INSTITUTO SUPERIOR MINERO METALÚRGICO

"DR. ANTONIO NÚÑEZ JIMÉNEZ".

FACULTAD DE METALURGIA - ELECTROMECÁNICA

MOA, HOLGUÍN

Trabajo de Diploma para optar por el Título de Ingeniería Informática

Título:

Diseño e Implementación del Sistema de Gestión y Control de la Ejecución de los Presupuestos para la Empresa UNEVOL S.A

Autor:

Luis A. Cortina Almeida

Tutor:

Yiezenia Rosario Ferrer

Moa, Cuba
Julio, 2011

DECLARACIÓN DE AUTORÍA

Declaro que soy el único autor de este trabajo y autorizo al Instituto Superior Minero Metalúrgico de Moa “Dr. Antonio Núñez Jiménez” y al Departamento de Informática para que hagan el uso que estimen pertinente con este trabajo.

Para que así conste firmamos la presente a los _____ días del mes de _____ del 2011.

Luis A. Cortina Almeida

_____

Firma autor

Dra. Yiezenia Rosario Ferrer

_____

Firma tutor

Agradecimientos

En primer lugar, a mi Dios, por llegar conmigo hasta aquí.

A mi mamá, por toda una vida de sacrificio.

A mi novia, por el cariño de este año y medio.

A mis hermanos.

A Javier y a Caridad.

A toda mi familia, mis amigos y vecinos que han estado ahí de una forma u otra.

A mis compañeros de aula.

A Ezequiel, por estar siempre ahí.

Especialmente a Amador por la paciencia, el consejo y la sabia palabra.

Dios (…) da vida a los muertos, y llama las cosas que no son, como si fuesen.

Romanos 4:17

Resumen

Las Tecnologías de la Información (TI) han cambiado la forma en que operan las organizaciones actuales. A través de su uso se logran importantes mejoras, pues automatizan los procesos operativos, suministran una plataforma de información necesaria para la toma de decisiones y, lo más importante, su implantación logra ventajas competitivas o reducir la ventaja de los rivales. La información se ha colocado en un buen lugar, como uno de los principales recursos que poseen las empresas actualmente. Los ejecutivos que se encargan de las tomas de decisiones, han comenzado a comprender que la información no es sólo un subproducto de la conducción empresarial, sino que a la vez alimenta a los negocios y puede ser uno de los tantos factores críticos para la determinación del éxito o fracaso de estos.

Uno de estos factores críticos en los negocios, es la gestión y control de los presupuestos. Los negocios actuales, enfrentan situaciones que pueden afectar negativamente el resultado de sus operaciones. Los presupuestos son importantes porque ayudan a minimizar el riesgo en estas operaciones y al mismo tiempo, sirven como mecanismo para la revisión de políticas y estrategias de la empresa y direccionarlas hacia lo que verdaderamente se busca.

Por la importancia que tiene en medio de la complicada situación económica internacional el control continuo de la ejecución del presupuesto en la Empresa UNEVOL S.A, así como las ventajas que presentaría para sus directivos el manejar la información de una forma más rápida y eficiente, surgió la necesidad de automatizar el proceso del control de la ejecución del presupuesto.

La tarea es construir una herramienta que brinde al usuario final la capacidad de gestionar datos y obtener informaciones que agilicen el proceso de análisis de los presupuestos, y además que mejore la calidad de trabajo de los encargados del mismo en el organismo.

Summary

The Information Technology (IT) has been changing the way the organizations work. Through IT, important improvements are achieved, such as automate operational processes, provide with the information platform needed to decision making and most of all, to reduce the competitor's advantage and increase the business ones. The information itself, it's placed as one of the main resources a modern enterprise could have. The executives that run the decision making process, are starting to be aware that information is not only a secondary product of business management, but also a feedback to the business itself and finally one of the so many critical agents on the failure or success of any business.

The management and control of budgets can be another agent on organizations success. The current businesses face many situations that can negatively affect the outcome of its operations. Budgets are important because they help to minimize the risk on these operations and at the same time, they are a mechanism to review the enterprise policies and strategies and direct them toward the actual goal of the company.

For the undeniable importance on the middle of a complicated international situation, and the advantages that would present for the executives to handle the information in a quicker and more efficient way, it became evident the need to automate the process of managing and controlling the budgets.

The task is to build a tool that provide to the end user the capability to manage data and obtain information that speed up the budget analysis process, besides, the improvement of the work quality of the people in it.

Índice

INTRODUCCIÓN

La tecnología de la información es esencial para mejorar la productividad de las empresas, aunque su aplicación debe llevarse a cabo de forma inteligente. El mero hecho de introducir tecnología en los procesos empresariales no es garantía de un aumento de la productividad. Para que la implantación de nueva tecnología produzca rentabilidad hay que cumplir varios requisitos: tener un conocimiento profundo de los procesos de la empresa, planificar detalladamente las necesidades de tecnología de la información e incorporar los sistemas tecnológicos paulatinamente, empezando por los más básicos. La mayor productividad se consigue mediante una reducción de los costes y el aumento de las ventas.

El presupuesto es un plan de acción dirigido a cumplir una meta prevista, expresada en valores y términos financieros que, debe cumplirse en determinado tiempo y bajo ciertas condiciones previstas, este concepto se aplica a cada centro de responsabilidad de la organización. La principal función de los presupuestos se relaciona con el control financiero de la organización. El control presupuestario es el proceso de descubrir qué es lo que se está haciendo, comparando los resultados con sus datos presupuestados correspondientes para verificar los logros o remediar las diferencias.

Los presupuestos pueden desempeñar tanto roles preventivos como correctivos dentro de la organización. Con estos debemos planear integral y sistemáticamente todas las actividades que la empresa debe desarrollar en un periodo determinado, además de controlar y medir los resultados cuantitativos, cualitativos y, fijar responsabilidades en las diferentes dependencias de la empresa para logar el cumplimiento de las metas previstas.

Debido a la crisis mundial del año 2010, la máxima dirección de la Revolución ha señalado la importancia de implantar medidas que afirmen la economía, que eleven los resultados a niveles aceptables de productividad y que se encamine a

un saludable crecimiento o al menos en un fortalecimiento progresivo de los principales factores en los cuales se cimenta la vida económica del país. *"El plan y el presupuesto son sagrados y se elaboran para ser cumplidos, no para conformarnos con justificaciones de cualquier tipo y hasta con imprecisiones y mentiras, intencionadas o no, cuando no se logran las metas trazadas"*.(Castro,R.) Con estas palabras apuntalaba el presidente Raúl Castro la importancia de un estricto control y al mismo tiempo de una responsabilidad puntual de las personas que tienen a su cargo dirigir y controlar como se reparten los recursos para lograr una empresa y un país más eficiente.

UNEVOL S.A. es una empresa mixta creada en 1998 con capital de Dalia Gestão e Serviços y del grupo empresarial UNECAMOTO del Ministerio de la Industria Sideromecánica (SIME), sus oficinas principales se encuentran situadas al oeste de Ciudad de la Habana, Cuba. Son importadores exclusivos de motores, recambios y tecnología de la marca Volvo Penta, importadores de equipos, motores, recambios y tecnología de la división Volvo Construction Equipment (Equipos de Construcción); y además, brindan asesoría y servicios de taller con todas las garantías que la marca impone tanto en el territorio nacional como el extranjero. Desde su creación han tenido la oportunidad de transferir y acumular vasta experiencia en las tecnologías de Volvo. La superación continua de sus técnicos e ingenieros directamente en fábrica y la implantación en Cuba de todas las facilidades tecnológicas e infraestructura les permite brindar un servicio de excelencia con tecnología de punta. (1)

En la actualidad la empresa UNEVOL S.A. presenta debilidades en la gestión de la información sobre la ejecución del presupuesto, las que limitan la obtención de informaciones vitales dentro del sistema de trabajo para la toma de decisiones a distintos niveles de dirección. Además hay un crecimiento excesivo en el número de documentos que contienen las informaciones de las diferentes etapas de los presupuestos, además de un procedimiento engorroso de obtención de datos para

los diferentes análisis a realizar por parte de cada directivo implicado en el proceso de controlar el movimiento de recursos y la toma de decisiones en temas claves como gastos y los análisis de ventas. Por otra parte, las herramientas y recursos informáticos son insuficientes, por lo que se aprecia que la gestión de la información no satisface las necesidades que necesitan los administrativos y trabajadores en general y la falta de un sistema automatizado para la organización de la información bien estructurada.

Es por eso que debido a la alta densidad de información que como resultado de este proceso de gestión de la información se acumula en la empresa y la necesidad de mejorar la calidad y rapidez de la obtención de la información, se desea automatizar el proceso de planificación y control de la ejecución de los presupuestos. Debido a esto surge el siguiente problema: La no existencia de un sistema automatizado que permita la gestión eficiente de la planificación y control de la ejecución del presupuesto en la empresa UNEVOL S.A. Este problema se enmarca en el objeto de estudio: Desarrollo de sistemas informáticos para la gestión de presupuesto en la empresa UNEVOL S.A. El campo de acción es el desarrollo con metodologías ágiles del sistema automatizado para la gestión de presupuestos en la Empresa UNEVOL S.A.

Para dar solución al problema planteado se propone como objetivo general desarrollar un sistema automatizado para la gestión eficiente de la planificación de los presupuestos a varios niveles de la empresa así como el control de la ejecución. La infraestructura informática de la empresa, nos permite realizar esta tarea mediante una aplicación cliente-servidor, donde la aplicación debe ser instalada en las computadoras de los empleados autorizados a utilizarla, y la base de datos de la misma se encontrará en el servidor MS SQL Server que utiliza la empresa para sus funciones. La idea a defender es la automatización del proceso de planificación y control de la ejecución del presupuesto, favorecerá la celeridad en la gestión y toma de decisiones gerenciales así como mejores métodos de trabajo en la empresa UNEVOL S.A.

De acuerdo a esta propuesta se derivan los siguientes objetivos específicos:

1. Establecer el estado del arte sobre la información disponible tanto nacional e internacional relacionada con los sistemas de gestión de presupuestos.
2. Seleccionar las herramientas a utilizar para el diseño del sistema de gestión de presupuesto.
3. Implementación del sistema de gestión de presupuesto.

Para el logro de los objetivos fue necesario plantearse las siguientes tareas:

1. Búsqueda de información nacional e internacional sobre los sistemas de gestión de presupuesto.
2. Estudio del basamento teórico.
3. Realizar un análisis del sistema de gestión de presupuesto.
4. A partir de la información disponible, diseñar la ingeniería de software para la automatización de dicho sistema.
5. Análisis de factibilidad y sostenibilidad.
6. Desarrollar el manual de usuario del sistema.

Para el cumplimiento de las tareas los métodos de investigación empleados fueron Métodos Teóricos y Métodos Empíricos.

Los siguientes Métodos Teóricos sustentan la investigación:

Analítico-Sintético:

Permitió integrar y descomponer el conocimiento, pues se hizo una breve investigación y estudio del sistema, determinando los aspectos esenciales y el arribo a conclusiones prácticas y teóricas y así identificar el problema concreto existente.

Inductivo-deductivo:

Permitió pasar de lo particular a lo general y viceversa favoreciendo objetivamente el enlace que se establece en la realidad entre lo singular y lo general ya que ambas se complementan mutuamente en el proceso de desarrollo científico.

Modelación:

Pues se crean abstracciones que explican la realidad, por ejemplo, todos los modelos y diagramas presentados.

Los Métodos Empíricos empleados son:

Entrevistas:

Nos permite obtener la información adecuada para adentrarnos más en el problema y en las necesidades del Departamento Económico y otros potenciales usuarios y así determinar los principales requerimientos del sistema.

Observación:

Para ver la funcionalidad que existe en el Departamento de Económico en cuanto al tratamiento de presupuestos y el comportamiento del sistema.

El documento consta de 5 capítulos:

Capítulo 1 "FUNDAMENTACIÓN TEÓRICA":

Se exponen conceptos y criterios principales necesarios para el entendimiento del problema, además se realiza un estudio general de los elementos principales de los presupuestos y una caracterización de las metodologías de desarrollo a utilizar, el lenguaje de programación, los sistemas gestores de base de datos las herramientas para el desarrollo del software y las técnicas empleadas durante el proceso de desarrollo.

Capítulo 2 "PLANIFICACIÓN - DEFINICIÓN":

En este capítulo se presenta parte de la solución de la investigación, haciendo uso de la metodología de desarrollo SXP propuesta por la Universidad de Ciencias Informáticas, en el que se aborda la fase de planificación y definición en la que se explica toda la dinámica del proyecto basándose en el expediente del proyecto.

Capítulo 3 "DESARROLLO":

En este capítulo se abordan los elementos pertenecientes a la fase de desarrollo. Se presenta las tarjetas CRC para apreciar el desarrollo orientado a objetos. También aparecen las tareas de ingeniería para llevar a cabo el desarrollo de las historias de usuario.

Capítulo 4 "PRUEBAS":

Este capítulo está dedicado a las pruebas que se les realizan al software, las pruebas de aceptación del cliente. Estas pruebas fueron llevadas a cabo antes de cada entrega que se realizó durante todo el desarrollo del proyecto.

