Universidad de las Ciencias Informáticas

Facultad 4

Título: Análisis del Método de Estimación empleado para el desarrollo del proyecto SIGEP

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas

Autores: Dayami Rodríguez Brito

Haydee Fernández Díaz

Tutor: Ing. Arturo César Arias Orizondo

Ciudad de La Habana, Julio del 2007

Caminante, son tus huellas
el camino, y nada más;
caminante, no hay camino,
se hace camino al andar.
Al andar se hace camino,
y al volver la vista atrás
se ve la senda que nunca
se ha de volver a pisar…

Antonio Machado

Declaración de Autoría

Declaramos que somos los únicos autores de este trabajo y autorizamos a la Facultad 4 de la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

Para que así conste firmamos la presente a los ___ días del mes de Julio de 2007.

Dayami Rodríguez Brito _____

Haydee Fernández Díaz _____

Ing. Arturo Cesar Arias Orizondo

_____

Datos de Contacto

Ing. Arturo Cesar Arias Orizondo

Profesor Instructor

Graduado en el 2003 de Ingeniero Informático del Instituto Superior Politécnico José Antonio Echeverría (ISPJAE), Título de Oro, promedio 5 puntos.

Ha impartido las asignaturas de Sistemas de Bases de Datos, Ingeniería de Software I y II, Gestión de Software en la Universidad de Ciencias Informáticas, participando en la elaboración de los programas de estas asignaturas.

Miembro del tribunal de acreditación de competencias del Departamento de Ingeniería de Software.

Vicedecano de producción e investigación de la UCI por dos cursos: 2004-2005, 2005-2006.

Miembro del consejo científico de la UCI.

Miembro de la comisión de carrera durante los cursos 2004-2005, 2005-2006.

Ha impartido cursos de postgrado sobre RUP y UML.

Participó en el Congreso y Feria TecnoInternet 2004 celebrado en Santiago de Chile, Chile.

Ha participado en evaluaciones de calidad de productos de software para Venezuela.

Diseñador de base de datos del proyecto SAFRE, primero de exportación de la UCI.

Líder del proyecto de desarrollo de software para el sistema penitenciario venezolano desde el 2005.

Correo electrónico: arturo@uci.cu

Agradecimientos Compartidos

A **la Revolución** por brindarnos la oportunidad de ser mejores y por confiar su futuro en nosotros.

A **la UCI** por permitir conectarnos al futuro y a la Revolución.

A **nuestro tutor Arturo** por su dirección, por mejorar nuestras ideas y los valiosos consejos que nos permitieron desarrollar este trabajo.

A **Elvio, Madelin, Lizandra** y **Erick** por su colaboración, apoyo y palabras de aliento.

A **los profesores** por compartir sus conocimientos, consejos y experiencia.

A la **FEU y la UJC** por ser una escuela inolvidable.

Y a **todos** los que aportaron su granito de arena.

Muchas Gracias…

Haydee

A **mis padres** por estar conmigo incondicionalmente, sin sus enseñanzas no estaría aquí ni sería quien soy ahora.

A **mis hermanos** por sus locas palabras de aliento.

A **mis abuelos** por ser tan especiales y hacerme sentir parte importante de su mundo.

A **mis tíos y primos** por su constante desvelo.

A **Dayami**, mi compañera de tesis por las noches de desvelo, por su confianza y optimismo, juntas alcanzamos este triunfo.

A mis **amigas Mayté, Elizabeth, Dinia, Dalkis, Yaislen, Nadiesda, Yary y Maglema,** gracias por estar conmigo estos 5 años, por aconsejarme, regañarme, compartir risas y llantos en todo los momentos.

Dayami

A **mami y papi**, por TODO el apoyo durante estos 23 años. Al fin les regalo lo que tanto esperaron. Los quiero mucho.

A **mi hermano querido**, por confiar siempre en mí.

A **mi esposo**, por su amor, apoyo y paciencia.

A **Haydee**, mi compañera de tesis, porque mejor no la hubiese querido para compartir este triunfo.

A **Diana Mabel,** por la amistad de tantos años y porque sé que puedo contar con ella.

A **Chirino**, por sus buenos consejos y estar siempre dispuesto a ayudarme.

A **Miguel Ángel**, por la amistad, ayuda y preocupación, aún cuando estaba lejos.

A **Heidy, Daimara, Mabel, Ana y Yenik** por la amistad en las buenas y en las malas.

Dedicatoria

A mis padres por creer en mí.

A mis hermanos por la confianza.

A mis sobrinos por ser mis más fieles inspiradores.

A mis abuelos por regalarme todo el amor del mundo.

Haydee

A mi hermano Yanko

A mi esposo Elio Júnior

Y en especial a mis padres con todo mi amor.

Dayami

Resumen

La planificación, en el mundo de la Industria del Software, se ha convertido en uno de los principales retos para la gestión de proyectos y una actividad fundamental para desarrollar software de alta calidad.

Se considera imprescindible para planificar, la aplicación de buenos métodos de estimación, por la creciente influencia que ejercen en el control preciso, predecible y repetido sobre los procesos de producción y los productos de software.

La Universidad de las Ciencias Informáticas (UCI) no cuenta con suficientes datos históricos de proyectos ya finalizados, que permitan realizar análisis estadístico, para acertar mejor en las estimaciones, por lo que para planificar, se elaboró un método de estimación totalmente empírico basado en la opinión de algunos especialistas con cierta experiencia.

El presente trabajo se centra en realizar un análisis del método de estimación, teniendo en cuenta los elementos que debe contemplar y los resultados obtenidos de la aplicación del método analizado, para identificar sus deficiencias y proponer mejoras para su futura aplicación.

Se realiza un estudio de los métodos de estimación y las métricas que se utilizan el en proceso de desarrollo de software, así como un análisis teórico y práctico del método de la UCI, a partir de la experiencia en el proyecto Sistema de Gestión Penitenciaria (SIGEP), ampliando el estudio, con la comparación de datos estimados y reales de otros proyectos como: Cuerpo de Investigaciones Científicas, Penales y Criminalísticas (CICPC) y Centro de Tratamiento y Análisis de Información de Seguridad Ciudadana (CTAISC), que también lo aplicaron.

Palabras Claves

Planificación, estimación, medición, método de estimación, métrica, riesgo.

Índice

Índice de Tablas y Figuras

Índice de Tablas

Índice de Figuras

Introducción

La Universidad de las Ciencias Informáticas (UCI), ha sido creada con un nuevo concepto de universidad productiva. Una de sus misiones es la producción de software y servicios informáticos a partir de la vinculación estudio-trabajo como modelo de formación, logrando una fuerte relación Universidad-Empresa. Esta universidad que convierte a la producción en sustento económico, político y social; es productora de grandes soluciones informáticas y está comprometida a ser la vanguardia del desarrollo de las empresas de software en Cuba, llevando la informatización a todos los sectores de la sociedad, de manera tal que convierta a la industria de software en un renglón fundamental de la economía del país.

La UCI tiene retos importantes, dados sus compromisos productivos reales, vinculando a la producción a todos sus estudiantes y profesores, en proyectos de alto valor, ya sea para el mercado nacional, como el internacional, concibiendo la docencia desde la producción.

Como entidad desarrolladora de software que pretende insertarse en el mercado internacional, la UCI debe lograr que la planificación se convierta en un elemento esencial a tener en cuenta en el proceso productivo, para obtener productos de alta calidad. El objetivo fundamental de la planificación de proyectos de software, es proporcionar un marco de trabajo, que permita hacer estimaciones razonables de recursos, costos y una planificación temporal. Estas estimaciones se hacen dentro de un marco de tiempo limitado al comienzo de un proyecto de software, y deberían actualizarse regularmente a medida que progresa el proyecto. Además las estimaciones deberán definir los escenarios del mejor y peor caso, de modo que los resultados del proyecto sean los más verídicos posible. Estimar es tener una visión al futuro y aceptar cierto grado de incertidumbre. Al estimar no solo se analiza el procedimiento técnico utilizado, sino que se toman en cuenta los recursos, costos, tiempo y planificación.

Cuando se planifica un proyecto de software, se deben realizar estimaciones de esfuerzo humano, la duración cronológica del esfuerzo humano, del proyecto y del costo. La planificación se logra entonces, mediante un proceso de descubrimiento de la información que lleve a estimaciones razonables.

En la mayoría de las definiciones analizadas de planificación, se pueden encontrar algunos elementos comunes, como el establecimiento de objetivos o metas y la elección de los medios más convenientes para alcanzarlos, por tanto, si una organización aspira a permanecer sana, debe plantearse objetivos

realistas, implementando planes más flexibles y efectivos que se adapten a las condiciones cambiantes del entorno de desarrollo.

Existen dos propósitos que la planificación debe cumplir: el protector y el afirmativo. El protector consiste en minimizar el riesgo reduciendo la incertidumbre que rodea al mundo de los negocios y definiendo las consecuencias de una acción determinada. El propósito afirmativo de la planificación consiste en elevar el nivel de éxito del proyecto. Otros de sus propósitos, consiste en coordinar los esfuerzos y los recursos dentro de las organizaciones. Estos elementos convierten a la planificación en una herramienta de posicionamiento anticipado, pues permite saber con anterioridad las fortalezas y debilidades que puede tener el proyecto, así como adecuarse a los cambios que puedan surgir en el mercado o en la sociedad.

El proceso de planeación tiene cinco pasos principales:

- Definición de los objetivos organizacionales.
- Controlar el cumplimiento de los objetivos.
- Desarrollar premisas considerando situaciones futuras.
- Identificar y escoger cursos alternativos de acción.
- Puesta en marcha de planes y evaluar los resultados.

