            stream = null;
           return s;
        }
//Función EscribirDatos
        public void EscribirDatos(string usuario,string clave)
        {
            // Borrar el Archivo existente.
            string file = "Archivo.txt";
            stream = new FileStream(file, FileMode.Create, FileAccess.Write);
            writer = new StreamWriter(stream);
                    writer.WriteLine(usuario);
                    writer.WriteLine(clave);
                    writer.Close();
                    writer = null;
                    stream.Close();
                    stream = null;
                    MessageBox.Show("La Contraseña fue Cambiada con     exito");
        }

        public string ObtenerUrl(string archivo)
        {
            string path = Path.GetFullPath(@"ArchivosHtml\\"+archivo);
            return path;
        }
}
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
//Nuevo Proyecto
public partial class DlgProyecto : Form
    {
        public int codi;
        string idcoordinadorP;
        public DlgProyecto()
        {
```

```
  InitializeComponent();
}

private void DlgProyecto_Load(object sender, EventArgs e)
{
    groupBox2.Enabled = false;
    groupBox3.Enabled = false;
    btGuardar.Enabled = false;
}

private void Buscar_Click(object sender, EventArgs e)
{
    DlgBuscarCoordinador bcor = new DlgBuscarCoordinador();
    bcor.ShowDialog();
    ctCoordinador.Text = bcor.nombreCoordinador;
    ctFacultad.Text = bcor.facultad;
    CoordinadorP.Text = bcor.NombreCoordinadorP;
    idcoordinadorP = bcor.idCoordinador;
}


private void btNuevo_Click(object sender, EventArgs e)
{
    string consulta = "Select *from Proyecto";
    CBaseDatos bases = new CBaseDatos();
    codi = bases.UltimaPosicion(consulta, "Proyecto");
    bases.CerrarConeccion();
    CoordinadorP.Text = " ";
    ctCodigo.Text = "";
    ctCodigo.Text = Convert.ToString(codi);
    ctCodigo.Focus();
    Area.Text = "< SELECCIONE DE LA LISTA >";
    cboTipo.Text = "< SELECCIONE DE LA LISTA >";
    ctPresupuestoS.Text = "";
    ctTitulo.Text = "";
    ctFacultad.Text = "";
    ctCoordinador.Text = "";
    ctAnyo.Text = "";
    ctAnyo.Focus();
    groupBox2.Enabled = true;
    groupBox3.Enabled = true;
    btGuardar.Enabled = true;
```

```
    }

    private void btGuardar_Click(object sender, EventArgs e)
    {
        CBaseDatos bases = new CBaseDatos();
        string consulta = "Insert into Proyecto Values('";
        string consulta3 = "Insert into Propuestas Values('";
        float presup = new float();
        if (ctAnyo.Text.Length != 0 && ctCoordinador.Text.Length != 0 &&
ctFacultad.Text.Length != 0 && Area.Text.Length != 0 && ctPresupuestoS.Text.Length != 0
&& ctTitulo.Text.Length != 0 && CoordinadorP.Text != "")
        {
            if (Area.Text != "< SELECCIONE DE LA LISTA >")
            {
                presup = (float)Convert.ToDouble(ctPresupuestoS.Text);
                consulta += codi + "','" + null + "','" + ctTitulo.Text + "','" + ctAnyo.Text + "','" +
Area.Text + "','" + false + "','" + false + "','" + Convert.ToString(presup) + "','" + null + "','" +
false + "','" + cboTipo.Text + "','" + idcoordinadorP + "')";
                bases.InsertarDatos(consulta);
                consulta3 += codi + "',1,0,0,0,0,'" + Fecha_R.Text + "','" + null + "','" + null + "','" +
"Introducir Nota  " + "', 0)";
                bases.InsertarDatos(consulta3);
                bases.CerrarConeccion();
                groupBox2.Enabled = false;
                groupBox3.Enabled = false;
                btGuardar.Enabled = false;
                MessageBox.Show("Registro Guardado");
            }
            else
                MessageBox.Show("Seleccione un Area");
        }
        else
        {
            MessageBox.Show("Introduzca los valores Correspondiente");
        }
    }

    private void btCerrar_Click(object sender, EventArgs e)
    {
        this.Close();
    }
}
```

//

//Formulario Lista de Actividades

