UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

FACULTAD 6

TRABAJO DE DIPLOMA PARA OPTAR POR EL TÍTULO DE

INGENIERO EN CIENCIAS INFORMÁTICAS

Título: Desarrollo de un sistema para el estudio de Factibilidad Económica en los proyectos del Centro GEYSED.

Autores:

Tahimí Font Fuentes

Leydis Torres Albo

Tutor:

Ing. Solangel Rodríguez Vázquez

La Habana, junio de 2013

"Año 55 de la Revolución"

DECLARACIÓN DE AUTORÍA

Declaramos que _____ y _____ somos los únicos autores de este trabajo y autorizamos a la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

Para que así conste firmamos la presente a los ____ días del mes de junio del 2013.

_____ _____

Firma del Autor
Tahimí Font Fuentes

Firma del Autor
Leydis Torres Albo

_____
Firma del Tutor
Ing. Solangel Rodríguez Vázquez

DATOS DE CONTACTOS

Tutora: Ing. Solangel Rodríguez Vázquez.

Institución: Universidad de las Ciencias Informáticas.

Dirección de la institución: Carretera a San Antonio de los Baños, Km. 2 ½, Reparto: Torrens.

Municipio: La Lisa.

Provincia: La Habana.

Correo electrónico: svazquez@uci.cu

Cargo del trabajador: Líder del Proyecto Calidad.

Título de la especialidad de graduado: Ingeniería en Ciencias Informáticas.

Año de graduación: 2008.

Institución donde se graduó: Universidad de las Ciencias Informáticas.

AGRADECIMIENTOS

Tahimí

Agradezco a Dios, por darme la salud y fuerzas que necesité en todo momento para llegar a este punto. Quiero agradecerle, a la persona más especial de mi vida, la única persona que sin dudar ni vacilar un segundo puedo decirle te amo, porque si existe algo verdadero: es su amor. Una persona que me dijo ¡Levántate! cuando estaba en el piso dándome todo el ánimo y apoyo del mundo; pero que también me dijo ¡Vas bien! motivándome a seguir adelante. Quiero agradecerle a esa persona porque sé que más de una noche no durmió pidiéndole a Dios por mis exámenes o porque había discutido conmigo y había una pequeña heridita sin sanar. A esa persona, va además de mi eterno agradecimiento todo el amor que mi ser pueda engendrar. Estoy muy orgullosa de ti mamita, algún día quiero ser como tú. Te amo.

A mi papá, por ser uno de esos motivos por lo que hoy me hago ingeniera, porque sé que me quiere más que nada en este mundo, por ser su eterna princesa y porque sé que este es también tu sueño. Te quiero.

A Albert, por ser además de una persona maravillosa mi segundo papá, por ayudarme en todo lo que pude necesitar los 5 años de mi carrera, pero sobre todas las cosas por cuidar a mi mami. Te quiero.

A mis tías. Tía Mary por su preocupación, su cariño, por estar al pendiente de mí. Tía Sandra por ser su hija hembra, por estar conmigo cuando el calor de mis seres queridos estaba ausente, gracias por darme la oportunidad de ser tu hija hembra. Las quiero.

A mi abuelo Pepe por confiar en mí y a mi abuela Mercedita que hoy no está conmigo físicamente, por enseñarme a creer y a tener fe.

A mi mejor amiga Arianna, por ser mi confidente, por aconsejarme, por convertirse en la hermana que nunca tuve. Gracias por estar en mi vida durante estos ocho años. Te quiero mucho.

A Leydis por ser la mejor compañera de tesis del mundo. A Yunet y Breissy por formar parte de esta inmensa familia que hemos formado.

A Aida, Grettel, Sandy, Emir y Renier por demostrarme su amistad cuando la necesites. A todas las personas que conocí durante estos 5 años, en fin a todas esas personas que hicieron mi estancia en la universidad un poco más agradable. A todos gracias

Leydis

Agradezco a Dios, darme la salud y fuerzas que necesité en todo momento para llegar al día de hoy.

A mi abuela que ha sido mi guía y mi consejera en todo el transcurso de mi vida y mi carrera, a mi papá por ser el hombre de mi vida que siempre ha estado a mi lado en todo momento. A Hainamit por estar presente cuando lo he necesitado, por darme tus puntos de vista y aceptar mis elecciones cuando los demás no han estado de acuerdo. A mis hermanas Yeney, Yuly, Nathali porque a pesar de la lejanía que nos separa siempre han estado a mi lado en cada paso que he dado en mi vida, a mis hermanos Juan Carlo y el Chino porque siempre han estado a mi lado apoyándome en cada momento de mi vida. A mis sobrinos porque a pesar de que sean muy pequeñines me alegran la vida con su existencia. A mis tíos Lola, Julio y Luiso por mostrar tanta preocupación hacia a mí.

A las tres fantásticas, mis hermanitas de la UCI, Breissy porque a pesar de ser tan matraquillosa con todo y celosa siempre ha estado a mi lado aconsejándome en los momentos que he necesitado y sé que me quiere como otra hermana más, pues tiene un corazón que vale oro, porque me lo demuestra en cada momento. Yunet porque siempre está presente en cada momento que uno la necesita, pues nunca me ha faltado un consejo cuando lo he necesitado, siempre la he llevado presente porque dice las cosas tal y como son y sin andar sin rodeo, gracias a ella he aprendido a ser un poco madura. A mi compañera de tesis Tahimi porque a pesar de que dicen que es media loca por su forma de ser, se puede contar con ella para lo que sea sin recibir nada a cambio, siempre está a mi lado en los momentos que la he necesitado. Gracias Breissy, Yunet y Tahimi por estar todas presentes en estos cinco años a pesar de que siempre nos estemos fajando pero sabemos que todas nos queremos y que siempre estaremos juntas a pesar de la distancia que nos separará siempre estaremos juntas en nuestros recuerdos. Las quiero a todas.

A mi amiga Jessica porque desde que nos conocimos siempre me ha demostrado que con ella se puede contar para lo que sea, porque es la amiga que todo ser humano quisiera tener.

A todas las personas que he conocido desde el primer día que puse los pies en esta universidad, y las que en el transcurso de la carrera me demostraron su amistad, de cada uno de ustedes he aprendido que la amistad está por encima de todo. Los quiero a todos y siempre los llevare presente en cada momento de mi vida.

DEDICATORIA

Tahimí

A la persona más tierna y luchadora que en la vida he conocido. Quien ha sido mi luz y mi guía durante mi infancia, adolescencia y mi juventud. A la única persona que amo en este mundo.

A mi madre, la mejor de mis guías, mi estrella y mi paradigma

Leydis

Dedicar esta tesis a las personas que más quiero en esta vida aunque así no lo demuestre.

A mi madre, que aunque físicamente no se encuentre entre nosotros ha sido la fuerza que me ha impulsado a seguir adelante durante todos estos años, ha sido la sombra que me apoya y me sigue en cada paso que doy. Gracias por alumbrarme el camino y brindarme el optimismo que en ocasiones me ha faltado.

A mi abuela, que es la mujer más fuerte y tierna que en la vida he conocido. Quien, ha sido un ejemplo para mí durante toda mi niñez, mi adolescencia y mi juventud. Siempre a mi lado en cada paso de la vida que doy, cuando las fuerzas me faltaron fueron sus consejos los que me ayudaron a seguir adelante y ser quien soy en este día. No me dieron la oportunidad de sentir el cariño de una madre pero me dieron una abuela que vale oro no la cambio por nada en esta vida. Gracias por existir y ser quien eres hoy en mi vida. Te quiero Mucho nunca lo dudes.

Al hombre que más quiero en esta vida, el mejor amigo, mejor hijo y mejor esposo, ese es mi papá y le doy gracias a Dios todos los días que me lo haya dado como papá, pues para mi es la persona más perfecta y más completa que ha creado la naturaleza y aunque fuera la persona más pobre del mundo siempre voy a quererlo y vivir orgullosa de él, y si vuelvo a nacer felizmente tendría el padre que tengo. Te Quiero Mucho papá.

RESUMEN

El estudio de factibilidad económica constituye hoy día un tema de gran interés y una necesidad para la ejecución de proyectos de inversión. La implementación de un sistema que contribuya a la toma de decisiones puede ser una tarea ardua, al igual que encontrar una herramienta libre que permita el trabajo con sistemas de factibilidad económica, proponiendo una solución a dichos problemas.

La presente investigación describe el desarrollo de un sistema que permite realizar de forma automatizada estudios de factibilidad económica en el centro GEYSED, dicha herramienta permite el cálculo de la ficha de costo y el flujo de caja de los proyectos así como los indicadores de evaluación económica como son: Valor Actual Neto (VAN) y la Tasa interna de retorno (TIR).

En el documento se abarcan las principales características de otros sistemas de estudio de factibilidad económica. Se recogen además un conjunto de herramientas y tecnologías que servirán de apoyo a la modelación e implementación de la solución. De conjunto con la aplicación desarrollada se generaron varios documentos los cuales serán de apoyo para el usuario final.

El cliente tendrá entonces un sistema que contribuirá al proceso de la toma de decisiones del centro de desarrollo GEYSED en aras de obtener impactos tanto económicos como sociales. La solución es multiplataforma lo cual indica que su uso no está restringido a ningún ámbito por lo que puede ser usada y modificada según las necesidades individuales del cliente.

Palabras claves: factibilidad, indicadores, proceso.

ÍNDICE

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

INTRODUCCIÓN

La planificación es un proceso fundamental para lograr determinados objetivos, debido a que la misma permite la toma de decisiones para alcanzar un futuro deseado, teniendo en cuenta diversos aspectos como la situación actual y los factores externos e internos que presenten las empresas, instituciones y proyectos sin importar su magnitud. Los proyectos de inversión constituyen una serie de propuestas que se llevan a cabo en conjunto con determinados recursos, logrando el crecimiento de la producción y la calidad en un período de tiempo.

La evaluación de proyectos de inversión constituye un tema de interés e importancia, mediante este proceso se valora cualitativa y cuantitativamente las ventajas y desventajas de destinar recursos a un determinado proyecto, convirtiéndose en un análisis fundamental a la hora de evaluar la conveniencia, o no, de ejecutar cualquier proyecto de inversión y el momento óptimo para ejecutarlo. Numerosas instituciones intensifican sus esfuerzos en la búsqueda de soluciones que permitan garantizar si sus proyectos serán factibles, a partir de un estudio adecuado de la factibilidad económica.

La factibilidad económica es la capacidad que tiene el proyecto para financiarse, es allí donde es necesario determinar la disponibilidad de los recursos necesarios para llevar a cabo los objetivos planteados. Un correcto estudio de la factibilidad económica viene dado por tres indicadores que garantizan la rentabilidad de determinado proyecto: valor actual neto (VAN), tasa interna de retorno (TIR) y el período de recuperación (PR) (Ministerio de Economía, 2007).

El desafío que muestra en la actualidad el desarrollo de las tecnologías de la información y las comunicaciones (TIC) es sustancial. A medida que los cálculos y proyecciones progresan se ha evidenciado el crecimiento de los estudios de factibilidad económica convirtiéndose en uno de los factores claves de la economía actual. Esto, conlleva que determinadas empresas e instituciones se vean cada vez más adentradas en la búsqueda de soluciones automatizadas para dejar a un lado el excesivo cúmulo de trabajo de forma manual, acudiendo para la solución de sus necesidades a compañías de software.

Cuba, ha sido sometida a un crudo bloqueo hace más de medio siglo por el gobierno de los Estados Unidos. Estas acciones propician que muchos adelantos tecnológicos, no lleguen a manos de la sociedad cubana. El gobierno cubano con el fin de elevar el nivel de acceso a la información, lleva a cabo la informatización de la sociedad. Para ello se han creado disímiles instituciones entre ellas se encuentra la

Empresa Nacional de Software (Desoft), la Empresa Cubana Productora de Software para la Técnica Electrónica (Softel) y la Universidad de las Ciencias Informática (UCI), creado bajo el albor de la batalla de idea y liderado por el Comandante en Jefe Fidel Castro entre otros líderes de la Revolución.

Entre los principales objetivos de la UCI se encuentra la informatización de la sociedad cubana y el desarrollo de software. Para ello cuenta con diversos centros de desarrollo, ejemplo de ello el centro de Geoinformática y Señales Digitales (GEYSED), que cuenta con dos departamentos destinados a la producción de software y cinco áreas para su asesoramiento entre las que se encuentra: tecnología, planificación y control, economía, mercadotecnia y calidad.

El área económica se hace responsable de la elaboración y control del presupuesto con el que cuenta el centro y la revisión y supervisión de las fichas de costo de cada uno de sus proyectos tanto nacionales como internacionales. La ficha de costo es uno de los elementos que se utilizan para calcular el flujo de caja de cada uno de los proyectos y realizar el estudio de factibilidad económica mediante los indicadores de evaluación financiera y conocer si los proyectos son factibles o no realizarlos y la importancia e impacto que tendrán para el Centro.

Otra de las actividades que se llevan a cabo en el centro como parte del control interno es la realización de controles mensuales a los materiales de oficina y a los activos fijos detallados en el plan anual de control, permitiendo a su vez conocer cómo se lleva a cabo el cumplimiento de la Ley 60 (Gaceta Oficial, 2011). Esta ley rige todos los parámetros que por control interno se deben tener y medir por puntos en las entidades para de esta forma mejorar el funcionamiento de las mismas.

Todas estas actividades antes mencionadas son de vital importancia para el funcionamiento del área económica del centro. Sin embargo uno de los procesos vitales dentro del área económica es el estudio de factibilidad, el cual no es posible realizarlo en GEYSED debido a que no se usan los indicadores de evaluación financiera, los mismos son el punto clave para determinar la viabilidad de un proyecto. En caso de realizarlo sería manualmente, lo que ocasionaría pérdida de tiempo y el no tener la información necesaria de forma puntual para presentar a un cliente determinado, pues el estudio de factibilidad permite que se pueda realizar incluso por componentes de desarrollo.

El problema de no realizar dicho estudio donde las soluciones económicas–financieras sean las más ventajosas para la entidad, limita la fundamentación y documentación de una inversión a acometer.

Además no es posible garantizar que los planes para la ejecución y puesta en explotación de la inversión responda a las necesidades reales de la economía nacional lo que permitiría a la alta dirección del centro tomar decisiones sobre la conveniencia o no de dicha inversión. Actualmente cuando se presenta una documentación al cliente no es posible presentarle un precio real que ante la entidad sea el adecuado ya una vez calculado las pérdidas y ganancias del mismo.

Teniendo en cuenta lo anteriormente planteado se define como **problema a resolver:** ¿Cómo contribuir en los procesos de evaluación económica-financiera de los proyectos del centro de desarrollo GEYSED para apoyar en el proceso de la toma de decisiones de sus directivos?

El **objeto de estudio** es: Los procesos de evaluación económica-financiera de proyectos. Enmarcando en el **campo de acción**: Sistemas informáticos para el estudio de la factibilidad económica en el centro GEYSED. Para dar solución al problema se define como **objetivo general:** Desarrollar un sistema informático que permita realizar el estudio de la factibilidad económica en los proyectos del centro GEYSED.

Se plantea como **idea a defender**: El desarrollo de un sistema informático que permita realizar el estudio de la factibilidad económica, posibilitará entender la capacidad que tiene el proyecto para generar beneficios y el impacto que estos pueden lograr sobre el centro de desarrollo GEYSED. Para dar cumplimiento al objetivo general se definen las siguientes **tareas de la investigación:**

1. Caracterizar los procesos involucrados en el estudio de la factibilidad económica del centro GEYSED.
2. Caracterizar a nivel nacional e internacional las herramientas asociadas al estudio de la factibilidad económica.
3. Determinar los métodos, herramientas y procedimientos factibles para el desarrollo del sistema.
4. Realizar el análisis y diseño de los artefactos que intervienen en la construcción del sistema.
5. Implementar la solución propuesta.
6. Diseñar y Aplicar las pruebas al sistema.

Entre los métodos de investigación que sostienen el presente trabajo se encuentran:
Dentro de los Métodos Teóricos:

Analítico-Sintético**:** Permitió dividir por partes la información obtenida a lo largo de la investigación para posteriormente obtener de ella los objetos de la investigación necesarios para fundamentar la misma.

Histórico-Lógico: Se utilizó para obtener un conocimiento de las distintas etapas del estudio de factibilidad en su sucesión cronológica y para conocer la evolución y desarrollo del mismo. Además se utilizó para extraer la esencia de la información obtenida en el transcurso de la investigación.

Modelación: Se utilizó para modelar toda la documentación de la aplicación y el análisis documental.

Dentro de los Métodos Empíricos:

Entrevista: Se realizó a través de una recopilación de información mediante una conversación profesional la cual además de adquirirse información acerca de lo que se investiga, tiene importancia educativa. Además permitió profundizar en el tema y en aspectos que fueron surgiendo a medida que se realizaban los encuentros con la asesora económica del centro. Se efectuó una entrevista no estructurada ya que permitió tener más flexibilidad así como formular las preguntas de manera informal, con el orden y profundidad requerida por el momento, posibilitó además explotar áreas que surgen espontáneamente durante la entrevista.

Análisis documental: Se realizó para buscar información sobre el objeto y campo de acción en documentos normativos propios de la institución como manuales y actas.

Estructura de la investigación:

Capítulo # 1_Fundamentación Teórica del estudio de la factibilidad económica: En este capítulo se abordaron los conceptos asociados al dominio del problema, se explican los indicadores que se tienen en cuenta para un adecuado estudio de la factibilidad económica, además de un análisis del estado del arte referente al objeto de estudio.

Capítulo # 2_Tendencias y tecnologías actuales a desarrollar: En este capítulo se describen las tecnologías y herramientas utilizadas en el sistema a desarrollar, además de comparar las tecnologías para escoger cual se asemeja o brinda más utilidad para la realización del mismo.