Estos pasos posibilitan que la planificación se lleve a cabo de una forma eficiente.

El mal manejo de la planificación puede ocasionar un impacto negativo en el proceso de desarrollo de software, lo que implica un aumento en el tiempo y los costos de producción, así como la mala calidad de un producto, lo cual sería perjudicial para el prestigio de la organización.

En estos momentos, la universidad utiliza para planificar, un método de estimación ideado por un grupo de especialistas de la Infraestructura Productiva (IP) con cierto grado de experiencia en la gestión y dirección de proyectos. Este método de estimación se elaboró sobre una base empírica e intuitiva, pues como la UCI lleva pocos años de creada, no cuenta con un registro histórico de datos de otros proyectos similares ya finalizados, de manera que las estimaciones se puedan hacer utilizando dichos datos. A partir del método se puede estimar el esfuerzo necesario para los proyectos productivos, siendo esta la base para calcular su costo.

Es por ello que la UCI necesita una herramienta precisa que permita estimar el esfuerzo necesario para desarrollar soluciones informáticas.

El método se está aplicando actualmente y sin modificación en los proyectos contratados en el marco de la 6ta Mixta Cuba-Venezuela, dentro de los que se incluye el proyecto Sistema de Gestión Penitenciaria (SIGEP), que da respuesta a las necesidades de gestión, información y apoyo a la toma de decisiones del Ministerio del Interior y Justicia y de la Dirección General de Custodia y Rehabilitación del Recluso (DGCRR) de Venezuela (ARIAS 2006). Un error en el método puede traer grandes consecuencias como pérdidas financieras, de recursos y tiempo, por lo que se hace necesario realizar un análisis de este método con el objetivo de conocer cuán preciso es y si ha dado resultados en la práctica. Surgen entonces las siguientes interrogantes: ¿Es correcto el método? ¿Se ajusta a las características de la UCI? ¿Tiene en cuenta los factores de riesgo que influyen en un desarrollo de software?

Toda esta situación lleva a definir que el **problema científico** de la investigación corresponde a la necesidad de analizar el grado de aceptación del método de estimación, por el alto impacto que puede traer basar la planificación de la producción en estimaciones poco fiables.

Partiendo del problema planteado, el **objeto de estudio** se enfocará hacia la estimación de proyectos de desarrollo de software.

El **campo de acción** lo constituyen los proyectos de desarrollo de software de la UCI, que se estimaron basados en el método que se elaboró en la universidad.

Para resolver los problemas que se plantearon con anterioridad, se propone como **objetivo general**: *Analizar el método de estimación elaborado en la UCI, teniendo en cuenta los elementos que contempla y los resultados obtenidos en la práctica*. Para ello se trazaron los siguientes **Objetivos Específicos** de la investigación:

- Realizar un estudio de las tendencias mundiales sobre la estimación de software.
- Realizar un análisis teórico y práctico del método, basado en la experiencia del proyecto SIGEP.
- Proponer elementos que deben ser considerados para perfeccionar el método.

Para lograr los objetivos trazados, se llevaron a cabo las siguientes **tareas de la investigación**:

- Estudiar los métodos de estimación y las métricas que se utilizan en el proceso de desarrollo de software.
- Estudiar y analizar los factores de riesgos que influyen en las estimaciones.
- Estudiar el método de estimación aplicado a los proyectos de exportación en la UCI.
- Describir los elementos que estuvieron ausentes en el método.
- Estudiar el caso del proyecto SIGEP.
- Obtener información sobre registros históricos de otros proyectos que aplicaron el método de estimación.
- Proponer los elementos que deben ser considerados para mejorar el método de estimación después del análisis teórico-práctico.

Capítulo 1: Fundamentación Teórica

Haciendo un estudio retrospectivo en la historia de la industria del software, se pudiera iniciar caracterizando los años 1960s como la era donde se aprendió a explotar la información tecnológica (KAN. 2000), comenzando a vincular los software con las operaciones diarias de las instituciones que ya para los años 1970s registraban demoras masivas en los cumplimientos de las planificaciones y el exceso de los costos, enfocando todas las actividades a partir de entonces en torno a la planificación.

Para los 1980s, la productividad incrementaba significativamente, disminuía el costo y la competencia en la industria crecía considerablemente. Venía madurándose la idea de predecir, analizar y evaluar los distintos atributos y características de los productos y procesos, que participan en el desarrollo y mantenimiento del software, mediante métricas e indicadores que expresaran la realidad lo más cuantificada posible y con mejoras continuas e incrementales que se refinarían a lo largo de los 1990s definida como la era de la calidad (KAN. 2000). A partir de entonces la competencia por obtener un producto óptimo y refinado comienza a exigir más precisión a la hora de estimar para planificar y medir cuánto esfuerzo, recursos, y tiempo supondrá construir un sistema o producto específico de software, durante la gestión de proyectos.

1.1 Planificación en la gestión de proyectos software.

La gestión de un proyecto es una tarea vital para obtener éxito (PRESSMAN 1998).

Comprender el ámbito del trabajo a realizar, los riegos en los que se puede incurrir, las tareas que han de llevarse a cabo, las etapas a recorrer, los recursos y el coste del proyecto, así como el plan a seguir, previéndose las posibles situaciones de encontrar escenarios con el mejor y peor de los casos, representan acciones esenciales para conseguir un alcance exitoso en un proyecto software.

Precisamente la gestión de proyectos software es la encargada de proporcionar este conocimiento, comenzando antes de iniciar el proceso de desarrollo, evolucionando a medida que se va de un nivel conceptual a la realidad y finaliza en el momento que se abandona el software.

La planificación va sucediendo como un proceso de descubrimiento de la información que de lugar a realizar estimaciones más razonables a medida que se avanza.

Los proyectos bien ejecutados pasan por tres etapas básicas para crear la planificación del software. Primero se estima el tamaño del producto, luego el esfuerzo necesario para construir un producto con este tamaño y por último la duración cronológica del proyecto.

Una planificación adecuada requiere:

- Todas las actividades bien definidas en el método.
- El esfuerzo y el tiempo se asigne inteligentemente a cada actividad.
- Las relaciones entre las actividades indicadas correctamente.
- Los hitos situados rigurosamente espaciados para que se pueda seguir el progreso.

Resulta importante describir actividades asociadas a la planificación de proyectos software.

1.1.1 Ámbito del Software.

El ámbito del software es la primera actividad llevada a cabo durante la planificación del proyecto de Software (PRESSMAN 1998).

En esta actividad el desarrollador del software y el cliente se encargan de definir el ámbito y los objetivos del proyecto. El ámbito se define como un pre-requisito para la estimación y la obtención de la información necesaria, identifica las funciones primordiales que debe llevar a cabo el software y, lo que es más importante, intenta limitar esas funciones de manera cuantitativa. Los objetivos identifican los fines globales del proyecto sin considerar cómo se llegará a esos fines.

Una vez entendidos los objetivos y el ámbito del proyecto, se han de considerar soluciones alternativas. Aunque se estudien con muy poco detalle, las alternativas han de permitir seleccionar el mejor enfoque, dadas las restricciones impuestas por las fechas tope de entrega, los límites presupuestarios, la disponibilidad de personal, las interfaces técnicas y multitud de otros factores.

1.1.2 Recursos.

La Segunda tarea de la planificación del desarrollo de Software es la estimación de los recursos requeridos para acometer el esfuerzo de desarrollo de Software (PRESSMAN 1998), esto simula a una pirámide donde las Herramientas (hardware y Software), son la base que proporciona la infraestructura de soporte al esfuerzo de desarrollo, en el segundo nivel de la pirámide se encuentran los Componentes

reutilizables y en la parte más alta de esta, se encuentra el recurso primario, las personas (el recurso humano).

Figura 1: Recursos del proyecto (PRESSMAN 1998)

Cada recurso queda especificado mediante cuatro características:

- Descripción del Recurso.
- Informes de disponibilidad.
- Fecha cronológica en la que se requiere el recurso.
- Tiempo durante el que será aplicado el recurso.

Recursos Humanos.

La cantidad de personas requeridas para el desarrollo de un proyecto de software sólo puede ser determinado después de hacer una estimación del esfuerzo de desarrollo (por ejemplo personas mes o personas años), seleccionando la posición dentro de la organización y la especialidad que desempeñará cada profesional.

Recursos o componentes de software reutilizables.

Cualquier estudio sobre recursos de software estaría incompleto sin estudiar la reutilización. Esto es la creación y la reutilización de bloques de construcción de software. Tales bloques se deben establecer en

catálogos para una consulta más factible, estandarizarse para una fácil aplicación y validarse además para una integración.

Recursos de entorno.

El entorno es donde se apoya el proyecto de software, llamado a menudo entorno de Ingeniería de Software, incorpora hardware y software.

El hardware proporciona una plataforma con las herramientas (software) requeridas para producir los productos que son el resultado de la buena práctica de la Ingeniería del Software. Un planificador de proyectos debe determinar recursos requeridos para el hardware y el software, y verificar que estos estén disponibles.

En esencia, el intento a la hora de planificar para predecir cuánto dinero, esfuerzo, recursos y tiempo; llevará realizar un producto software determinado, implica estimar constantemente, siendo base también en futuras planificaciones.

2.1 La estimación en la Gestión de Proyectos.

La estimación de la duración de las actividades que conforman el desarrollo de software es un tema que concierne a la gestión y control de proyectos para asegurar el éxito.