```
public partial class DlgListaActividades : Form
   {
      bool verificar;
      public DlgListaActividades()
      {
         InitializeComponent();
      }

      public string fecha(int anyo, int mes, int dia)
      {
         string fechaActualizada = "";
         fechaActualizada = Convert.ToString(anyo) + "-" + Convert.ToString(mes) + "-" + dia;
         return fechaActualizada;
      }
      public void CargarDataGrid()
      {
         int fila = 0;
         ListaA.Rows.Clear();
         CBaseDatos bases = new CBaseDatos();
         SqlDataReader lector;
         DateTime fechaanterior = DateTime.Now.Subtract(TimeSpan.FromDays(5));
         DateTime fechaposterior = DateTime.Now.AddDays(5);
         DateTime fechaactual = DateTime.Now;

         bool Subejecucion = false;
         string consulta = "Select Proyecto.Codigo,Codigo_C,Titulo,POA.Numero_Accion,Revision,Actividad_Subejecucion from Proyecto INNER JOIN POA ON Proyecto.Codigo = POA.Codigo where(POA.fecha_Inicio BETWEEN '" + fecha(fechaanterior.Year, fechaanterior.Month, fechaanterior.Day) + "'AND '" + fecha(fechaposterior.Year, fechaposterior.Month, fechaposterior.Day) + "') and(Proyecto.Estado_Concluido=0)OR(POA.Actividad_Subejecucion=1)";
         bases.InsertarDatos("Update Poa set Actividad_Ejecucion=0,Actividad_Subejecucion=1 where(fecha_Fin <'" + fecha(fechaactual.Year, fechaactual.Month, fechaactual.Day) + "')and(Actividad_Ejecucion=1)and(Gasto_Generado=0)");

         lector = bases.ObtenerDatos(consulta);
```

```
   while (lector.Read())
          {
             string[] lista ={ Convert.ToString(lector[0]), Convert.ToString(lector[1]),
Convert.ToString(lector[3]), Convert.ToString(lector[2]) };
             ListaA.Rows.Add(lista);

             verificar = Convert.ToBoolean(lector[4]);
             Subejecucion = Convert.ToBoolean(lector[5]);
             if (verificar == true)
             {
                ListaA.Rows[fila].DefaultCellStyle.BackColor = Color.Khaki;
             }
             if (Subejecucion == true)
             {
                ListaA.Rows[fila].DefaultCellStyle.BackColor = Color.Red;
                ListaA.Rows[fila].DefaultCellStyle.ForeColor = Color.White;
             }
             fila++;
          }


          lector.Close();
             bases.InsertarDatos("Update Poa set Actividad_Ejecucion=1
where(fecha_Inicio="'+fecha(fechaactual.Year,fechaactual.Month,fechaactual.Day)+"')");

       }

       private void btCerrar_Click(object sender, EventArgs e)
       {
          this.Close();
       }

       private void DlgListaActividades_Load(object sender, EventArgs e)
       {
          string[] codigos;
          DateTime[] fechas;
          CBaseDatos datos = new CBaseDatos();
          DateTime fechaActual = DateTime.Now.Subtract(TimeSpan.FromDays(1));

          string cadena = "SELECT MAX(POA.fecha_Fin) AS fecha,POA.Codigo FROM
POA,Proyecto where(POA.Codigo = Proyecto.Codigo)and"+
```

```
"(Proyecto.Estado_ejecucion = 1) AND (Proyecto.Estado_Concluido = 0)GROUP BY POA.Codigo";
            int numero=0,contador=0;
            SqlDataReader lector = datos.ObtenerDatos(cadena);
            numero = lector.FieldCount;
            if (numero > 0)
            {
                codigos=new string[numero];
                fechas=new DateTime[numero];

                while (lector.Read())
                {
                    fechas[contador] = Convert.ToDateTime(lector[0]);
                    codigos[contador] = Convert.ToString(lector[1]);
                    contador++;
                }
                lector.Close();

                for (int i = 0; i < numero; i++)
                {
                    if (fechas[i].Day == fechaActual.Day && fechas[i].Month == fechaActual.Month && fechas[i].Year == fechaActual.Year)