Capítulo 5 "ESTUDIO DE FACTIBILIDAD":

En este capítulo se realiza un estudio de factibilidad del proyecto, se utilizará la Metodología Costo Beneficio para determinar si es factible realizar este proyecto.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

1.1 Introducción

En este capítulo se realiza un proceso investigativo de los aspectos teóricos necesarios para la elaboración y concepción del Trabajo de Diploma. Se describen los principales conceptos asociados al problema y que son necesarios para un mejor entendimiento y darle solución al mismo. Además se realiza un estudio del proceso de gestión y control de la ejecución de los presupuestos en Cuba y la estructura que debe tener con sus funciones asociadas, estableciendo la estrategia a llevar a cabo y los beneficios esperados. Se hace una caracterización de cada tipo de herramientas y del lenguaje de programación, así como las metodologías de desarrollo escogida para el desarrollo del software.

1.2 Presupuestos. Conceptos principales.

Se le llama presupuesto al cálculo anticipado de los ingresos y gastos de una actividad económica (personal, familiar, un negocio, una empresa, una oficina) durante un período, por lo general en forma anual. (2)

Es la previsión de gastos e ingresos para un determinado lapso, por lo general un año. Permite a las empresas, los gobiernos, las organizaciones privadas y las familias establecer prioridades y evaluar la consecución de sus objetivos. (3)

El presupuesto es un plan de acción cuantitativo y auxiliar de la administración y el control y abarca todas las fases de las operaciones (ventas, producción, distribución, financiamiento). (Horgren C, 1971).

Un presupuesto es primordialmente el esquema de un plan proyectado de acción de una empresa, para un período definido. (Lang T, 1958).

El presupuesto es un estimado cuidadosamente preparado de las condiciones futuras de los negocios. (John J.W. Neuner).

Elaborar un presupuesto permite a las empresas, los gobiernos, las organizaciones privadas o las familias establecer prioridades y evaluar la consecución de sus objetivos. Para alcanzar estos fines, puede ser necesario incurrir en déficit (que los gastos superen a los ingresos) o, por el contrario, puede ser posible ahorrar, en cuyo caso el presupuesto presentará un superávit (los ingresos superan a los gastos).En el ámbito del comercio, presupuesto es también un documento o informe que detalla el coste que tendrá un servicio en caso de realizarse. El que realiza el presupuesto se debe atener a él, y no puede cambiarlo si el cliente acepta el servicio.

Los presupuestos son importantes porque ayudan a minimizar el riesgo en las operaciones de la organización. Por medio de los presupuestos se mantiene el plan de operaciones de la empresa en unos límites razonables. Sirven como mecanismo para la revisión de políticas y estrategias de la empresa y direccionarlas hacia lo que verdaderamente se busca. Cuantifican en términos financieros los diversos componentes de su plan total de acción. Las partidas del presupuesto sirven como guías durante la ejecución de programas de personal en un determinado período, y sirven como norma de comparación una vez que se hayan completado los planes y programas. Los procedimientos inducen a los especialistas de asesoría a pensar en las necesidades totales de las compañías, y a dedicarse a planear de modo que puedan asignarse a los varios componentes y alternativas la importancia necesaria. Los presupuestos sirven como medios de comunicación entre unidades a determinado nivel y verticalmente entre ejecutivos de un nivel a otro. Una red de estimaciones presupuestarias se filtran hacia arriba a través de niveles sucesivos para su ulterior análisis.

1.2.1 Funcionamiento de los presupuestos en Cuba y la Empresa.

En Cuba, el estado es un estado socialista de economía planificada, mediante la cual se realiza el control sobre todo el funcionamiento económico basado en el predominio de la propiedad social, ejercida principalmente a través de las empresas de propiedad estatal (denominadas corrientemente entes autónomos o empresas públicas en otros contextos) esta ha constituido, de hecho, la principal forma de intervención del estado en el accionar integral de la economía.

El Estado organiza, dirige y controla la actividad económica nacional conforme a un plan que garantice el desarrollo programado del país, a fin de fortalecer el sistema socialista, satisfacer cada vez mejor las necesidades materiales y culturales de la sociedad y los ciudadanos, promover el desenvolvimiento de la persona humana y de su dignidad, el avance y la seguridad del país. *El Estado administra directamente los bienes que integran la propiedad socialista de todo el pueblo, o podrá crear y organizar empresas y entidades encargadas de su administración, cuya estructura, atribuciones, funciones y el régimen de sus relaciones son regulados por la ley*. (4)

En el caso de las empresas mixtas como es UNEVOL S.A., el presupuesto se elabora contando con las proyecciones de las diferentes áreas (sucursales, centros de costo, etc.) y agrupándolas en un presupuesto corporativo para dividir los resultados planificados y reales a varios niveles de dirección. Después, se somete a la junta de accionistas que evaluará el documento anterior, realizará cambios si se estima pertinente y aprobará dicho presupuesto. La junta puede realizar cambios a lo largo del año al mismo presupuesto y estos cambios deben ser archivados y permanecer como constancia para futuros análisis.

El departamento económico es responsable por la gestión de esta información, ya sea manipularla o distribuirla a los directivos que estime pertinentes o a las auditorías de las diferentes instituciones que la requieran. También es responsable

por preparar los informes que se utilizarán para evaluar la situación de la empresa y tomar las decisiones pertinentes en consecuencia con los resultados obtenidos.

Como soporte informático para el registro de sus operaciones, la empresa cuenta con un ERP (Enterprise Resource Planning) que ofrece un amplio espectro de funcionalidades. Todos los datos contables que se utilizan para la elaboración y control de los presupuestos en la empresa se obtienen de dicho sistema.

Antes de la implementación de este trabajo, el proceso de elaboración del presupuesto consistía en:

1. Obtener el Estado de Resultado de la Empresa del ERP, exportarlo a Microsoft Excel para tener la posibilidad de manipular los datos.
2. Basado en una plantilla del mismo Microsoft Excel, realizar las operaciones pertinentes hasta obtener la presentación del informe deseada.

Este proceso es extremadamente lento e ineficiente, por lo cual se abre la puerta a alguna solución automatizada que cumpliera con los requisitos propios del negocio y además aporte más dinamismo y eficiencia a un proceso con tanta importancia para la empresa. La tarea es buscar la solución más apropiada para este problema.

1.2.2 Evaluación de sistemas existentes.

Al iniciar la investigación, se encuentra que el ERP que la empresa utiliza, presentaba entre sus opciones un módulo de Presupuestos, lo cual parece la primera opción a revisar para determinar si es factible usar el mismo, basado en las necesidades reales del proceso. Al indagar sobre el tema se informa que nunca se compró este módulo por razones económicas y además en aquel momento no brindaba todas las funcionalidades que se necesitaban para el dinamizar el flujo de trabajo, es decir, se desecha esta posibilidad por cuestiones financieras y porque no cumple con todas las necesidades de la empresa.

La interacción que el sistema a escoger debe tener con los datos del ERP de la empresa, es otra cuestión de peso, pues se debía simplificar el trabajo de los empleados. Para eso, tendría que existir cierto tipo de integración con la

información existente para minimizar la entrada de datos y al final obtener la información completa y correcta. Estas circunstancias nos detuvieron de buscar en otros sistemas existentes, ya que la solución óptima era construir uno que se adaptara fácilmente a las características y necesidades actuales de la Empresa.

Por lo tanto se decidió desarrollar una aplicación Cliente-Servidor paralela al ERP de la entidad debido a:

- La posibilidad casi nula de obtener un sistema que se adaptara perfectamente al ERP establecido en la empresa.
- Por las condiciones actuales de la economía, es más factible para la empresa construir un sistema paralelo de buenos resultados que comprar el módulo.
- Las necesidades específicas de la Entidad en el flujo de trabajo en el que está enmarcado esta tarea.

  La construcción de un sistema permitiría un nivel más profundo de participación de los involucrados en el proceso, permitiendo cubrir un rango mayor de aspectos que el módulo ofrecido por los constructores del ERP. (Manejo de diferentes presupuestos a distintos niveles de dirección, obtención de reportes para la toma de decisiones y el control de la distribución y el acceso a la información a cada nivel de la organización)
- La automatización de este proceso resultaría en ventajas a los diferentes trabajadores y directivos que están involucrados en este proceso. Por un lado, un mejor manejo de la información y más facilidades en las operaciones de entrada de datos; y por otro, la rapidez de obtención de los datos para la realización de análisis financieros y gerenciales.

Para realizar este tipo de tarea con éxito, necesitamos herramientas de gestión que nos permitan planificar, dinamizar y administrar recursos, con el fin de culminar todo el trabajo requerido y cumplir con el alcance, límites de tiempo y con los costos definidos para el mismo.

1.3 Aspectos generales sobre las Metodologías Usadas

Actualmente para desarrollar un proyecto con éxito, debe estar regido por una metodología de desarrollo, la cual puede seguir uno o varios modelos de ciclo de vida, o sea, el ciclo de vida indica que es lo que hay que obtener a lo largo del desarrollo del proyecto. Una metodología de desarrollo de software es un conjunto de técnicas, herramientas, procedimientos y soporte documental que permite a los desarrolladores definir los elementos necesarios para la construcción de un nuevo producto de software. Es la que durante el proceso de desarrollo del software define "quién está haciendo qué, cuándo y cómo para alcanzar un determinado objetivo." Mediante la metodología de desarrollo de software se van indicando paso a paso todas las actividades a realizar para lograr el producto informático deseado, indicando además qué personas deben participar en el desarrollo de las actividades y qué papel deben de tener. Además detallan la información que se debe producir como resultado de una actividad y la información necesaria para comenzarla.

1.3.1 Metodologías tradicionales

Teniendo en cuenta la filosofía de desarrollo de las metodologías, aquellas con mayor énfasis en la planificación y control del proyecto, en especificación precisa de requisitos y modelado, reciben el apelativo de Metodologías Tradicionales o Pesadas.

Estas metodologías tradicionales imponen una disciplina de trabajo sobre el proceso de desarrollo del software, con el fin de conseguir un software más eficiente. Para ello, se hace énfasis en la planificación total de todo el trabajo a realizar y una vez que está todo detallado, comienza el ciclo de desarrollo del producto software. Se centran especialmente en el control del proceso, mediante una rigurosa definición de roles, actividades, artefactos, herramientas y notaciones para el modelado y documentación detallada. Además, las metodologías tradicionales no se adaptan adecuadamente a los cambios, por lo que no son

métodos adecuados cuando se trabaja en un entorno, donde los requisitos no pueden predecirse o bien pueden variar. (5)

Constituye un ejemplo de esta metodología:

1.3.1.1 RUP (Rational Unified Procces)

El Proceso Racional Unificado (Rational Unified Process en inglés, habitualmente resumido como RUP) es un proceso de desarrollo de software y junto con el Lenguaje Unificado de Modelado UML, constituye la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos. También se conoce por este nombre al software desarrollado por Rational, hoy propiedad de IBM, el cual incluye información entrelazada de diversos artefactos y descripciones de las diversas actividades. Está incluido en el Rational Method Composer (RMC), que permite la personalización de acuerdo con las necesidades. El RUP es un producto de Rational (IBM). Se caracteriza por ser iterativo e incremental, estar centrado en la arquitectura y guiado por los casos de uso. Incluye artefactos (que son los productos tangibles del proceso como por ejemplo, el modelo de casos de uso, el código fuente, etc.) y roles (papel que desempeña una persona en un determinado momento, una persona puede desempeñar distintos roles a lo largo del proceso). (6)

1.3.1.2 Lenguaje Unificado de Modelado

Lenguaje Unificado de Modelado (LUM o UML, por sus siglas en inglés, Unified Modeling Language) es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad; está respaldado por el OMG (Object Management Group). Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema. UML ofrece un estándar para describir un "plano" del sistema (modelo), incluyendo aspectos conceptuales tales como procesos de negocio y funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes

reutilizables. Es importante resaltar que UML es un "lenguaje de modelado" para especificar o para describir métodos o procesos. Se utiliza para definir un sistema, para detallar los artefactos en el sistema y para documentar y construir. En otras palabras, es el lenguaje en el que está descrito el modelo.

Se puede aplicar en el desarrollo de software entregando gran variedad de formas para dar soporte a una metodología de desarrollo de software (tal como el Proceso Unificado Racional o RUP), pero no especifica en sí mismo que metodología o proceso usar. (7)

1.3.2 Metodologías ágiles

Hoy en día con el auge de la tecnología, y con el objetivo de agilizar y automatizar los procesos en el desarrollo de software, nos vemos en la necesidad de implantar Metodologías de Desarrollo de Software que nos ayuden a entregar un producto de calidad en tiempo y costo estimados, las metodologías ágiles de desarrollo de software han despertado interés gracias a que proponen simplicidad y velocidad para la creación de aplicaciones. Las metodologías tradicionales no se adaptan a las nuevas necesidades o expectativas que tienen los usuarios hoy en día, en parte que los métodos usados no son flexibles ante la posibilidad de la exigencia de nuevos requerimientos. Estos cambios generalmente implican altos costos, demanda de tiempo y la reestructuración total del proyecto que se esté llevando; en contraparte, los métodos ágiles permiten un desarrollo iterativo y adaptable que permite la integración de nuevas funcionalidades a lo largo del desarrollo del proyecto; para que tanto el cliente como el desarrollador queden satisfechos porque el producto final tiene una calidad adecuada.