En efecto, Ana María Moreno Sánchez Capuchino dice: La primera tarea en la gestión de proyectos es la estimación. (CAPUCHINO 1996)

La estimación, como actividad antecesora y constante de lo que luego será la planificación, proporciona valores aproximados de costos, tiempos y esfuerzos que se necesitarán para el desarrollo del producto a construir. Esa aproximación, requiere experiencia, acceder a una buena información histórica y el coraje de confiar en predicciones (medidas) cuantitativas cuando todo lo que existe son datos cualitativos (PRESSMAN); sin embargo, contar con dicha información en etapas tempranas de desarrollo, permite tomar decisiones importantes antes llevarlo a cabo, es decir que de cierta forma nos estamos atreviendo a predecir el futuro y así lo expresa también Ana Sánchez Capuchino definiendo la estimación como: el proceso que proporciona un valor, a un conjunto de variables para la realización de un trabajo, dentro de un “rango aceptable de tolerancia”. (CAPUCHINO 1996)

Se han desarrollado varias técnicas de estimación para el desarrollo de software, que tienen en común los siguientes aspectos:

- Se ha de establecer de antemano el ámbito del proyecto.
- Como base para la realización de estimaciones, se usan las métricas del software.
- El proyecto se desglosa en partes más pequeñas que se estiman individualmente.

La estimación es siempre difícil de realizar por diversas razones, algunas de ellas son (HURTADO 2007):

- No existe un modelo de estimación universal.
- Son muchas las personas implicadas en el proyecto, desde la alta dirección de la empresa a los ejecutivos del proyecto, que precisan de las estimaciones.
- La utilidad de una estimación varía con la etapa de desarrollo en que se encuentra el proyecto.
- Las estimaciones precisas son difíciles de formular, sobre todo al inicio del proyecto.
- La estimación suele hacerse superficialmente, sin tener en cuenta el esfuerzo necesario para hacer el trabajo.
- La rapidez del cambio de las metodologías y las tecnologías no permiten la estabilización del proceso de estimación.
- Los estimadores pueden no tener experiencias sobre aquello que pretenden estimar.
- El estimador suele hacer la estimación en función del tiempo que a él le llevaría en realizar el trabajo, sin tener en cuenta la experiencia y formación de la persona que realmente lo realiza.
- El estimador tiende a reducir en alguna medida sus estimaciones para hacer más aceptable su oferta.

1.2.1 Tamaño del Software.

El tamaño del software representa el primer reto a cumplir por el planificador, siendo una producción cuantificable del proyecto de software.

La precisión de una estimación del proyecto de software se predice basándose en (PRESSMAN 1998):

1. El grado en el que el planificador ha estimado adecuadamente el tamaño del producto a construir.

2. La habilidad para traducir la estimación del tamaño en esfuerzo humano, tiempo y dinero (una función de la disponibilidad de métricas fiables de software de proyectos anteriores).
3. El grado en que el plan del proyecto refleja las habilidades del equipo de software.
4. La estabilidad de los requisitos del software y el entorno que soporta el esfuerzo de la ingeniería del software.

1.2.2 Factores que repercuten en la incertidumbre de las estimaciones.

La estimación conlleva un riesgo inherente y es este riesgo el que lleva a la incertidumbre (PRESSMAN 1998).

Podemos citar una serie de factores que repercuten en la incertidumbre como la *complejidad del proyecto*, que también está muy vinculada en la planificación, pues resulta que cuando se habla de este término hay que tener en cuenta los esfuerzos relacionados con la experiencia en prácticas anteriores, porque lógicamente un equipo que lleve tiempo trabajando en proyectos con la misma complejidad, va a llegar el momento que vean la respuesta mucho más fácil, en comparación con equipos que se lanzan por primera vez.

El tamaño del proyecto es otro factor importante que puede afectar la precisión y la eficiencia de las estimaciones (PRESSMAN 1998).

Desde un principio el tamaño del proyecto va aumentado y paralelamente la dependencia de los elementos que lo componen se ven afectados en este crecimiento, por lo que para obtener mejores estimaciones se divide en partes lógicas con vistas a ver la solución más enfocada y lo más sencilla posible, aunque todo dependa de la forma en la que el encargado de descomponer lo vea, porque aún esta puede tornarse compleja y grande.

El grado de incertidumbre estructural tiene también efecto en el riesgo de la estimación (PRESSMAN 1998). Cuando nos referimos a la estructura estamos analizando el grado con que han sido definidos los requisitos y funciones, la facilidad con que se pueden compartimentar las funciones, así como la estructura por orden de prioridad de la información a procesar.

Resulta conveniente mostrar cuales son las opciones que se presentan a la hora de realizar estimaciones en el marco del desarrollo de proyectos de software. (OVEJERO 2006). Basar la estimación en proyectos

similares: esta alternativa funciona bien cuando el proyecto tiene parecido con otros que ya se hayan desarrollado. La desventaja de su implementación es la de requerir información de mediciones efectuadas en proyectos pasados que en todo momento no están disponibles y que además no siempre representan un buen indicador.

- Utilizar técnicas de descomposición relativamente sencillas para generar estimaciones: se basa en la descomposición del problema redefiniéndolo en conjuntos más pequeños. Posee dos puntos de vista: descomposición del problema y descomposición del proceso. (PRESSMAN 1998)
- Desarrollar un modelo empírico: utiliza fórmulas derivadas empíricamente para predecir costos, esfuerzos y tiempos. (PRESSMAN 1998). Los datos empíricos que soportan la mayoría de los modelos de estimación se obtienen de una muestra limitada de proyectos; razón por la cual este tipo de modelo no es adecuado para todas las clases de software ni para todos los entornos de desarrollo.

1.2.3 Características de un buen elemento de estimación.

Los métodos de estimación deben basarse en un parámetro o elemento de estimación que tenga las siguientes características:

- Objetivo.
- Fácilmente identificable.
- Apto para ser valorado numéricamente.
- Válido.
- Apto para ser refinado a medida que se obtiene mayor información.

Se puede encontrar en la literatura existente, muchas definiciones de lo que es la estimación; como la siguiente: “es la predicción del personal, del esfuerzo, de los costos y del tiempo que se requerirán para realizar todas las actividades y construir todos los productos asociados con el proyecto”. (MARÍA 1998)

Por otra parte una definición no técnica de estimación dice que “es un conjunto aproximado de valores para algo que ha de ser hecho”.

Se comienza la estimación haciendo uso de pocas variables en un nivel alto de abstracción, permitiéndose obtener valores aproximados de costo, tiempo y esfuerzo como para estudiar la viabilidad del proyecto. Una vez comenzado el proyecto y obtenidos estos valores se pueden efectuar comparaciones, detectar desvíos en el plan y realizar los ajustes correspondientes en lo que respecta a una buena planificación.

El desarrollo del software requiere de la estimación para controlar y administrar los recursos que se necesitan utilizar antes y durante el proyecto. No se puede considerar la estimación como una ciencia exacta ya que existen numerosas variables humanas, técnicas, del entorno y políticas, que intervienen en su proceso y que pueden afectar los resultados finales. Sin embargo, cuando es llevada a cabo en forma sistemática, se pueden lograr resultados con un grado aceptable y convertirla en un instrumento útil para la toma de decisiones.

Cualquier cronograma de tareas pendientes implica estimaciones. Cualquier intención de calcular tiempos, recursos o costos al futuro, implica estimaciones. Sin estimaciones no podría haber gestión de proyectos porque el proyecto es, como su nombre lo indica, una proyección.

1.2.4 Pasos a seguir para estimar proyectos software.

1. Obtener información acerca del proyecto y el entorno, la cual es relevante para determinar las restricciones y objetivos a seguir.
2. Seleccionar las métricas a estimar para el proyecto y los métodos mediante los que realizaremos esa estimación. Las métricas a estimar deben reflejar las restricciones y objetivos del proyecto, a fin de evaluar si pueden ser satisfechas.
3. Planificar cómo medir y recolectar los datos necesarios para analizar la precisión de las estimaciones.
4. Se realizan las estimaciones para evaluar si están siendo cumplidas o no. Los resultados son retroalimentados tanto para el desarrollo del sistema como a la realización de futuras estimaciones, proporcionando a los desarrolladores la oportunidad de tomar decisiones correctivas basándose en datos nuevos.
5. Evaluación de las métricas usadas para hacer las estimaciones aprendiendo de la experiencia. Comprender las razones por la que las estimaciones son imprecisas y buscar las soluciones, acompañadas de sugerencias, mejorando así precisiones futuras.

6. Guardar la experiencia adquirida a fin de que sea usada para el futuro. Esto es importante en el proceso de estimación, ya que las estimaciones precisas solo pueden ser esperadas cuando hay experiencia previa. Las estimaciones hechas sin referencia o experiencia previa serán poco menos que conjeturas.

1.3 Métodos de Estimación.

En el marco de trabajo durante el proceso de desarrollo de software se clasifican los siguientes métodos de estimación:

- Empíricos.
- Analógicos.
- Teóricos.
- Heurísticos.
- Las estimaciones global y detallada.
- Juicio del experto.
- Método Puntos de Casos de Uso.
- Modelo COCOMO.
- Puntos de Función.
- COCOMO II y los Puntos de Función.

1.3.1 Empíricos.

Cualquier estimación se debe basar en un modelo empírico, o sea, algo subjetivo producto de la experiencia, que relacione un atributo de interés con otros atributos mensurables. Este modelo empírico es el punto de partida para cada método de estimación.