                        datos.InsertarDatos("Update Proyecto set Estado_Concluido=1 where(Codigo='" + codigos[i]+ "')");
                }
            }

            CargarDataGrid();
        }

        private void verActividadToolStripMenuItem_Click(object sender, EventArgs e)
        {
            int numeroF = ListaA.RowCount;
            if (numeroF > 0)
            {
                int numero = ListaA.CurrentRow.Index;
                object valorCelda = ListaA.Rows[numero].Cells[0].Value;
```

```
 DlgInformacionActividades info = new DlgInformacionActividades(Convert.ToInt32(ListaA.Rows[numero].Cells[2].Value), Convert.ToInt32(ListaA.Rows[numero].Cells[0].Value));
            info.ShowDialog();
            CargarDataGrid();
        }
        else
            MessageBox.Show("No hay datos para seleccionar");
    }

    private void generarGastoToolStripMenuItem_Click(object sender, EventArgs e)
    {


        CBaseDatos datos = new CBaseDatos();
        SqlDataReader lector;
        bool verificar = false,verificargasto=false;
        int numeroF = ListaA.RowCount;
        if (numeroF > 0)
        {
            int indice = ListaA.CurrentRow.Index;
            lector = datos.ObtenerDatos("Select Monto,Gasto_Generado from POA where(Codigo='" + Convert.ToString(ListaA.Rows[indice].Cells[0].Value) + "')and(Numero_Accion='" + Convert.ToString(ListaA.Rows[indice].Cells[2].Value) + "')");
            while (lector.Read())
            {
                if (Convert.ToSingle(lector[0]) > 0)


                    verificar = true;
                if (Convert.ToBoolean(lector[1]) == true)
                    verificargasto = true;
            }
            lector.Close();
            if (verificargasto != true)
            {
                if (verificar)
                {
                    DlgGastos gasto = new DlgGastos(Convert.ToString(ListaA.Rows[indice].Cells[0].Value), Convert.ToString(ListaA.Rows[indice].Cells[3].Value), Convert.ToString(ListaA.Rows[indice].Cells[2].Value));
```

```
                gasto.ShowDialog();
                CargarDataGrid();
            }
            else
            {
                datos.InsertarDatos("Update Poa set Gasto_Generado=1where(Codigo='" +
Convert.ToString(ListaA.Rows[indice].Cells[0].Value) + "')and(Numero_Accion='" +
Convert.ToString(ListaA.Rows[indice].Cells[2].Value) + "')");
                MessageBox.Show("No se puede efectuar un gasto por que esta accion no
produce gasto");
            }
        }
        else
            MessageBox.Show("No se Puede Realizar un gasto por que ya fue efectuado o
la accion no genera gasto");
    }
    else
        MessageBox.Show("No hay datos para seleccionar");
  }
}

////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

public partial class DlgInformacionActividades : Form
{
    private int pos, codigo;
    public DlgInformacionActividades()
    {
        InitializeComponent();


    }
    public DlgInformacionActividades(int posicion, int cod)
    {
        InitializeComponent();
        pos = posicion;
        codigo = cod;
    }
    private void DlgInformacionActividades_Load(object sender, EventArgs e)
    {
        CBaseDatos bases = new CBaseDatos();
        SqlDataReader lector1, lector2, lector3;
```

```
        string idcordinador = "", idfacultad = "";

        lector1 = bases.ObtenerDatos("Select Area,Id_Coordinador From Proyecto
where(Codigo='" + codigo + "')");
        while (lector1.Read())
        {
            areaTextBox.Text = Convert.ToString(lector1[0]);
            idcordinador = Convert.ToString(lector1[1]);
        }
        lector1.Close();

        lector1 = bases.ObtenerDatos("Select *From Coordinador_Proyecto
where(Id_Coordinador='" + idcordinador + "')");
        while (lector1.Read())
        {
            ctcoP.Text = Convert.ToString(lector1[1]);
            CtTpP.Text = Convert.ToString(lector1[2]);
            CtTOP.Text = Convert.ToString(lector1[3]);
            ctExtCP.Text = Convert.ToString(lector1[4]);
            CtEmail.Text = Convert.ToString(lector1[5]);
            idfacultad = Convert.ToString(lector1[6]);
        }
        lector1.Close();