El desarrollo ágil de software es un marco de trabajo conceptual de la ingeniería de software que promueve iteraciones en el desarrollo a lo largo de todo el ciclo de vida del proyecto. Existen muchos métodos de desarrollo ágil; la mayoría minimiza riesgos desarrollando software en cortos lapsos de tiempo. El software desarrollado en una unidad de tiempo es llamado una iteración, la cual debe durar

de una a cuatro semanas. Cada iteración del ciclo de vida incluye: planificación, análisis de requerimientos, diseño, codificación, revisión y documentación. Una iteración no debe agregar demasiada funcionalidad para justificar el lanzamiento del producto al mercado, pero la meta es tener un demo (sin errores) al final de cada iteración. Al final de cada iteración el equipo vuelve a evaluar las prioridades del proyecto. (8)

Las Metodologías Ágiles se basan en los siguientes principios:

- Realizar entregas cortas en el tiempo y continuas.
- Dar la bienvenida a los cambios.
- Entregas periódicas y frecuentes que funcionen.
- La comunicación directa es el método más eficiente y efectivo para comunicar información.
- La medida principal de progreso es el software que funciona.
- Buen diseño y calidad técnica.
- La simplicidad es algo básico.

Actualmente existen varias metodologías ágiles de desarrollo de software, las cuales se deben estudiar con detenimiento para definir cuál es la más adecuada a usar en el software a construir.

1.3.2.1 Dynamic Systems Development Method (DSDM)

El método de desarrollo de sistemas dinámicos (en inglés Dynamic Systems Development Method o DSDM) es un método que provee un framework (marco de trabajo) para el desarrollo ágil de software, apoyado por su continua implicación del usuario en un desarrollo iterativo y creciente que sea sensible a los requerimientos cambiantes, para desarrollar un sistema que reúna las necesidades de la empresa en tiempo y presupuesto. Es uno de un número de métodos de desarrollo ágil de software y forma parte de la alianza ágil. DSDM fue desarrollado en el Reino Unido en los años 90 por un consorcio de proveedores y de expertos

en la materia del desarrollo de sistemas de información (IS), el consorcio de DSDM, combinando sus experiencias de mejores prácticas. Como extensión del Desarrollo rápido de aplicaciones (RAD), DSDM se centra en los proyectos de sistemas de información que son caracterizados por presupuestos y agendas apretadas. DSDM trata los problemas que ocurren con frecuencia en el desarrollo de los sistemas de información en lo que respecta a pasar sobre tiempo y presupuesto y otras razones comunes para la falta en el proyecto tal como falta de implicación del usuario y de la comisión superior de la gerencia. (9)

1.3.2.2 Adaptive Software Development (ASD)

Adaptive Software Development es el modelo de implementación de patrones ágiles para desarrollo de software, diseñado por Jim Highsmith.

Características básicas de ASD:

- Trabajo orientado y guiado por la misión del proyecto.
- Basado en la funcionalidad.
- Desarrollo iterativo.
- Desarrollo acotado temporalmente.
- Guiado por los riesgos.
- Trabajo tolerante al cambio. (10)

1.3.2.3 Agile Unified Process (AUP)

El Proceso Unificado Ágil de Scott Ambler o Agile Unified Process (AUP) en inglés es una versión simplificada del Proceso Unificado de Rational (RUP). Este describe de una manera simple y fácil de entender la forma de desarrollar aplicaciones de software de negocio usando técnicas ágiles y conceptos que aún se mantienen válidos en RUP. El AUP aplica técnicas ágiles incluyendo Desarrollo Dirigido por Pruebas (Test Driven Development- TDD), Modelado Ágil, Gestión de Cambios Ágil, y Refactorización de Base de Datos para mejorar la productividad. (11)

1.3.2.4 Extreme Programming (XP)

La programación extrema o eXtreme Programming (XP) es un enfoque de la ingeniería de software formulado por Kent Beck, autor del primer libro sobre la materia, Extreme Programming Explained: Embrace Change (1999). Es el más destacado de los procesos ágiles de desarrollo de software. Al igual que éstos, la programación extrema se diferencia de las metodologías tradicionales principalmente en que pone más énfasis en la adaptabilidad que en la previsibilidad. Los defensores de XP consideran que los cambios de requisitos sobre la marcha son un aspecto natural, inevitable e incluso deseable del desarrollo de proyectos. Creen que ser capaz de adaptarse a los cambios de requisitos en cualquier punto de la vida del proyecto es una aproximación mejor y más realista que intentar definir todos los requisitos al comienzo del proyecto e invertir esfuerzos después en controlar los cambios en los requisitos. Se puede considerar la programación extrema como la adopción de las mejores metodologías de desarrollo de acuerdo a lo que se pretende llevar a cabo con el proyecto, y aplicarlo de manera dinámica durante el ciclo de vida del software.

Los Valores originales de la programación extrema son: simplicidad, comunicación, retroalimentación (feedback) y coraje. Un quinto valor, respeto, fue añadido en la segunda edición de Extreme Programming Explained. Los cinco valores se detallan a continuación:

1.2.3.4.1 Simplicidad:

La simplicidad es la base de la programación extrema. Se simplifica el diseño para agilizar el desarrollo y facilitar el mantenimiento. Un diseño complejo del código junto a sucesivas modificaciones por parte de diferentes desarrolladores hace que la complejidad aumente exponencialmente. Para mantener la simplicidad es necesaria la refactorización del código, ésta es la manera de mantener el código simple a medida que crece. También se aplica la simplicidad en la documentación,

de esta manera el código debe comentarse en su justa medida, intentando eso sí que el código esté autodocumentado. Para ello se deben elegir adecuadamente los nombres de las variables, métodos y clases. Los nombres largos no decrementan la eficiencia del código ni el tiempo de desarrollo gracias a las herramientas de autocompletado y refactorización que existen actualmente. Aplicando la simplicidad junto con la autoría colectiva del código y la programación por parejas se asegura que cuanto más grande se haga el proyecto, todo el equipo conocerá más y mejor el sistema completo.

1.2.3.4.2 Comunicación:

La comunicación se realiza de diferentes formas. Para los programadores el código comunica mejor cuanto más simple sea. Si el código es complejo hay que esforzarse para hacerlo inteligible. El código autodocumentado es más fiable que los comentarios ya que éstos últimos pronto quedan desfasados con el código a medida que es modificado. Debe comentarse sólo aquello que no va a variar, por ejemplo el objetivo de una clase o la funcionalidad de un método. Las pruebas unitarias son otra forma de comunicación ya que describen el diseño de las clases y los métodos al mostrar ejemplos concretos de cómo utilizar su funcionalidad. Los programadores se comunican constantemente gracias a la programación por parejas. La comunicación con el cliente es fluida ya que el cliente forma parte del equipo de desarrollo. El cliente decide qué características tienen prioridad y siempre debe estar disponible para solucionar dudas.

1.2.3.4.3 Retroalimentación (feedback):

Al estar el cliente integrado en el proyecto, su opinión sobre el estado del proyecto se conoce en tiempo real. Al realizarse ciclos muy cortos tras los cuales se muestran resultados, se minimiza el tener que rehacer partes que no cumplen con los requisitos y ayuda a los programadores a centrarse en lo que es más importante. Considérense los problemas que derivan de tener ciclos muy largos.

Meses de trabajo pueden tirarse por la borda debido a cambios en los criterios del cliente o malentendidos por parte del equipo de desarrollo. El código también es una fuente de retroalimentación gracias a las herramientas de desarrollo. Por ejemplo, las pruebas unitarias informan sobre el estado de salud del código. Ejecutar las pruebas unitarias frecuentemente permite descubrir fallos debidos a cambios recientes en el código.

1.2.3.4.4 Coraje o valentía:

Los puntos anteriores parecen tener sentido común, entonces, ¿por qué coraje? Para los gerentes la programación en parejas puede ser difícil de aceptar, porque les parece como si la productividad se fuese a reducir a la mitad ya que solo la mitad de los programadores está escribiendo código. Hay que ser valiente para confiar en que la programación por parejas beneficia la calidad del código sin repercutir negativamente en la productividad. La simplicidad es uno de los principios más difíciles de adoptar. Se requiere coraje para implementar las características que el cliente quiere ahora sin caer en la tentación de optar por un enfoque más flexible que permita futuras modificaciones. No se debe emprender el desarrollo de grandes marcos de trabajo (frameworks) mientras el cliente espera. En ese tiempo el cliente no recibe noticias sobre los avances del proyecto y el equipo de desarrollo no recibe retroalimentación para saber si va en la dirección correcta. La forma de construir marcos de trabajo es mediante la refactorización del código en sucesivas aproximaciones.

1.2.3.4.5 Respeto:

El respeto se manifiesta de varias formas. Los miembros del equipo se respetan los unos a otros, porque los programadores no pueden realizar cambios que hacen que las pruebas existentes fallen o que demore el trabajo de sus compañeros. Los miembros respetan su trabajo porque siempre están luchando por la alta calidad en el producto y buscando el diseño óptimo o más eficiente para la solución a

través de la refactorización del código. Los miembros del equipo respetan el trabajo del resto no haciendo menos a otros, sino orientándolos a realizarlo mejor, obteniendo como resultado una mejor autoestima en el equipo y elevando el ritmo de producción en el equipo.

Las características fundamentales del método son:

- Desarrollo iterativo e incremental: pequeñas mejoras, unas tras otras.

- Pruebas unitarias continuas, frecuentemente repetidas y automatizadas, incluyendo pruebas de regresión. Se aconseja escribir el código de la prueba antes de la codificación.
- Programación en parejas: se recomienda que las tareas de desarrollo se lleven a cabo por dos personas en un mismo puesto. Se supone que la mayor calidad del código escrito de esta manera -el código es revisado y discutido mientras se escribe- es más importante que la posible pérdida de productividad inmediata.

- Frecuente integración del equipo de programación con el cliente o usuario. Se recomienda que un representante del cliente trabaje junto al equipo de desarrollo.

- Corrección de todos los errores antes de añadir nueva funcionalidad. Hacer entregas frecuentes.

- Refactorización del código, es decir, reescribir ciertas partes del código para aumentar su legibilidad y mantenibilidad pero sin modificar su comportamiento. Las pruebas han de garantizar que en la refactorización no se ha introducido ningún fallo.

- Propiedad del código compartida: en vez de dividir la responsabilidad en el desarrollo de cada módulo en grupos de trabajo distintos, este método

promueve el que todo el personal pueda corregir y extender cualquier parte del proyecto. Las frecuentes pruebas de regresión garantizan que los posibles errores serán detectados.

- Simplicidad en el código: es la mejor manera de que las cosas funcionen. Cuando todo funcione se podrá añadir funcionalidad si es necesario. La programación extrema apuesta que es más sencillo hacer algo simple y tener un poco de trabajo extra para cambiarlo si se requiere, que realizar algo complicado y quizás nunca utilizarlo.

La simplicidad y la comunicación son extraordinariamente complementarias. Con más comunicación resulta más fácil identificar qué se debe y qué no se debe hacer. Cuanto más simple es el sistema, menos tendrá que comunicar sobre éste, lo que lleva a una comunicación más completa, especialmente si se puede reducir el equipo de programadores. (12)

Hay diversas prácticas inherentes a la Programación Extrema, en cada uno de las fases de desarrollo del proyecto.

- Planificación del proyecto: Historias de usuario, planificación de entregas, iteraciones, velocidad del proyecto, reuniones diarias.
- Diseño: Diseños simples, Riesgos, Tarjetas C.R.C.
- Codificación: refactorizar, programación en pareja.
- Pruebas: Test de aceptación. (13)

1.3.2.5 SCRUM

SCRUM es un marco de trabajo para la gestión y desarrollo de software basada en un pr-oceso iterativo e incremental utilizado comúnmente en entornos basados en el desarrollo ágil de software. SCRUM es un modelo de referencia que define un conjunto de prácticas y roles, y que puede tomarse como punto de partida para definir el proceso de desarrollo que se ejecutará durante un proyecto. SCRUM permite la creación de equipos auto organizados impulsando la co-localización de

todos los miembros del equipo, y la comunicación verbal entre todos los miembros y disciplinas involucrados en el proyecto.

Un principio clave de SCRUM es el reconocimiento de que durante un proyecto los clientes pueden cambiar de idea sobre lo que quieren y necesitan, y que los desafíos impredecibles no pueden ser fácilmente enfrentados de una forma predictiva y planificada. Por lo tanto, SCRUM adopta una aproximación pragmática, aceptando que el problema no puede ser completamente entendido o definido, y centrándose en maximizar la capacidad del equipo de entregar rápidamente y responder a requisitos emergentes.

Existen varias implementaciones de sistemas para gestionar el proceso de SCRUM, que van desde notas amarillas "post-it" y pizarras hasta paquetes de software. Una de las mayores ventajas de SCRUM es que es muy fácil de aprender, y requiere muy poco esfuerzo para comenzarse a utilizar.

Aunque surgió como modelo para el desarrollo de productos tecnológicos, también se emplea en entornos que trabajan con requisitos inestables y que requieren rapidez y flexibilidad; situaciones frecuentes en el desarrollo de determinados sistemas de software.