1.3.2 Analógicos.

Este método hace la estimación de un proyecto nuevo por analogía con las estimaciones de proyectos anteriores comparables y que estén terminados. En la analogía pueden variar los siguientes factores como el tamaño, complejidad, usuarios. Utiliza medidas de los atributos del modelo empírico a fin de caracterizar

el caso actual, para el que se realiza la estimación. Las medidas conocidas para el caso actual son usadas para buscar un conjunto de datos que identifiquen casos análogos. La predicción se hace intercalando desde uno o varios casos análogos al caso actual.

Las ventajas que proporciona este método es un menor costo en tiempo y recursos que el método del juicio del experto. Como desventajas cabría destacar que las estimaciones de proyectos anteriores no siempre se ajustan a nuevos proyectos, ya que muchos de los factores de estas estimaciones no siempre se mantienen. (ESCORIAL 2006)

1.3.3 Teóricos.

Los métodos de estimación teóricos proponen un modelo numérico basado en el modelo empírico. Los modelos teóricos deben ser validados empíricamente, por comparación con los datos actuales de las medidas.

1.3.4 Heurísticas.

Los métodos heurísticos suelen usarse como extensiones de otros métodos. Las heurísticas son reglas empíricas, desarrolladas mediante experiencia, que obtienen conocimiento acerca de relaciones entre atributos del modelo empírico. Las heurísticas se pueden utilizar para ajustar estimaciones realizadas con otros métodos.

1.3.5 Las estimaciones global y detallada.

La estimación global, conocida también como descendente, se hace teniendo en cuenta las funcionalidades del producto, pasándose posteriormente al detalle.

La estimación detallada o ascendente empieza por la estimación de los esfuerzos individuales, los cuales se suman para obtener el esfuerzo del proyecto. (ESCORIAL 2006).

1.3.6 Juicio del experto.

Las opiniones de los expertos se basa principalmente en juicios emitidos por uno o varios expertos avalados por su experiencia en entornos similares y apoyados, en algunos casos, en datos objetivos obtenidos de proyectos anteriores y almacenados (ESCORIAL 2006), aunque no se incluyen dentro del marco de trabajo para seleccionar métodos de estimación, ya que estos métodos no se pueden caracterizar fácilmente.

Existen algunos métodos más específicos como el Experto Puro y el Wideband Delphies.

La desventaja de este método es el alto costo en tiempo y recursos humanos necesarios para su implantación, así como la subordinación al nivel de experiencia y conocimientos en el entorno que puedan aportar los técnicos. Las ventajas que posee indican que las estimaciones parciales son neutralizadas y se presenta una estimación global. Por otro lado las estimaciones suministradas por este grupo de expertos difícilmente pueden ser obviadas gracias a la trascendencia que la organización otorga a este proceso, al proporcionar costosos recursos a esta tarea (ESCORIAL 2006).

Durante la práctica de los métodos mencionados, se han originado otros más formales y eficientes, reconocidos y utilizados por su nivel de precisión:

- Basados en el tamaño: Método de Putnam, Método COCOMO (Modelo de Construcción de Costo).
- Basados en las funciones: Puntos de Función, Puntos de casos de uso y Puntos de Objeto.
- Combinados: Puntos de Función y de Objetos con COCOMO.

1.3.5 Puntos de Casos de Uso

Este método estima el esfuerzo de desarrollo de un producto de software a partir de los Casos de Uso y algunos factores de complejidad técnica y ambiente que influyen en el desarrollo. Fue propuesto originalmente por Gustav Karner y posteriormente refinado por muchos otros autores.

Este método exige la existencia de un modelo de casos de uso, por lo que se deberá comenzar a aplicar, una vez que se tenga algún entendimiento del dominio del problema o cuando se estén realizando las labores de arquitectura y dimensionamiento del tamaño del sistema (HERNÁNDEZ 2002).

El método utiliza los actores y casos de uso identificados para calcular el esfuerzo que costará desarrollarlos. A los casos de uso se les asigna una complejidad basada en transacciones, que son pares de pasos acción-usuario->respuesta-sistema de los escenarios de los casos de uso. A los actores se les asigna una complejidad basada en el tipo de actor, es decir, si son interfaces con usuarios o si son interfaces con otros sistemas (API o Protocolo). También se utilizan factores de entorno y de complejidad técnica para afinar el resultado (WIKIPEDIA1 2006).

Una vez asignada la complejidad, se calculan los puntos de caso de uso no ajustados, el TCF (factor de complejidad técnica) y el EF (factor del entorno). Con ellos, se calculan los puntos de caso de uso o UCP, que finalmente se traducen a esfuerzo en horas-hombre con un sencillo cálculo (WIKIPEDIA1 2006).

Una de las principales desventajas de este método es que no existe una teoría de cómo escribir o estructurar correctamente los casos de uso, por lo que todas las medidas de tamaño y estimación serán afectadas por la rigurosidad de los analistas (HERNÁNDEZ 2002).

1.3.6 Modelo COCOMO.

COCOMO fue propuesto y desarrollado por Barry Boehm en 1981, es uno de los modelos de estimación de costo mejor documentado, estudiado y utilizado en la industria de software. El modelo permite, basándose en un grupo de ecuaciones no lineales obtenidas mediante técnicas de regresión a través de un histórico de proyectos ya realizados (MARCELO 2006); estimar el esfuerzo, costo y tiempo que se requiere en un proyecto de software a partir de una medida del tamaño del mismo, expresada en el número de líneas de código que se estimen generar para la creación del producto software. El modelo original ha evolucionado a un modelo más completo llamado COCOMO II.

1.3.7 Puntos de Función.

El método de puntos de función fue creado por Allan Albretch y se basa principalmente en la identificación de los componentes del sistema informático en términos de transacciones y grupos de datos lógicos que son relevantes para el usuario en su negocio. A cada uno de estos componentes les asigna un número de puntos por función basándose en el tipo de componente y su complejidad; y la sumatoria de esto, da los puntos de función sin ajustar. El ajuste es un paso final basándose en las características generales de todo el sistema informático que se está contando.

Los objetivos de calcular Puntos de Función son:

- Medir lo que el usuario pide y lo que el usuario recibe.
- Medir atributos independientemente de la tecnología utilizada en la implantación del sistema.
- Proporcionar una métrica de tamaño que de soporte al análisis de la calidad y la productividad.
- Proporcionar un medio para la estimación del software.
- Proporcionar un factor de normalización para la comparación de distintos software.

El análisis de los Puntos de Función se desarrolla considerando cinco parámetros básicos externos del Sistema (PRESSMAN 1998):

1. Entrada (EI, External Input).

2. Salida (EO, External Output).

3. Consultas (EQ, External Query).

4. Ficheros Lógicos Internos (ILF, Internal Logic File).

5. Ficheros Lógicos Externos (EIF, External Interface File).

Con estos parámetros, se determinan los puntos de función sin ajustar (PFsA), luego se utiliza COCOMO para calcular el esfuerzo de desarrollo y otros indicadores a partir de la conversión de los puntos de función sin ajustar en líneas de código fuente. A este valor, se le aplica un Factor de Ajuste obtenido en base a unas valoraciones subjetivas sobre la aplicación y su entorno; es decir, las características generales del sistema. Se debe aplicar el coeficiente de conversión de acuerdo a la experiencia para obtener el esfuerzo de desarrollo.

Podemos decir entonces que los puntos de función aparecen con ventajas substanciales por sobre las líneas de código, para fines de estimación temprana del tamaño del software, y por ende, del esfuerzo de desarrollo. Además es una medida ampliamente utilizada, y con éxito, en muchas organizaciones que desarrollan software en forma masiva (HURTADO 2007).

1.3.8 COCOMO II y los Puntos de Función.

Debido a la complejidad de los proyectos de software, el modelo original COCOMO, fue modificado, denominándose al modelo actual COCOMO II. El nuevo modelo permite determinar el esfuerzo y tiempo de un proyecto de software a partir de los puntos de función sin ajustar, lo cual supone una gran ventaja, dado que en la mayoría de los casos es difícil determinar el número de líneas de código de que constará un nuevo desarrollo, en especial cuando se tiene poca o ninguna experiencia previa en proyectos de software. Esto hace que ambos modelos, Puntos de Función y COCOMO sean perfectamente compatibles y complementarios. Su triunfo depende ampliamente de la adaptación del modelo a las necesidades de la organización, usando datos históricos; los cuales no siempre están disponibles.

1.4 Mediciones.

"Cuando pueda medir lo que está diciendo y expresarlo con números, ya conoce algo sobre ello; cuando no pueda medir, cuando no pueda expresar lo que dice con números, su conocimiento es precario y deficiente: puede ser el comienzo del conocimiento, pero en tus pensamientos apenas estás avanzando hacia el escenario de la ciencia" (KELVIN 2000).

Es una afirmación indiscutible que las mediciones son cruciales en el progreso de todas las ciencias. El progreso científico se logra a través de observaciones y generalizaciones basadas en datos y mediciones y la derivación de teorías como resultado de estas (KAN. 2000). Sin la verificación a través de los datos y las mediciones, las teorías y las proposiciones permanecerían en un nivel abstracto.

La clave para manejar los riesgos es la medición y se sabe que negocios exitosos tienen mediciones exitosas. Una medida proporciona una indicación cuantitativa de extensión, cantidad, dimensiones, capacidad o tamaño de algunos atributos de un proceso o producto.