        lector2 = bases.ObtenerDatos("Select *from Coordinador_Facultativo
where(Id_Facultad='" + idfacultad + "')");
        while (lector2.Read())
        {
            facultadTextBox.Text = Convert.ToString(lector2[1]);
            nombre_CoordinadorTextBox.Text = Convert.ToString(lector2[2]);
            telefono_ParticularTextBox.Text = Convert.ToString(lector2[3]);


            telefono_oficinaTextBox.Text = Convert.ToString(lector2[4]);
            emailTextBox.Text = Convert.ToString(lector2[5]);
            ctExtCF.Text = Convert.ToString(lector2[6]);
        }
        lector2.Close();
```

```
lector3 = bases.ObtenerDatos("Select Pasos_Accion,Actividad,Observaciones,fecha_Inicio,fecha_Fin,mes_Ejecucion,monto,Responsable_Pri,Responsable_Otr From POA where(Codigo='" + codigo + "') AND (Numero_Accion='" + pos + "')");
            while (lector3.Read())
            {
                pasos_AccionTextBox.Text = Convert.ToString(lector3[0]);
                actividadTextBox.Text = Convert.ToString(lector3[1]);
                observacionesTextBox.Text = Convert.ToString(lector3[2]);
                fechainicio.Text = Convert.ToString(lector3[3]);
                fecha_Fin.Text = Convert.ToString(lector3[4]);
                mes_EjecucionTextBox.Text = Convert.ToString(lector3[5]);
                montoTextBox.Text = Convert.ToString(lector3[6]);
                responsable_PriTextBox.Text = Convert.ToString(lector3[7]);
                responsable_OtrTextBox.Text = Convert.ToString(lector3[8]);
            }
            lector3.Close();
            bases.CerrarConeccion();
            btCerrar.Focus();
        }

        private void btCerrar_Click(object sender, EventArgs e)
        {
            CBaseDatos Base = new CBaseDatos();
            Base.InsertarDatos("UPDATE POA SET Revision=1  Where(Codigo='" + codigo + "')AND (Numero_Accion='" + pos + "')");
            Base.CerrarConeccion();
            this.Close();
        }
    }

////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

public partial class DlgBuscarTodosProyectos : Form
    {


        DataTable Mitabla;
        public string codigoProyecto="";
        public bool verificar = true;
```

```
public DlgBuscarTodosProyectos()
{
    InitializeComponent();

}

public void Busqueda(string columna, string dato)
{
    DataRow[] filas;
    filas = Mitabla.Select(columna + "='" + dato + "'");

    if (filas.Length > 0)
    {
        foreach (DataGridViewRow fila in this.datosProyecto.Rows)
        {
            if (fila.Cells[columna].Value == null || fila.Cells == null)
            {
                break;
            }

            string s = fila.Cells[columna].Value.ToString();
            if (String.IsNullOrEmpty(s) == false)
            {
                if (s == dato)
                {
                    // Seleccionamos la fila
                    datosProyecto.Rows[fila.Index].Selected = true;
                    // nos aseguramos de que sea visible
                    datosProyecto.FirstDisplayedScrollingRowIndex = fila.Index;
                    codigoProyecto                                                    =
Convert.ToString(datosProyecto.Rows[fila.Index].Cells[0].Value);
                    break;
                }
            }
        }
    }
    else
        MessageBox.Show("El Parametro de busqueda es Invalido");


}
```

```
private void Buscar_Click(object sender, EventArgs e)
{
    if (CtCodigo.Text != "")
        Busqueda("Codigo", CtCodigo.Text);
    else if (ctCodigo_C.Text != "")
        Busqueda("Codigo_C", ctCodigo_C.Text);
    else if (ctTitulo.Text != "")
        Busqueda("Titulo", ctTitulo.Text);
    else
        MessageBox.Show("Tiene que seleccionar una de las opciones de busqueda");
}

private void CtCodigo_KeyPress(object sender, KeyPressEventArgs e)
{
     if (e.KeyChar == 13)
    {
        if (CtCodigo.Text != "")
            Busqueda("Codigo", CtCodigo.Text);
        else if (ctCodigo_C.Text != "")
            Busqueda("Codigo_C", ctCodigo_C.Text);
        else if (ctTitulo.Text != "")
            Busqueda("Titulo", ctTitulo.Text);
        else
            MessageBox.Show("Tiene que seleccionar una de las opciones de busqueda");
    }
}

private void btCerrar_Click(object sender, EventArgs e)
{
    this.Close();
}

private void btAceptar_Click(object sender, EventArgs e)
{
    if (codigoProyecto == "")
    {
        int numero = datosProyecto.CurrentRow.Index;
        codigoProyecto =Convert.ToString(datosProyecto.Rows[numero].Cells[0].Value);
    }
    this.Close();
```