Capítulo # 3_Análisis y Diseño: En este capítulo se realizó el análisis y diseño del sistema a desarrollar.

Capítulo # 4_Implementación: Se abordó todo el proceso de construcción, así como el diseño y la realización de pruebas al sistema.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA DEL ESTUDIO DE LA FACTIBILIDAD ECONÓMICA

Introducción

En el presente capítulo se recogen los aspectos fundamentales para el entendimiento del dominio del problema. Se relacionan los conceptos asociados al estudio de la factibilidad económica y se realiza un resumen de las características fundamentales de las soluciones existentes. Se describe de forma general el objeto de estudio haciendo énfasis en la situación problemática existente en el centro GEYSED referente al estudio de la factibilidad económica.

1.1. Conceptos asociados al dominio del problema

Factibilidad

Se refiere a la disponibilidad de recursos para llevar a cabo determinado objetivo. La factibilidad permite conocer si determinado proyecto, proceso o tarea tendrá o no alcances futuros. Existen diferentes tipos de factibilidad: económica, técnica y operacional u organizacional (Kendall, 2010).

Economía

Se considera el estudio de la manera en que las sociedades utilizan los recursos escasos para producir mercancías valiosas y distribuirlas entre los diferentes individuos (Paul, y otros, 2002).

Lionel Robbins expresa “que la economía es la ciencia que se encarga del estudio de la satisfacción de las necesidades humanas mediante bienes que siendo escasos tienen usos alternativos entre los cuales hay que optar” (Méndez, 2009)

En síntesis se puede decir que la economía es la prudente administración de los recursos de una sociedad, familia o individuo, con la finalidad de satisfacer sus necesidades en lo material. Así como el estudio de la manera o el modo en que las sociedades gestionan sus recursos para satisfacer las necesidades en productos y servicios. Además, la economía explica el cómo los individuos y organizaciones logran sus ingresos y de qué forma lo invierte.

Factibilidad Económica

La factibilidad económica consiste en la investigación, elaboración y desarrollo de un modelo operativo viable para solucionar problemas, requerimientos necesidades de organizaciones o grupos sociales que pueden referirse a la formulación de políticas, programas, tecnologías, métodos, o procesos. El proyecto debe tener el apoyo de una investigación de tipo documental, y de campo, o un diseño que incluya ambas modalidades (Flores, 2003). Del mismo modo Fidias Arias expresa que: “se trata de una propuesta de

acción para resolver un problema práctico o satisfacer una necesidad. Es indispensable que dicha propuesta se acompañe de una investigación, que demuestre su factibilidad o posibilidad de realización" (Arias, 2006).

Se puede decir entonces que la factibilidad económica consta de un estudio u investigación que se lleva a cabo para solucionar problemas prácticos y satisfacer necesidades. También es la disponibilidad de los recursos necesarios para llevar a cabo los objetivos o metas señaladas.

Proyecto de inversión

Un proyecto de inversión es una propuesta de acción que implica utilización de un conjunto determinado de recursos para el logro de resultados esperados (Sanín, 2009). Colin F. Bruce expresa: "un paquete discreto de inversiones, insumos y actividades, diseñados con el fin de eliminar o reducir varias restricciones al desarrollo, para lograr uno o más productos o beneficios, en términos del aumento de la productividad y del mejoramiento de la calidad de vida de un grupo de beneficiarios dentro de un determinado período de tiempo" (Bruce, 1982).

Se puede expresar que un proyecto de inversión es un conjunto de actividades con objetivos y trayectorias organizadas para la resolución de problemas en un período de tiempo determinado.

Evaluación económica de proyectos

Es un método de análisis útil para adoptar decisiones ante diferentes alternativas. Tiene como objetivo identificar las ventajas y desventajas asociadas a la inversión de un proyecto antes de la ejecución del mismo. Integra en su análisis tanto los costes monetarios como los beneficios expresados en otras unidades relacionadas con las mejoras en las condiciones de vida de un grupo (Santos, 2005).

Evaluación financiera de proyectos

Tiene como finalidad aportar una estrategia que permita al proyecto allegarse a los recursos necesarios para su implantación, y contar con la suficiente liquidez y solvencia, para desarrollar ininterrumpidamente operaciones productivas y comerciales. El estudio financiero aporta la información necesaria para estimar la rentabilidad de los recursos que se utilizarán. Se considera la vertiente monetaria de un proyecto con el objetivo de considerar su rentabilidad en términos de flujos de dinero (Fernández, 2007).

Valor Actual Neto

Consiste en encontrar la diferencia entre el valor actualizado de los flujos de beneficio y el valor, también actualizado, de las inversiones y otros egresos de efectivos (Jiménez, y otros, 2007). El valor actual neto,

es el valor monetario que resulta de restar la suma de los flujos o entradas descontadas del proyecto a la inversión inicial. La tasa de descuento o actualización es la tasa mínima aceptable (Bonilla, 2007).
El valor actual neto no es más que la suma monetaria resultante de descontar a la inversión inicial del valor de los egresos y las inversiones realizadas por los proyectos.

Tasa Interna de Retorno

La tasa interna de retorno (TIR) representa la tasa de interés más alta que un productor podría pagar sin dinero, si todos los fondos para el financiamiento de la inversión se tomaran prestados y éste se pagara con las entradas en efectivos de la inversión medida que se fuesen produciendo (Miniño, 1994). La TIR se define como la tasa de descuento que hace que el valor presente de los flujos de efectivos esperados de un proyecto sea igual que el monto inicial invertido (Brigham, y otros, 2008).
La tasa interna de retorno consiste en representar el valor más alto que pueda llegar un proyecto sin tener fondos monetarios, pagándose así con la inversión en efectivo que pueda tener el proyecto al iniciarse.

1.2. Objeto de estudio

1.2.1. Descripción General

La evaluación de proyectos de inversión es un análisis que se lleva a cabo mediante un proceso de varias aproximaciones en las que intervienen técnicos, financistas y administradores, realizando así, una evaluación del estudio de la factibilidad económica de los proyectos productivos. El estudio de la factibilidad económica es de vital importancia debido a que controla los fondos suficientes para la realización de cualquier proyecto, ya sea porque no se cuente con los recursos necesarios o capital humano.
Son disímiles las empresas en las que se realizan procesos para el estudio de la factibilidad económica, pues le permite garantizar si un proyecto es viable o no. El centro de desarrollo GEYSED, dedicado a la producción de software también requiere de un estudio de factibilidad económica enfocado a brindar apoyo en las áreas contable-financiera, administrativa y de mercadeo. De acuerdo con las necesidades detectadas en el mercado se permite realizar diagnósticos de viabilidad, encontrar las debilidades, dotar de herramientas precisas para la correcta ejecución de los procesos y proyectar el centro hacia la consecución de sus objetivos, orientados siempre a generar valor.

Actualmente en el centro GEYSED no se realizan estudios de factibilidad económica utilizando indicadores de evaluación financiera, por su parte existe una plantilla dentro de la ofertas que se le

entrega al cliente donde se definen las políticas y estrategias de precios que se han de aplicar, pero esto no define si el proyecto será factible o no, pues los indicadores básicos del estudio de la factibilidad económica están ausentes (VAN y TIR).

En el centro se desarrollan un determinado número de proyectos, de los mismos, a pesar de su impacto y utilidad en la sociedad no se conocen si serán rentables o no, convirtiéndose en uno de los problemas que actualmente se presenta, pues se están asignando recursos tanto humanos como materiales, sin obtener beneficios económicos. En caso de que el proyecto no sea rentable económicamente solo se ejecuta en caso de ser un proyecto que tenga impacto social, ya que una de las razones de la UCI es la informatización de la sociedad.

Actualmente el área económica del centro necesita una herramienta destinada a la realización del estudio de factibilidad económica para el apoyo a la toma de decisiones, por lo que se hace necesario el uso de los mismos, teniendo en cuenta los indicadores tradicionales y los procesos esenciales para su correcta elaboración.

1.3. Procesos para realizar la evaluación del estudio de la factibilidad económica

Flujo de Caja

El flujo de caja es la expresión en dinero líquido de los costos y los beneficios esperados y su estimación, se considera el paso más crítico y difícil en la evaluación de proyectos. Es además un componente imprescindible de la presupuestación de capital o plan de inversiones de la empresa. El flujo de caja para un período dado es la diferencia entre el flujo monetario recibido y el flujo monetario emitido (pagada) si un proyecto de inversión es desarrollado (Jácome, y otros, 2012).

El flujo de caja no es lo mismo que el beneficio o utilidad contable, la razón radica en que la utilidad se calcula como ingresos menos gastos mientras el flujo de caja es resultado de la diferencia entre entradas y salidas de efectivo. Debido a la existencia de fenómenos como el crédito o procedimientos contables como la depreciación, ingresos no es sinónimo de cobros ni gastos lo es de pagos. Dado que el proceso de inversión se expresa como una corriente de cobros y pagos que se producen en el tiempo, la evaluación de una inversión se realiza tomando en consideración dicha corriente (Jácome, y otros, 2012).

Se puede determinar que el flujo de caja no es más que la acumulación neta de activos líquidos en un período determinado, por lo que constituye un indicador importante de la liquidez de los proyectos. El flujo

de caja es la base para analizar la viabilidad de proyectos de inversión, pues los mismos son la base para el cálculo del valor actual neto (VAN) y de la tasa interna de retorno (TIR).

El presupuesto de efectivo es un informe de las entradas y salidas de efectivo planeadas de la empresa que se utiliza para calcular sus requerimientos de efectivo a corto plazo, con particular atención a la planeación en vista de excedentes y faltantes de efectivo. Una empresa que espera un excedente de efectivo puede planear inversiones a corto plazo, en tanto que una empresa que espera faltantes de efectivo debe disponer del financiamiento a corto plazo. Se utiliza para analizar la viabilidad de proyectos, siendo éste la base de cálculo del Valor Actual Neto (VAN) y la Tasa Interna de Retorno (TIR) (Peña, 2007).

Usualmente el Flujo de caja se calcula con una matriz con columnas y filas. En las columnas se disponen los períodos de tiempo que generalmente son de meses y en las filas los ingresos y salidas de dinero.
Las entradas es todo el dinero que ingresa a la empresa por sus actividades productivas, ya sea por venta de servicios o productos. Las salidas es todo el dinero que sale de la empresa y que es necesario para llevar a cabo su actividad productiva (Peña, 2007).

$$Flujo\ de\ caja\ =\ Entradas\ (Ingresos) - Salidas\ (Costos).$$

La siguiente tabla indica los factores a tener en cuenta para el cálculo del flujo de caja para los proyectos que se desarrollan en la Universidad de las Ciencias Informáticas (UCI):

	Año/Mes $_0$	**Año/Mes $_1$**	**Año/Mes $_n$**
Inversiones			
Ingresos por Ventas			
(-) Gastos de desarrollo UCI			
(-) Reutilización de Componentes			
(-) Know How			
(-) Subcontratación			

(-) Calidad			
(-) Administración UCI			
Flujo de Caja			

Tabla 1: Plantilla de Flujo de Caja. Elaboración Conjunta.

Para estimar el costo de los proyectos de la UCI se debe determinar el total de gastos del proyecto, el cual, según el esquema propuesto está formado por la suma de los seis elementos que se describen a continuación:

Gastos de desarrollo UCI: Cuando el personal se encuentre trabajando en la UCI, se calcula multiplicando el tiempo de desarrollo (Se calcula según el método desarrollado por la Empresa de Calidad de Software (Calisoft) y la experiencia acumulada por cada centro de desarrollo) en horas/hombres necesarias para cumplir el alcance pactado por el costo unitario por hora de cada hombre (12 $/hora, la cual fue aprobada por la Red de Centros de Desarrollo en el año 2010) (Gestión de Proyectos, 2012).
Cuando el personal se encuentre trabajando en el exterior y siempre que los gastos de vida y pasaje, sean asumidos por una Empresa Comercializadora u otro cliente, entonces solo se tendrá en cuenta el gasto de personal de los recursos que laboran en el extranjero (Gestión de Proyectos, 2012).
Usualmente en los proyectos requieren las dos variantes, por lo que lo normal es sumar los resultados de ambas variantes para obtener el gasto de desarrollo UCI.

Reutilización de Componentes: Se calcula sobre la base de la amortización del costo de los componentes reutilizados en la solución de la oferta, los cuales usualmente pertenecen a varios centros de desarrollo.

Know How: Se determina sobre la base de la experiencia del equipo encargado de tomar las decisiones del centro, teniendo en cuenta la experiencia y el dominio logrado en las tecnologías empleadas. Este se tiene en cuenta en forma de porciento en un rango entre el 10 y el 25% del gasto de desarrollo UCI. Esto fue acordado en la Red de Centros de Desarrollo (RCD).

Subcontratación: Su valor se determina sobre la base de las cotizaciones o contratos previos de entidades externas a la UCI que oferten y/o presten servicios necesarios para el desarrollo de soluciones

y/o servicios. En el caso de las soluciones en que la empresa comercializadora asume estas subcontrataciones, no se debe incluir en el costo este concepto.

Calidad: Es el 3,09% del gasto de desarrollo UCI, según acuerdo de la RCD.

Administración UCI: Es hasta 5% sobre las partidas de gasto de desarrollo UCI, más la Reutilización de componentes y la calidad (Gestión de Proyectos, 2012).

1.4. Cálculo de los indicadores

Para evaluar la viabilidad y/o factibilidad de un proyecto los indicadores que se han determinado usar para el estudio de factibilidad económica coinciden con los utilizados por los expertos, los mismos son: el valor actual neto (VAN) y la tasa interna de retorno (TIR). Estos indicadores de evaluación permiten dar una medida, más o menos ajustada, de la rentabilidad que se puede obtener con el proyecto de inversión, antes de ponerlo en marcha. También permiten compararlo con otros proyectos similares y en su caso, realizar los cambios en el que se consideren oportunos para hacerlo más rentable.

Existen fórmulas que permiten calcular los indicadores antes mencionados, las cuales se detallan a continuación:

Valor Actual Neto (VAN)

El Valor Actual Neto (VAN) se suele definir como el valor actual de los flujos de caja esperados, entendiéndose por flujos de caja, el flujo de ingresos y egresos en efectivo. Una definición más explícita correspondería, entonces, a la que lo define como el valor actualizado del saldo entre el flujo de ingresos y egresos en efectivo generados por un proyecto durante su vida útil. Para hallar dicho saldo se descuentan los flujos a una tasa de descuento constante durante el período de vida útil del proyecto, lo que permite simplificar el modelo de cálculo del VAN de la forma siguiente:

$$VAN = -A + [FC_1/(1+k)^1] + [FC_2/(1+k)^2] + \cdots + [FC_n/(1+k)^n]$$

Siendo:

A: desembolso inicial.

FC**:** flujos de caja.

n**:** número de años (1,2,...,n).

k**:** tipo de interés ("la tasa de descuento").

$1/(1+r)^n$ **:** factor de descuento para ese tipo de interés y ese número de años.

Un aspecto primordial para el cálculo del VAN es definir la tasa de descuento a utilizar. La tasa de descuento es la rentabilidad mínima que se le exige al proyecto. En la determinación de la misma, se deben tener en cuenta factores objetivos como: las tasa de interés a que la empresa y el país reciben recursos financieros, los niveles de rentabilidad de la rama económica a que pertenece el proyecto, el riesgo financiero, y también criterios subjetivos relacionados a la experiencia y al buen juicio de quien evalúa la inversión (Rodríguez, 2006).

Criterios de decisión:

Si VAN> 0: El proyecto es rentable.

Si VAN = 0: El proyecto es postergado.

Si VAN< 0: El proyecto no es rentable.

Tasa Interna de Retorno (TIR)

La Tasa Interna de Rendimiento (TIR), se define como aquella tasa de actualización o descuento (r), que hace cero la rentabilidad absoluta neta de la inversión. Es decir, aquella tasa de descuento que iguala el valor actual de la corriente de cobros con el valor actual de la corriente de pagos. Analíticamente el criterio de la TIR se expresa como sigue:

$$VAN = -A + [FC_1/(1+r)^1] + [FC_2/(1+r)^2] + \cdots + [FC_n/(1+r)^n] = 0$$

Donde se requiere hallar aquel valor de r que iguala a 0 los flujos de caja del proyecto y que representa, por tanto, su tasa interna de rendimiento. En síntesis, la TIR se calcula igual que el VAN, la única diferencia es que se estiman aquellas tasas de actualización que hacen el VAN = 0, lo que se alcanza en un proceso de aproximaciones sucesivas (Rodríguez, 2006).

El criterio general para saber si es conveniente realizar un proyecto es el siguiente:

$r > k$ Se debe realizar el proyecto debido a que la rentabilidad del proyecto es mayor que el costo de oportunidad del capital.

$r < k$ No se realizar el proyecto, solo se aceptaría para proyectos de financiamiento, en los cuales el propósito del proyecto es pedir un determinado préstamo por el cual hay que pagar intereses a una tasa k.

$r = k$ El decisor es indiferente entre realizar el proyecto o no.

1.5. Análisis de factibilidad económica

El estudio financiero aporta una estrategia que permite al proyecto allegarse a los recursos necesarios para su implantación, y contar con la suficiente liquidez y solvencia, para desarrollar ininterrumpidamente operaciones productivas y comerciales. Este aporta la información necesaria para estimar la rentabilidad de los recursos que se utilizarán.

El análisis de factibilidad económica culmina el estudio de factibilidad económica, el mismo se encarga de expresar los resultados que brindan los indicadores del VAN y TIR sobre la base de los criterios de decisión de cada uno de ellos para determinar si un proyecto será o no factible apoyando así el proceso de toma de decisiones.