Para lograr una buena calidad del producto es necesario identificar las mediciones y los criterios que serán utilizados para identificar el nivel deseado de la calidad y determinar si se está alcanzando, siendo los métodos de estimación y las métricas de software una de las vías más utilizadas para cuantificar y valorar resultados. **Anexo 1** y **Anexo 2.**

1.5 Métricas de software.

A la hora de planificar un proyecto es esencial hacer las estimaciones del esfuerzo humano, a través de las mediciones de software, obteniendo registros históricos de datos. Se mide el software con el objetivo de indicar la calidad del producto y evaluar la productividad de las personas involucradas durante el desarrollo del software, para evaluar los beneficios derivados del uso de nuevas técnicas y herramientas; para establecer una línea base para la estimación y para justificar el uso de nuevas herramientas y la necesidad de información. Las palabras claves a la hora de medir son: señalar, predecir, evaluar y mejorar. Las mediciones de software han venido estudiándose desde hace ya varios años y precisamente son conocidas como métricas de software.

Es muy común asociar la palabra métrica con los términos medida y medición, aunque son completamente diferentes, es por ello que, existe una confusión en la utilización de los términos métricas

de software y mediciones de software, sin embargo, en la literatura los términos métrica, medida o medición son usados como sinónimos (ZUSE 1995).

1.5.1 Relación entre mediciones y métricas.

Una *medida* proporciona una indicación cuantitativa de la extensión, cantidad, dimensiones, capacidad o tamaño de algunos atributos de un proceso o producto.

La *medición* es el acto de determinar una medida.

Una *métrica* es una medida cuantitativa del grado en que un sistema, componente o proceso posee un atributo dado.

Las medidas son las que se utilizan para comparar. En otras palabras podemos decir que:

- La medida captura una característica individual.
- La medición permite capturar dicha característica.
- La métrica permite relacionar y comparar mediciones.

Las métricas definen que es lo que se va a estimar y los métodos de estimación definen como una métrica es estimada.

Un indicador es una métrica o combinación de métricas que proporcionan una visión profunda del proceso de desarrollo del software, del proyecto de software y del producto en si (esfuerzo, costo), por tanto las métricas son el fundamento de los indicadores.

1.5.2 ¿Qué entendemos por métricas de software?

Las métricas de software las han definido como: “La aplicación continua de mediciones basadas en técnicas para el proceso de desarrollo del software y sus productos para suministrar información relevante a tiempo, así el planificador junto con el empleo de estás técnicas mejorará el proceso y sus productos”.

Figura 2: Métricas de software

Se utilizan fundamentalmente para (ZUSE 1995):

- Obtener las bases para la estimación.
- Seguir el progreso de los proyectos.
- Determinar la complejidad (relativa).
- Ayudar a comprender cuando se ha alcanzado un estado deseado de calidad.
- Analizar los defectos.
- Validar experimentalmente las mejores prácticas.

La funcionalidad principal de las métricas, como lo indica la **Figura 1.3**, es medir aspectos o particularidades específicas de los productos obtenidos. Las métricas del software responden a dos objetivos fundamentales, la valoración y la estimación. Las principales magnitudes dentro de la valoración son la calidad, fiabilidad y la productividad; mientras que las magnitudes de la estimación corresponden al esfuerzo y al tiempo.

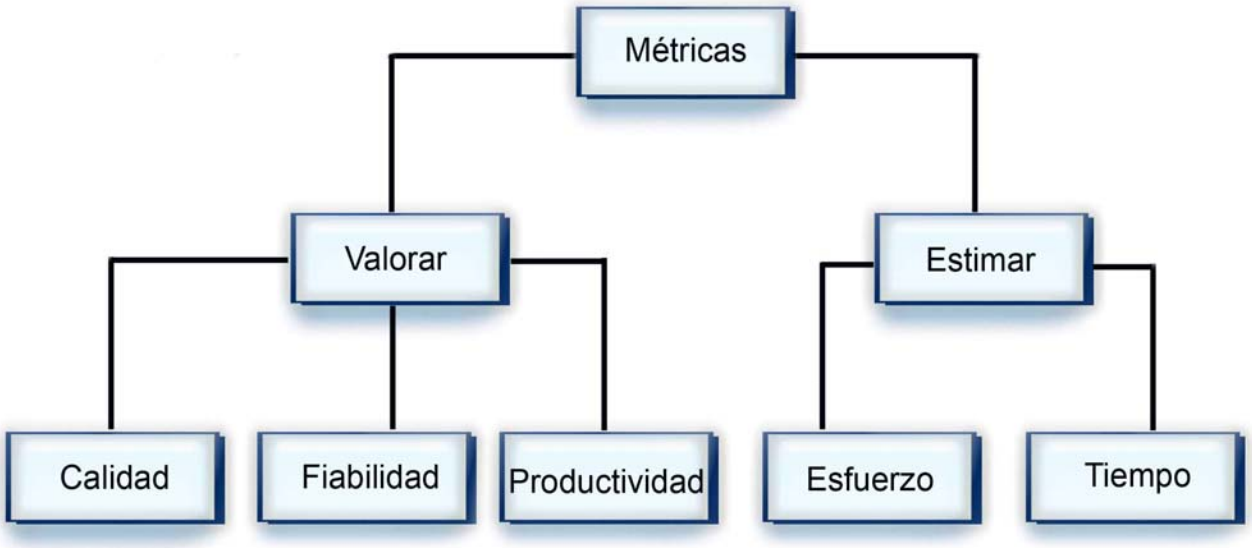

Figura 3: Funcionalidad de las métricas

1.5.3 Propiedades de las métricas.

Cientos de métricas han sido propuestas para el software pero no todas proporcionan suficiente soporte práctico para su desarrollo, pues algunas de ellas exigen mediciones demasiado complejas, de otras no se tienen los conocimientos necesarios como para atreverse a aplicarlas y otras violan las nociones básicas de lo que realmente es el software de alta calidad. Es por ello que se han definido una serie de propiedades que deben acompañar a las métricas de software para que sean efectivas, por tanto la métrica obtenida y las medidas que conducen a ello deben cumplir con las siguientes características fundamentales:

- Simple y fácil de calcular: debería ser relativamente fácil de aprender a obtener la métrica y su cálculo no obligará a un esfuerzo o a una cantidad de tiempo inusuales.
- Empírica e intuitivamente persuasiva: la métrica debería satisfacer las nociones intuitivas del ingeniero de software sobre el atributo del producto en cuestión (por ejemplo: una métrica que mide la cohesión de un módulo debería aumentar su valor a medida que crece el nivel de cohesión).

- Consistente en el empleo de unidades y tamaños: el cálculo matemático de la métrica debería utilizar medidas que no lleven a extrañas combinaciones de unidades. Por ejemplo, multiplicando el número de personas de un equipo por las variables del lenguaje de programación en el programa resulta una sospechosa mezcla de unidades que no son intuitivamente concluyentes.
- Independiente del lenguaje de programación: las métricas deberían apoyarse en el modelo de análisis, modelo de diseño o en la propia estructura del programa. No deberían depender de los caprichos de la sintaxis o semántica del lenguaje de programación.
- Un mecanismo eficaz para la retroalimentación de la calidad: la métrica debería suministrar al desarrollador de software información que le lleve a un producto final de superior calidad.

1.5.4 Métricas del Producto.

Los productos son artefactos que pueden ser documentos, componentes, modelos, diagramas, módulos, a los cuales se les aplican métricas para obtener mediciones de cada uno de estos productos. Las métricas del producto describen características como el tamaño, complejidad, rasgos del diseño, rendimiento y nivel de calidad (KAN. 2000).

En general las características a medir de los artefactos del producto son (RUP 2003):

- Tamaño: Las métricas del tamaño del producto se refieren generalmente al volumen del producto desarrollado. Incluyen líneas de código (LOC), número de ficheros, páginas de la documentación.
- Calidad:
    - Defectos: indicadores de que un artefacto no funciona como ha sido especificado, o cualquier otra característica indeseable.
    - Complejidad de una estructura o un algoritmo: mientras mayor sea la complejidad y más difícil sea de comprender y modificar la estructura del sistema, mayor probabilidad habrá de que falle.
    - Acoplamiento: mediciones de cuántos elementos del sistema están interconectados y cuán extensivamente.
    - Cohesión: mediciones de cuán bien un elemento o un componente cumple con los requerimientos de tener un sólo y bien definido propósito.

  - Primitividad: el grado en el cual las operaciones o métodos de una clase pueden estar compuestos por otros de la misma clase.

- Totalidad: medición de la magnitud en la cual un artefacto cumple con todos los requerimientos (plan / real).
- Rastreabilidad: Indicadores de que los requerimientos de determinado nivel se están satisfaciendo por determinados artefactos, o que todos los artefactos tengan razón de existir.
- Volatilidad: el grado de cambio de un artefacto debido a defectos o a cambios en los requerimientos.
- Esfuerzo: medición del trabajo (Unidad de tiempo del personal) que es requerido para producir un artefacto.

1.5.5 Métricas del Proyecto.

El proyecto debe ser caracterizado en términos de tamaño, tipo, complejidad y formalidad porque condicionan expectativas sobre las distintas tendencias a seguir a la hora de las mediciones. Algunas de las características que describen son: el número de programadores de un software, el costo, la planificación, la productividad del equipo, entre otros.

Las principales métricas a tener en cuenta son (RUP 2003):

- Modularidad: Promedio de daños debido a cambios perfectivos o correctivos en la implementación.
- Adaptabilidad: promedio de esfuerzo debido a cambios perfectivos o correctivos en la implementación.
- Madurez: tiempo de prueba activo / número de cambios correctivos.
- Mantenimiento: mantenimiento productivo / desarrollo productivo.
- Progreso del proyecto: debe reportarse basándose en el plan del proyecto desde la perspectiva del valor devengado.