```
}

private void Codigo_CheckedChanged(object sender, EventArgs e)
{
    if (Codigo.Checked)
    {
        CtCodigo.Enabled = true;
        ctCodigo_C.Enabled = false;
        ctTitulo.Enabled = false;
        ctTitulo.Text = "";
        ctCodigo_C.Text = "";
        CtCodigo.Focus();
    }
    else if (Codigo_C.Checked)
    {
        ctCodigo_C.Enabled = true;
        CtCodigo.Enabled = false;
        ctTitulo.Enabled = false;
        CtCodigo.Text = "";
        ctTitulo.Text = "";
        ctCodigo_C.Focus();
    }
    else if (Titulo.Checked)
    {
        CtCodigo.Enabled = false;
        ctTitulo.Enabled = true;
        ctCodigo_C.Enabled = false;
        CtCodigo.Text = "";
        ctCodigo_C.Text = "";
        ctTitulo.Focus();
    }
}

private void DlgBuscarTodosProyectos_Load(object sender, EventArgs e)
{
    ctTitulo.Enabled = false;
    ctCodigo_C.Enabled = false;
    CtCodigo.Focus();
    CBaseDatos bases = new CBaseDatos();
    Mitabla = new DataTable();
    bases.LLenarTabla(Mitabla, "Select Codigo,Codigo_C,Titulo From Proyecto");
    bases.CerrarConeccion();
```

```
            datosProyecto.DataSource = Mitabla;
            datosProyecto.Columns[0].Width = 60;
            datosProyecto.Columns[1].Width = 65;
        }
    }
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

public partial class DlgReporteArea : Form
    {
        public DlgReporteArea()
        {
            InitializeComponent();
        }

        private void DlgReporteArea_Load(object sender, EventArgs e)
        {
            CBaseDatos datos = new CBaseDatos();
            string consulta = " SELECT COUNT(Codigo) AS numero, SUM(Presupuesto_Aprobado) AS Presupuesto_Aprobado, Area, Anyo FROM VistaPropuestasProyectos WHERE(Aprobado = 1) GROUP BY Area, Anyo";
            datos.CargarAdactador(consulta,dsArea.Area);

            datos.CargarAdactador("Select *from VistaPropuestasProyectos where(Aprobado=1)",dsProyectosPropuestas.Proyectos);


this.reportViewer1.SetDisplayMode(Microsoft.Reporting.WinForms.DisplayMode.PrintLayout);
            this.reportViewer1.ZoomMode = Microsoft.Reporting.WinForms.ZoomMode.Percent;
            this.reportViewer1.RefreshReport();
            this.reportViewer2.RefreshReport();
        }


    }
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

public partial class DlgactualizaPoa : Form
    {
        public DlgactualizaPoa()
        {
            InitializeComponent();
        }
```

```
        public string fecha(int anyo, int mes, int dia)
        {
            string fechaActualizada = "";
            fechaActualizada = Convert.ToString(anyo) + "-" + Convert.ToString(mes) + "-" + dia;
            return fechaActualizada;
        }

        private void btbuscar_Click(object sender, EventArgs e)
        {
            DlgBuscarproyecto pai = new DlgBuscarproyecto();
            pai.ShowDialog();
            ctColdigo.Text = pai.Codigoproyecto;
            CBaseDatos datos = new CBaseDatos();
            SqlDataReader lector;