Jeff Sutherland aplicó el modelo SCRUM al desarrollo de software en 1993 en Easel Corporation. En 1995 lo presentó junto con Ken Schwaber como proceso formal, también para gestión del desarrollo de software en OOPSLA 95. Más tarde, en 2001 serían dos de los promulgadores del Manifiesto ágil. En el desarrollo de software SCRUM está considerado como modelo ágil por la Agile Alliance.

La ficha adjunta incluye una descripción sinóptica del proceso y sus elementos que son:

- Roles: Propietario del producto, Gestor o Manager del SCRUM, Equipo e Interesados.

- Componentes del proceso: Pila del producto (Product Backlog), Pila del sprint (Sprint Backlog), Incremento.
- Reuniones: Planificación del sprint, Revisión diaria, Revisión del sprint.

El Sprint es el período en el cual se lleva a cabo el trabajo en sí. Es recomendado que la duración de los sprints sea constante y definida por el equipo en base a su propia experiencia. Se puede comenzar con una duración de sprint en particular (2 o 3 semanas) e ir ajustándolo en base al ritmo del equipo, aunque sin relajarlo demasiado. Al final de cada sprint, el equipo deberá presentar los avances logrados, y deberían entregar un producto con características de utilizable por el cliente. (14)

1.3.2.6 SXP

SXP es un híbrido cubano de metodologías ágiles que tiene como base las metodologías SCRUM y XP que permiten actualizar los procesos de desarrollo de software para el mejoramiento de su producción. Consta de 4 fases: Planificación-Definición, Desarrollo, Entrega y Mantenimiento, cada una desglosada en flujos de trabajo y actividades que generan artefactos. Esta metodología ayuda a fortalecer el trabajo en equipo, enfocados en una misma dirección, permitiendo además seguir de forma clara el avance de las tareas a realizar, a partir de la inserción de procedimientos ágiles que permitan actualizar los procesos de software para el mejoramiento de la producción, aumentando el nivel de interés del equipo. (15)

XP fue la metodología candidata para guiar el proceso ingenieril, puesto que le precedía su alto grado de aceptación por la comunidad internacional de desarrollo ágil, además que nos facilitaba una documentación más discreta y mayor dinamismo para el desarrollo; la idea de las duplas de desarrollo para el grupo de investigadores resultó muy interesante, pues en pequeñas iteraciones dos desarrolladores lograrían hacer, lo que antes un equipo especializado en cada tema debía hacer (analista, arquitecto, diseñador, desarrollador, probador). SCRUM es entonces la metodología ideal para toda

la gestión de proyectos, serviría de soporte para acelerar el dinamismo que se identificó en XP, la identificación de los pequeños sprint (iteraciones) y las reuniones con el SCRUM Máster todos los días se acercaba más a la disciplina que se quería alcanzar en el grupo, donde líderes de solución y equipo de desarrollo se reunieran y controlaran los avances e identificaran los posibles riesgos que afectaban de una manera u otra la correcta ejecución del proyecto. (16)

1.3.3 ¿Por qué aplicar SXP?

La tendencia hoy en día, es obtener productos de software en el menor tiempo posible y elaborar la documentación necesaria.

Con el uso de esta metodología, se pretende evitar errores clásicos como:

- Planificaciones a muy largo plazo que nunca se cumplen a tiempo.
- Las desviaciones sobre el plan sólo se detectan al final, cuando ya es tarde y la única opción es codificar por horas y horas para cumplir los plazos definidos.
- Desarrollos que no cumplen con la calidad deseable, muchos errores en gran medida provocados por las prisas para entregar algo y cumplir con la planificación inicial.

Evidentemente ningún desarrollo está exento de dificultades, no existe ninguna metodología mágica que resuelva todas las situaciones negativas que se nos presentan a lo largo del camino, pero los métodos ágiles ofrecen herramientas para tratar de resolver algunos de estos problemas:

- Entregas periódicas en ciclos cortos, es decir, dar pasitos pequeños pero seguros. Los **Sprint** de SCRUM se ajustan perfectamente a esta idea.
- Pruebas unitarias, integración continua, definir una guía de estilo, refactorizaciones, muchas de las prácticas de XP nos parecían un buen modo de mejorar la calidad de nuestros desarrollos.
- Las reuniones de SCRUM diarias son muy efectivas para el control del estado del desarrollo en un momento determinado.

Lo más importante para el desarrollo de las aplicaciones a construir es su funcionalidad es decir, que el sistema pueda estar en explotación lo más pronto posible. Se piensa añadirle funcionalidad al mismo en futuras versiones pero lo que se desea de inmediato es una funcionalidad mínima del mismo por la importancia que tiene. Se necesita la documentación mínima necesaria para los futuros mantenimientos del producto final. El desarrollo del software se hizo en la misma empresa donde se va a implantar, por lo que los usuarios finales se encontraban en la misma y era fácil la presencia de ellos durante el desarrollo de la aplicación.

1.4 Tecnologías a utilizar

En este caso, el sistema a construir se debe adaptar a cierto sistema ya existente lo que disminuye el rango de búsqueda en cuanto a servidor de base de datos se trata. La empresa utiliza el Microsoft SQL Server, por lo que se decide utilizar el mismo para lograr mayor compatibilidad con la fuente de datos existente. Además, por las ventajas que presenta la integración ofrecida por los productos de Microsoft, se decidió utilizar el .Net Framework como marco de trabajo base del sistema a desarrollar, seleccionando el lenguaje CSharp para implementar la aplicación.

1.4.1 Microsoft SQL Server

Microsoft SQL Server es un sistema para la gestión de bases de datos producido por Microsoft basado en el modelo relacional. Sus lenguajes para consultas son T-SQL y ANSI SQL. Microsoft SQL Server constituye la alternativa de Microsoft a otros potentes sistemas gestores de bases de datos como son Oracle, PostgreSQL o MySQL.

Incluye entre sus características:

- Soporte de transacciones.
- Escalabilidad, estabilidad y seguridad.

- Soporta procedimientos almacenados.
- Incluye también un potente entorno gráfico de administración, que permite el uso de comandos DDL y DML gráficamente.
- Permite trabajar en modo cliente-servidor, donde la información y datos se alojan en el servidor y los terminales o clientes de la red sólo acceden a la información.
- Además permite administrar información de otros servidores de datos.

Este sistema incluye una versión reducida, llamada MSDE con el mismo motor de base de datos pero orientado a proyectos más pequeños, que en sus versiones 2005 y 2008 pasa a ser el SQL Express Edition, que se distribuye en forma gratuita.

T-SQL (Transact-SQL) es el principal medio de programación y administración de SQL Server. Expone las palabras clave para las operaciones que pueden realizarse en SQL Server, incluyendo creación y modificación de esquemas de la base de datos, introducir y editar datos en la base de datos, así como supervisión y gestión del propio servidor. Las aplicaciones cliente, ya sea que consuman datos o administren el servidor, aprovechan la funcionalidad de SQL Server mediante el envío de consultas de T-SQL y declaraciones que son procesadas por el servidor y los resultados (o errores) regresan a la aplicación cliente. SQL Server permite que sean administrados mediante T-SQL. Para esto, expone tablas de sólo lectura con estadísticas del servidor. La funcionalidad para la administración se expone a través de procedimientos almacenados definidos por el sistema que se pueden invocar desde las consultas de T-SQL para realizar la operación de administración. También es posible crear servidores vinculados (Linked Servers) mediante T-SQL. Los servidores vinculados permiten el funcionamiento entre múltiples servidores con una consulta.

El cliente Nativo de SQL es la biblioteca de acceso a datos para los clientes de Microsoft SQL Server versión 2005 en adelante. Implementa nativamente soporte

para las características de SQL Server, incluyendo la ejecución de la secuencia de datos tabular, soporte para bases de datos en espejo de SQL Server, soporte completo para todos los tipos de datos compatibles con SQL Server, conjuntos de operaciones asíncronas, las notificaciones de consulta, soporte para cifrado, así como recibir varios conjuntos de resultados en una sola sesión de base de datos. El cliente Nativo de SQL se utiliza como extensión de SQL Server plug-ins para otras tecnologías de acceso de datos, incluyendo ADO u OLE DB. El cliente Nativo de SQL puede también usarse directamente, pasando por alto las capas de acceso de datos genéricos. (17)

1.4.2 Microsoft Visual CSharp (C#)

C# es un lenguaje de programación orientado a objetos desarrollado y estandarizado por Microsoft como parte de su plataforma .NET, que después fue aprobado como un estándar por la ECMA e ISO. (18)

Su sintaxis básica deriva de C/C++ y utiliza el modelo de objetos de la plataforma .NET, similar al de Java aunque incluye mejoras derivadas de otros lenguajes (entre ellos Delphi). El nombre C Sharp fue inspirado por la notación musical, donde ♯ (sostenido, en inglés sharp) indica que la nota (C es la nota do en inglés) es un semitono más alta, sugiriendo que C♯ es superior a C/C++. (19)

C♯, como parte de la plataforma.NET, está normalizado por ECMA desde diciembre de 2001 (C# Language Specification "Especificación del lenguaje C♯"). El 7 de noviembre de 2005 salió la versión 2.0 del lenguaje, que incluía mejoras tales como tipos genéricos, métodos anónimos, iteradores, tipos parciales y tipos anulables. El 19 de noviembre de 2007 salió la versión 3.0 de C#, destacando entre las mejoras los tipos implícitos, tipos anónimos y LINQ (Language Integrated Query -consulta integrada en el lenguaje).

Aunque C♯ forma parte de la plataforma.NET, ésta es una interfaz de programación de aplicaciones (API), mientras que C♯ es un lenguaje de

programación independiente diseñado para generar programas sobre dicha plataforma. Ya existe un compilador implementado que provee el marco de DotGNU - Mono que genera programas para distintas plataformas como Win32, UNIX y Linux.

1.5 Conclusiones

En este capítulo se hace un estudio sobre el funcionamiento de los presupuestos particularizando nuestro problema en especial y se exponen los conceptos necesarios para el entendimiento del software, se toman decisiones importantes, luego de un estudio de las distintas herramientas para el desarrollo de software tales como la elección de los lenguajes de programación y metodologías a utilizar.

CAPÍTULO 2: PLANIFICACIÓN-DEFINICIÓN

2.1 Introducción

En este capítulo, se introduce la fase de planeación y diseño, donde se detallan las necesidades del cliente, se describen las funcionalidades que serán objeto de automatización mediante el empleo de las historias de usuarios (HU), se realiza una estimación del esfuerzo necesario para las mismas y se establece un plan de iteraciones necesarias sobre el sistema para su terminación.

2.2 Solución propuesta

La solución al problema propuesto es desarrollar un sistema informático que permita la gestión y el control de los presupuestos. El sistema a implementar debe ser capaz de gestionar toda la información que se necesita a lo largo del flujo de trabajo actual, la gestión de Sucursales y Centros de Costo, la gestión de los Presupuestos y la visualización de la información teniendo en cuenta la estructura organizativa de la empresa, además debe ser capaz de controlar que los accesos de los usuarios del sistema estén de acuerdo a su función dentro del organismo, además proteger la información de personas no autorizadas.

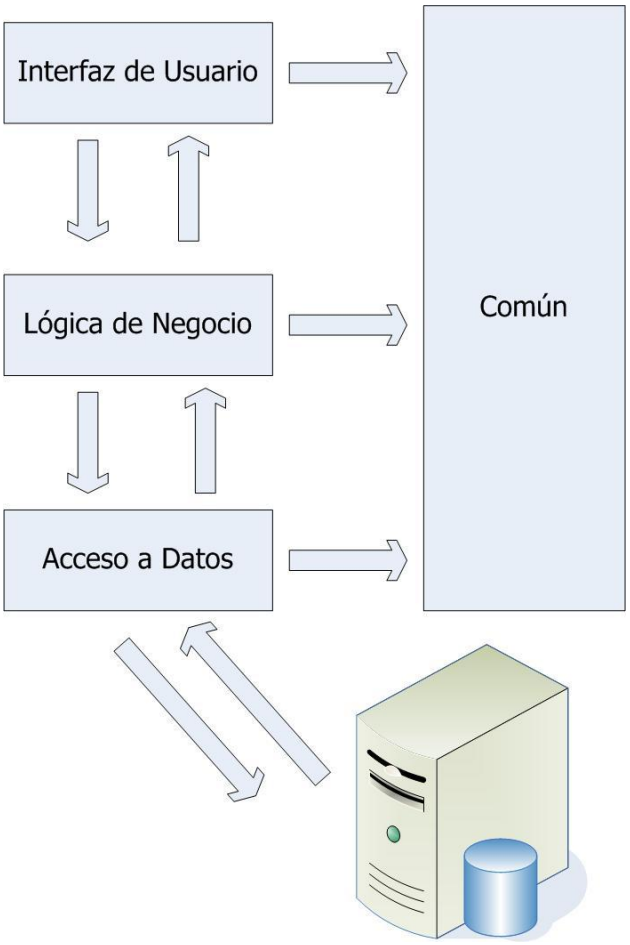
Se decidió utilizar una arquitectura cliente servidor de dos niveles donde el cliente (cada PC donde esté instalada la aplicación) solicita recursos y el servidor (servidor de base de datos) responde directamente a la solicitud, con sus propios recursos. La programación por capas es un estilo de programación en el que el objetivo primordial es la separación de la lógica de negocios de la lógica de diseño; un ejemplo básico de esto consiste en separar la capa de datos de la capa de presentación al usuario. La ventaja principal de este estilo es que el desarrollo se puede llevar a cabo en varios niveles y, en caso de que sobrevenga algún cambio, sólo se ataca al nivel requerido sin tener que revisar entre código mezclado. Además, permite distribuir el trabajo de creación de una aplicación por niveles; de

este modo, cada grupo de trabajo está totalmente abstraído del resto de niveles. (20)