Los indicadores que se tienen en cuenta dentro del proyecto, permiten:

- Evaluar el estado del proyecto en curso.

- Seguir la pista de riesgos potenciales.
- Detectar áreas problemáticas antes de que se conviertan en críticas.
- Ajustar el flujo y las áreas de trabajo.
- Evaluar la habilidad del equipo de proyecto en controlar la calidad de los productos de trabajo de la IS.

La utilización fundamental de las métricas del proyecto es para minimizar la planificación de desarrollo, guiando los ajustes necesarios que eviten retrasos y mitiguen problemas y riesgos potenciales; así como para evaluar la calidad de los productos en el momento actual, modificando el enfoque técnico para mejorar la calidad.

1.5.6 Métricas del Proceso.

Los procesos de software pueden definirse como los pasos definidos para determinar quién, cuándo, cómo y dónde, debe hacer cada actividad dentro del proceso de desarrollo de software. Las métricas del proceso brindan un mayor enfoque sobre la calidad lograda como consecuencia de un proceso repetible y ordenado. Este proceso es un factor clave y controlable para mejorar la calidad del software y el rendimiento del trabajo. Es importante conocer que las métricas del proceso se van a obtener de las métricas del proyecto.

Las métricas del proceso dependen esencialmente del entorno de desarrollo. Es necesario medir atributos específicos de los propios procesos, como el tiempo empleado, su coste y el esfuerzo requerido. La relación entre las medidas de los resultados obtenidos en un proceso y los recursos usados en él permitirá medir la productividad, atributo este clave para estimar costo y esfuerzo (ESCORIAL 2006).

Un ejemplo de este tipo de métricas es el tiempo empleado para desarrollar un elemento software que depende de distintos factores externos (dificultad del problema, capacidad del personal, metodología empleada).

Para definir completamente un proceso, las métricas deben ser especificadas con el nivel de formalidad más bajo en las actividades planificadas. Las actividades deben ser planificadas por el administrador del proyecto, usando un conjunto inicial de estimados. Por lo tanto, se debe mantener un registro de los valores reales durante el transcurso del tiempo y cualquier modificación de lo estimado que se haga. Es

importante que en área donde se detecten problemas, no se utilicen las métricas en contra de los desarrolladores pues a partir de entonces, los datos obtenidos no serán reales y no se cumplirá el objetivo. Tiene una importancia vital el hecho de que los desarrolladores comprendan la necesidad de las mediciones y las vean como una justificación para mejorar.

Los indicadores utilizados dentro del proceso permiten:

- Al gestor evaluar lo que funciona y lo que no.
- A la organización, tener una visión profunda de la eficacia de un proceso ya existente.

Las métricas del proceso son estratégicas porque determinan el curso del proceso de producción de software. Las métricas del proyecto son tácticas porque establecen el curso del proyecto actual. La primera aplicación de las métricas del proyecto ocurre durante la estimación (a través de datos históricos).

1.5.7 Métricas de Recursos.

Las métricas de los recursos medirán los elementos que incluyen a personas (experiencia, habilidades, coste, funcionamiento), los métodos y las herramientas (en términos de efecto sobre productividad y calidad, coste), tiempo, esfuerzo, y presupuesto (recursos consumidos, recursos restantes). El modelo de COCOMO requiere la caracterización de la experiencia del personal y del ambiente del desarrollo de la capacidad y del software, y es un buen marco en el cual guardar estas métricas. El gasto, el presupuesto, y la información del horario vendrán del plan del proyecto (RUP 2003).

Las métricas de recursos se aplican, fundamentalmente, a las horas de labor, el principal recurso del desarrollo de software. Aquí lo que concierne son las horas trabajadas, categorías de trabajo y realización de tareas.

Cualquier interrupción tiene tres costos potenciales: la pérdida de tiempo, el tiempo adicional que toma reconstruir el momento en que fue interrumpido y la probabilidad incremental de cometer errores. Un foco principal de cualquier esfuerzo para mejorar la productividad o el ciclo de tiempo debe identificar y reducir estas distracciones.

1.5.8 Métricas de Productividad.

La productividad se va a definir como la razón entre entradas/salidas. Para el caso de la ingeniería de software, se define como la cantidad de esfuerzo requerido para lograr un cierto grado de productividad.

Las métricas orientadas a la productividad se centran en el rendimiento del proceso de la ingeniería de software, o sea, en que tan productivo es el software que se va a diseñar. Cuando se trata de establecer métricas de productividad y calidad en la construcción de software, o para realizar estimaciones de coste o duración, es imprescindible disponer de una medida fiable y comprensible del tamaño de lo que se construye. Por este motivo cambios bruscos en las medidas de productividad entre proyectos, es una indicación de que no se está siguiendo un proceso estándar. En la medida de que los grupos de trabajo se consoliden en un proceso estándar de desarrollo, los rangos de productividad se deben estabilizar y ser más concientes.

En la actualidad, la industria del software necesita tener un control y seguimiento más preciso, predecible y repetido sobre los procesos de producción y los productos de software. Con el objetivo de planear el progreso de los proyectos, mediante las métricas se obtienen las bases para estimar, ayudando a analizar la complejidad y obtener mejores prácticas con vistas a lograr mejores estimaciones en futuras decisiones.

En la UCI la planificación de proyectos de software es una tarea que se consolida con la práctica. Como universidad joven, el poco tiempo de experiencia no alcanza aún a obtener una base de registros históricos que permitan plantear una estrategia común para estimar y planificar con precisión todos los proyectos. El método más reconocido y aplicado mundialmente en las estimaciones de proyectos de software es COCOMO, pero como este basa sus estimaciones a partir de registros históricos obtenidos de proyectos ya finalizados, los especialistas se vieron imposibilitados en utilizarlo, por el inconveniente de no contar en la universidad con un registro de datos, recurriendo a otros métodos de estimación.

Especialistas con cierto grado de experiencia en la UCI elaboraron un método de estimación, dando una respuesta cuantitativa a las estimaciones durante el desarrollo de los proyectos software. Los métodos de estimación que se tuvieron en cuenta según las clasificaciones referenciadas anteriormente, fueron el juicio del experto, la analogía, los métodos empíricos y teóricos; consiguiendo fusionarlos en correspondencia a las características de la metodología utilizada en la universidad.

Capítulo 2: Método de estimación

El capítulo se inicia abordando los factores de riesgo que influyen en la elaboración de un buen método de estimación. Se explica detalladamente en qué consiste el método de estimación elaborado en la UCI para las etapas de levantamiento de requisitos y prototipo, haciendo una breve descripción de las actividades que se identifican en cada una, con los roles que intervienen. Finalmente, se realiza un análisis teórico del método de estimación.

2.1 ¿Cómo elaborar un buen método de estimación?

Para elaborar un buen método de estimación, previendo la viabilidad de los atributos: tamaño y esfuerzo, hay que hacer un análisis también de algunos factores de riesgo que inciden en las decisiones de las estimaciones antes y/o durante el proceso de desarrollo, y gestionarlos para planear un futuro más controlado, que brinde ayuda al equipo de desarrollo, gestionando la incertidumbre de lo que se conoce y representa como un problema potencial. El objetivo de esta actividad es estimar aproximándose más a la realidad, cumpliendo con las características que identifican un buen método de estimación.

A continuación se enumeran algunas características deseables en todo método de estimación que pretenda apoyar la etapa de planificación de proyectos de desarrollo de software (VARAS 1995):

- Poder adaptarse a la productividad de la organización.
- Considerar la comunicación entre personas.
- Incorporar guías útiles para estimar aquellos parámetros que son subjetivos o no, que se deducen en forma explícita a partir del modelo.
- Usable.
- Constar de etapas simples de entender y definidas en forma precisa.
- Objetivo.
- Proveer medios para adaptarse a cambios en el ambiente de desarrollo.

2.1.1 Factores de riesgo que inciden en la estimación de un buen método.

El riesgo es la posibilidad de que existan consecuencias indeseables o inconvenientes, de un acontecimiento, cuya aparición no se puede determinar con prioridad. Algunos de los riesgos que influyen en las estimaciones son:

- Evaluación de la organización: relacionado con la definición de procesos y otras características de la organización.
- Definición del proceso: grado de definición del proceso de software y su seguimiento por la organización de software.
- Complejidad del sistema a construir: depende del número de elementos que interactúan entre sí y no sólo de su cantidad sino también de su calidad.
- Tecnología a construir: asociados con la complejidad del sistema a construir y la tecnología de punta que contiene el sistema.
- Entorno de desarrollo: disponibilidad y calidad de las herramientas que se van a emplear en la construcción del producto.
- Experiencia del equipo de desarrollo: experiencia técnica y de proyectos de los desarrolladores que van a realizar el trabajo.

2.1.2 Gestionar riesgos.

Cuando se considera el riesgo en el contexto de la ingeniería del software, los tres pilares conceptuales se basan en que (MENÉNDEZ 2004):

- El riesgo afecta a los futuros acontecimientos.
- El riesgo implica cambio, que puede venir dado por cambios de opinión, de acciones, de lugares.
- El riesgo implica elección y la incertidumbre que entraña la elección.