            DatosPoa.Rows.Clear();
            lector = datos.ObtenerDatos("Select *from Poa where(Codigo='" + ctColdigo.Text +
"')");

            while (lector.Read())
            {
                DateTime fecahinico, fechafin;
                fecahinico = DateTime.Parse(Convert.ToString(lector[6]));
                fechafin = DateTime.Parse(Convert.ToString(lector[7]));
                string[] cadena ={ Convert.ToString(lector[1]), Convert.ToString(lector[2]),
Convert.ToString(lector[5]), Convert.ToString(lector[3]), Convert.ToString(lector[4]),

fecahinico.ToShortDateString(),fechafin.ToShortDateString(),Convert.ToString(lector[8]),Conv
ert.ToString(lector[9]),Convert.ToString(lector[10]),Convert.ToString(lector[11]),
                Convert.ToString(lector[12]),Convert.ToString(lector[13])};
                DatosPoa.Rows.Add(cadena);
            }
            lector.Close();
            datos.CerrarConeccion();
        }

        private void btGuardar_Click(object sender, EventArgs e)
        {
             CBaseDatos datos = new CBaseDatos();
            DateTime fecahinicio, fechafin;
            string consulta = "";
            int numfilas = DatosPoa.Rows.Count;
```

```
        if (numfilas == 0)
            MessageBox.Show("No hay Datos presione el boton Buscar");
        else
        {
            for (int i = 0; i < numfilas; i++)
            {
                fecahinicio = DateTime.Parse(Convert.ToString(DatosPoa.Rows[i].Cells[5].Value));
                fechafin = DateTime.Parse(Convert.ToString(DatosPoa.Rows[i].Cells[6].Value));
                consulta = "UPDATE POA SET Pasos_Accion='" + Convert.ToString(DatosPoa.Rows[i].Cells[1].Value) + "',Responsable_Pri ='" + Convert.ToString(DatosPoa.Rows[i].Cells[3].Value) +
                "',Responsable_Otr ='" + Convert.ToString(DatosPoa.Rows[i].Cells[4].Value) + "', Actividad='" + Convert.ToString(DatosPoa.Rows[i].Cells[2].Value) + "',fecha_Inicio='" + fecha(fecahinicio.Year, fecahinicio.Month, fecahinicio.Day) +
                "',fecha_Fin='" + fecha(fechafin.Year, fechafin.Month, fechafin.Day) + "',Tiempo='" + Convert.ToString(DatosPoa.Rows[i].Cells[7].Value) + "',Mecanismo_Retroa='" + Convert.ToString(DatosPoa.Rows[i].Cells[8].Value) +
                "',Monto='" + Convert.ToString(DatosPoa.Rows[i].Cells[9].Value) + "',Rubro='" + Convert.ToString(DatosPoa.Rows[i].Cells[10].Value) + "',mes_Ejecucion='" + Convert.ToString(DatosPoa.Rows[i].Cells[11].Value) +
                "',Observaciones='" + Convert.ToString(DatosPoa.Rows[i].Cells[12].Value) + "' WHERE(Codigo='" + ctColdigo.Text + "')and(Numero_Accion='" + Convert.ToString(DatosPoa.Rows[i].Cells[0].Value) + "')";
                datos.InsertarDatos(consulta);
            }
            datos.CerrarConeccion();
            MessageBox.Show("Los datos fueron Actualizados");
        }
    }

    private void btCerrar_Click(object sender, EventArgs e)
    {
        this.Close();
    }


  }
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

public partial class DlgExtrafinaciamiento : Form
  {
```

```
        public DlgExtrafinaciamiento()
        {
            InitializeComponent();
        }

        public string fecha(int anyo, int mes, int dia)
        {
            string fechaActualizada = "";
            fechaActualizada = Convert.ToString(anyo) + "-" + Convert.ToString(mes) + "-" + dia;
            return fechaActualizada;
        }

        public void ActualizarEstadoCuenta(int numero_gasto, int codigo)
        {
            CBaseDatos Bases = new CBaseDatos();
            SqlDataReader lector1, lector2;

            float valoractual = 0;
            float presupuestoA = 0;
            float saldo = 0, monto = 0;

            if (numero_gasto == 1)
            {
                lector1 = Bases.ObtenerDatos("Select Presupuesto_Aprobado from Proyecto
where(Codigo='" + codigo + "')");
                while (lector1.Read())
                {
                    presupuestoA = Convert.ToSingle(lector1[0]);
                }
                lector1.Close();
                lector2 = Bases.ObtenerDatos("Select Monto_Gasto from Estado_Cuenta
where(Codigo='" + codigo + "') AND (Numero_Gasto='" + numero_gasto + "')");
                while (lector2.Read())
                {
                    monto = Convert.ToSingle(lector2[0]);
                }
                lector2.Close();
                saldo = presupuestoA - monto;
                Bases.InsertarDatos("Update Estado_Cuenta set Saldo_Actual ='" + saldo +
"'where(Codigo='" + codigo + "') AND (Numero_Gasto='" + numero_gasto + "')");