1.6. Soluciones existentes que permiten determinar el estudio de la factibilidad económica

EvalAs

EvalAs es un sistema que permite la evolución económica de proyectos productivos. Este es un software que permite la introducción de datos de productos y producción, inversiones e impuestos, costos fijos y variables. A partir de los datos ingresados se calcula la matriz de flujos de caja y luego los principales indicadores financieros: valor actual neto (VAN), tasa interna de retorno (TIR) y el período de recuperación (PR).

El software permite detallar la información de los créditos solicitados para financiar el proyecto además de calcular automáticamente las cuotas para los préstamos con sistemas de amortización francés y alemán, permitiendo adicionalmente el ingreso manual de los montos de las cuotas, si se utiliza otro sistema. A partir de estos datos, se calculan nuevamente los principales indicadores del proyecto. Exporta todo el contenido del proyecto que se está trabajando a un archivo de formato HTML, siendo una alternativa práctica y económica.

En la última versión se puede realizar análisis de sensibilidad y de escenarios. Permite la comparación con otro proyecto para determinar cuál alternativa es la más rentable. Puede utilizarse para determinar rentabilidad de proyectos de producción de software, industrial, forestal y agropecuaria (EvalAs, 2012).

Managerial Analyzer

Es una herramienta financiera creada para el análisis económico-financiero empresarial, que facilita la confección, interpretación y realización del estudio y análisis total de la situación económico-financiera de cualquier empresa. Puede realizar un diagnóstico preciso y eficaz sobre la situación económico-financiera de las instituciones. Previene mediante el diagnóstico precoz los posibles problemas presentes y futuros. Analiza con tiempo la evolución a futuro de su empresa.

La aplicación realiza diferentes tipos de análisis entre los que se encuentra:

Análisis de balance realizándose las siguientes actividades: balance de situación, balance de pérdidas-ganancias, análisis del activo disponible-realizable, análisis del activo real-ficticio, análisis del exigible a corto-largo plazo, análisis de los resultados de explotación y el análisis del balance por masas patrimoniales.

Rentabilidad: rentabilidad económica y financiera, umbral de rentabilidad, cálculo del apalancamiento financiero, análisis y cálculo del fondo de maniobra, análisis de la capacidad de autofinanciación, análisis de la capacidad de crecimiento, períodos de maduración y el ciclo de caja (Analyzer, 2012).

Financial Analysis

Ésta herramienta ofrece gran variedad de análisis para evaluar la salud y el rendimiento de una empresa. Posee treinta y tres modelos financieros, capacidad de crear diagramas y gráficos, capacidad de exportación de ficheros, ajustes de conversión e inflación. Sin lugar a duda Financial Analysis es una herramienta indispensable para la optimización de empresas y/o negocio (Analysis, 2007).

VERSAT

Sistema económico integrado. Permite la gestión de los recursos económicos de la empresa en sus diferentes áreas de contabilidad general, activos fijos, finanzas, costos por procesos, inventario, facturación, presupuesto maestro entre otros, con una gama de funcionalidades relacionada entre sí, facilitando la consolidación de los mismos (VERSAT, 2012).

1.6.1. ¿Por qué ninguno de los sistemas anteriores?

Los sistemas anteriores independientemente del lugar donde hayan sido desplegados; son reconocidos en la esfera económica, estos a pesar de su impacto no son recomendados a ser utilizados en el centro debido a que no se ajustan al esquema económico que sigue la universidad; los mismos no pueden ser modificados puesto que trabajan sobre plataformas propietarias, no cumpliendo así con las políticas de migración a software libre que se llevan a cabo a nivel nacional. En el caso de VERSAT es uno de los sistemas nacionales, este cuenta con un amplio contenido económico pero no realiza un estudio de factibilidad económica, por su repercusión en diferentes empresas nacionales se le realizó un estudio pensando en requisitos que se pudieran ajustar a las necesidades del centro. El sistema a desarrollar debe permitir futuras modificaciones en caso de que el cliente desee realizar nuevas estimaciones de los costos de proyectos e incluirle nuevos requisitos según las necesidades del cliente.

1.7. Conclusiones parciales

Tras el análisis realizado teniendo en cuenta la fundamentación teórica de la presente investigación se ha determinado que un estudio de factibilidad económica aporta una estrategia que permite a las entidades allegarse a los recursos necesarios para su implantación, y contar con la suficiente liquidez y solvencia, para desarrollar ininterrumpidamente operaciones productivas y comerciales. Este aporta la información necesaria para estimar la rentabilidad de los recursos que se utilizarán en la realización de cualquier proyecto de inversión.

Luego de detallar los procesos de un correcto estudio de factibilidad económica se determina que es importante hacer constar que todas las técnicas de evaluación económica independientemente de su contenido deben considerarse en conjunto, pues garantizan una parte más sólida y segura en la toma de decisiones.

Se pudo determinar que a pesar de la existencia de herramientas de gran uso a nivel internacional y nacional, no se encontró una herramienta que pueda adaptarse a las necesidades del centro de desarrollo de GEYSED. Por lo que lo que se hace necesario desarrollar un sistema que sea capaz de calcular los indicadores necesarios para el estudio de la factibilidad económica.

CAPÍTULO 2: TENDENCIAS Y TECNOLOGÍAS ACTUALES A DESARROLLAR

Introducción

En el presente capítulo se realiza un análisis de las tecnologías actuales que se utilizan en el desarrollo de sistemas que estudien la factibilidad económica, haciendo énfasis en la que se utilizan para el desarrollo de la propuesta de dicha solución. Para ello, se tiene en cuenta la estandarización de tecnologías que se lleva a cabo en el centro para el cual se realizará la aplicación. Se abordan temas relacionados con las herramientas, lenguaje de programación a utilizar, así como el lenguaje de modelado y la metodología a emplear.

2.1. Metodología de Desarrollo de Software

Durante el desarrollo de software se corren riesgos que pueden repercutir en la solución final del producto, en este punto se hace necesario el uso de una metodología de software, la cual plantea una serie de procedimientos, técnicas y herramientas que deben tenerse en cuenta en el desarrollo de un producto. Puede seguir uno o varios modelos de ciclo de vida los cuales indican que es lo que hay que obtener a lo largo del desarrollo del software.

La metodología de software tiene como objetivo elevar la calidad del software que se produce en todas y cada una de sus fases de desarrollo (Canós, y otros, 2002).

Extreme Programming (XP)

Esta metodología de desarrollo de software es una de las más exitosas de la actualidad, se basa en la idea de que existen cuatro variables que guían el desarrollo de sistemas: costo, tiempo, calidad y alcance. La metodología consiste en una programación rápida o extrema, cuya particularidad es tener como parte del equipo, al usuario final, pues es uno de los requisitos para llegar al éxito del proyecto. La manera de encarar los desarrollos avalados por este modelo de desarrollo es permitir a las fuerzas externas (gerencia, clientes) manejar hasta tres de estas variables, quedando el control de la restante en manos del equipo de desarrollo (Beck, 2000).

XP, se basa en una constante retroalimentación entre el cliente y el equipo de desarrollo haciendo pequeñas iteraciones cada dos semanas, donde la única documentación existente es el código en sí, cada una de estas versiones cuenta con las modificaciones necesarias. Una de las desventajas que posee esta metodología es que no cuenta con una documentación del proyecto, lo más similar a la documentación son las historias de usuarios, pero al finalizar el proyecto se descartan. Se recomienda la realización de

dos secciones una con todas las historias de usuario que faltan por desarrollar, y otra donde se archiven las concluidas, esto aproximará el estado de avance del proyecto (Beck, 2000).

Proceso Unificado de Rational (RUP)

El Proceso Unificado de desarrollo es un marco de trabajo genérico que puede especializarse para una gran variedad de sistemas de software, para diferentes áreas de aplicación, diferentes tipos de organización, diferentes niveles de aptitud y diferentes tamaños de proyecto.

Es un proceso que define claramente quién, cuándo, cómo y qué debe hacerse, y como su enfoque está basado en modelos, utiliza un lenguaje bien definido para tal fin, el UML. El Proceso Unificado de desarrollo está basado en componentes, lo cual quiere decir que el sistema de software en construcción está formado por componentes de software interconectados a través de interfaces bien definidas (Jacobson, y otros, 2000).

La metodología RUP se divide en cuatro fases el desarrollo del software:

- Inicio: el objetivo en esta etapa es determinar la visión del proyecto.
- Elaboración: en esta etapa el objetivo es determinar la arquitectura óptima.
- Construcción: en esta etapa el objetivo es llevar a obtener la capacidad operacional inicial.
- Transmisión: el objetivo es llegar a obtener el reléase del proyecto.

Proceso Unificado Ágil (AUP)

El Proceso Unificado Ágil describe de una manera simple y fácil de entender la forma de desarrollar aplicaciones de software. Propone que aquellos elementos con alto riesgo obtengan prioridad en el proceso de desarrollo y sean abordados en etapas tempranas del mismo. Para ello, se crean y mantienen listas identificando los riesgos desde etapas iníciales del proyecto.

Desde un punto de vista tradicional, esta metodología puede verse más simple que las versiones más clásicas del proceso unificado, mientras que desde el punto de vista ágil, puede entenderse como una versión más pesada que las metodologías ágiles. Este puede ser un punto a favor a la hora determinar que metodología de desarrollo utilizar, por la incertidumbre que puede producir el elegir una metodología a la que el proyecto no se adapte completamente (Metodologías, 2010).

A continuación se muestra una tabla comparativa basada en los principales elementos que se deben tener en cuenta para la selección de la metodología a usar durante el desarrollo del sistema.

Elementos a comprar	**Metodologías**		
	XP	**RUP**	**AUP**
Artefactos generados	No genera artefactos.	Genera gran cantidad artefactos.	Flexibilidad en cuanto a los artefactos generados.
Importancia de la selección de tecnologías.	Resta importancia a las tecnologías.	Tiene en cuenta el uso de las tecnologías a usar.	Es adaptable a las tecnologías que se usen en el centro.
Calidad del producto	No está orientada a la calidad del producto.	Le atribuye gran importancia a la calidad del producto.	Le atribuye gran importancia a la calidad del producto.
Dominio del equipo de desarrollo	No tiene dominio.	Tiene dominio.	No tiene dominio.
Estandarización de tecnologías y tendencias en el centro GEYSED	No es de las más utilizadas.	Es la utilizada.	No se utiliza.

Tabla 2: Comparación de las metodologías. Elaboración propia.

2.1.1. ¿Qué metodología usar?

Luego de un análisis realizado teniendo en cuenta los elementos expuestos en la tabla comparativa y ajustándose a las particularidades del sistema a desarrollar se determina el uso de RUP ya que permite adaptarse a las características del centro como parte del proceso de estandarización de tecnologías y

herramientas. RUP es la metodología con la que el equipo de desarrollo se encuentra más familiarizado por lo que su uso y entendimiento se hace más fácil. Aporta además gran documentación entre los que se encuentran los modelos de diseño, las descripciones de los casos de uso del sistema y el manual de usuario, pues sirve como base para futuras modificaciones. También le atribuye gran importancia a la tecnología a usar como es el caso del lenguaje de modelado.

2.2. Lenguaje Unificado de Modelado: UML

El lenguaje unificado de modelado o notación (UML) sirve para especificar, visualizar y documentar esquemas de sistemas de software orientado a objetos. UML no es un método de desarrollo, lo que significa que no sirve para determinar qué hacer en primer lugar o cómo diseñar el sistema, sino que simplemente le ayuda a visualizar el diseño y a hacerlo más accesible para otros.

UML se compone de muchos elementos de esquematización que representan las diferentes partes de un sistema de software. Los elementos UML se utilizan para crear diagramas, que representa alguna parte o puntos de vista del sistema. Posee formas de modelar conceptos como por ejemplo las funciones del sistema, además de otras particularidades como la de escribir clases en un lenguaje determinado, esquemas de bases de datos y componentes de software reusables. Usa procesos de otras metodologías, aprovechando la experiencia de sus creadores (Larman, 2000)

Características que presenta UML:

- Permite modelar sistemas utilizando técnicas orientadas a objetos (OO).
- Permite especificar todas las decisiones de análisis, diseño e implementación, construyéndose así modelos precisos, no ambiguos y completos.
- Puede conectarse con lenguajes de programación (Ingeniería directa e inversa).
- Permite documentar todos los artefactos de un proceso de desarrollo (requisitos, arquitectura, pruebas y versiones).
- Cubre las cuestiones relacionadas con el tamaño, propias de los sistemas complejos y críticos.
- Existe un equilibrio entre expresividad y simplicidad, pues no es difícil de aprender ni de utilizar.
- UML es independiente del proceso, aunque para utilizarlo óptimamente se debería usar en un proceso que fuese dirigido por los casos de uso, centrado en la arquitectura, iterativo e incremental.

Se utiliza UML en su versión 2.0, no solo por ser el lenguaje de modelado más conocido, sino por ser el lenguaje que utiliza la metodología seleccionada. Además de ser el lenguaje de modelado que se utiliza en el centro y con el que el equipo de trabajo tiene mayor desempeño.

2.3. Herramientas CASE[1]

Las Herramientas CASE son diversas aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de software reduciendo el coste de las mismas en términos de tiempo y de dinero.

Según Almenares cada una de estas herramientas persigue nueve objetivos principales (Almenares, y otros, 2007):

- Mejorar la productividad en el desarrollo y mantenimiento del software.
- Aumentar la calidad del software.
- Mejorar el tiempo y coste de desarrollo y mantenimiento de los sistemas informáticos.
- Mejorar la planificación de un proyecto.
- Aumentar la biblioteca de conocimiento informático de una empresa ayudando a la búsqueda de soluciones para los requisitos.
- Automatizar, desarrollo del software, documentación, generación de código, pruebas de errores y gestión del proyecto.
- Ayuda a la reutilización del software, portabilidad y estandarización de la documentación.
- Gestión global en todas las fases de desarrollo de software con una misma herramienta.
- Facilitar el uso de las distintas metodologías propias de la ingeniería del software.

Visual Paradigm

Visual Paradigm para UML, es una herramienta diseñada para desarrollar software con programación orientada a objetos (POO), busca reducir la duración del ciclo de desarrollo brindando ayuda tanto a arquitectos, analistas, diseñadores y desarrolladores. La herramienta ayuda al equipo de desarrollo a agilizar el modelado del software, aumentando al máximo y acelerando el trabajo en equipo y las contribuciones individuales. Además, posee una buena cantidad de productos o módulos para facilitar el trabajo durante la confección de un software así como garantizar la calidad del producto final.

Permite dibujar todos los tipos de diagramas, código inverso, generar código desde diagramas y generar documentación.

[1] CASE: Ingeniería de Software Asistida por Computación.

Facilita a las organizaciones, integrar y desplegar sus aplicaciones empresariales y sus bases de datos. Incorpora el soporte para trabajo en equipo, permite a varios desarrolladores trabajar a la vez en el mismo diagrama y ver en tiempo real los cambios realizados por sus compañeros.

Además Visual Paradigm ofrece:

- Entorno de creación de diagramas para UML 2.0.
- Diseño centrado en casos de uso y enfocado al negocio que generan un software de mayor calidad.
- Uso de un lenguaje estándar común a todo el equipo de desarrollo que facilita la comunicación.
- Capacidades de ingeniería directa (versión profesional) e inversa.
- Modelo y código que permanece sincronizado en todo el ciclo de desarrollo.
- Disponibilidad de múltiples versiones, para cada necesidad.
- Disponibilidad de integrarse en los principales IDEs[2].
- Disponibilidad en múltiples plataformas.

Rational Rose.

Es una de las herramientas de modelado visual para sistemas basados en objetos. Se utiliza para modelar un sistema antes de proceder a construirlo. Cubre todo el ciclo de vida de un proyecto: concepción y formalización del modelo, construcción de los componentes, transición a los usuarios y certificación de las distintas fases. También utiliza un proceso de desarrollo iterativo controlado ("*controlled iterative process development"*), donde el desarrollo se lleva a cabo en una secuencia de iteraciones. Otra característica importante es que usando Rose se puede generar código en distintos lenguajes de programación a partir de un diseño en UML (Kruchten, 2004).

Rose permite que haya varias personas trabajando a la vez en el proceso iterativo controlado, para ello posibilita que cada desarrollador opere en un espacio de trabajo privado que contiene el modelo completo y tenga un control exclusivo sobre la propagación de los cambios en ese espacio de trabajo. También es posible descomponer el modelo en unidades controladas e integrarlas con un sistema para realizar el control de proyectos que permite mantener la integridad de dichas unidades.

[2] IDEs: es un entorno de desarrollo integrado, programa informático compuesto por un conjunto de herramientas de programación.

A continuación se muestra una tabla comparativa basada en los principales elementos que se deben tener en cuenta para la selección de la herramienta CASE a usar durante el desarrollo del sistema:

Elementos a comparar	**Herramientas CASE**	
	Visual Paradigm	**Rational Rose**
Herramienta multiplataforma	Es una herramienta multiplataforma.	No es una herramienta multiplataforma.
Estandarización de tecnologías y tendencias en el centro GEYSED	Es la herramienta utilizada en el centro.	No es utilizada en el centro.
Compatibilidad con la metodología a utilizar	Es compatible con la metodología de desarrollo a utilizar.	Es compatible con la metodología de desarrollo a utilizar.
Compatibilidad con Lenguaje de Modelado a utilizar	Es compatible con el lenguaje de modelado a utilizar.	Es compatible con el lenguaje de modelado a utilizar.
Dominio del equipo de desarrollo	Tiene dominio con la herramienta.	No tiene dominio con la herramienta.

Tabla 3: Comparación de las herramientas. Elaboración propia.

2.3.1. ¿Qué herramienta usar?

Luego de un análisis realizado teniendo en cuenta los elementos expuestos en la tabla comparativa y ajustándose a las particularidades del sistema a desarrollar se determina el uso de Visual Paradigm en su versión 8.0 ya que permite adaptarse a las características del centro como parte del proceso de estandarización de tecnologías y herramientas. Visual Paradigm es la herramienta con la que el equipo de desarrollo se encuentra más familiarizado por lo que su uso y entendimiento se hace más fácil. Es una herramienta multiplataforma lo cual es adecuado teniendo en cuenta las características del centro como

parte del proceso de migración que realiza. La herramienta seleccionada tiene gran compatibilidad con la metodología a usar al igual que con el lenguaje de modelado.