Por las ventajas que ofrece, decidimos utilizar este estilo de programación en nuestro proyecto, dividiendo el mismo en cinco capas:

1. **Interfaz de Usuario:** como lo dice el nombre, es la que ve el usuario, en la misma presenta el sistema al usuario, le comunica la información y captura la información del usuario en un mínimo de proceso (realiza un filtrado previo para comprobar que no hay errores de formato). También es conocida como interfaz gráfica y debe tener la característica de ser "amigable" (entendible y fácil de usar) para el usuario. Esta capa se comunica únicamente con la capa de lógica de negocio.
2. **Lógica de Negocio:** Es aquí donde se establecen todas las reglas del negocio que deben cumplirse. Esta capa se comunica con la interfaz de usuario, para recibir las solicitudes y presentar los resultados, y con la capa de acceso datos para solicitar el almacenamiento y recuperación de datos.
3. **Acceso a Datos:** En esta capa, solo se realiza el acceso a los datos. De esta manera cuando es necesario cambiar el motor de base de datos, solamente tendremos que corregir esta capa.
4. **Común:** En esta capa se encuentra la declaración de las entidades de la aplicación, de manera que se pueden referenciar desde otras capas sin entrar en ciclos recursivos de compilación.
5. **Base de Datos:** En esta capa es donde están los datos y se corresponde directamente con la definición de esquemas, tablas, vistas, procedimientos almacenados y todo lo que se pueda o deba poner en un motor de base de datos.

Cada capa contiene funciones específicas dentro de la aplicación, y cada una de ellas interactúa solo con la capa de inferior nivel dentro de la arquitectura como se muestra en la figura a continuación.



2.3 Planificación del Proyecto

La planificación-definición constituye la primera fase de la metodología SXP, en esta se definen todos los procesos a automatizar, así como el tiempo que se determinará para realizar cada una de estas aplicaciones. Por tal motivo es la encargada de generar toda la documentación correspondiente a la concepción inicial del sistema, su definición, además incluye algunos vinculados a la primera fase de los procesos de ingeniería de Software. Por otra parte involucra documentos relacionados con la estimación inicial de esfuerzos y la valoración de los riesgos. Las planillas surgen a partir de cada actividad en específico y constituyen un baluarte para la documentación del Software, estas cobran suma

importancia ya que en ellas se detalla la razón de su existencia, la descripción de los objetivos a tener en cuenta.

2.3.1 Plantilla Concepción del sistema

Primer documento generado en la fase de Planificación-Definición, este recoge los detalles relacionados con las actividades entrevistas y/o encuestas al cliente. Este documento, además de reflejar la visión general del producto a implementar, también recoge los diferentes roles que intervendrán en el desarrollo del software, se documenta el tipo de proyecto, las herramientas que serán utilizadas para el desarrollo de la aplicación, el alcance que va a tener, una descripción de los involucrados en el negocio, cuales son los motivos de la necesidad del desarrollo del software y la propuesta de solución. Esta es la plantilla principal dentro de la documentación por ser la guía para los demás documentos que se generan durante el ciclo de desarrollo de software.

Para el caso del presente proyecto se realizó una reunión con el cliente, en la cual se especificó con más claridad el flujo de trabajo, la información que se genera desde la planificación de los presupuestos hasta el control de su ejecución y otras necesidades que no podían ser satisfechas en ese momento debido a lo engorroso lento que resultaría el proceso, lo cual debe ser corregido en nuestra solución al problema y brindar al cliente esta funcionalidad. A partir de los resultados arrojados por la reunión realizada, se determinó que el producto a desarrollar debe estar acorde con las necesidades imperantes para los usuarios finales. Para ello se realizó un análisis de todos los usuarios que participan en el proceso y sus formas independientes de interactuar con el mismo, con el objetivo de cubrir todos los requerimientos previstos por cada uno de ellos y además reconocer los roles que se pueden crear para agruparlos y tener una perspectiva de las necesidades de cada uno en nuestra aplicación.

Roles: Analista y Cliente.

2.3.2 Plantilla Modelo de Historias de usuario del negocio

La plantilla del Modelo Historias de usuario del negocio, es un artefacto que se genera del Juego de la planificación, luego de estar definida la concepción del sistema, se hace mucho más fácil comprender el negocio.

Se definen las características específicas del negocio, así como la forma en que interactúa el sistema con el cliente y viceversa. El Modelo de negocio cuando se trabaja con metodologías ágiles, es diferente al ya conocido en el proceso unificado, ya que en este caso se trabaja con historias de usuarios, en vez de con casos de uso.

Roles: Analista

2.3.3 Plantilla Lista de Reserva del Producto (LRP)

La plantilla de Lista de Reserva del Producto, es el primer artefacto generado en la etapa de captura de requisitos, está conformada por una lista priorizada que define el trabajo que se va a realizar en el proyecto. Cuando un proyecto comienza es muy difícil tener claro todos los requerimientos sobre el producto. Sin embargo, suelen surgir los más importantes que casi siempre son más que suficientes para una iteración.

Esta lista puede crecer y modificarse a medida que se obtienen más conocimientos acerca del producto y del cliente. Con la restricción de que sólo puede cambiarse entre iteraciones. El objetivo es asegurar que el producto definido al terminar la lista es el más correcto, útil y competitivo posible y para esto la lista debe acompañar los cambios en el entorno y el producto. Esta lista puede estar conformada por requerimientos técnicos y del negocio, funciones, errores a reparar, defectos, mejoras y actualizaciones tecnológicas requeridas.

Roles: Analista y Cliente

2.3.4 Plantilla Historias de usuario

Una historia de usuario es una representación de un requerimiento de software escrito en una o dos frases utilizando el lenguaje común del usuario. Las historias de usuario son utilizadas en las metodologías de desarrollo ágiles para la especificación de requerimientos (acompañadas de las discusiones con los usuarios y las pruebas de validación). Cada historia de usuario debe ser limitada, esta debería poderse escribir sobre una nota adhesiva pequeña. Dentro de la metodología XP las historias de usuario deben ser escritas por los clientes.

Las historias de usuario son una forma rápida de administrar los requerimientos de los usuarios sin tener que elaborar gran cantidad de documentos formales y sin requerir de mucho tiempo para administrarlos. Las historias de usuario permiten responder rápidamente a los requerimientos cambiantes.

Beneficios

- Al ser muy corta, ésta representa requisitos del modelo de negocio que pueden implementarse rápidamente (días o semanas)
- Necesitan poco mantenimiento
- Mantienen una relación cercana con el cliente
- Permite dividir los proyectos en pequeñas entregas
- Permite estimar fácilmente el esfuerzo de desarrollo
- Es ideal para proyectos con requerimientos volátiles o no muy claros (21)

Las historias de usuario se encargan de dirigir la construcción de las pruebas de aceptación (deben generarse una o más pruebas para verificar que la historia ha sido correctamente implementada). En el momento de implementar una historia de usuario, se debe detallar a través de la comunicación con el cliente.

Roles: Analista y Cliente

2.3.4.1 Estimación de esfuerzo

Las historias de usuario servirán para crear el plan estimado de entrega. Se convocará una reunión para crear el plan de entregas, que se usará para crear los planes de iteración para cada iteración. Con cada historia de usuario previamente evaluada en tiempo de desarrollo ideal, el cliente las agrupará en orden de importancia. Una semana ideal es el tiempo que costaría implementar dicha historia si no tenemos nada más que hacer, incluyendo la parte de test correspondiente. Por otra parte, se mantiene un registro de la "velocidad" de desarrollo, establecida en puntos por iteración, basándose principalmente en la suma de puntos correspondientes a las historias de usuario que fueron terminadas en la última iteración. La planificación se puede realizar basándose en el tiempo o el alcance. La velocidad del proyecto es utilizada para establecer cuántas historias se pueden implementar antes de una fecha determinada o cuánto tiempo tomará implementar un conjunto de historias.

De esta forma se puede trazar el plan de entregas en función de estos dos parámetros: tiempo de desarrollo ideal y grado de importancia para el cliente. Las iteraciones individuales son planificadas en detalle justo antes de que comience cada iteración.

2.3.5 Plantilla Lista de riesgos

La plantilla de Lista de riesgos, es el documento que se genera de la actividad de Valoración de riesgos. En ella quedan definidos los posibles riesgos que actuarán sobre el proceso de desarrollo de software, así como la estrategia trazada, además de un plan de contingencia que describe que curso seguirán las acciones si el riesgo se materializa.

Roles: Gerente

2.4 Diseño del proyecto

El Diseño de Sistemas se define el proceso de aplicar ciertas técnicas y principios con el propósito de definir un dispositivo, un proceso o un Sistema, con suficientes detalles como para permitir su interpretación y realización física. (22)

El Diseño es la única manera de materializar con precisión los requerimientos del cliente, es un proceso y un modelado a la vez, con un conjunto de pasos repetitivos que permiten al diseñador describir todos los aspectos del sistema a construir. El diseño debe implementar todos los requisitos explícitos contenidos en la Lista de Reserva. Debe ser una guía que puedan leer y entender los que construyan el código y los que prueban y mantienen el software. El diseño debe proporcionar una completa idea de lo que es el software, enfocando los dominios de datos, funcional y comportamiento desde el punto de vista de la Implementación.

La Metodología XP sugiere que hay que conseguir diseños simples y sencillos. Hay que procurar hacerlo todo lo menos complicado posible para conseguir un diseño fácilmente entendible e implementable que a la larga costará menos tiempo y esfuerzo desarrollar.

2.4.1 Plantilla Modelo de diseño

La plantilla del Modelo de diseño, es el documento que se genera del diseño con las metáforas, donde se debe diseñar la solución más simple que pueda funcionar y ser implementada en un momento determinado del proyecto. En XP no se enfatiza la definición temprana de una arquitectura estable para el sistema. Dicha arquitectura se asume de forma evolutiva y los posibles inconvenientes que se generarían por no contar con ella explícitamente en el comienzo del proyecto se solventan con la existencia de una metáfora.

Roles: Diseñador

2.5 Planificación de las iteraciones

En este plan se establece cuántas iteraciones serán necesarias realizar sobre el sistema para su terminación. El plan de iteraciones puede incluir indicaciones sobre cuáles Historias de Usuario se incluirán en un release, lo cual debería ser consistente con el contenido de una o dos iteraciones.

2.5.1 Primera iteración: Gestionar Sucursales, Gestionar Centros de Costo, Gestionar Presupuestos y Gestionar Corporativos.

Esta primera iteración tiene como objetivo darle cumplimiento a las Historias de Usuario (HU) que se consideraron de importancia inicial para el desarrollo del sistema. Las HU mencionadas brindan funcionalidades que gestionan la información más importante con que trataremos en el sistema, es decir, la información base de todo el proceso de gestión de presupuestos.

En esta etapa solo se tendrán las funcionalidades mínimas requeridas por el cliente. Además se tendrá la primera versión de prueba, que contará con una vista inicial del sistema que proveerá al cliente de una base para definir con más exactitud sus necesidades. Esta primera versión se le presentará al cliente con el objetivo de obtener una retroalimentación del mismo para posteriores iteraciones del producto. Esta versión solo será distribuida a un mínimo de usuarios para su prueba inicial.

Gestionar las Sucursales y los Centros de Costo, permitirá definir la estructura organizativa de la empresa dentro del sistema, y la gestión de los Presupuestos y de los Presupuestos Corporativos brindará al cliente la posibilidad de manejar los datos de la planificación de los mismos en correspondencia con esta estructura.

2.5.2 Segunda iteración: Visualizar Reportes, Visualizar Dashboard y Gestionar Usuarios.

En esta iteración se les dará cumplimiento a las Historias de Usuario que son requisitos complementarios de la aplicación. La implementación de los reportes permitirá visualizar la información gestionada en la iteración anterior, de una manera rápida y eficiente, además permitirá al cliente presentar la información de

diferentes maneras que hasta ese momento se dificultaba por lo lento y engorroso que resultaba el manejo de los datos. El dashboard es un elemento que hará nuestra aplicación más llamativa e interesante al usuario, pues brindará información general en forma de gráficas y otros elementos visuales que permitirá conocer cómo se comportan los parámetros básicos de los presupuestos de un vistazo. La gestión de los usuarios permitirá agruparlos por perfiles y niveles de acceso que garantizará la seguridad de la información de manera transparente para el usuario. Solamente se distribuirá la aplicación al resto de los usuarios después de esta iteración.