}
```

```
        Else
        {
            int numeroAnterior = numero_gasto - 1;
            lector2 = Bases.ObtenerDatos("Select Saldo_Actual from Estado_Cuenta where(Codigo='" + codigo + "') AND (Numero_Gasto='" + numeroAnterior + "')");
            while (lector2.Read())
            {
                valoractual = Convert.ToSingle(lector2[0]);
            }
            lector2.Close();

            lector1 = Bases.ObtenerDatos("Select Monto_Gasto from Estado_Cuenta where(Codigo='" + codigo + "') AND (Numero_Gasto='" + numero_gasto + "')");
            while (lector1.Read())
            {
                monto = Convert.ToSingle(lector1[0]);
            }
            lector1.Close();

            saldo = valoractual - monto;
            Bases.InsertarDatos("Update Estado_Cuenta set Saldo_Actual='" + saldo + "'where(Codigo='" + codigo + "') AND (Numero_Gasto='" + numero_gasto + "')");
        }
        Bases.CerrarConeccion();

    }

    private void btCerrar_Click(object sender, EventArgs e)
    {
        this.Close();
    }

    private void btBuscar_Click(object sender, EventArgs e)
    {
        groupBox3.Enabled = true;
        ctnota.Enabled = true;
        CBaseDatos bases = new CBaseDatos();
        DlgBuscarproyecto buscar = new DlgBuscarproyecto();
        buscar.ShowDialog();
```

```
        if (buscar.verificar == true)
      {
          if (buscar.Codigoproyecto.Length != 0)


          {
              ctcodigo.Text = buscar.Codigoproyecto;
              lbtitulo.Text = buscar.titulo;
              lbtitulo.Refresh();

              int cuenta = bases.UltimaPosicion("Select No_Financiamiento From Extrafinaciamiento where(Codigo='" + buscar.Codigoproyecto + "')", "Extrafinaciamiento");

              SqlDataReader lector = bases.ObtenerDatos("Select Presupuesto_Aprobado From Proyecto where(Codigo='" + buscar.Codigoproyecto + "')");

              while (lector.Read())
              {
                  presupuesto.Text = Convert.ToString(lector[0]);
              }
              lector.Close();
              if (cuenta == 1)
              {
                  ctno_Financiamiento.Text = "1";
                  Nota_A.Text = "No hay Notas";
              }
              else
              {
                  ctno_Financiamiento.Text = Convert.ToString(cuenta);
                  cuenta--;
                  lector = bases.ObtenerDatos("Select Nota From Extrafinaciamiento where(Codigo='" + buscar.Codigoproyecto + "') and(No_Financiamiento='" + cuenta + "')");
                  while (lector.Read())
                  {
                      Nota_A.Text = Convert.ToString(lector[0]);
                  }
              }
          }
          else
          {
```

```
                MessageBox.Show("No existen Proyectos aun ");
            }
        }
        bases.CerrarConeccion();

    }

    private void btNuevo_Click(object sender, EventArgs e)
    {
        groupBox1.Enabled = true;
        ctnota.Text = "";
        ctno_Financiamiento.Text = "";
        ctextrafinaciamiento.Text = "";
        ctcodigo.Text = "";
        btBuscar.PerformClick();
    }

    private void btGuardar_Click(object sender, EventArgs e)
    {
        DateTime ff = DateTime.Parse(fecha_financiamiento.Text);
        CBaseDatos bases = new CBaseDatos();
        SqlDataReader lector1;

        if (ctextrafinaciamiento.Text.Length > 0)
        {
            string consulta = "Insert Into Extrafinaciamiento Values('" + ctcodigo.Text + "','" +
ctno_Financiamiento.Text + "',' " + fecha(ff.Year, ff.Month, ff.Day) + "','" +
ctextrafinaciamiento.Text + "','" + ctnota.Text + "')";
            float valor, nuevovalor, valoractual = 0;
            int cuenta = 0;