2.4. Lenguaje de Programación

Las computadoras necesitan de un lenguaje propio para poder interpretar las instrucciones que se les dan y para que los usuarios puedan controlar su comportamiento. Ese lenguaje que permite esta relación con las computadoras es el lenguaje de programación.

Un lenguaje de programación es aquel elemento dentro de la informática que permite crear programas mediante un conjunto de instrucciones, operadores y reglas de sintaxis; que pone a disposición del programador para que este pueda comunicarse con los dispositivos hardware y software existentes además puede ser utilizado para controlar el comportamiento de una máquina, particularmente una computadora (Terrence, 2007).

C++

Es una extensión del lenguaje de programación C, se considera un lenguaje de alto nivel que al mismo tiempo se basa en instrucciones cercanas a la máquina. Proporciona facilidades con respecto a la creación de estructuras de datos integradas fuertemente al lenguaje. Esto facilita el trabajo de los programadores en cuanto a la creación de tipos de datos con operaciones asociadas. Además dichas estructuras constituyen una extensión natural de los tipos de datos primitivos, lo que contribuye a elevar el grado de claridad y entendimiento (Stroustrup, 2000).

Es un lenguaje versátil, potente a la hora de realizar sistemas complejos y muy empleados debido a la documentación que posee para su entendimiento y utilización. Existen muchos algoritmos y librerías implementadas en C++, por lo que se puede adaptar fácilmente.

2.4.1. ¿Por qué C++?

Se determinó el uso de C++, pues es uno de los lenguajes que se utiliza en el centro debido al dominio que poseen los desarrolladores sobre el mismo. Se espera un sistema de software que sea adaptable por lo que no se debe prescindir del lenguaje en cuestión. Además C++ tiene a su favor que es utilizado por un conjunto de herramientas libres de desarrollo, es multiplataforma, orientado a objetos y en cuanto a ejecución es uno de los más rápidos y eficientes.

2.5. Framework de desarrollo

Es una estructura conceptual y tecnológica de soporte definido, normalmente con artefactos o módulos de software concretos, que puede servir de base para la organización y desarrollo de software. Típicamente, puede incluir soporte de programas, bibliotecas, y un lenguaje interpretado, entre otras herramientas, para así ayudar a desarrollar y unir los diferentes componentes de un proyecto (Sánchez, 2006).

Framework Qt

Qt es un framework para el desarrollo de aplicaciones multiplataforma. Algunas de sus características son: compatibilidad multiplataforma con un solo código fuente, comunicación con bases de datos, manejo de cadenas de caracteres además está programado en C++ y brinda soporte para un número elevado de lenguajes de programación (Martínez, 2011).

Qt dispone de grandes ventajas destacando:

- Qt es completamente gratuito para aplicaciones de código abierto aunque también se utiliza para aplicaciones comerciales.
- Las herramientas, librerías y clases están disponibles para casi todas las plataformas Unix y sus derivados (como Linux, MacOS X, Solaris).
- Qt tiene una extensa librería con clases y herramientas para la creación de elegantes aplicaciones.

2.5.1. ¿Por qué Qt?

Se considera para darle solución a la presente investigación el uso del framework Qt en su versión 4.8. Esto se debe principalmente a las facilidades que brinda a la hora de programar e incluye clases, librerías y herramientas para la producción de aplicaciones de interfaz gráfica. Está distribuida bajo los términos de GNU Lesser General PublicLicense, es software libre y de código abierto. Qt es multiplataforma, además dispone de una amplia gama de herramientas que facilitan entre otras cosas la creación de formularios, botones y ventanas de diálogo con el uso del ratón.

2.6. IDE de Desarrollo

Entorno de Desarrollo Integrado o "*Integrated Development Environment* (IDE)". Los IDE no son más que programas compuestos por herramientas que son necesarias al programador a la hora de implementar. Puede ocurrir que el programa se dedique únicamente a un solo lenguaje de programación o a varios. Es

un entorno de programación que ha sido empaquetado como un programa de aplicación, es decir, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica GUI. Los IDEs pueden ser aplicaciones por si solas o pueden ser parte de aplicaciones existentes (Blanco, 2012).

Qt Creator

Qt Creator es un entorno completo de desarrollo integrado para la creación de aplicaciones con el framework Qt, está diseñado para desarrollar aplicaciones e interfaces de usuario una vez y desplegarlas a múltiples sistemas operativos. El objetivo principal de este entorno de desarrollo es conocer las necesidades de desarrollo de los programadores que buscan la simplicidad, facilidad de uso, productividad, extensibilidad y apertura, mientras se baja la barrera de entrada para los recién llegados a Qt.

Las características clave de Qt Creator permiten a los programadores realizar diferentes tareas como desarrollar aplicaciones rápida y fácilmente con el asistente de proyectos, acceder rápidamente a proyectos recientes y sesiones, diseñar aplicaciones de interfaz de usuario basadas en "*widget"*, compilar, correr y desarrollar proyectos que se dirigen a múltiples plataformas, usar herramientas de análisis de código para verificar la administración de memoria en sus aplicaciones así como acceder fácilmente a información con el módulo del sistema de ayuda contextual de Qt (Guzmán, 2011).

2.6.1. ¿Por qué QT Creator?

Teniendo en cuenta las características expuestas anteriormente de Qt Creator en su versión 2.4 se determina que es el adecuado para la realización del sistema una vez identificado el lenguaje de programación así como el framework por la gran compatibilidad que presentan estas herramientas y tecnología. Además de ser el que actualmente se utiliza en los proyectos del centro de desarrollo de software GEYSED.

2.7. Gestor de Base dato

Un sistema de gestión de bases de datos es un paquete de software diseñado para almacenar y administrar base de datos. Estos permiten el acceso directo a la información y son capaces de manipularla ya que funcionan como una interfaz entre el usuario, la base de datos y las aplicaciones que la utilizan. Estos brindan un independiente y eficiente acceso a los datos reduciendo el tiempo de ejecución e integridad y seguridad de los mismos (Domínguez, 2008).

SQLite

SQLite es un sistema de gestión de bases de datos relacional, contenida en una pequeña biblioteca escrita en C. A diferencia de los sistemas de gestión de bases de datos cliente-servidor, el motor de SQLite no es un proceso independiente con el que el programa principal se comunica. En lugar de eso, la biblioteca SQLite se enlaza con el programa pasando a ser parte integral del mismo. El programa utiliza la funcionalidad de este ligero gestor a través de llamadas simples a subrutinas y funciones. Esto reduce la latencia en el acceso a la base de datos, debido a que las llamadas a funciones son más eficientes que la comunicación entre procesos. El conjunto de la base de datos (definiciones, tablas, índices, y los propios datos), son guardados como un sólo fichero estándar en la máquina host. Este diseño simple se logra bloqueando todo el fichero de base de datos al principio de cada transacción (SQLite, 2011).

2.7.1. ¿Por qué SQLite?

Se selecciona el gestor de base de datos SQLite en su versión 3.0 por ser un gestor ligero que no necesita de un servidor al cual conectarse. SQLite además es una biblioteca de software que implementa una en sí misma, sin servidor y sin necesidad de configuración. El código fuente para SQLite es de dominio público y por tanto es libre.

2.8. Conclusiones Parciales

Tras el análisis realizado teniendo en cuenta las tendencias y tecnologías actuales para desarrollar un sistema que estudie la factibilidad económica, donde se tuvo en cuenta las principales características de cada uno de los aspectos tratados se determinó que luego de una comparación realizada entre las posibles metodologías a utilizar RUP es la apropiada en el marco del sistema que se desarrollará, como lenguaje de modelado se utiliza UML en su versión 2.0 por su gran compatibilidad con la metodología seleccionada al igual que con la herramienta CASE a utilizar, Visual Paradigm en su versión 8.0. En el caso del lenguaje de programación, framework y entorno de desarrollo integrado se utiliza C++, framework Qt en su versión 4.8 y Qt Creator en su versión 2.4 respectivamente, para la selección de los mismos no se realizaron comparaciones debido a que se trató de ajustar de forma directa estas tecnologías con las características del centro donde se realizará el despliegue de dicho sistema. Como gestor de desarrollo se utilizará SQLite en su versión 3.0 por ser un gestor ligero y que no necesita un servidor para su correcto funcionamiento. El uso de las herramientas y tecnologías seleccionadas brindará una solución al problema planteado, ofreciendo además facilidad y dominio de las mismas para el equipo de trabajo.

CAPÍTULO 3: ANÁLISIS Y DISEÑO DE LA PROPUESTA DE SOLUCIÓN

Introducción

En el presente capítulo se hace una descripción de la solución propuesta a través del modelo de dominio representando los principales conceptos asociados al trabajo. Se realiza un análisis de los requisitos funcionales y no funcionales, casos de usos que se generan a partir de los requisitos funcionales y una explicación de cada uno de ellos a través de las descripciones textuales. Se define además la arquitectura del sistema y los artefactos que se generan en los flujos de trabajo del análisis y diseño.

3.1. Modelo del Dominio

Un modelo de dominio captura los tipos de objetos más importantes en el contexto del sistema. Los objetos del dominio representan las “cosas” que existen o los eventos que suceden en el entorno en el que trabaja el sistema. Las clases del dominio aparecen en tres formas típicas:

- Objetos del negocio que representan cosas que se manipulan en el negocio.
- Objetos del mundo real y conceptos de los que el sistema debe hacer un seguimiento.
- Sucesos que ocurrirán o han ocurrido.

El modelo de dominio se describe mediante diagramas UML, especialmente mediante diagrama de clases. Estos diagramas muestran las clases del dominio y cómo se relacionan unas con otras mediante asociaciones (Jacobson, y otros, 2000).

¿Cuándo se aplica un modelo de dominio?

El Modelo de Dominio se aplica cuando:

- Los flujos de información son difusos (múltiples orígenes, sólo eventos, sucesos).
- Imposibilidad de determinar subsistemas (exceso de interconexiones).
- Solapamiento de responsabilidades.
- Múltiples responsabilidades.
- Difícil establecimiento de reglas de funcionamiento.

3.1.1. Diagrama Modelo de Dominio

Luego de un estudio de todo lo anteriormente planteado se llega a la conclusión que debido a fronteras bien establecidas, donde se logren ver claramente quiénes son las personas que lo inician, quiénes son

los beneficiados, pero además quiénes son las personas que desarrollan las actividades en cada uno de estos procesos, se plantea el modelo de dominio.

Se realizará a través de un diagrama de clases UML. Los conceptos representados en el diagrama se mostrarán en un glosario de términos; logrando que el personal económico del centro y todo el interesado adquiera un entendimiento mínimo de un correcto estudio de factibilidad económica.

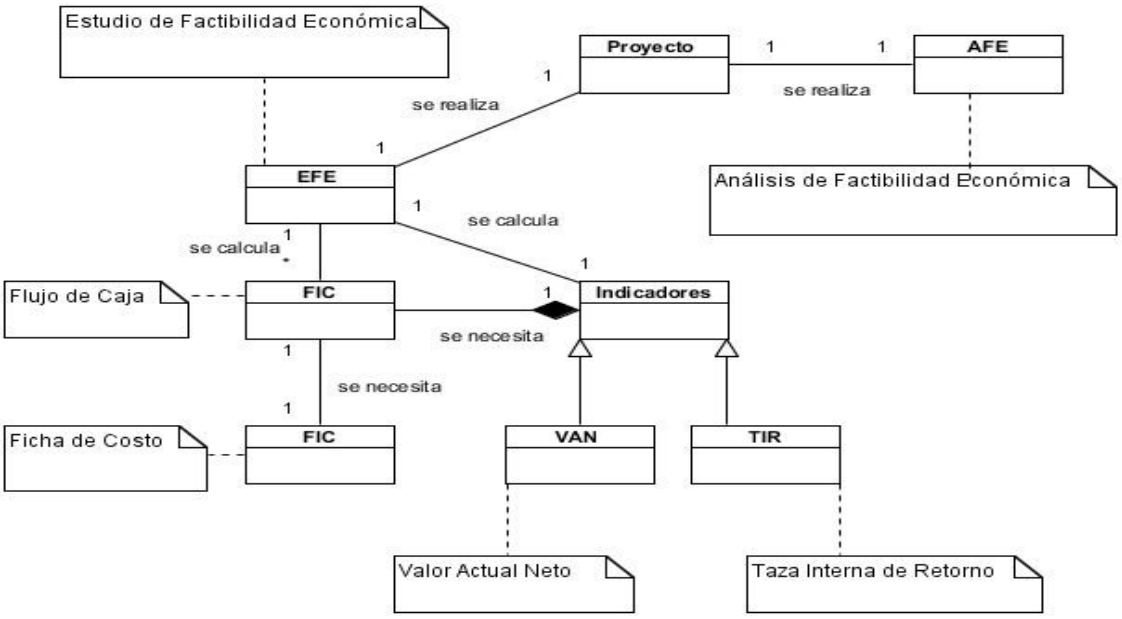

Figura 1: Diagrama de Clases de Dominio.

3.1.2. Conceptos principales del Modelo de Dominio

Un **proyecto** es un conjunto de personas que trabajan y se relacionan entre sí para lograr uno o varios objetivos, existen diferentes tipos de proyecto entre ellos los proyectos productivos. A cada uno de estos proyectos productivos, se le realiza un **estudio de factibilidad económica** (EFE) que permitirán conocer si la realización del proyecto será o no factible.

Un adecuado estudio de factibilidad económica está dado por dos factores: el cálculo de un **flujo de caja** que no es más que la acumulación neta de activos líquidos en un período determinado, para la elaboración del mismo se necesita una **ficha de costo** que constituye un instrumento de control que influye en la toma de decisiones y por el cálculo de **indicadores**: **valor actual neto** (VAN) y la **tasa interna de retorno** (TIR) para ello es indispensable el uso de la ficha de costo.

Con los procesos mencionados anteriormente se realiza un **análisis de factibilidad económica**, que permita determinar si el proyecto es o no factible y/o viable.

3.2. Requisitos

Los requerimientos o requisitos son la pieza fundamental en un proyecto de desarrollo de software, pues marcan el punto de partida para actividades como la planeación, básicamente en lo que se refiere a las estimaciones de tiempos y costos, así como la definición de recursos necesarios y la elaboración de cronogramas que será uno de los principales mecanismos de control con los que se contará durante la etapa de desarrollo. Además la especificación de requerimientos es la base que permite verificar si se alcanzaron o no los objetivos establecidos en el proyecto ya que estos son un reflejo detallado de las necesidades de los clientes o usuarios del sistema y es contra lo que se va a estar verificando si se están cumpliendo las metas trazadas.

Los requerimientos pueden dividirse en requerimientos funcionales y requerimientos no funcionales.

- Los requerimientos funcionales: definen las funciones que el sistema será capaz de realizar. Describen las transformaciones que el sistema realiza sobre las entradas para producir salidas.
- Los requerimientos no funcionales: tienen que ver con características que de una u otra forma puedan limitar el sistema, como por ejemplo, el rendimiento (en tiempo y espacio), interfaces de usuario, fiabilidad (robustez del sistema, disponibilidad de equipo), mantenimiento, seguridad, portabilidad y estándares.

3.2.1. Requisitos Funcionales

RF1: Crear Proyecto: Permite crear un nuevo proyecto al cual se le realizará el estudio de la factibilidad.

RF2: Abrir Proyecto: Permite abrir un proyecto determinado.

RF3: Guardar Proyecto: Permite guardar un proyecto determinado.

RF4: Generar Reporte: Permite exportar la ficha de costo o el flujo de caja a un determinado formato.

RF3.1: Exportar a PDF.

RF3.2: Exportar a Excel.

RF5: Administrar Flujo de Caja: Permite adicionar y eliminar flujo de caja.

RF4.1: Adicionar Flujo de Caja.

RF4.2: Eliminar Flujo de Caja.

RF6: Gestionar datos del Flujo de Caja: Permite adicionar, eliminar y modificar los datos del flujo de caja.

RF5.1: Adicionar datos Flujo de Caja.

RF5.2: Eliminar datos Flujo de Caja.

RF5.3: Modificar datos Flujo de Caja.

RF7: Calcular Indicadores: Permite realizar el cálculo de los indicadores de evaluación financieros, Valor Actual Neto (VAN) y Taza Interna de Retorno (TIR).

RF6.1: Calcular VAN.

RF6.2: Calcular TIR.

RF8: Realizar copias de Seguridad: Permite realizar copias de seguridad al sistema encriptando el archivo de la bases de dato a utilizar.

RF9: Modelar gráfica: Permite realizar gráficas del flujo de caja por año.

RF8.1: Realizar gráficos del flujo de caja por año.

3.2.2. Requisitos No Funcionales

RNF1: Requisitos de Seguridad

El sistema deberá contar con un grupo de políticas de accesibilidad a las diferentes funcionalidades del mismo en dependencia del nivel de autorización que presente determinado usuario, en este caso el responsable del área económica del centro.

- La persona encargada de la parte económica, tendrá una contraseña única de cada proyecto para entrar al sistema.
- Se han creado archivos binarios que permitirá la seguridad del sistema. Los archivos que son creados por el gestor de base de dato utilizado se guardarán encriptados para posibilitar mayor seguridad referente a la información de cada proyecto.

RNF2: Requisitos de Fiabilidad

El sistema deberá permitir la recuperación de datos, así como la recuperación frente a fallos del sistema, debe posibilitar reiniciar el sistema, realizando copias de seguridad con determinada frecuencia.