El futuro es lo que preocupa, ¿qué riesgos podrían hacer que el proyecto fracase? El cambio es la preocupación ¿cómo afectarán los cambios en los requisitos del cliente, en las tecnologías de desarrollo, en los ordenadores a los que van dirigidos, el proyecto y todas las entidades relacionadas con él, al cumplimiento de la planificación temporal y al éxito en general? Y por último la elección ¿qué métodos y

herramientas se deberían emplear, cuánta gente debería estar implicada, qué importancia hay que darle a la calidad?

Se han considerado dos formas de clasificar estrategias para controlar los riesgos: reactivas y proactivas.

La estrategia reactiva es la que reacciona en el momento que ocurren los problemas, para a partir de ahí combatirlos. En el mejor de los casos, la estrategia reactiva supervisa el proyecto en previsión de posibles riesgos. Los recursos se ponen aparte, en caso de que pudieran convertirse en problemas reales, pero lo más frecuente es que el equipo de software no haga nada respecto a los riesgos hasta que algo esté mal. Después el equipo se agiliza para corregir el problema rápidamente. La gestión entra en crisis, encontrándose el proyecto en peligro real.

La otra estrategia considerada como la más inteligente para el control del riesgo es ser proactivo. La estrategia proactiva empieza mucho antes de que comiencen los trabajos técnicos. Se identifican los riesgos potenciales, se valoran su probabilidad y su impacto y se establece una prioridad según su importancia. Después el equipo de software establece un plan para controlar el riesgo. El primer objetivo es evitar el riesgo, aunque es poco común que todos puedan ser detectados. Entonces el equipo trabaja para desarrollar un plan de contingencia que le permita responder de una manera eficaz y controlada.

2.1.2.1 Riesgos del software.

Aunque ha habido amplios debates sobre la definición adecuada para riesgo de software, hay acuerdo común en que el riesgo siempre implica dos características (PRESSMAN 1998):

- Incertidumbre: El acontecimiento que caracteriza al riesgo puede o no ocurrir; por ejemplo, no hay riesgos de un 100 por ciento de probabilidad.
- Pérdida: Si el riesgo se convierte en una realidad, ocurrirán consecuencias no deseadas o pérdidas.

Cuando se analizan los riesgos es importante analizar el nivel de incertidumbre y el grado de pérdidas asociado con cada riesgo. Para hacerlo, se consideran diferentes categorías de riesgos.

Los riesgos del proyecto amenazan al plan del proyecto. Es decir, si los riesgos del proyecto se hacen realidad, es probable que la planificación temporal del proyecto se retrase y que los costos aumenten. Los

riesgos del proyecto identifican los problemas potenciales de presupuesto, planificación temporal, personal (asignación y organización), recursos, clientes y requisitos y su impacto en un proyecto de software.

Los riesgos técnicos amenazan la calidad y la planificación temporal del software que hay que producir. Si un riesgo técnico se convierte en realidad, la implementación puede llegar a ser difícil o imposible. Los riesgos técnicos identifican problemas potenciales de diseño, implementación, de interfaz, verificación y de mantenimiento. Además las ambigüedades de especificaciones, incertidumbre técnica, técnicas anticuadas y las "tecnologías de punta" son también factores de riesgo. Los riesgos técnicos ocurren porque el problema es más difícil de resolver de lo que pensábamos.

Los riesgos del negocio amenazan la viabilidad del software a construir. A menudo ponen en peligro el proyecto o el producto. Los candidatos para los principales riesgos del negocio son (MENÉNDEZ 2004):

- Construir un producto o sistema excelente que no quiere nadie en realidad (riesgo de mercado).
- Construir un producto que no encaja en la estrategia comercial general de la compañía (riesgo estratégico).
- Perder el apoyo de una gestión experta debido a cambios de enfoque o a cambios de personal (riesgo de dirección).
- Perder presupuesto o personal asignado (riesgos de presupuesto).

Es importante recalcar que no siempre funciona una categorización tan sencilla. Algunos riesgos son simplemente imposibles de predecir.

Los riesgos conocidos son todos aquellos que se pueden descubrir después de una cuidadosa evaluación del plan del proyecto del entorno técnico y comercial en el que se desarrolla el proyecto y otras fuentes de información fiables (MENÉNDEZ 2004), ejemplo: fechas de entrega poco realistas, falta de especificación de requisitos o de ámbito del software, o un entorno pobre de desarrollo. Los riesgos predecibles se extrapolan de la experiencia en proyectos anteriores, ejemplo: cambio de personal, mala comunicación con el cliente, disminución del esfuerzo del personal a medida que atienden peticiones de mantenimiento, pueden ocurrir pero son extremadamente difíciles de identificar por adelantado.

2.1.2.2 Identificación del riesgo.

La identificación del riesgo es un intento sistemático para especificar las amenazas de la planificación del proyecto (estimaciones, planificación temporal, carga de recursos). Identificando los riesgos conocidos y predecibles, el gestor del proyecto da un paso adelante para evitarlos cuando sea posible y controlarlos cuando sea necesario (MENÉNDEZ 2004).

Existen dos tipos diferentes de riesgos: genéricos y específicos del producto.

Los riesgos genéricos son una amenaza potencial para todos los proyectos de software. Los específicos de producto sólo los pueden identificar los que tienen una clara visión de la tecnología, el personal y el entorno específico del proyecto en cuestión. Para identificar los riesgos específicos del producto se da una respuesta a la siguiente pregunta: ¿Qué características especiales de este producto pueden estar amenazadas por el plan del proyecto?

La clave del éxito está en adelantarse a los problemas y tener acciones contempladas para evitar que sucedan o disminuir su impacto y tanto los riesgos genéricos como los específicos del producto se deberían identificar sistemáticamente, porque: "Si no atacas activamente a los riesgos, ellos te atacarán activamente a ti" (BASTERRA 2006).

Un método para identificar riesgos es crear una lista de comprobación de elementos de riesgo. La lista de comprobación se puede utilizar para identificar riesgos definidos al inicio para estimar un buen método.

La lista de comprobación de factores de riesgo puede organizarse de diferentes maneras, respondiendo a cuestiones relevantes de cada una de los factores de riesgo para estimar, permitiendo valorar su impacto. Finalmente, se lista un conjunto de "componentes y controladores del riesgo" junto con sus probabilidades de aparición. Los controladores del rendimiento, el soporte, el coste y la planificación temporal del proyecto se estudian como respuesta a las preguntas.

Riesgos del impacto en el negocio.

La siguiente lista de comprobación de elementos de riesgo identifica riesgos genéricos asociados con el impacto en el negocio(MENÉNDEZ 2004):

- ¿Efecto de este producto en los ingresos de la compañía?
- ¿Viabilidad de este producto para los gestores expertos?

- ¿Es razonable la fecha límite de entrega?
- ¿Número de clientes que usarán este producto y la consistencia de sus necesidades relativas al producto?
- ¿Número de otros productos/sistemas con los que este producto debe tener interoperatividad?
- ¿Sofisticación del usuario final?
- ¿Cantidad y calidad de la documentación del producto que debe ser elaborada y entregada al cliente?
- ¿Costos asociados por un retraso en la entrega?
- ¿Costos asociados con un producto defectuoso?

Cada respuesta para el producto a desarrollar debe compararse con la experiencia anterior. Si se obtiene una gran desviación del porcentaje o si las magnitudes son similares, pero los resultados anteriores fueron poco satisfactorios, el riesgo es grande.

Riesgos relacionados con el cliente.

No todos los clientes son iguales. "*Los clientes tienen diferentes necesidades. Algunos saben lo que quieren; otros saben lo que no quieren. Algunos están deseando saber todos los detalles, mientras que otros se quedan satisfechos con vagas promesas"* (PRESSMAN 1998).

Los clientes tienen diferentes personalidades. Algunos disfrutan siendo clientes (la tensión, la negociación, las recompensas psicológicas de un buen producto).

Otros preferirían no ser clientes en absoluto. Algunos aceptarían felizmente cualquier cosa que se les entregara y le sacarían el mejor provecho a un producto pobre. Otros se quejarán amargamente cuando les falte calidad; algunos darán las gracias cuando la calidad es buena; otros se quejarán por todo.

Los clientes se contradicen a menudo. Quieren todo para ayer y gratis. A menudo, el producto se ve atrapado entre las propias contraindicaciones del cliente.

Un "mal" cliente puede tener un profundo impacto en la habilidad del equipo de software para completar el proyecto a tiempo, dentro del presupuesto y también representa una amenaza significativa al plan del proyecto.

La siguiente lista de comprobación de elementos de riesgo identifica riesgos genéricos asociados con diferentes clientes (MENÉNDEZ 2004):

- ¿Ha trabajado con el cliente anteriormente?
- ¿Tiene el cliente una idea formal de lo que se requiere? ¿Se ha molestado en escribirlo?
- ¿Aceptará el cliente gastar su tiempo en reuniones formales de requisitos para identificar el ámbito del proyecto?
- ¿Está dispuesto el cliente a establecer una comunicación fluida con el desarrollador?¿Está dispuesto el cliente a participar en las revisiones?
- ¿Está dispuesto el cliente a dejar a su personal hacer el trabajo? Es decir, ¿resistirá la tentación de mirar por encima del hombro durante el trabajo técnico?
- ¿Entiende el cliente el proceso del software?

Si la respuesta a alguna de estas preguntas es "no", se debería hacer una investigación más profunda para valorar el potencial de riesgo.

Riesgos del proceso.

Si el proceso del software no está bien definido; si el análisis, diseño y pruebas se realizan sobre la marcha; si la calidad es un concepto que todo el mundo estima importante, pero por la que nadie actúa de manera tangible para alcanzarla, entonces el proyecto está en peligro (MENÉNDEZ 2004).