El resultado de esta iteración integrada a la iteración anterior dará como resultado la segunda versión del sistema, además de la segunda versión de prueba del mismo. Esta versión será entregada al cliente para verificar si cumple con los requisitos acordados.

2.5.3 Tercera iteración: Gestión de la Conexión a la Base de Datos

En esta iteración se tiene como objetivo darle cumplimiento las Historias de Usuario Gestión de la Conexión a la Base de Datos. Esto es una función especial solicitada por el cliente para evitar redistribuir la aplicación en caso de que se tenga que mover la base de datos para un servidor nuevo, dándole la posibilidad a los usuarios de gestionar la conexión a la base de datos.

El resultado alcanzado en esta iteración será integrado a los resultados de las iteraciones anteriores obteniéndose la tercera versión del sistema.

2.5.4 Plan de duración de las iteraciones

Partiendo de las historias de usuario planteadas en la planilla correspondiente a la misma se realiza una planificación en tres iteraciones basándose en el tiempo y procurando agrupar la funcionalidad relacionada en la misma iteración.

Iteración	Orden de implementación por historia de usuario	Duración total de la iteración en semanas
1	✓ Gestionar Sucursales ✓ Gestionar Centros de Costo ✓ Gestionar Presupuestos ✓ Gestionar Corporativos	1.5 + 1 + 2.5 +1 = 6
2	✓ Visualizar Reportes ✓ Visualizar Dashboard ✓ Gestionar Usuarios	3 + 2 + 1.5 = 6.5
3	✓ Gestión de la Conexión a la Base de Datos	1

Tabla 2.1: Planificación de la iteración en semanas

2.6 Tarjetas CRC

Las tarjetas CRC (clase, responsabilidad y colaboración) son una metodología para el diseño de software orientado a objetos creada por Kent Beck y Ward Cunningham. (23)

Las tarjetas CRC permiten que el equipo completo contribuya en la tarea del diseño. Una tarjeta CRC representa un objeto. El nombre de la clase se coloca a modo de título en la tarjeta, los atributos y las responsabilidades más significativas se colocan a la izquierda, y las clases que están implicadas en cada responsabilidad a la derecha, en la misma línea que su requerimiento correspondiente. En este documento solo expondremos un ejemplo del uso de dicha metodología, el resto se documenta en el expediente del proyecto.

Partiendo de la arquitectura del sistema propuesta, encontraremos tres tipos de clases: las Utilitarias o Comunes, que nos permitirán la persistencia de los datos de la base de datos; las de Acceso a Datos, que se encargarán de utilizar las clases Utilitarias para enviar y recibir estos datos y las clases de Lógica de Negocio, que

utilizarás las clases Utilitarias para validar las reglas del negocio planteadas por el cliente.

Nombre de la clase: Sucursal	
Tipo de la clase: Utilitaria	
Responsabilidades:	**Colaboradores:**
Identificador de la Sucursal	CentroCosto
Nombre de la Sucursal	
Lista de CentroCosto	

Nombre de la clase: Sucursal_DAL	
Tipo de la clase: Acceso a Datos	
Responsabilidades:	**Colaboradores:**
Insertar Sucursal	Sucursal
Modificar Sucursal	CentroCosto
Eliminar Sucursal	
Listado de Sucursales	
Adicionar CentroCosto a la Lista	
Eliminar CentroCosto de la Lista	

Nombre de la clase: Sucursal_BLL	
Tipo de la clase: Lógica de Negocios	
Responsabilidades:	**Colaboradores:**
Validar Datos de la Sucursal	Sucursal
	Sucursal_DAL

2.7 Conclusiones

En este capítulo hemos abordado la fase de planificación y definición en el cual se explicó parte de la dinámica del proyecto basándose en el expediente del proyecto, se planificaron las historias de usuario que se deben tener en cuenta para la construcción del sistema, así como las iteraciones a partir de la estimación de esfuerzos de cada historia de usuario. El desarrollo del sistema fue planificado en tres iteraciones. Se concluye que se tiene la información necesaria para pasar a la siguiente fase del proyecto.

CAPÍTULO3: DESARROLLO DEL SISTEMA

3.1 Introducción

En este capítulo se construye la solución propuesta de forma iterativa, tal y como indica la metodología XP. Una iteración es una entrega del proyecto al cliente que está incompleta, pero tiene implementadas algunas funcionalidades. La siguiente iteración es otra entrega con alguna funcionalidad más y así sucesivamente hasta llegar a la iteración final, en la que se entrega el software acabado. En este proyecto vale destacar que el desarrollo de las iteraciones ha sido ajustado de forma tal que al final de cada una de ellas, se tenga un entregable del sistema que el cliente puede probar y validar que cumple con sus requerimientos establecidos en la Lista de Reserva del Producto.

3.2 Desarrollo de las Iteraciones

Durante la Fase de Planificación - Definición se detallaron las Historias de Usuario correspondientes a cada una de las iteraciones a desarrollar, teniendo en cuenta las prioridades y restricciones de tiempos previstas por el cliente. Para darle cumplimiento a cada Historias de Usuario, primeramente se realiza una revisión del plan de iteraciones y de ser necesario se realizan modificaciones. Dentro del contenido de este plan se descomponen las Historias de Usuario en Tareas Ingeniería (TI) o tareas de programación, asignándole de esta forma un equipo de desarrollo (o una persona) que será el responsable de su implementación, con ellas se pretende cumplir con las funcionalidades básicas que luego conformarán las funcionalidades generales de cada historia. Las tareas no tienen que necesariamente ser entendidas por el cliente, pueden ser escritas en lenguajes técnicos, pues las mismas son usadas únicamente por los programadores. En este documento, se presentaran ejemplos de las tareas de ingeniería, el resto será expuesto en el documento correspondiente en el expediente del proyecto.

3.2.1 Primera iteración.

En esta iteración se da cumplimiento a la implementación de las Historias de Usuario que se consideraron de importancia primordial para el desarrollo y funcionamiento básico del sistema, con el fin de obtener una versión inicial del producto con algunas de las funcionalidades básicas del usuario. Se incluyen las Historias de Usuarios de la siguiente tabla, además de la estimación del tiempo de desarrollo y el tiempo real.

Historias de Usuario	Tiempo de Implementación (semanas)	
	Estimación	Real
Gestionar Sucursales	1.5	1
Gestionar Centros de Costo	1.5	1.5
Gestionar Presupuestos	2.5	2.5
Gestionar Corporativos	1	1

Tabla 3.1: Historias de Usuarios de la 1ra Iteración

3.2.1.1 Historia de Usuario Gestionar Sucursales.

Una sucursal es "dicho de un establecimiento: Que, situado en distinto lugar que la central de la cual depende, desempeña las mismas funciones que esta" (24). UNEVOL S.A, con el objetivo de ampliar sus frentes de trabajo en todo el país, ha creado distintas sucursales en diferentes puntos importantes de la economía cubana, como lo muestra la foto a continuación.

Los presupuestos de la empresa, se dividen precisamente por sucursales para facilitar su interpretación para los directivos de cada sucursal. La gestión de la información de las sucursales es primordial para establecer la base del control de la ejecución de los presupuestos en nuestro sistema. A continuación se describe una de las tareas de ingeniería perteneciente a esta historia de usuario.

Tarea de Ingeniería	
Número Tarea: 1	**Número Historia de Usuario:** 1
Nombre Tarea: Adicionar Sucursal	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 1
Fecha inicio: 21 de marzo 2011	**Fecha fin:** 25 de marzo 2011
Programador responsable: Luis A. Cortina Almeida	
Descripción: El sistema debe presentar un formulario para insertar los datos de la Sucursal.	

Tabla 3.2: Tarea de ingeniería # 1 Adicionar Sucursales

3.2.1.2 Historia de Usuario Gestionar Centros de Costo.

Las organizaciones pueden optar por clasificar las diferentes áreas en centros de costo. Existen ventajas de clasificar simple y sencillamente a las divisiones de las

sucursales como centros de costo ya que con esto los costos son más fácilmente medibles. Eso nos permite obtener un nivel más detallado de los problemas que enfrenta o puede enfrentar la organización en general, además de mejorar el control sobre las operaciones que se realizan en cada sucursal en particular.

Tarea de Ingeniería	
Número Tarea: 4	**Número Historia de Usuario:** 2
Nombre Tarea: Adicionar Centro de Costo a Sucursal	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 1.5
Fecha inicio: 28 de marzo 2011	**Fecha fin:** 6 de abril 2011
Programador responsable: Luis A. Cortina Almeida	
Descripción: El sistema debe presentar un formulario para insertar los datos de los Centros de Costo y definir a que sucursal pertenece.	

Tabla 3.3: Tarea de ingeniería # 4 Adicionar Centro de Costo a Sucursal

3.2.1.3 Historia de Usuario Gestionar Presupuestos.

La gestión de los presupuestos, es la columna vertebral de esta aplicación. En esta historia de usuario, se definen un conjunto de funcionalidades que el sistema debe implementar que forman parte del proceso de trabajo actual, o son cambios que mejoran el flujo de trabajo creando facilidades adicionales al cliente que eran imposibles de tener antes de la utilización de esta herramienta.

Tarea de Ingeniería	
Número Tarea: 7	**Número Historia de Usuario:** 3
Nombre Tarea: Adicionar Presupuesto	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 2.5
Fecha inicio: 6 de abril 2011	**Fecha fin:** 22 de abril 2011
Programador responsable: Luis A. Cortina Almeida	
Descripción: El sistema debe presentar un formulario para insertar los datos de los presupuestos.	

Tabla 3.4: Tarea de ingeniería # 7 Adicionar Presupuesto

3.2.1.4 Historia de Usuario Gestionar Presupuestos Corporativos.

El presupuesto corporativo, agrupa un presupuesto de cada sucursal con el objetivo de medir el comportamiento de los presupuestos de una manera consolidada. Con estos se puede obtener una vista clara de la empresa en general y permite su presentación a juntas de accionistas y otras reuniones que requieran consolidar la información de la entidad en general.

Tarea de Ingeniería	
Número Tarea: 12	**Número Historia de Usuario:** 4
Nombre Tarea: Adicionar Presupuesto Corporativo	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 1
Fecha inicio: 25 de abril 2011	**Fecha fin:** 29 de abril 2011
Programador responsable: Luis A. Cortina Almeida	
Descripción El sistema debe presentar un formulario para introducir los datos del presupuesto corporativo y permitir la selección de los presupuestos que formaran parte del mismo.	

Tabla 3.5: Tarea de ingeniería # 12 Adicionar Presupuesto Corporativo

3.2.2 Segunda iteración.

En esta iteración se da cumplimiento a otras historias de usuarios que brindan funcionalidades complementarias al sistema.

Historias de Usuario	**Tiempo de Implementación (semanas)**	
	Estimación	**Real**
Visualizar Reportes	3	2
Visualizar Dashboard	3	3
Gestionar Usuarios	1.5	1.5

Tabla 3.6: Historias de Usuarios de la 2da Iteración

3.2.2.1 Historia de Usuario Implementar Reportes.

Los reportes son una manera muy eficaz de presentar la información de un sistema. Permiten al usuario obtener modelos que posteriormente pueden ser utilizados en reuniones, informes y en documentos de texto que pueden ser fácilmente manipulados por otros sistemas, debido a su capacidad de exportar a formatos de documentos de trabajo que son utilizados por la mayoría de las empresas. En nuestro caso, tendremos reportes de la ejecución de los presupuestos y de los presupuestos corporativos, además de otros reportes solicitados por el cliente que se desprenden de la información de este flujo de trabajo.

Tarea de Ingeniería	
Número Tarea: 15	**Número Historia de Usuario:** 5
Nombre Tarea: Visualizar reportes de Presupuestos.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 1
Fecha inicio: 2 de mayo 2011	**Fecha fin:** 13 de mayo 2011
Programador responsable: Luis A. Cortina Almeida	
Descripción El sistema debe presentar una lista de los presupuestos para que el usuario seleccione el que va a visualizar, además un formulario para la entrada de los parámetros del reporte.	

Tabla 3.7: Tarea de ingeniería # 15 Visualizar reportes de Presupuestos

3.2.2.2 Historia de Usuario Visualizar Dashboard.

El nombre Dashboard se refiere al tablero de un automóvil, el cual ofrece al conductor información permanente sobre el estado del vehículo. El mundo de los negocios toma la palabra con un sentido similar pero en lugar de aplicarlo a los automóviles lo refiere a la empresa. Así Dashboard es una página desarrollada en base a tecnología web (en nuestro caso basado en un componente de aplicaciones

de escritorio)[1] mediante la cual se despliega en tiempo real información de la empresa extraída de varias fuentes o bases de datos. Su característica de tiempo real otorga a los usuarios un conocimiento completo sobre la marcha de la empresa y permite hacer análisis instantáneos e inteligencia de negocios. (25)

En nuestro caso, estará en enfocada en la ejecución del presupuesto de manera general, permitiendo de un vistazo conocer cómo está la empresa y determinar debilidades y deficiencias a la hora de ejecutar lo ya planificado.

Tarea de Ingeniería	
Número Tarea: 17	**Número Historia de Usuario:** 6
Nombre Tarea: Visualizar Dashboard.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 3
Fecha inicio: 16 de mayo 2011	**Fecha fin:** 27 de mayo 2011
Programador responsable: Luis A. Cortina Almeida	
Descripción El sistema debe presentar un formulario para que el usuario defina los datos del presupuesto que quiere mostrar.	