RNF3: Requisitos de Usabilidad

El sistema deberá ser de uso sencillo, contar con una interfaz sugerente sin cúmulo de imágenes que distraigan al cliente del objetivo de su empleo. Deberá brindar además, facilidades que permitan interactuar con el sistema de forma rápida. Permitiendo que personas con pocas habilidades informáticas manejen el sistema.

RNF4: Hardware

El sistema debe de contener un microprocesador de 2.20GHz y la memoria RAM de 1.00 GB. Estas características antes mencionadas deben ser iguales o superiores a las propiedades donde se realizará el despliegue del sistema.

3.3. Descripción del Sistema. Modelos de Casos de Uso del Sistema

Tras haber realizado una pequeña descripción del sistema, se da paso a la descripción del modelo de casos de uso del sistema haciendo uso de las ventajas que brinda el lenguaje de modelado UML, se formulan las funcionalidades del sistema y representación mediante un diagrama, para ello es de vital importancia definir los actores y los casos de uso que representarán las responsabilidades del mismo.

3.3.1. Determinación y justificación de los actores del sistema

Un actor es cualquier individuo con los que el sistema interactúa. Lo que se modela como actor, es el rol que se juega cuando se interactúa con el sistema para beneficiarse de sus resultados (Jacobson, y otros, 2000). En el sistema a realizar se ve reflejado los dos casos, la explicación se describe a continuación.

Actor	Justificación
Usuario	Representa a la persona (personal económico) que interactúa con el sistema.

Tabla 4: Actor del Sistema. Funcionalidad.

3.3.2. Casos de Uso del Sistema

Un caso de uso constituye una técnica utilizada para describir el comportamiento del sistema, a través de un documento narrativo que define la secuencia de acciones que obtienen resultados de valor para un

actor que utiliza un sistema para completar un proceso, sin importar los detalles de la implementación (Arias, 2006).

Prioridad de casos de usos críticos.

Caso de Uso del Sistema	Prioridad
Administrar Flujo de Caja.	Crítica
Gestionar datos de Flujo de Caja.	Crítica
Calcular los Indicadores.	Crítica

Tabla 5: Prioridad de los Casos de Uso del Sistema.

3.3.3. Diagrama de Casos de Uso del Sistema

Los diagramas de casos de uso sirven para especificar la comunicación y el comportamiento de un sistema mediante su interacción con los usuarios y/u otros sistemas; o lo que es igual, un diagrama que muestra la relación entre los actores y los casos de uso en un sistema. Los diagramas de casos de uso se utilizan para ilustrar los requerimientos del sistema al mostrar cómo reacciona una respuesta a eventos que se producen en el mismo (Jacobson, y otros, 2000).

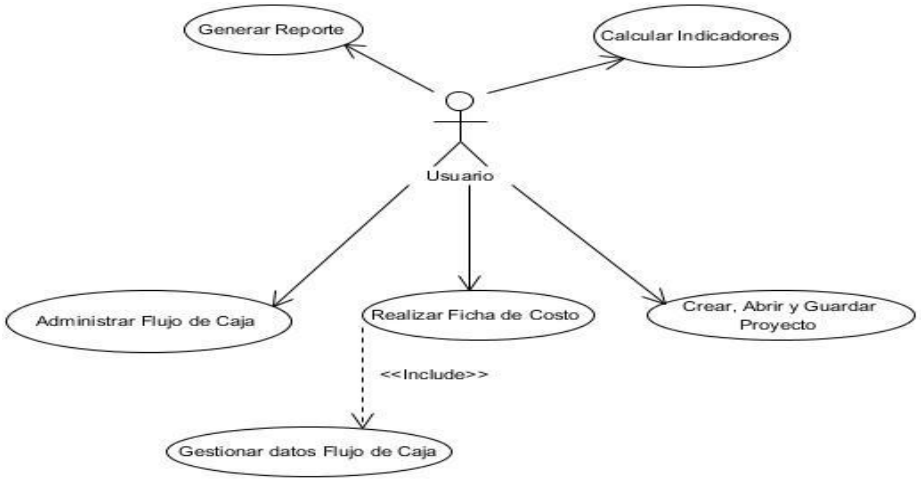

Figura 2 Diagrama de Caso de Uso del Sistema.

3.3.4. Expansión de los Casos de Uso

Se muestra a continuación la descripción para el Caso de Uso Administrar Flujo de Caja por la importancia que representa para comprender el funcionamiento del sistema, las descripciones correspondientes al resto de los casos de usos se pueden consultar en los **(Anexos 1).**

Descripción del Caso de Uso. Administrar Flujo de Caja.

Objetivo	Administrar Flujo de Caja.	
Actores	Usuario:(Inicia).	
Resumen	El caso de uso inicia cuando el Usuario crea o abre un proyecto, termina cuando se realiza el cálculo del flujo de caja.	
Complejidad	Alta	
Prioridad	Crítico	
Precondiciones	Se debe tener abierto el proyecto al cual se le administrará el flujo de caja.	
Postcondiciones	Se adiciona o elimina de forma satisfactoria.	
Flujo de eventos		
Flujo básico: Administrar Flujo de Caja.		
	Actor	**Sistema**
1.	Selecciona la opción “Archivo” (A) en la barra de menú de la ventana principal.	
2.		Muestra las siguientes opciones: • Crear Proyecto • Abrir Proyecto • Guardar Proyecto • Salir
3.	Puede seleccionar las opciones Crear o Abrir Proyecto siempre y cuando tenga la contraseña del proyecto a abrir.	

4.		Habilita las opciones de Administrar Flujo de Caja. • Adicionar Flujo de Caja (B). Ver Sección 1: "Adicionar Flujo de Caja". • Eliminar Flujo de Caja (C). Ver Sección 2: "Eliminar Flujo de Caja. Termina caso de uso.
Sección 1:"Adicionar Flujo de Caja"		
Flujo Básico: Adicionar Flujo de Caja.		
	Actor	**Sistema**
4.	Selecciona la opción Adicionar Flujo de Caja (B).	
5.		Muestra el "Nuevo Flujo de Caja" a seleccionar.
6.	Selecciona el año y el mes, o el año completo (D) perteneciente al nuevo flujo de caja y oprime el botón "Aceptar" (E).	
7.		Se adiciona el Nuevo Flujo de Caja guardándose en la Base de Datos que contiene el sistema, mostrándose un mensaje "Se adicionó el nuevo flujo de caja". Termina el caso de uso.

Sección 2: "Eliminar Flujo de Caja"			
Flujo Básico: Eliminar Flujo de Caja.			
	Actor		**Sistema**
4.	Selecciona el Flujo de Caja que desea eliminar eligiendo la fila que desea eliminar de ese Flujo de Caja y la opción eliminar (F).		
5.			Busca en la Base de Datos todo lo relacionado con el Flujo de Caja seleccionado y lo elimina. Termina el caso de uso.
Relaciones	**CU Incluidos**	Gestionar datos del Flujo de Caja. Ver CU Gestionar datos del Flujo de Caja.	
	CU Extendidos	Ninguno.	
Requisitos no funcionales	Ninguno.		
Asuntos pendientes	No procede.		

Tabla 6: Descripción del Caso de Uso Administrar Flujo de Caja.

3.4. Patrones

A grandes rasgos se puede decir que un patrón es un modelo a seguir para realizar una actividad en específico. Estos surgen con la experiencia de los seres humanos de tratar de lograr ciertos objetivos. Por lo tanto capturan las experiencias existentes y probadas para promover buenas prácticas. Los patrones:

- Son una abstracción de "problema–solución".
- Se ocupan de problemas recurrentes.
- Identifican y especifican abstracciones de niveles más altos que componentes o clases individuales.
- Proporcionan vocabulario y entendimiento común.

3.4.1. Patrones de Diseño

Los patrones de diseño son la base para la búsqueda de soluciones a problemas comunes en el desarrollo de software y otros ámbitos referentes al diseño de interacción o interfaces. Un patrón de diseño resulta ser una solución a un problema de diseño. Para que una solución sea considerada un patrón debe poseer ciertas características. Una de ellas es que debe haber comprobado su efectividad resolviendo problemas similares en ocasiones anteriores. Otra es que debe ser reutilizable, lo que significa que es aplicable a diferentes problemas de diseño en distintas circunstancias.

Para lograr un diseño eficiente de la aplicación se tuvieron en cuenta varios patrones de diseño, que una vez aplicados estos patrones los diseños se hicieron más flexibles, modulares y reutilizables. Para la asignación de responsabilidades se tuvieron en cuenta algunos de los Patrones GRASP. Según Jacobson la responsabilidad es un "contrato u obligación de un tipo de clase" (Jacobson, y otros, 2000). Los patrones de asignación de responsabilidades usados para el desarrollo de la aplicación fueron:

- Experto: Este patrón, está diseñado para que la responsabilidad de realizar una función determinada sea de la clase que tiene o puede tener la información requerida. Está presente en la clase controladora *CProyecto* y en la clase *CFlujoCaja* de manera que cada responsabilidad está asignada a la clase que cuenta con la información necesaria para realizarla.

- Creador: Está diseñado para asignar responsabilidades relacionadas con la creación de objetos. Su intención es encontrar un creador que necesite conectarse al objeto creado. Está presente en la clase *CProyecto* que es el encargado de crear instancias de la clase *CFlujoCaja*.

- Controlador: Está diseñado para asignar la responsabilidad de controlar el flujo de eventos del sistema, a clases específicas, esto facilita la centralización de actividades. Presente en la clase *CProyecto*, que es la encargada de atender los eventos del sistema que son generados por un actor externo y se asocian a operaciones del sistema.

- Alta Cohesión: Este patrón se encarga de que las clases del diseño cumplan con las tareas que tienen definida, aplicando atributos y métodos de manera sencilla para implementar dichas tareas. Presente en las clases *CProyecto, CFlujoCaja*.

- Bajo Acoplamiento: Este patrón se encarga de que las clases del diseño colaboren unas con otras, siempre y cuando esta colaboración se mantenga en un mínimo aceptable, reduciendo el impacto de posibles cambios en la herramienta. Presentes en todas las clases del sistema.

3.4.2. Patrones de Arquitectura

Arquitectura Modelo Vista Controlador (MVC)

Es un estilo de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos. Esta arquitectura es ventajosa ya que se le pueden aplicar opciones como el multilenguaje, distintos diseños de presentación sin alterar la lógica de negocio. La separación de capas como presentación, lógica de negocio, acceso a datos es fundamental para el desarrollo de arquitecturas consistentes, reutilizables y más fácilmente mantenibles, lo que al final resulta en un ahorro de tiempo en futuras modificaciones. Esta arquitectura está formada por 3 capas:

El **Modelo** es el objeto que representa, maneja y controla los datos así como todas las transformaciones que se realicen, este no tiene conocimiento específico de los controladores o de las vistas, ni siquiera contiene referencias a ellos, es el propio sistema el que tiene encomendada la responsabilidad de mantener los enlaces, y notificar a las vistas cuando cambia el modelo. El modelo en el sistema a desarrollar son las entidades en la base de datos SQLite.

La **Vista** es el objeto que maneja la presentación visual de los datos representados por el modelo, generando su representación visual y mostrando los datos al usuario. Interactúa con el modelo a través de una referencia al propio modelo. La capa vista en el sistema serían las interfaces.

El **Controlador** es el objeto que proporciona significado a las órdenes del usuario, actuando sobre los datos representados por el modelo, cuando se realiza algún cambio, entra en acción, bien sea por cambios en la información del modelo o por alteraciones de la vista. El controlador interactúa con el modelo a través de una referencia al propio Modelo. En el sistema esta capa es representada específicamente en la clase *CProyecto* que es la que contiene y administra los flujos de caja almacenados en el modelo.

3.4.3. Arquitectura Final

Para el desarrollo de la arquitectura del sistema que permita el estudio de la factibilidad económica se utiliza los patrones antes mencionado. La Arquitectura Modelo Vista Controlador se aplica ya que por un

lado se definen los componentes para la representación de la información, y por otro para la interacción del usuario, además de facilitar la tarea del desarrollo de la aplicación y su posterior mantenimiento.

3.5. Modelo del Diseño

Es un modelo de objeto que describe la realización física de los casos de uso, centrándose en como los requisito funcionales y no funcionales, junto con otras restricciones relacionadas con el entorno de implementación, tienen impacto en el sistema a considerar, constituyendo la principal actividad para el proceso de implementación.

RUP propone que el artefacto Modelo de Diseño contenga una introducción, que no es más que una descripción textual que sirve como breve introducción al modelo; paquetes y subsistemas de diseño; contiene también diagramas, específicamente los diagramas de clases del diseño y diagramas de interacción del diseño, estos últimos también llamados realización de casos de uso; además contiene clases, interfaces y relaciones contenidas en los paquetes.

3.5.1. Clases de Diseño

Son una abstracción de una clase o construcción similar en la implementación del sistema. El lenguaje utilizado en una clase de diseño es el mismo lenguaje de programación. Se especifican la visibilidad de los atributos y las operaciones.

La relación entre las clases generalmente tiene un significado directo cuando la clase es implementada, los métodos tienen correspondencia directa con los métodos utilizados en la implementación. Puede posponer el manejo de algunos requisitos para las subsiguientes actividades de implementación, indicándolos como requisitos de implementación de la clase.

Generalmente aparece con un estereotipo sin costura que se corresponde con una construcción en el lenguaje de programación dado. Puede realizar y proporcionar interfaces si es necesario en el lenguaje de programación. Puede activarse indicando que objetos de la clase mantengan su propio hilo de control y se ejecuten concurrentemente con otros objetos activos (Jacobson, y otros, 2000).

3.5.2. Diagrama de Clases del Diseño

Es una representación más concreta que el diagrama de clases del análisis. Representa la parte estática del sistema, así como las clases y sus relaciones.

Los diagramas de clases también son la base para los diagramas de componentes, son importantes no sólo para visualizar, especificar y documentar modelos estructurales, sino también para construir sistemas

ejecutables, aplicando ingeniería directa e inversa. A continuación se presenta el diagrama de clase por paquetes según la arquitectura aplicada en el sistema, ver **(Anexo 3)** diagrama de clase más general.

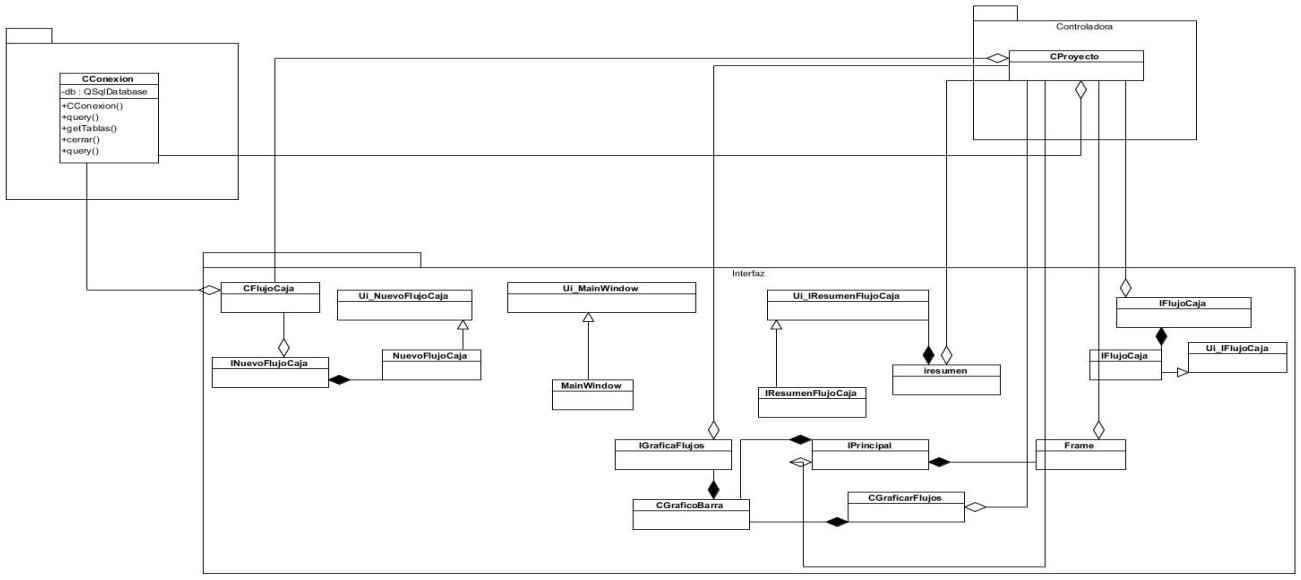

Figura 3: Diagrama de Clases del Diseño.

La clase *CProyecto* es la clase controladora, la misma contiene los flujos de caja que se realizan, el cálculo de los indicadores VAN y TIR son funcionalidades implementadas en esta clase ambas claves para el desarrollo del sistema. *CFlujoCaja* es la clase persistente, pues contiene toda la información de los flujos de cajas que se realizan. Las clases interfaces (*IPrincipal. IResumen, IFlujoCaja, IGraficarBarra, INuevoFlujoCaja*) son de uso sencillo y son las responsables de hacerle llegar al usuario la información requerida. La clase *CConexión* es la encargada de la conexión a la base de datos, con el propósito de acceder a los datos que se necesiten de acuerdo a la circunstancias.

3.6. Diagramas de Interacción

Los diagramas de interacción muestran una interacción y valga la redundancia, que consiste de un conjunto de objetos y sus relaciones, incluyendo los mensajes que puedan ser realizados entre ellos. Son importantes para modelar los aspectos dinámicos de un sistema y para construir sistemas ejecutables a través de ingeniería hacia adelante e ingeniería inversa. Comúnmente contienen:

- Objetos.
- Enlaces.
- Mensajes.

Pueden servir para visualizar, especificar, construir y documentar los aspectos dinámicos de una sociedad particular de objetos, o pueden ser usados para modelar un flujo particular de control de un caso de uso. Los diagramas de interacción están conformados por los diagramas de colaboración y los diagramas de secuencia.