            bases.InsertarDatos(consulta);
            lector1 = bases.ObtenerDatos("Select Presupuesto_Aprobado from Proyecto
where(Codigo='" + ctcodigo.Text + "')");
            while (lector1.Read())
            {
                valoractual = Convert.ToSingle(lector1[0]);
            }
            valor = Convert.ToSingle(ctextrafinaciamiento.Text);
            nuevovalor = valoractual + valor;
            lector1.Close();
```

```
bases.InsertarDatos("Update Proyecto set Presupuesto_Aprobado ='" + nuevovalor + "' where(Codigo='" + ctcodigo.Text + "')");
            cuenta = bases.UltimaPosicion("Select Numero_Gasto from Estado_Cuenta where(Codigo='" + ctcodigo.Text + "')", "Estado_Cuenta");
            bases.CerrarConeccion();
            groupBox1.Enabled = false;

            for (int i = 1; i < cuenta; i++)
            {
                ActualizarEstadoCuenta(i, Convert.ToInt32(ctcodigo.Text));
            }
        }
        else
            MessageBox.Show("Ingrese el monto del Financiamiento");
    }

}
```

Reportes

Universidad Nacional Autonoma de Nicaragua-Leon

SISTEMA DE MONITOREO DE SUBPROYECTOS DE INVESTIGACION (SiMSI)

UNAN-Leon
Edificio Central
Ap. 68, Leon, Nicaragua
Tel. / Fax:
(505) 311-5013 ext 1076
e-mail: Vip@unanleon.edu.ni

Vicerrectoría de Investigación y Postgrado

Año	Resepcionados	Precentados	Pendientes de Presentacion	Aprobados
2004	1	0	1	0
2005	2	0	2	0
2006	25	0	25	0
2007	33	0	33	0

Universidad
Nacional
Autonoma de
Nicaragua-León

Vicerrectoría de Investigación y Postgrado

SISTEMA DE MONITOREO
DE SUBPROYECTOS
DE INVESTIGACION
(SiMSI)

UNAN-Leon
Edificio Central
Ap. 68, Leon, Nicaragua
Tel. / Fax:
(505) 311-5013 ext 1076
e-mail: Vip@unanleon.edu.ni

SUB PROYECTOS DE PEQUEÑAS AYUDAS DE INVESTIGACION RECEPCIONADOS AL 12/11/2007

Resepcionados por Año		Titulo del Subproyecto	Presupuesto Solicitado	Facultad
2004	1	Seroprevalencia de brucelosis en caninos, porcinos, equinos y humanos en la población suburbana de la ciudad de León	U$ 3414	VETERINARIA / UNAN – León
2005	2	Valoración de una intervención educacional para mejorar los conocimientos, actitudes y prácticas respecto al cáncer cervico uterino en la comunidad de los Lechecuagos	U$ 567.75	Facultad de Ciencias Médicas
	3	Investigación fitoquimica de la uncaria tomentosa Nicaragüense	U$ 1400	Facultad de Ciencias Químicas
2006	4	Diseño de un plan de capacitación en mercadeo para las pyme dedicadas a la artesanía de barro en el municipio de la paz centro en el 2006	U$ 453.75	Facultad de Ciencias Económicas y Empresariales (incluye TURISMO)
	5	Diagnostico de las poblaciones de P. TECUNUMANI de la zona norte de Nicaragua, afectadas por d. frontalis y evaluación de su diversidad genética empleando marcadores moleculares	U$ 3296	Facultad de Ciencias
	6	Estudio epidemiológico de la leishmaniasis tegumentar americana en las comarcas los puyules y el naranjo, municipio de waslala, dpto. de Matagalpa en el periodo mayo a agosto del año 2006	U$ 312.53	Facultad de Ciencias Médicas
	7	Obtención de microcapsulas de acetaminofen	U$ 2992.41	Facultad de Ciencias Químicas
	8	Tipificación fenotipica de staphylococcus aureus causantes de mastitis subclínica bovina	U$ 3600	VETERINARIA / UNAN – León