Tabla 3.8: Tarea de ingeniería # 17 Visualizar Dashboard.

3.2.2.3 Historia de Usuario Gestionar Usuarios.

La seguridad en el sistema a implementar es primordial debido a que se requiere un alto nivel de control de acceso a la información, por lo que se crearon perfiles para hacer más fácil este trabajo. Básicamente, se cuenta con cuatro capas de seguridad.

1. Al iniciar la aplicación, se solicitarán las credenciales del usuario y se autenticará contra el dominio de la empresa, si el usuario no pertenece al dominio entonces se le denegara el acceso a la aplicación.

[1] Nota del autor

2. Si el usuario pertenece al dominio, se solicitará en la base de datos si pertenece o no a la lista de usuarios autorizados a utilizar la aplicación, si no está autorizado entonces se le denegara el acceso a la aplicación.
3. Si está autorizado, se obtendrán en la base de datos los permisos que tiene dicho usuario en la aplicación y antes de cualquier operación en el sistema se chequeará que tenga acceso para realizar dicha operación.
4. Se guardarán en cada tabla de la base de datos los usuarios que insertaron y modificaron filas y la fecha, además de tener un log de entrada y salida de la aplicación de cada usuario.

No se realizará encriptación de paquetes debido a la confiabilidad del medio en que la aplicación va a funcionar.

Tarea de Ingeniería	
Número Tarea: 18	**Número Historia de Usuario:** 7
Nombre Tarea: Adicionar usuario.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 1.5
Fecha inicio: 30 de mayo 2011	**Fecha fin:** 8 de junio 2011
Programador responsable: Luis A. Cortina Almeida	
Descripción El sistema debe presentar un formulario para que el usuario defina los datos del nuevo usuario a insertar, incluyendo el perfil a que pertenece y las opciones de acceso del mismo.	

Tabla 3.9: Tarea de ingeniería # 18 Adicionar Usuario.

3.2.3 Tercera iteración.

En esta iteración se da cumplimiento a otra historia de usuario que brinda una funcionalidad complementaria al sistema, permite reconfigurar la base de datos para no tener que redistribuir la aplicación en caso de que se mueva la base de datos de servidor.

Historias de Usuario	Tiempo de Implementación (semanas)	
	Estimación	Real
Gestión de la Conexión a la Base de Datos	1	1

Tabla 3.10: Historias de Usuarios de la 3ra Iteración

3.2.3.1 Gestionar la Conexión a la Base de Datos.

En esta historia de usuario se implementan funcionalidades que permiten la localización correcta de la base de datos del sistema y de la base de datos del ERP con el que estamos trabajando. Debido a que la base de datos de la empresa está dividida en tres bases de datos diferentes, se hizo definir un mecanismo para cambiar rápidamente de base de datos y tener acceso a las informaciones que cada una de ellas brinda. Por esta razón, debemos dar la oportunidad al usuario de seleccionar con que base de datos quiere trabajar. Previendo cambios de servidores o algún tipo de situación parecida, también es necesario darle la posibilidad al usuario de cambiar la configuración de la conexión a la base de datos, evitando tener que redistribuir la aplicación ante un suceso de este equipo.

Tarea de Ingeniería	
Número Tarea: 20	**Número Historia de Usuario:** 8
Nombre Tarea: Seleccionar Base de Datos.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 2
Fecha inicio: 11 de junio 2011	**Fecha fin:** 25 de junio 2011
Programador responsable: Luis A. Cortina Almeida	
Descripción El sistema debe presentar una lista de las diferentes bases de datos con las que la empresa trabaja para que el usuario seleccione con la que va a trabajar.	

Tabla 3.11: Tarea de ingeniería # 20 Seleccionar Base de Datos.

3.3 Conclusiones

Con el desarrollo de este capítulo se ha dado una breve descripción de todas las iteraciones presentes para este proyecto, las tareas de ingeniería a implementar en cada una de ellas para realizar las entregas y cumplir con los requerimientos de funcionalidad del sistema.

CAPÍTULO 4: PRUEBAS

4.1 Introducción

El presente capítulo servirá para dar una visión de los resultados de las pruebas de aceptación realizadas al sistema, recuérdese que estas son anteriormente confeccionadas por el cliente y su resultado determina en gran medida la capacidad de cumplir con los requisitos solicitados y la calidad del sistema implementado. En nuestro proyecto en particular, las pruebas se realizaban después de cada entrega por solicitud del cliente.

4.2 Pruebas.

En la metodología XP es esencial el desarrollo de las pruebas, esto permite constantemente probar la calidad del código. Cada vez que se quiere implementar las funcionalidades que tendrá el software, XP propone una redacción sencilla de prueba, para ser pasada por el código posteriormente. El desarrollo constante de las pruebas permite que se desarrolle un producto con mayor calidad dando a los programadores una mayor seguridad en el trabajo que desarrollan.

4.2.1 Pruebas de aceptación.

El objetivo de las pruebas de aceptación es validar que un sistema cumple con el funcionamiento esperado y permitir al usuario de dicho sistema que determine su aceptación, desde el punto de vista de su funcionalidad y rendimiento. Las pruebas de aceptación son definidas por el usuario del sistema y preparadas por el equipo de desarrollo, aunque la ejecución y aprobación final corresponden al usuario. (26)

Lo que caracteriza un escrito formal de caso de prueba es que hay una entrada conocida y una salida esperada, los cuales son formulados antes de que se ejecute la prueba. La entrada conocida debe probar una precondición y la salida esperada debe probar una postcondición (27).

Las pruebas de aceptación:

- Son el termómetro de los desarrolladores, fundamentalmente de los programadores a la hora de medir la calidad de su trabajo.
- Garantizan la entrega de un producto con calidad, que responde a las necesidades del cliente.

Durante las iteraciones las HU seleccionadas serán traducidas a pruebas de aceptación. En ellas se especifican, desde la perspectiva del cliente, los escenarios para probar que una HU ha sido implementada correctamente. Esta puede tener todas las pruebas de aceptación que necesite para asegurar su correcto funcionamiento. El objetivo final es garantizar que las funcionalidades requeridas por el cliente hayan sido cumplidas. Una HU no se considera completa hasta que no ha pasado por sus pruebas de aceptación. En este documento, solo se presentara una parte de estas pruebas de aceptación, el resto se encuentra en el documento correspondiente en el expediente de proyecto.

4.2.1.1 Pruebas de aceptación Primera Iteración.

Caso de Prueba de Aceptación	
Código Caso de Prueba: HU1_P1	Nombre Historia de Usuario: Gestionar Sucursales.
Nombre de la persona que realiza la prueba: Departamento Económico UNEVOL S.A.	
Descripción de la Prueba: Se gestionan los datos de las sucursales (adicionar, modificar, eliminar) además se visualiza la información gestionada.	
Condiciones de Ejecución: Ejecutar la aplicación desde el escritorio de Windows, comprobar los requisitos del sistema y comprobar la conexión a la base de datos y su velocidad.	
Entrada / Pasos de ejecución: Abrir la aplicación y cargar las pantallas de gestión de	

sucursales, seleccionar las opciones de adicionar, modificar y eliminar respectivamente y ver los resultados de cada una de las operaciones.
Resultado Esperado: se insertó, modificó y eliminó correctamente cada sucursal.
Evaluación de la Prueba: Satisfactoria

Tabla 4.1: Caso de Prueba de Aceptación Historia de Usuario Gestionar Sucursales.

4.2.1.2 Pruebas de aceptación Segunda Iteración.

Caso de Prueba de Aceptación	
Código Caso de Prueba: HU5_P1	Nombre Historia de Usuario: Visualizar Reportes
Nombre de la persona que realiza la prueba: Departamento Económico UNEVOL S.A.	
Descripción de la Prueba: Se visualizan todos los reportes que el usuario necesita, se comprueba la veracidad de los datos y la estructura. Se observa la rapidez de ejecución de los mismos.	
Condiciones de Ejecución: Ejecutar la aplicación desde el escritorio de Windows, comprobar los requisitos del sistema y comprobar la conexión a la base de datos y su velocidad, además la correcta información que brinden los reportes.	
Entrada / Pasos de ejecución: Abrir la aplicación y cargar las pantallas de visualizar reportes, escribir los parámetros especificados para cada uno y revisar la veracidad de la información con pruebas aleatorias sobre el cuerpo del reporte y sobre los resultados finales.	
Resultado Esperado: Se muestran correctamente todos los reportes de la forma esperada por el usuario, con una velocidad aceptable para el mismo y sin errores en los datos o totales.	
Evaluación de la Prueba: Satisfactoria	

Tabla 4.2: Caso de Prueba de Aceptación Historia de Usuario Visualizar Reportes.

4.2.1.3 Pruebas de aceptación Tercera Iteración.

Caso de Prueba de Aceptación	
Código Caso de Prueba: HU8_P1	Nombre Historia de Usuario: Gestionar Conexión a la Base de Datos
Nombre de la persona que realiza la prueba: Departamento Económico UNEVOL S.A.	
Descripción de la Prueba: Se modifica la cadena de conexión a la base de Datos.	
Condiciones de Ejecución: Ejecutar la aplicación desde el escritorio de Windows, comprobar los requisitos del sistema y comprobar la conexión a la base de datos y su velocidad.	
Entrada / Pasos de ejecución: Abrir la aplicación y cargar las pantallas de gestión de Configuración de la Base de Datos, insertar los datos sobre el servidor y la base de datos nueva y comprobar la conexión.	
Resultado Esperado: Se cambió correctamente la cadena de Conexión de la Base de Datos.	
Evaluación de la Prueba: Satisfactoria	

Tabla 4.3: Caso de Prueba de Aceptación Historia de Usuario Gestionar Conexión a la Base de Datos.

4.3 Conclusiones del capítulo

Con la culminación de este capítulo el cliente se asegura de que el producto desarrollado cumple con las funcionalidades para lo cual fue concebido, realizando una prueba para cada una de las HU implementadas y dejando a su vez la valoración de cada una de ellas. Las pruebas realizadas arrojaron resultados satisfactorios tanto para el cliente como para el equipo de desarrollo.

CAPÍTULO 5: ESTUDIO DE FACTIBILIDAD

5.1 Introducción

Con el desarrollo de este capítulo se pretende ofrecer un análisis sobre el estudio de factibilidad del proyecto, para lo cual se utilizó la técnica de Análisis de Costo – Beneficio. El análisis de Costo - Beneficio (CBA) es el acercamiento de la escala de peso para la toma de decisiones. Todos los elementos positivos (los movimientos de efectivos y otras ventajas intangibles) se ponen en un lado del equilibrio y todos los elementos negativos (los costos y las desventajas) se ponen en el otro. Cualquiera sea el peso, el más pesado gana.

5.2 Metodología Costo Beneficio para Proyectos de Software

El objetivo de aplicar esta metodología es proporcionar una medida de los costos en que se incurre en la realización de un proyecto y comparar dichos costes previstos con los beneficios esperados de la realización de dicho proyecto. Esta medida o estimación servirá:

- Para valorar la necesidad y oportunidad de acometer la realización del proyecto.
- Para seleccionar la alternativa más beneficiosa para la realización del proyecto.
- Para estimar adecuadamente los recursos económicos necesarios en el plazo de realización del proyecto. (28)

Para los proyectos de software en el lado del costo estaría el precio del software, el costo de consultores para instalar y para poner el software en ejecución y el costo de entrenamiento para los usuarios del software. Sin embargo en el lado de las ventajas, estaría los procesos mejorados del negocio (que conducen a una disminución del costo anual), debido a contar con mejor información disponible, la compañía podrá tomar mejores decisiones (que conducen a

adicionales movimientos de efectivos) y el incremento creciente de la moral del personal, debido al uso de nuevas herramientas modernas para apoyar el negocio. (29)

Lo relevante es que podemos estar en condiciones de tomar decisiones oportunas con un nivel de certeza muy razonable, sin invertirle tiempo valioso en analizar muchas ideas cuando pocas son acogidas e implementadas. Buscar la practicidad (efectividad) en un tiempo de grandes cambios, nos puede llevar a ser más eficientes, que en otro sentido tratando de conseguir la solución más óptima y perfecta pero poco realizable o a destiempo.

Los elementos a tener en cuenta para la realización de este proyecto son:

Costos

A. Precio del Software.- Generalmente se contempla el Licenciamiento y Soporte
B. Infraestructura.- Toda los componentes de Hardware y Software requeridos.
C. Implantación.- Consultoría para instalación y puesta en funcionamiento
D. Entrenamiento.- Dirigido a los Usuarios de la Aplicación

Costo Total de la Solución (CTS) = A + B + C + D

Beneficios

A. Mejora de Procesos.- Conducen a reducción de tiempo y recursos
B. Disponer de Sistemas de Información.- Mejora la toma de decisiones y obtención de ingresos.
C. Personal Motivado.- Creciente moral del personal al funcionar en un entorno de herramientas modernas para el negocio.
D. Intangibles.- Otros beneficios intangibles que sean identificados y cuantificables.