3.6.1. Diagrama de Colaboración

Los diagramas de colaboración destacan la organización de los objetos que participan en una interacción. Un diagrama de colaboración se construye colocando en primer lugar los objetos que participan en la colaboración como nodos del grafo. A continuación se representan los enlaces que conectan esos objetos como arcos del grafo. Por último, estos enlaces se adornan con los mensajes que envían y reciben los objetos. Los diagramas de colaboración tienen dos características que los distinguen de los diagramas de secuencia.

- En primer lugar, el camino. Para indicar cómo se enlaza un objeto a otro, se puede asociar un estereotipo de camino al extremo más lejano de un enlace.
- En segundo lugar, está el número de secuencia. Para indicar la ordenación temporal de un mensaje, se precede de un número (comenzando con el mensaje número 1), que se incrementa secuencialmente por cada nuevo mensaje en el flujo de control.

A continuación se muestra el diagrama de la Sección 1 Adicionar Flujo de Caja del Caso de Uso (CU) Administrar Flujo de Caja. Para ver el resto de los diagramas dirigirse a la Sección (**Anexos 4)**.

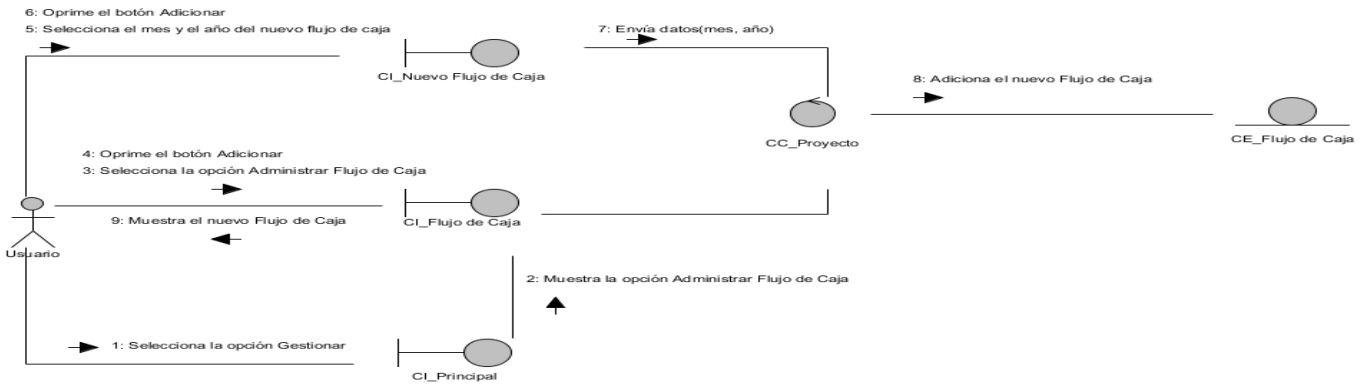

Figura 4: Diagrama de Colaboración. CU Administrar Flujo de Caja: Sección 1 Adicionar Flujo de Caja.

3.6.2. Diagrama de Secuencia

Los diagramas de secuencia destacan el orden temporal de los mensajes. Un diagrama de secuencia se forma colocando en primer lugar los objetos que participan en la interacción en la parte superior del diagrama, a lo largo del eje X. Normalmente, se coloca a la izquierda el objeto que inicia la interacción, y los objetos subordinados a la derecha. A continuación, se colocan los mensajes que estos objetos envían y reciben a lo largo del eje Y, en orden de sucesión en el tiempo, desde arriba hasta abajo. Esto ofrece al lector una señal visual clara del flujo de control a lo largo del tiempo.

A continuación se muestra el diagrama de la Sección 1 Adicionar Flujo de Caja del Caso de Uso (CU) Administrar Flujo de Caja. Para ver el resto de los diagramas dirigirse a los **(Anexos 5)**.

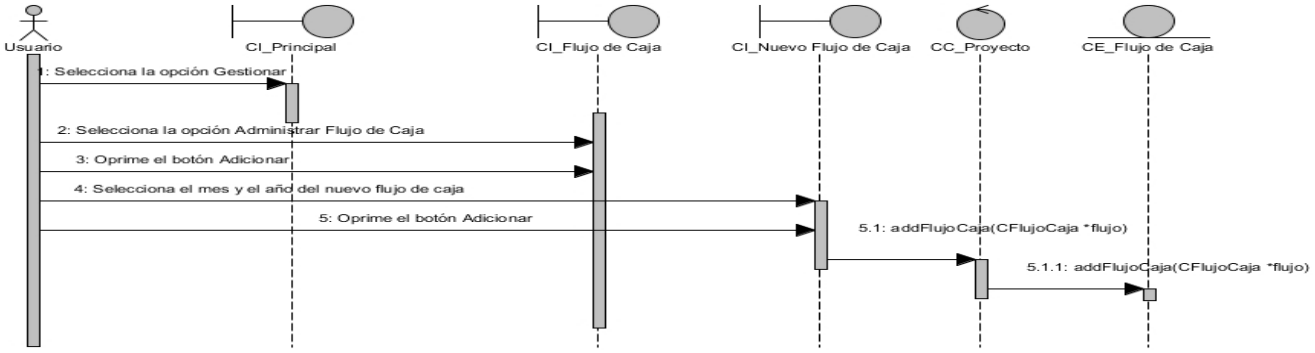

Figura 5: Diagrama de Secuencia. CU Administrar Flujo de Caja. Sección 1 Adicionar Flujo de Caja.

3.7. Conclusiones Parciales

Con la realización de este capítulo se desarrolló el análisis de la solución describiendo los procesos realizados y documentando los artefactos definidos por la metodología de desarrollo seleccionada; obteniéndose como resultado los diagramas de clases, de colaboración, secuencia y la descripción de los casos de uso arquitectónicamente significativos. Se concluye además que una vez definidos los requisitos funcionales y no funcionales que posee una visión general de lo que debe hacer y poseer un sistema. Por otra parte el modelo de diseño es la base fundamental para la implementación, ya que se enfoca en cómo va a estar estructurado e implementado el software. La aplicación de patrones de diseño garantizó una mayor organización a la hora de implementar la solución y permitió fomentar la reutilización. En este contexto se utilizaron para identificar las clases e instancias participantes, sus roles y colaboraciones y la distribución de responsabilidades.

CAPÍTULO 4: IMPLEMENTACIÓN Y PRUEBA

Introducción

En el presente capítulo se realiza la implementación de las clases y objetos correspondientes a la solución propuesta en capítulos anteriores. Se describe el modelo de implementación, despliegue y las pruebas realizadas para evaluar la eficiencia y calidad del producto desarrollado.

4.1. Modelo de Implementación

El Modelo de Implementación es comprendido por un conjunto de componentes y subsistemas que constituyen la composición física de la implementación del sistema, entre los componentes se pueden encontrar datos, archivos, ejecutables, código fuente y los directorios. El modelo de implementación comienza con los resultados obtenidos en el diseño y se crean los componentes físicos de la aplicación que se traducen en ficheros de código fuente .cpp y .h debido a su implementación en el lenguaje C++ (Letelier, 2006).

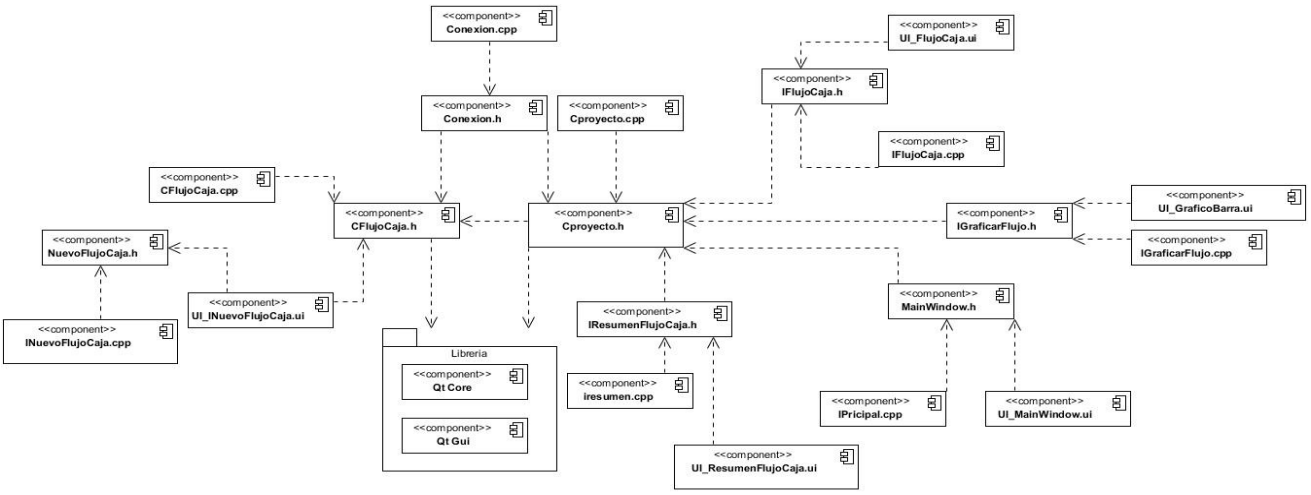

Figura 6: Diagrama de Componente.

4.2. Modelo de Datos

El Modelo de datos describe de forma abstracta las entidades y sus características, además de las relaciones entre estas dentro de la base de datos. Las entidades son objetos que guardan información necesaria para el sistema. Su símbolo es un rectángulo. Los atributos son características de una entidad, se representan colocando su nombre dentro del rectángulo de la entidad. Los atributos se clasifican en: obligatorios, opcionales, claves foráneas y claves primarias (estas se dividen en simples y compuestas). Gráficamente la clave primaria se representa con un signo de suma y la foránea con un símbolo de número. Las relaciones muestran la asociación entre dos entidades, representadas por una línea que une a las entidades involucradas (Escoba, y otros, 2010).

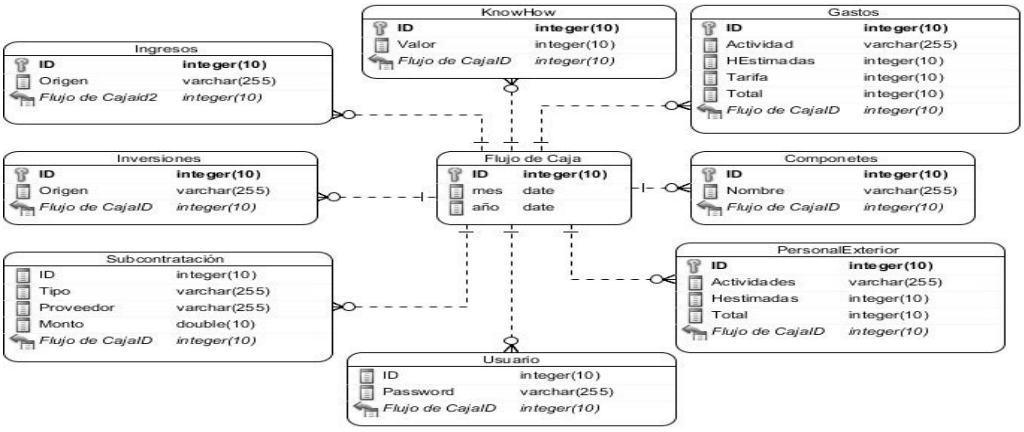

Figura 7: Modelo Entidad Relación.

4.3. Modelo de Despliegue

En todo sistema software es necesario describir la distribución física de los elementos que lo componen, mostrando la configuración del hardware en la que se desplegará el sistema, identificando los nodos, que pueden ser procesadores o dispositivos de hardware (Zambreno, y otros, 2012). El Diagrama de Despliegue muestra la disposición física de los distintos nodos que componen un sistema y el reparto de los componentes sobre dichos nodos. En el caso del sistema en cuestión las relaciones físicas finales entre los componentes de hardware y software son representados a través de dos nodos, los cuales serían una computadora (PC) y una Impresora. La conexión de la impresora con la PC puede ser por el puerto USB que contiene la PC o red inalámbrica.

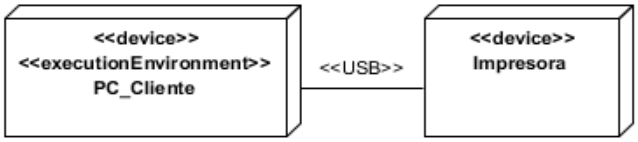

Figura 8: Diagrama de Despliegue.

4.4. Pruebas al Sistema Propuesto

Pruebas del Software

Las pruebas de software son un elemento crítico para la garantía de la calidad de software y representan una revisión final de las especificaciones, del diseño y de la codificación (Pressman, 2012). Una vez finalizado el desarrollo de un sistema se procede a realizarle diferentes tipos de pruebas en función de garantizar que cumpla con los requisitos planteados por los clientes y que sean corregidos los posibles errores antes de ser desplegado.

4.4.1. Métodos de Pruebas

Con el objetivo de encontrar la mayor cantidad de errores posibles en un producto de software se han definido diferentes tipos de pruebas que responden a puntos de vistas diferentes. Para la validación de la solución propuesta se utiliza el método de Caja Negra, el mismo responde a que una vez conocida la funcionalidad específica para la que fue diseñado el producto, se puedan llevar a cabo pruebas que demuestren que cada funcionalidad es operativa y se conoce como Prueba de Caja Negra (Pressman, 2012). Los métodos de prueba de software dictan de qué manera probar, pero no qué es lo que se debe probar.

Pruebas de Caja Negra

Las pruebas de Caja Negra, se centran en los requisitos funcionales del software con el objetivo de garantizar que el funcionamiento es completamente operativo. Las pruebas de Caja Negra permiten realizar un estudio desde el punto de vista de las entradas que recibe y las salidas o respuesta que produce, sin tener en cuenta el trabajo interno que realiza (Pressman, 2012).

- **Método Partición Equivalente**

Es un método de caja negra que divide el campo de entrada de un programa en clases de datos de los que se pueden derivar casos de pruebas. El diseño de caso de pruebas para la partición equivalente se

basa en una evaluación a determinado conjunto de estados válidos o no válidos para condiciones de entradas (Pressman, 2012).

- **Caso de Prueba**

Un caso de prueba específica como probar un caso de uso o un escenario específico de un caso de uso. Incluye la verificación del resultado de la interacción entre los actores y el sistema, que se satisfacen las precondiciones y poscondiciones especificadas por el caso de uso y sigue las acciones especificadas en el mismo (Jacobson, y otros, 2000).

A continuación se presenta el caso de prueba correspondiente al CU Administrar Flujo de Caja, los restantes se pueden encontrar en los **(Anexos 2)** de la presente investigación.

Sesión	**Escenario**	**Acción Realizada**	**Reacción del Sistema**	**Resultados Satisfactorios**
SC1 Administrar Flujo de Caja	EC 1.1 Se adiciona satisfactoriamente el flujo de caja.	Se escoge el botón adicionar, se muestra una interfaz, se escoge la fecha del flujo de caja deseado y para guardar el nuevo flujo de caja se oprime el botón Aceptar.	El sistema guarda el flujo de caja adicionado.	Satisfactorio.
	EC 1.2 No se adiciona satisfactoriamente el flujo de caja.	Se escoge el botón adicionar, se muestra una interfaz, se escoge la fecha del flujo de caja deseado y para guardar el nuevo flujo de caja se oprime el botón Aceptar.	El sistema no guarda el flujo de caja adicionado.	Satisfactorio.
	EC 2.1 Se elimina satisfactoriamente el flujo de caja.	Se escoge del menú donde se encuentran los flujos de caja el flujo de caja que sea desea eliminar, luego se selecciona el botón eliminar.	El sistema elimina el flujo de caja seleccionado.	Satisfactorio

	EC 2.2 Se elimina satisfactoriamente el flujo de caja.	Se escoge del menú donde se encuentran los flujos de caja el flujo de caja que sea desea eliminar, luego se selecciona el botón eliminar.	El sistema no elimina el flujo de caja seleccionado.	Satisfactorio

Tabla 7: Caso de Prueba. CU Administrar Flujo de Caja.

Durante las pruebas de caja negras se realizaron dos iteraciones, una primera iteración donde se detectaron tres inconformidades, dos de ellas en el caso de uso Gestionar Datos de Flujo de Caja donde existían valores que no se guardaban en la base de datos y en ocasiones no se agregaban entradas a la tabla perteneciente al factor gastos. Otra de las no conformidades fue encontrada en el caso de uso Calcular VAN donde el valor introducido en la tasa de descuento admitía caracteres extraños. En una segunda iteración los resultados mostrados fueron satisfactorios. La gráfica presentada a continuación muestra las dos iteraciones realizadas al sistema.

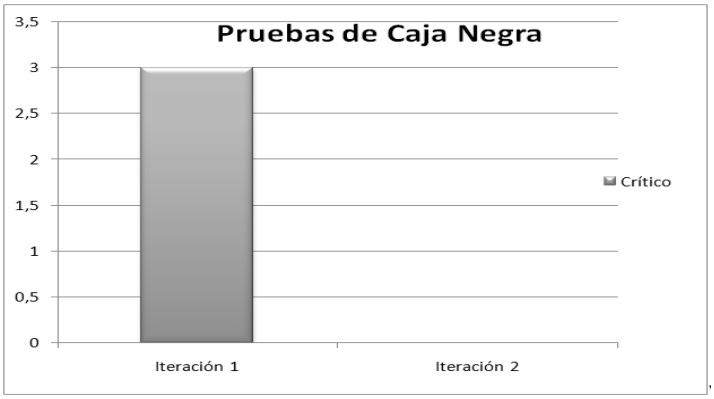

Figura 9: Representación de las Iteraciones de las Pruebas de Caja Negra.

Pruebas de Aceptación

Las pruebas de aceptación tienen como objetivo la evaluación del producto y la revisión de la documentación final en conjunto con el cliente. Estas pruebas no se realizan durante la etapa de desarrollo, el objetivo es que el cliente pruebe el sistema. En las pruebas de aceptación el cliente puede aceptar lo implementado o realizar pedidos de cambios. Durante la realización de las pruebas de aceptación al sistema Proyecto de Factibilidad en GEYSED, se detectó en una primera iteración que el sistema no validaba de forma correcta los datos requeridos para la realización del flujo de caja y permitía

la entrada de valores alfanuméricos en el campo de texto perteneciente a los valores de gastos, ingresos e inversiones, además permitía la entrada de caracteres extraños en el campo de texto perteneciente al Know-How. Otros de los cambios solicitados por el cliente fue que el sistema en un inicio no contaba con los permisos necesarios para el acceso a los diferentes proyectos creados.

Estas inconformidades fueron corregidas y luego se volvió a realizar otra iteración de las pruebas para el caso de uso Gestionar Datos de Flujo de Caja así como para el caso de uso Crear, Abrir y Guardar un proyecto, arrojando resultados satisfactorios. La gráfica presentada a continuación muestra las dos iteraciones realizadas al sistema.

Figura 10: Representación de las Iteraciones de las Pruebas de Aceptación del Sistema.

4.5. Conclusiones Parciales

En este capítulo se realizó una descripción de la implementación y las pruebas realizadas al sistema. En el mismo se generaron los artefactos necesarios para la implementación y las pruebas del sistema, por lo que los diagramas de componentes permitieron comprender la interacción entre los componentes que conforman el sistema y mostrar las dependencias que existen entre los mismos. Mediante las pruebas de caja negra seleccionadas para validar el cumplimiento de los requisitos funcionales establecidos en el análisis del sistema, se lograron encontrar problemas funcionales presentes en el sistema, con los resultados satisfactorios en la iteración de prueba se puede concluir que el software no presenta ningún error funcional, por lo que su etapa de desarrollo fue totalmente eficiente.

CONCLUSIONES GENERALES

En este punto se consideran cumplidos los objetivos trazados al tener desarrollado el sistema para el estudio de la factibilidad económica. El mismo cumple todos los objetivos planteados por lo que se convierte en un sistema cuyas funcionalidades básicas muestran los resultados esperados.

Se creó además una documentación relacionada con el sistema, la cual ayudará al cliente en su uso, en la misma se da una explicación de cada proceso implementado para lograr una mayor sencillez en el trabajo con el sistema desarrollado.

Esta investigación representa un aporte importante al área económica del centro GEYSED, ya que permite determinar si un proyecto será o no factible contribuyendo de esta forma a la toma de decisiones, puede ser utilizado además por otros centros de desarrollo lo cual sería satisfactorio ya que impediría la pérdida de tiempo pues se deja a un lado el trabajo de forma manual, además de que evita la dependencia de software privativos que realicen estas operaciones.

Se cuenta entonces con un sistema flexible, seguro, que por su diseño y estar desarrollado en un sistema multiplataforma permite su modificación para adaptarlo a las futuras necesidades de la universidad.

RECOMENDACIONES

Se recomienda:

- Analizar qué otras funcionalidades se le pueden ofrecer a los usuarios para apoyar el estudio de la factibilidad en el centro.
- Tener en cuenta este trabajo para proyectos futuros, siempre y cuando se cumplan con las normas de confidencialidad establecidas.
- Valorar la experiencia del equipo de desarrollo para la implementación de un producto similar a nivel Nacional.

REFERENCIA BIBLIOGRÁFICA

Almenares, Kiosmy y De León, Rubén. 2007. *Herramientas Case.* 2007.

Analysis, Financial. 2007. Los avances en la Ciencia de la Administración Computacional. [En línea] 2007. [Citado el: 25 de 11 de 2011.] http://www.springerlink.com. 978-3-540-36626-3.

Analyzer, Managerial. 2012. Managerial Analyzer. *Managerial Analyzer.* [En línea] 2012. [Citado el: 15 de 11 de 2012.] http://www.managerialanalyzer.com.

Arias, Fidias G. 2006. *El Proyecto de Investigación: Introducción a la Metodología Científica.* Caracas - Venezuela : Episteme, 2006.

Arias, Michael. 2006. *La ingeniería de requerimientos y su importancia en el desarrollo de proyectos de software.* 2006.

Beck, Kent. 2000. *Extreme Programming Explained.* Madrid : Addison Wesley, 2000. ISBN 201616416..

Blanco, Carlos. 2012. *IDE de Desarrollo.* 2012.

Bonilla, Juan Carlos Leiva. 2007. *Emprendedores y la creación de empresas. .* Costa Rica : Tecnología de Costa Rica, 2007. ISBN 9977661944, 9789977661940.

Brigham, Eugene y Besley, Scott. 2008. *Fundamentos de administración financiera .* s.l. : Cengage Learning , 2008. ISBN 9708300144, 9789708300148.

Bruce, Colin. 1982. *Estudio de la factibilidad de un proyecto de inversion: etapas en su estudio.* 1982.

Canós, José H, Letelier, Patricio y Penadés, María Cramen. 2002. *Métodologías Ágiles en el Desarrollo de Software.* Valencia. España : Universidad Politécnica de Valencia, 2002.

Domínguez, Danilo. 2008. Introducción a la base de datos. [En línea] 2008. [Citado el: 20 de 12 de 2012.] http://www.programacion.net/tutorial/sql.

Escoba, Claudia y Pérez, Joselín. 2010. *Desarrollo de las consultas de cristalino.* La Habana : s.n., 2010.

EvalAs. 2012. Evalas. *Evalas.* [En línea] 2012. [Citado el: 15 de 11 de 2012.] http://www.elsitioagricola.com. 506866..

Fernández, Saúl. 2007. *Los proyectos de inversión.* 2007. ISBN 9977661855.

Flores, Maria. 2003. *Manual de Tesis de Grado y Especialización y Maestría y Tesis Doctorales de la Universidad Pedagógica Libertador.* 2003.

Gaceta Oficial. 2011. *Gaceta Oficial de la República de Cuba. Ministerio de Justicia. Cuba.* La Habana. Cuba : s.n., 2011.

Gestión de Proyectos. 2012. *Procedimiento para la estimación de costos de los proyectos de la UCI. Dirección de Proyectos, Universidad de las Ciencias Informáticas.* 2012.

Guzmán, Douglas Humberto. 2011. *Intregración de OPENGL en QT.* 2011.

Jacobson, Ivar, Booch, Grady y Rumbaugh, James. 2000. *El Proceso Unificado de Desarrollo de Software.* Madrid : Addison Wesley : s.n., 2000. ISBN: 84-7829-036-2..

Jácome, Rocha y Hernando, William. 2012. *La construcción del flujo de caja de un proyecto de inversión.* 2012.

Jiménez, Francisco Javier, Espinoza Gutiérrez, Carlos Luis y Fonseca Renata, Leonel. 2007. *Ingeniería Económica.* Costa Rica : Tecnología de Costa Rica, 2007. ISBN 9789977661889.

Kendall. 2010. Buenas Prátcicas. [En línea] 2010. [Citado el: 5 de 11 de 2012.] http://www.buenaspracticas.com.

Kruchten, Philippe. 2004. *The Rational Unified Process.* s.l. : Addison-Wesley Professional, 2004. ISBN: 0321197704, 9780321197702.

Larman, Craig. 2000. *Larman, Craig. UML y PatronesLarman, Craig. UML y Patrones UML y Patrones. Introducción al análisis y diseño orientado a objetos.* México : México : PRENTICE HALL, 2000. ISBN: 970-17-0261-1.

Letelier, Patricio. 2006. Rational Unified Process (RUP). *Rational Unified Process (RUP).* [En línea] 2006. https://pid.dsic.upv.es.

Martínez, León Alberto. 2011. *Implementación de un editor gráfico de circuitos eléctricos con Qt.* Cartagena : s.n., 2011.

Méndez, Silvestre José. 2009. *Fundamentos de Economía.* 2009.

Metodologías. 2010. Metodologías Ágiles. [En línea] 2010. http://es.scribd.com/doc/84327749/Metodologias-Agiles.

Miniño, Fabio Herrera. 1994. *Fundamentos de análisis económico: Guía para investigación y extensión rural.* 1994. ISBN 9977571783.

Ministerio de Economía. 2007. Departamento Corporativo. Ministerio de Economía, Fomento y Turismo. Chile. [En línea] 2007. [Citado el: 5 de 11 de 2012.] http://www.decoop.cl.

Paul, Samuel y Willian, Nordhaus. 2002. *Economía.* 2002. ISBN 9788448136321.

Peña, J. 2007. *Planeación del Efectivo.* s.l. : Universidad Autónoma De Santo Domingo, 2007.

Pressman, Roger S. 2012. Software Engineering. A partitioner´s Approach. 2012.

Rodríguez, Gonzalo M. 2006. *LA EVALUACIÓN FINANCIERA Y SOCIAL DE PROYECTOS DE INVERSIÓN.* 2006.

Sánchez, Jordi. 2006. Framework de desarrollo. [En línea] 2006. [Citado el: 4 de 11 de 2012.] http://jordisan.net/blog/2006/que-es-un-framework.

Sanín. 2009. *Guía metodológica general para la Preparación y Evaluación de Proyectos del Instituto Latinoamericano y del Caribe de Planificación Económica y Social, ILPES.* 2009.

Santos, Eduardo Días de. 2005. *Evaluación económica.* 2005.

SQLite. 2011. SQLite. [En línea] 2011. [Citado el: 20 de 12 de 2012.] http://www.sqlite.com.

Stroustrup, Bjarne. 2000. .*El lenguaje de programación C++.* Madrid : Addison Wesley, 2000. ISBN 84-7829-019-2..

Terrence, William. 2007. *Lenguaje de Programación. El Mundo Informático.* 2007.

VERSAT. 2012. VERSART. [En línea] 2012. [Citado el: 25 de 11 de 2012.] http://www.expomatanzas.cu.

Zambreno, Joseph y Narahari, Bhagirath. 2012. The George Washington University. *The George Washington University.* [En línea] 7 de Noviembre de 2012. http://www.seas.gwu.edu/.

BIBLIOGRAFÍA CONSULTADA

Abreu, Martín. 2006. *Formulación y Evaluación de Inversión en México.* 2006.

Aspectos Económicos de la Biotecnología. Dpto. de Economía Aplicada. *Evalucón Financiera de Proyectos de Inversion.*

Espinosa, Alibel. 2004. *Univercidad de los Andes Facultad de Ciencia Económicas y Sociales. El Enfoque estrategíco en los estudios preliminares y de mercado y la evalución financiera en un proyecto de inversión.* s.l. : Merida. Venezuela., 2004.

Jacobson, Ivar, Booch, Grady y Rumbaugh, James. 2000. *El Proceso Unificado de Desarrollo de Software.* Madrid : Addison Wesley : s.n., 2000. ISBN: 84-7829-036-2..

Koch, Josefina. 2012. Un estudio de la Factibilidad Económica. [En línea] 2012. http://www.eumed.net/libros-gratis.

Larman, Craig. 1999. *ntroducción al análisis y diseño orientado a objetos.* Mexico : PRENTICE HALL : s.n., 1999. ISBN: 970-17-0261-1..

Luna, Rafael. 1999. *Manual para determinar la factibilidad.* Centroamérica - Estados Unidos : s.n., 1999.

Masini, José, Briceño, Gustavo y Misle, Pedro. 2001. *Evaluación de Proyectos. Módulo 3.* 2001.

Miranda, Juan. *La Evaluación Financiera.*

Sáez, José, Molina, Jesús y Jiménes, Pedro. 2008. *Una Arquitectura para una Herramienta.* 2008. ISBN: 30071.

Santiestevan, Medellín. 2006. *Evaluación Financiera de Proyectos de Inversión.* 2006.

Sarmiento S., Julio. 2000. *Evaluación de proyectos.* 2000.

En línea. Universidad de las Ciencias Informáticas.Entorno Virtual de Aprendizaje (EVA). Curso Virtual de Ingeniería de Software. [En línea] En línea. eva.uci.cu.

ANEXOS

Anexo 1

Descripción detallada de los Casos de Usos (CU) arquitectónicamente significativo.

Objetivo	Gestionar datos del Flujo de Caja.	
Actores	Usuario:(Inicia).	
Resumen	El caso de uso inicia cuando el Usuario comienza a llenar los factores que debe tener un Flujo de Caja, termina cuando se muestra los valores del flujo de caja.	
Complejidad	Alta	
Prioridad	Crítico	
Precondiciones	Se debe tener seleccionado el Flujo de Caja al cual se le llenaran los datos y una vez llenados los datos se le debe de oprimir la tecla enter para que sean guardados los datos.	
Postcondiciones	Se llenaron los datos satisfactoriamente del Flujo de Caja seleccionado.	
Flujo de eventos		
Flujo básico: Gestionar datos del Flujo de Caja.		
	Actor	**Sistema**
1.	Selecciona el factor a tener en cuenta para el cálculo del Flujo de Caja (A).	

2.		Muestra las opciones que tiene dichos factores: • Adicionar entradas (filas) (B) al factor seleccionado. Ver Sección 1 “Adicionar Entrada”. • Eliminar entradas (filas) (C) del factor seleccionado. Ver Sección 2 “Eliminar Entrada”. • Modificar entradas (filas) del factor seleccionado. Ver Sección “Modificar Entrada”.
Sección 1:“Adicionar Entrada”		
Flujo Básico: Adicionar Entrada.		
	Actor	**Sistema**
3.	Selecciona el factor al cual se le agregaran las entradas (filas) (D) y luego la opción adicionar “Entradas” (B).	
4.		Adiciona una entrada (fila) (A) al factor seleccionado.
5.	Comienza a llenar los datos del factor seleccionado, en correspondencia con los datos que cuenta cada factor.	
6.		Guarda los datos de forma automática en la Base de Datos que contiene el sistema, mostrándose un mensaje “Los datos fueron guardados satisfactoriamente”. Termina caso de uso.

Sección 2:"Eliminar Entrada"		
Flujo Básico: Eliminar Entrada.		
	Actor	**Sistema**
3.	Selecciona el factor al cual se le eliminarán las entradas (fila), luego la entrada (fila) que desea eliminar del factor seleccionado y por último la opción eliminar "Entrada" (C).	
4.		Busca en la Base de Datos toda la entrada (fila) que esté relacionado con el factor seleccionado y elimina la fila completa, se muestra un mensaje "Se han eliminado los datos". Termina caso de uso.
Sección 3:"Modificar Entrada".		
Flujo Básico: Modificar Entrada.		
	Actor	**Sistema**
3.	Selecciona el factor que desea modificar.	
4.	Modifica los datos que desee.	
5.		Guarda dichos cambios en la Base de Datos que contiene el sistema, mostrándose un mensaje "Los datos fueron modificados y guardados satisfactoriamente". Termina caso de uso.

Relaciones	**CU Incluidos**	No procede.
	CU Extendidos	No procede.

Requisitos no funcionales	Ninguna
Asuntos pendientes	No procede.

Tabla 8 CU Incluido: Gestionar datos del Flujo de Caja.

Objetivo	Calcular los Indicadores.	
Actores	Usuario:(Inicia).	
Resumen	El caso de uso inicia cuando el Usuario selecciona la opción Calcular de la interfaz principal teniendo en cuenta el indicador que desea y termina cuando se muestra el valor.	
Complejidad	Alta	
Prioridad	Crítico	
Precondiciones	Se debe tener el proyecto al cual se le calculará los indicadores una vez llenado sus datos y el mismo debe de contener más de un flujo de caja.	
Postcondiciones	Se mostró de forma satisfactoria el valor del indicador seleccionado.	
Flujo de eventos		
Flujo básico: Calcular Indicadores		
	Actor	**Sistema**
1.		Muestra en la interfaz principal las opciones: • Calcular VAN. • Calcular TIR. Ver Sección 1. "Calcular TIR".
2.	Selecciona la opción Calcular VAN de la interfaz principal	

3.		Verifica que existan más de un flujo de caja del proyecto seleccionado para poder realizar los cálculos del VAN.
5.		Muestra una interfaz donde se debe introducir la tasa de descuento.
6.	Introduce la tasa de descuento.	
7.		Muestra una interfaz donde se debe introducir el intervalo de tiempo por año.
8.	Introduce el intervalo de tiempo.	
9.		Muestra un mensaje con el valor del VAN. Termina Caso de Uso.
Flujos Alternos.		
3a Evento. Validar VAN.		
	Actor	**Sistema**
1.		Muestra un mensaje informando que no existen datos suficientes para el cálculo del VAN.
2.	Corrige el proyecto y continúa a partir del paso 5 del flujo básico.	
Sección 1: "TIR"		
Flujo básico: TIR.		
	Actor	**Sistema**

1.		Verifica que exista más de un flujo de caja del proyecto seleccionado para poder realizar los cálculos de la TIR.
2.		Muestra una interfaz donde se debe introducir el intervalo de tiempo por año.
3.	Introduce el intervalo de tiempo.	
4.		Muestra un mensaje con el valor de la TIR.
Flujos Alternos.		
1a Evento. Validar TIR		
	Actor	**Sistema**
1.		Muestra un mensaje informando que no existen datos suficientes para el cálculo de la TIR.
2.	Corrige el proyecto y continúa a partir del paso 3 del flujo básico.	

Relaciones	**CU Incluidos**	Ninguno.
	CU Extendidos	Ninguno.
Requisitos no funcionales	Ninguno.	
Asuntos pendientes	No procede.	

Tabla 9 CU: Calcular VAN y TIR.