Beneficio Total de la Solución (BTS) = A + B + C + D

Resultado

• Si CTS < BTS entonces la Solución es Viable, caso contrario no es recomendable.

Normalmente las organizaciones asignan importantes recursos en este complejo proceso de decisión, que en muchos casos terminan sin concretarse o se desvanecen en el tiempo. Peor aún es que no se costean o registran todos los gastos en tiempo y personal dedicado a estas labores. (30)

5.3 Estimación de Costos y Beneficios del Proyecto

5.3.1 Costos

Costos en Moneda Libremente Convertible:

- Costos Directos.

  1. Compra de equipos de cómputo: $2000.00
  2. Alquiler de equipos de cómputo: No procede.
  3. Compra de licencia de Software: No procede.
  4. Depreciación de equipos: $ 280.50
  5. Materiales directos: No procede.

  Total: $ 2280.50 CUC

- Costos Indirectos.

  1. Formación del personal que elabora el proyecto: No procede.
  2. Gastos en llamadas telefónicas: $25.00
  3. Gastos para el mantenimiento del centro: No procede.
  4. Know How: No procede.
  5. Gastos en representación: $100.00.

  Total: $125.00

- Gastos de distribución y venta.

  1. Participación en ferias o exposiciones: No procede.
  2. Gastos en transportación: No procede.

3. Compra de materiales de propagandas: No procede.

Total: $0.00.

Total CUC: $2405.50

Costos en Moneda Nacional: No Procede

5.3.2 Beneficios

Los efectos económicos pueden clasificarse como:

- Efectos directos.
- Efectos indirectos
- Efectos externos
- Intangibles

5.3.2.1 Efectos Directos

- ✓ Positivos
    - Se gestiona la información necesaria a la que los usuarios finales del sistema podrán acceder.
    - Mayor integración usuario - artefactos, ya que por medio de este el usuario siente necesidades de interactuar con el sistema, debido a que este facilita el trabajo y le brinda la información necesaria.
    - Se cuenta con una herramienta capaz de mantener la seguridad e integridad de la información.
    - Permite al usuario estar informado de todo lo concerniente a los presupuestos, y brinda información para los diferentes niveles dentro de la empresa.
    - Facilita a usuarios con diferentes roles en el sistema gestionar y obtener información sobre los presupuestos mediante la aplicación.
- ✓ Negativos

- Para usar la aplicación es necesario la utilización de un ordenador conectado a la red, paralelo a los gastos de consumo de electricidad y mantenimiento que conlleva.

5.3.2.2 Efectos indirectos

- Los efectos económicos observados que pudiera repercutir sobre otros mercados no son perceptibles.

5.3.2.3 Efectos Externos

- Se contará con una herramienta que permitirá a los usuarios finales gestionar la información de los presupuestos de una manera rápida, fácil y segura, además la obtención de reportes sobre las diferentes áreas de la empresa y sobre el manejo de los presupuestos en cada una de ellas.

5.3.2.4 Intangibles

En la valoración económica siempre hay elementos perceptibles por una comunidad como perjuicio o beneficio, pero al momento de ponderar en unidades monetarias esto resulta difícil o prácticamente imposible. A fin de medir con precisión los efectos, deberán considerarse dos situaciones:

Costos:

- ✓ Resistencia al cambio.

Beneficios:

- ✓ Mayor comodidad, organización e información para los usuarios.
- ✓ Mayor integración usuarios-artefactos.
- ✓ Mejora en la calidad y visibilidad de la información.

5.4 Conclusiones sobre el estudio de factibilidad

Tomando en cuenta que el control del presupuesto es una tarea de importancia diaria para la empresa, que al implementar esta herramienta se logrará agilizar el proceso de gestión de los mismos, mejorar la calidad y la rapidez con que se manejan las informaciones, además de brindar una herramienta para la consulta de directivos de diferentes niveles y mejorar la calidad de trabajo de los involucrados en el proceso, el cliente estima que los beneficios que reportará la implementación de este sistema tienen más peso que los costos planificados para llevar a cabo este proyecto.

La empresa recibió ofertas de varios proveedores los cuales estimaban el precio de la construcción de este software cercano a los $5000.00 CUC. La dirección de la empresa decidió utilizar este problema y su solución como el trabajo de tesis del autor debido a que:

1. Los costos estimados por las ofertas recibidas son superiores a los costos estimados para este proyecto.
2. Los equipos de cómputo que se adquieran para este proyecto, serán utilizados como activos fijos de la empresa después de terminada la tarea.
3. El entrenamiento de los usuarios se realiza sin costo.

5.5 Conclusiones del Capítulo

Este capítulo realizó el estudio de factibilidad mediante la metodología Costo-Beneficio, se expusieron los criterios manejados con el cliente para la evaluación de la factibilidad del proyecto teniendo en cuenta su importancia, beneficios y los costos en que se incurrirán de realizarlos. La respuesta del cliente es que es factible por el peso de los beneficios de la implementación de un sistema con estas características sobre los costos en que se incurre de realizar este proyecto.

CONCLUSIONES GENERALES

Por medio de la culminación del proyecto se logró:

- Desarrollar una aplicación de escritorio capaz de gestionar la información necesaria sobre el control de los presupuestos y su ejecución que satisface las necesidades de la Empresa UNEVOL S.A.
- El desarrollo del proyecto permitió realizar una valoración sobre los conceptos principales asociados al objeto de estudio y campo de acción del mismo.
- Además estuvo presente el cliente vinculado al área de producción, durante el período de planeación y desarrollo del software.
- Se realizó el análisis y diseño de la aplicación, partiendo de las características del territorio en el que se encuentra enmarcada la empresa, su distribución a través del país y las necesidades del personal involucrado en el proceso.
- Se identificaron las principales funcionalidades que debería brindar la aplicación, para ello se definieron los requerimientos funcionales y no funcionales del sistema, los cuales se tomaron en cuenta en la construcción del mismo, así como las fases de planificación y definición del sistema como parte de la metodología de desarrollo utilizada.
- Las pruebas de aceptación dieron una visión de la valoración del cliente sobre el trabajo desarrollado, este resultó ser satisfactorio, ya que se cumplió con la totalidad de las Historias de Usuarios propuestas.
- Con el análisis del estudio de factibilidad, se demostró que los beneficios que traería la realización del proyecto para el organismo eran superiores a los costos, siendo estos de $ 2405.50 CUC.

RECOMENDACIONES

El proceso de implementación de productos requiere de una solución integral en las organizaciones buscando una eficiencia de base en las tecnologías de la información, por lo tanto es una actividad que se divide por fases, etapas y se realiza de manera cíclica en la medida que la los productos logran mayores alcances.

La utilización de una metodología para ejecutar los procesos de implementación en la empresa permitirá lograr una buena comunicación durante las distintas etapas del proyecto en todos los niveles de la organización, así como la disminución del tiempo de implementación de los Sistemas y conseguir la estandarización de un método de trabajo.

La utilización de las herramientas de negocio actuales para sembrar ideas que mejoren el trabajo ya realizado, a partir de sugerencias que ayuden a mejorar o a ampliar el espectro de funcionamiento de esta herramienta.

Bibliografía

1. UNEVOL S.A. [En línea] 2009. [Citado el: 25 de Enero de 2011.] http://www.unevol.co.cu/index.php?option=com_content&view=article&id=1&Itemid=5.

2. Wikipedia. [En línea] 2010. [Citado el: 8 de Febrero de 2011.] <http://es.wikipedia.org/wiki/Presupuesto>.

3. Monografias.com. Glosario de términos contables. [En línea] 2009. [Citado el: 8 de Febrero de 2011.] http://www.monografias.com/trabajos64/glosario-terminos-contables/glosario-terminos-contables.shtml.

4. Cuba.cu CONSTITUCIÓN DE LA REPUBLICA DE CUBA Art. 17. [En línea] 2011. [Citado el: 8 de Febrero de 2011.] http://www.cuba.cu/gobierno/cuba.htm.

5. EcuRed. Metodologías Tradicionales. [En línea] 2010. [Citado el: 25 de Mayo de 2011.] http://www.ecured.cu/index.php/Metodologías_Tradicionales.

6. Wikipedia. Proceso Racional Unificado. [En línea] 2010. [Citado el: 25 de Mayo de 2011.] http://es.wikipedia.org/wiki/ Proceso_Unificado_de_Rational.

7. Wikipedia. UML. [En línea] 2010. [Citado el: 25 de Mayo de 2011.] http://es.wikipedia.org/wiki/ UML.

8. Wikipedia. Desarrollo ágil de software . [En línea] 2010. [Citado el: 25 de Mayo de 2011.] http://es.wikipedia.org/wiki/ Desarrollo_ágil_de_software.

9. Wikipedia. Método de desarrollo de sistemas dinámicos. [En línea] 2010. [Citado el: 25 de Mayo de 2011.] http://es.wikipedia.org/wiki /Método_de_desarrollo_de_sistemas_dinámicos.

10. developerWIKI. ASD (Adaptive Software Development). [En línea] 2010. [Citado el: 25 de Mayo de 2011.] https://sites.google.com/a/egafutura.com/developerwiki/glosario/metodologia-agil-asd-adaptive-software-development.

11. Wikipedia. Agile Unified Process. [En línea] 2010. [Citado el: 25 de Mayo de 2011.] http://es.wikipedia.org/wiki/Agile_Unified_Process.

12. Wikipedia. Programación extrema. [En línea] 2010. [Citado el: 25 de Mayo de 2011.] http://es.wikipedia.org/wiki/Programación_extrema.

13. Programacion Extrema. Fases de la Programación Extrema. [En línea] 2010. [Citado el: 25 de Mayo de 2011.] http://programacionextrema.tripod.com/fases.htm.

14. Wikipedia SCRUM. [En línea] 2010. [Citado el: 25 de Mayo de 2011.] http://es.wikipedia.org/wiki/Scrum.

15. USBVirtual. SXP, metodología ágil para el desarrollo de software. [En línea] 2010. [Citado el: 25 de Mayo de 2011.] http://usbvirtual.usbcali.edu.co/ijpm/index.php?option=com_content&view=article&id=31:sxp-metodologia-agil-para-el-desarrollo-de-software&catid=2:volumen2&Itemid=2.

16. **Peñalver, G. Meneses, A. García, S.** *SXP, METODOLOGÍA ÁGIL PARA EL DESARROLLO DE SOFTWARE.* Antofagasta, Chile : s.n., 2010.

17. Wikipedia. Microsoft SQL Server. [En línea] 2008. [Citado el: 21 de Febrero de 2011.] http://es.wikipedia.org/wiki/Microsoft_SQL_Server.

18. Wikipedia. C Sharp. [En línea] 2010. [Citado el: 21 de Febrero de 2011.] http://es.wikipedia.org/wiki/C_Sharp.

19. **Kovacs, James.** C#/.NET History Lesson (en inglés). [En línea] 7 de Septiembre de 2007. [Citado el: 21 de Febrero de 2011.] http://jameskovacs.com/2007/09/07/cnet-history-lesson/.

20. Wikipedia. Programación por Capas. [En línea] 2010. [Citado el: 25 de Junio de 2011.] http://es.wikipedia.org/wiki/ Programación_por_capas.

21. Wikipedia. Historias de usuario. [En línea] 2010. [Citado el: 8 de Junio de 2011.] http://es.wikipedia.org/wiki/Historias_de_usuario.

22. **Concepción, P.** Monografias.com. Análisis y diseño de sistemas. [En línea] 2010. [Citado el: 21 de Junio de 2011.] http://www.monografias.com/trabajos/anaydisesis/anaydisesis.shtml.

23. Wikipedia. Tarjetas CRC. [En línea] 2010. [Citado el: 21 de Junio de 2011.] http://es.wikipedia.org/wiki/Tarjetas_CRC.

24. Real Academia Española. Sucursal . [En línea] 2011. [Citado el: 9 de Mayo de 2011.] http://buscon.rae.es/draeI/.

25. GestioPolis.com El diseño del Dashboard. [En línea] 2010. [Citado el: 21 de Junio de 2011.] http://www.gestiopolis.com/administracion-estrategia/dashboard-kpi-metricas.htm.

26. **Rojas, J., Barrios, E.** Portal Universidad Francisco José de Caldas. [En línea] 2007. [Citado el: 21 de Junio de 2011.] http://gemini.udistrital.edu.co/comunidad/grupos/arquisoft/fileadmin/Estudiantes/Pruebas/HTML%20-%20Pruebas%20de%20software/node55.html.

27. Wikipedia. Caso de prueba. [En línea] 2010. [Citado el: 21 de Junio de 2011.] http://es.wikipedia.org/wiki/Caso_de_prueba.

28. **Cervantes, Enid Martín.** *La metodología métrica. Una herramienta para el desarrollo de intranets corporativas.* La Habana : s.n., 2000.

29. 12Manage. Análisis Costo Beneficio. [En línea] 2009. [Citado el: 21 de Junio de 2011.] http://www.12manage.com/methods_cost-benefit_analysis_es.html.

30. **Sánchez, L.** CENTRO DE ESTUDIOS FISCALES. [En línea] 24 de Noviembre de 2008. [Citado el: 21 de Junio de 2011.] http://cef.sri.gov.ec/virtualcef/file.php/1/Publicaciones/ensayo.pdf.