Anexo 2

Descripciones de los casos de pruebas para los distintos CU

Sesión	Escenario	Acción Realizada	Reacción del Sistema	Resultados Satisfactorios
SC 1 Gestionar Datos de Flujo de Caja.	EC 1.1 Se adiciona satisfactoriamente los datos requeridos.	Se escoge el factor al cual se le adicionara la nueva fila, se escoge la opción adicionar fila. Comienza a adicionar los datos requeridos : ✓ Componentes: • Visor: 665. • Recuperador: 665. • Gestor: 480. • Ingesta: 665. • Cliente Grabación: 600. • Analytic: 665. • Cliente Analytic: 600. ✓ Inversiones:7500 ✓ Gastos: • Visor: $ 7.980,00 • Recuperador:$7.980,00. • Gestor: $ 5.760,00. • Ingesta: $ 7.980,00. • Cliente Grabación: $ 7.200,00. • Analytic: $ 7.980,00. • Cliente Analytic: $ 7.200,00 ✓ Know How: 15.	Se adiciona la fila correctamente para cada factor seleccionado, los datos introducidos se adicionaron de forma correcta.	Satisfactorio.

	SC 1.2 Datos introducidos incorrecto.	No se realiza ninguna acción.	El sistema no permite la entrada de datos incorrectos.	Satisfactorio.
	EC 2.1 Se elimina satisfactoriamente la fila escogida.	Se selecciona la fila que se desea eliminar, luego se escoge la opción eliminar fila.	Se elimina de forma satisfactoria.	Satisfactorio.
	EC 3.1 Se modifica satisfactoriamente la fila escogida.	Se selecciona la fila que se desea modificar. Se comienzan a realizar las modificaciones.	Se modifica satisfactoriamente.	Satisfactorio.
	SC 3.2 Datos modificados incorrectos.		El sistema no permite la entrada de datos incorrectos.	Satisfactorio.

Tabla 10: Caso de Pruebas del CU Gestionar datos del Flujo de Caja.

Sesión	**Escenario**	**Acción Realizada**	**Reacción del Sistema**	**Resultados Satisfactorios**
ES 1Calcular Indicadores.	EC 1.1 Se calcula el indicador VAN satisfactoriamente.	Se selecciona la opción Calcular VAN de la interfaz principal.	Muestra una interfaz para introducir la tasa de descuento, seguidamente el intervalo de tiempo y calcula el VAN satisfactoriamente.	Satisfactorios.

	EC 1.2 No se puede realizar el cálculo del VAN	Se selecciona la opción Calcular VAN de la interfaz principal.	Muestra un mensaje especificando que el VAN no se puede realizar debido a que existe suficientes datos para realizar lo operación.	Satisfactorios.
	EC 2.1 Se calcula el indicador TIR satisfactoriamente.	Se selecciona la opción Calcular TIR de la interfaz principal.	Muestra una interfaz para introducir la tasa de descuento, seguidamente el intervalo de tiempo y calcula el TIR satisfactoriamente.	Satisfactorios.
	EC 2.2 No se puede realizar el cálculo del TIR	Se selecciona la opción Calcular TIR de la interfaz principal.	Muestra un mensaje especificando que el TIR no se puede realizar debido a que existe suficientes datos para realizar lo operación.	Satisfactorios.

Tabla 11: Caso de Prueba del CU Calcular VAN y TIR.

Anexo 3

Diagrama de Clases del Diseño

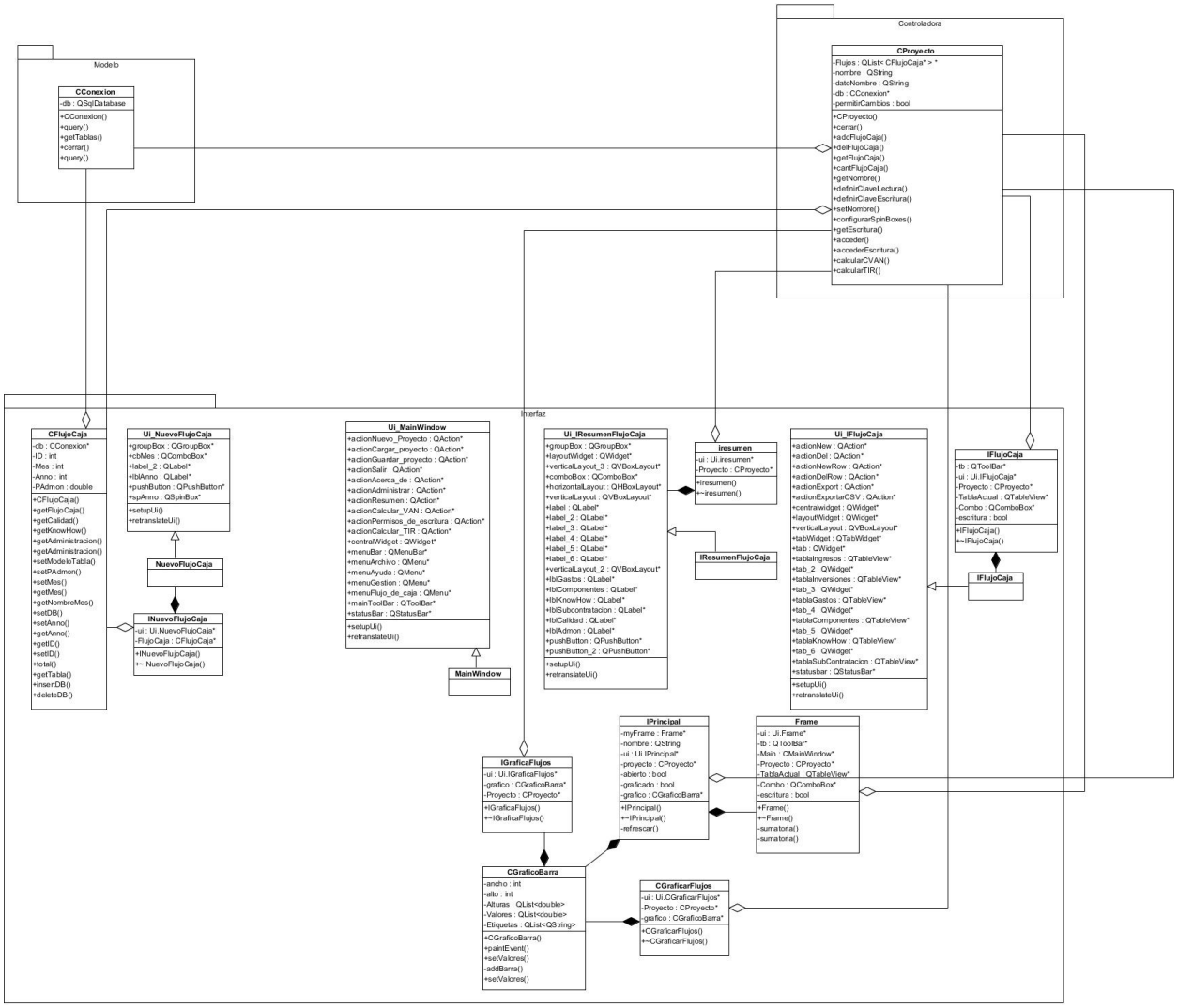

Figura 11: Diagrama de Clases del Diseño.

Anexo 4
Diagramas de Colaboración del Diseño

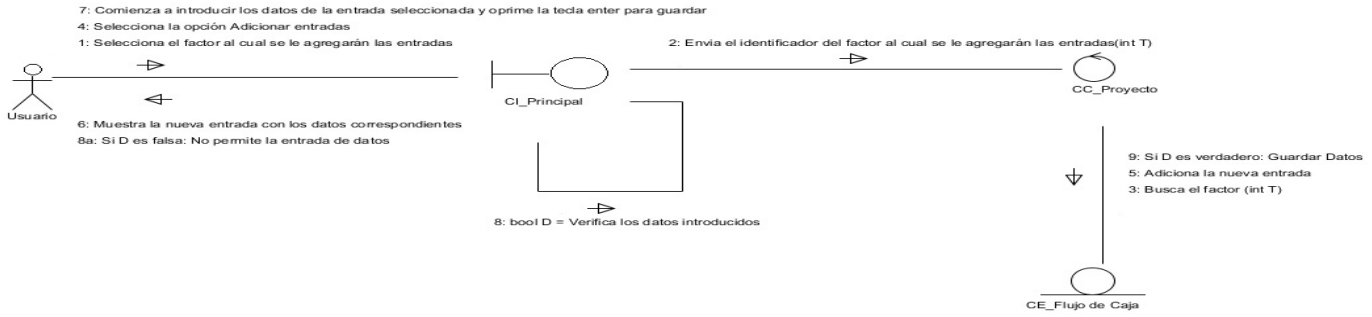

Figura 12: CU Gestionar Datos del Flujo de Caja. Sección 1 Adicionar Entradas.

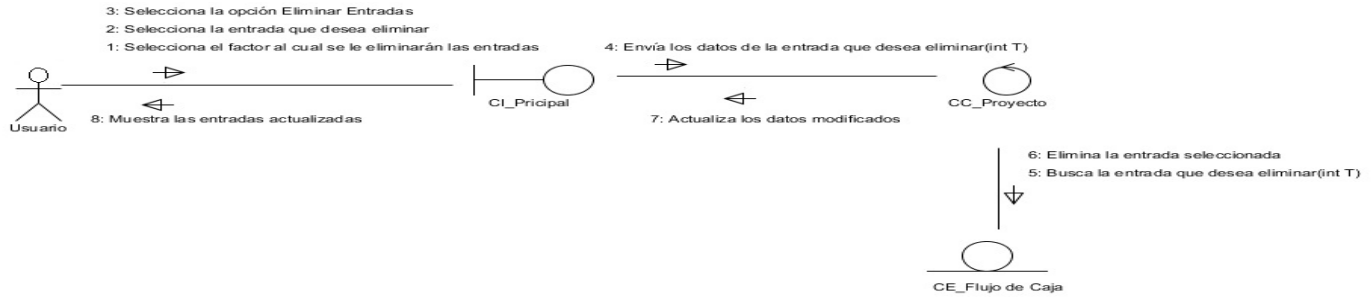

Figura 13: CU Gestionar Datos del Flujo de Caja. Sección 2 Eliminar Entrada.

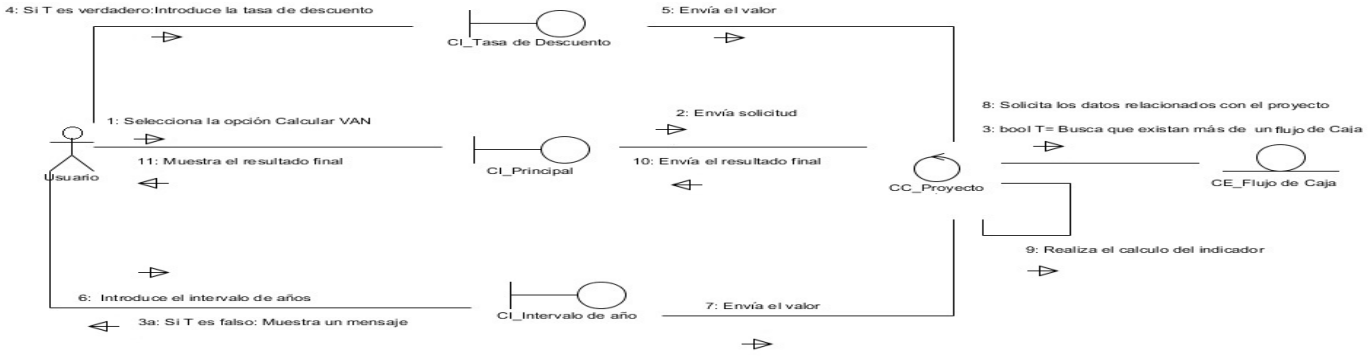

Figura 14: Calcular Indicadores.

Anexo 5
Diagrama de Secuencia

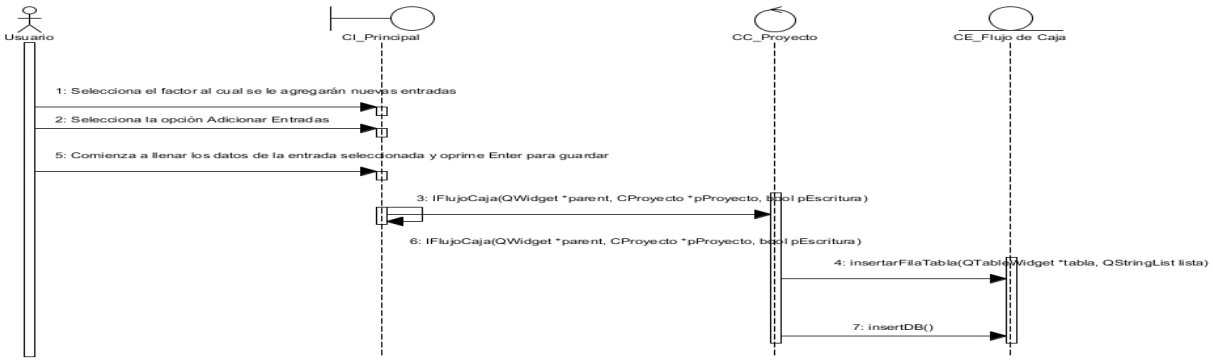

Figura 15: CU Gestionar Datos de Flujo de Caja. Sección 1 Adicionar.

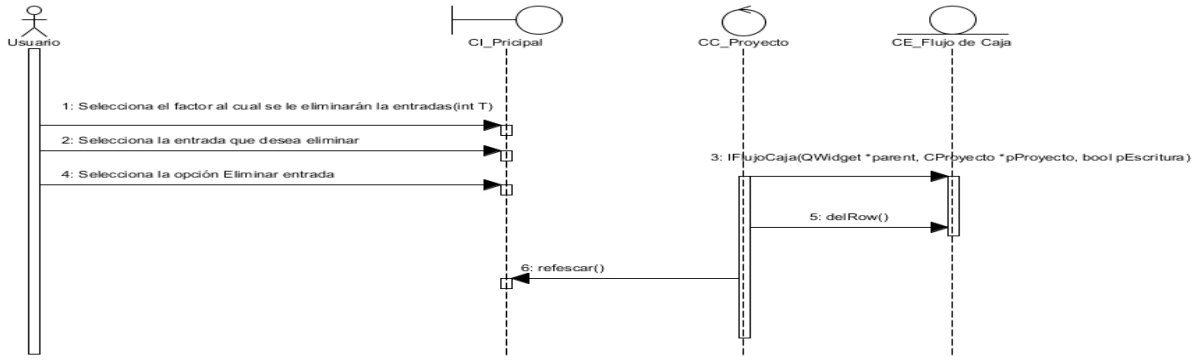

Figura 16: CU Gestionar Datos del Flujo de Caja. Sección 2 Eliminar.

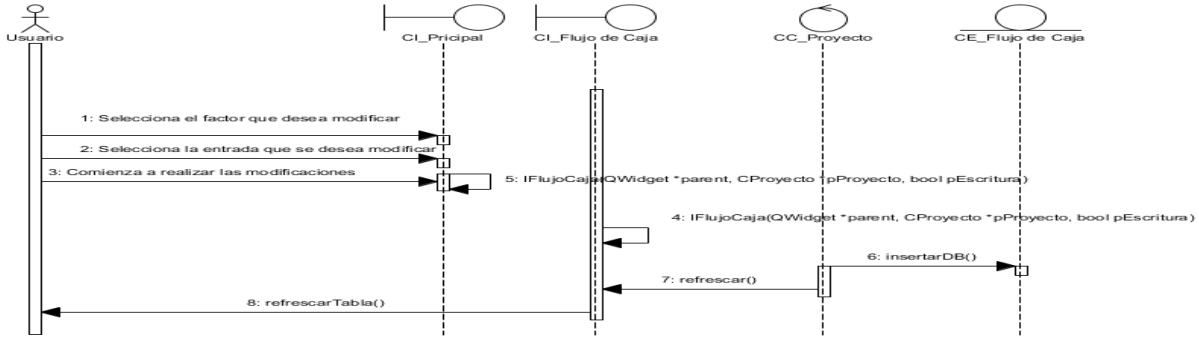

Figura 17: Calcular Indicadores.

GLOSARIO DE TÉRMINOS

Aquí se incluye una explicación de determinados términos y siglas utilizados en el texto para facilitar su comprensión. Generalmente se incluyen los términos que tienen menor difusión en el campo profesional, los de otro campo profesional, o aquellos términos conocidos, pero que se usan con un significado diferente en el texto. Además, pueden ser incluidas las siglas utilizadas en el documento para facilitar su lectura.

API: *Application Programming Interface - Interfaz de Programación de Aplicaciones* es el conjunto de funciones y procedimientos (o métodos si se refiere a programación orientada a objetos) que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción.

Applets: Componente de una aplicación que se ejecuta en el contexto de otro programa, por ejemplo un navegador web. El applet debe ejecutarse en un contenedor, que lo proporciona un programa anfitrión, mediante un plugin, o en aplicaciones como teléfonos móviles que soportan el modelo de programación por applets.

Economía: Es la extracción, producción, intercambio, distribución y consumo de bienes y servicios. Forma o medios de satisfacer las necesidades humanas mediante los recursos (que se consideran escasos). Es la forma en que individuos y colectividades sobreviven, prosperan y funcionan.

Factibilidad: Se refiere a la disponibilidad de los recursos necesarios para llevar a cabo los objetivos o metas señaladas. Generalmente la factibilidad se determina sobre un proyecto.

Feedback: Es un proceso por el cual cierta proporción de la señal de salida de un sistema se pasa (alimentado de nuevo) a la entrada. Esta se usa a menudo para controlar el comportamiento dinámico del sistema. Ejemplos de información se puede encontrar en la mayoría de los sistemas complejos, tales como la ingeniería, arquitectura, economía, la termodinámica y la biología

GUI: Graphical User Interface- Interfaz Gráfica de Usuario es el artefacto tecnológico de un sistema interactivo que posibilita, a través del uso y la representación del lenguaje visual, una interacción amigable con un sistema informático. Utiliza un conjunto de imágenes y objetos gráficos para representar la información y acciones disponibles en la interfaz.