Aspectos del proceso:

- ¿Se ha desarrollado en la organización una descripción escrita del proceso del software a emplear en este proyecto?
- ¿Están de acuerdo los miembros del personal con el proceso del software tal y como está documentado y están dispuestos a usarlo?
- ¿Se emplea este proceso del software para otros proyectos?
- ¿Se ha proporcionado una copia de los estándares de ingeniería del software publicados a cada desarrollador y gestor de software?

- ¿Se han desarrollado diseños de documentos y ejemplos para todas las entregas definidas como parte del proceso del software?
- ¿Se llevan a cabo regularmente revisiones técnicas formales de las especificaciones de requisitos, diseño y código?
- ¿Se llevan a cabo regularmente: revisiones técnicas de los procedimientos de prueba y de los casos de prueba?
- ¿Se documentan todos los resultados de las revisiones técnicas, incluyendo los errores encontrados y recursos empleados?
- ¿Existe algún mecanismo para asegurarse de que el trabajo realizado en un proyecto se ajusta a los estándares de ingeniería de software?
- ¿Se emplea una gestión de configuración para mantener la consistencia entre los requisitos del sistema/software, diseño, código y casos de prueba?
- ¿Hay algún mecanismo de control de cambios de los requisitos del cliente que impacten en el software?

Aspectos técnicos:

- ¿Se emplean técnicas de especificación de aplicaciones para ayudar en la comunicación entre el cliente y el desarrollador?
- ¿Se emplean métodos específicos para el análisis del software?
- ¿Emplea un método específico para el diseño de datos?
- ¿Se han definido y empleado reglas específicas para la documentación del código?¿Emplea métodos específicos para el diseño de casos de prueba?
- ¿Se emplean herramientas de software para apoyar la planificación y el seguimiento de las actividades?
- ¿Se emplean herramientas de gestión de configuración para controlar y seguir los cambios a lo largo de todo el proceso del software?

- ¿Se emplean herramientas de software para apoyar los procesos de análisis y diseño del software?
- ¿Se emplean herramientas para crear prototipos software?
- ¿Se emplean herramientas de software para dar soporte a los procesos de prueba?
- ¿Se emplean herramientas de software para soportar la producción y gestión de la documentación?
- ¿Se han establecido métricas de calidad para todos los proyectos de software?
- ¿Se han establecido métricas de productividad para todos los proyectos de software?

Si la mayoría de las cuestiones anteriores se han respondido negativamente, el proceso del software es débil y el riesgo es alto.

Riesgos tecnológicos.

Alcanzar las fronteras de la tecnología actual es un reto interesante, porque impulsa al profesional a utilizar al máximo todo su ingenio y talento, aunque puede ser muy arriesgado también. La siguiente lista de comprobación de elementos de riesgo identifica riesgos genéricos asociados con la técnica a construir:

- ¿Es nueva para su organización la tecnología a construir?
- ¿Demandan los requisitos del cliente la creación de nuevos algoritmos o tecnología de entrada o salida?
- ¿El software interactúa con hardware nuevo o no probado?
- ¿Interactúa el software a construir con productos software suministrados por el vendedor que no se hayan probado?
- ¿Interactúa el software a construir con un sistema de base de datos cuyo funcionamiento y rendimiento no se han comprobado en esta área de aplicación?
- ¿Demandan los requisitos del producto una interfaz de usuario especial?
- ¿Demandan los requisitos del producto la creación de componentes de programación distintos de; los que su organización haya desarrollado hasta ahora?

- ¿Demandan los requisitos el empleo de nuevos métodos de análisis, diseño o pruebas?
- ¿Imponen excesivas restricciones de rendimiento los requisitos del producto? ¿No está seguro el cliente de que la funcionalidad pedida sea factible?

Si la respuesta a alguna de estas preguntas es afirmativa, se debería realizar una investigación más profundidad para valorar el riesgo potencial.

Riesgos del entorno de desarrollo.

El esfuerzo, incluso de experimentados profesionales puede verse afectada en determinado momento por la utilización de algunas herramientas inadecuadas.

El entorno de ingeniería del software soporta al equipo del proyecto, al proceso y al producto, pero si el entorno es malo, puede ser una fuente de riesgos significativa.

La siguiente lista de comprobación de elementos de riesgo identifica riesgos genéricos asociados con el entorno de desarrollo (MENÉNDEZ 2004):

- ¿Tenemos disponible una herramienta de gestión de proyectos de software?
- ¿Tenemos disponible una herramienta de gestión del proceso del software?
- ¿Existen herramientas de análisis y diseño disponibles?
- ¿Proporcionan las herramientas de análisis y diseño, métodos apropiados para el producto a construir?
- ¿Hay disponible compiladores o generadores de código apropiados para el producto a construir?
- ¿Hay disponibles herramientas de pruebas apropiadas para el producto a construir?
- ¿Tenemos disponibles herramientas de gestión de configuración software?
- ¿Hace uso el entorno de bases, de datos o información almacenada?
- ¿Están todas las herramientas de software integradas entre sí?
- ¿Se ha formado a los miembros del equipo del proyecto en todas las herramientas?
- ¿Existen expertos disponibles para responder todas las preguntas que surjan sobre las herramientas?

- ¿Es adecuada la ayuda en línea y la documentación de las herramientas?

Si se ha contestado negativamente a la mayoría de las preguntas anteriores, el entorno de desarrollo es débil y el riesgo es alto.

Riesgos asociados con la experiencia de los desarrolladores.

Se sugieren las siguientes cuestiones para valorar los riesgos asociados con la experiencia de los desarrolladores (MENÉNDEZ 2004):

- ¿Disponemos de la mejor gente?
- ¿Tiene el personal todos los conocimientos adecuados?
- ¿Tenemos suficiente personal?
- ¿Se ha asignado al personal para toda la duración del proyecto?
- ¿Habrá parte del personal del proyecto que trabaje sólo durante parte de él?
- ¿Dispone el personal de las expectativas correctas sobre el trabajo?
- ¿Ha recibido el personal la formación adecuada?
- ¿Será mínimo el movimiento del personal para permitir la continuidad?

Si la respuesta a alguna de estas preguntas es "no", se debería hacer una investigación más profunda para valorar el potencial de riesgo.

2.1.3 Controladores del riesgo.

Es importante identificar los controladores de riesgo que afectan a los componentes software (rendimiento, costo, soporte y planificación temporal), ver **Anexo 1**

- Riesgo de rendimiento: el grado de incertidumbre con el que el producto encontrará sus requisitos y se ajuste para su empleo.
- Riesgo de costo: el grado de incertidumbre que mantendrá el presupuesto del proyecto.
- Riesgo de soporte: el grado de incertidumbre de la facilidad del software para corregirse, adaptarse y ser mejorado.

- Riesgo de la planificación temporal: el grado de incertidumbre con que se podrá mantener la planificación temporal y de que el producto se entregue a tiempo.

2.1.4 Estimación del riesgo.

La estimación del riesgo, intenta medir cada riesgo de dos maneras. La probabilidad de que el riesgo sea real y las consecuencias de los problemas asociados con el riesgo, en caso de ocurrir. El jefe del proyecto, junto con otros gestores y personal técnico, realiza cuatro actividades de proyección del riesgo (MENÉNDEZ 2004):

1. Establecer una escala que refleje la probabilidad percibida del riesgo.
2. Definir las consecuencias del riesgo.
3. Estimar el impacto del riesgo en el proyecto y en el producto.
4. Apuntar la exactitud general de la proyección del riesgo de manera que no haya confusiones.

Una tabla de riesgo (**Tabla 1**) le proporciona al proyecto una sencilla técnica para la estimación del riesgo.

Un equipo de proyecto empieza por listar todos los riesgos (no importa lo remotos que sean) en la primera columna de la tabla. Cada riesgo es categorizado en la segunda columna (Ejemplo: PS implica un riesgo del tamaño del proyecto, BU implica un riesgo de negocio). La probabilidad de aparición de cada riesgo se introduce en la siguiente columna de la tabla. El valor de la probabilidad de cada riesgo puede estimarse por cada miembro del equipo individualmente. De los valores individuales se obtiene la media para obtener una probabilidad consensuada. A continuación se valora el impacto de cada riesgo.

Cada componente de riesgo se valora usando la caracterización presentada en el **Anexo 3** determinando el impacto de los riesgos. Las categorías para cada uno de los cuatro componentes de riesgo: rendimiento, soporte, coste y planificación temporal, son promediados para determinar un valor general de impacto. A continuación se ilustra una tabla de riesgo como ejemplo (MENÉNDEZ 2004).

Riesgos	Categorías	Probabilidad	Impacto
La estimación del tamaño puede ser significativamente baja.	PS	60%	2
Mayor número de usuarios de los previstos.	PS	30%	3
Menos reutilización de la prevista.	PS	70%	2
Los usuarios finales se resisten al sistema.	BU	40%	3
La fecha de entrega estará muy ajustada.	BU	50%	2
Se perderán los presupuestos.	CU	40%	1
El cliente cambiará los requisitos.	PS	80%	2
La tecnología no alcanzará las expectativas.	TE	30%	1
Falta de formación en las herramientas.	DE	80%	3
Personal sin experiencia.	ST	30%	2
Habrá muchos cambios de personal.	ST	60%	2

Valores de Impacto	
1	Catastrófico
2	Crítica
3	Marginal
4	Despreciable

Tabla 1: Desarrollo de una tabla de riesgo

Una vez que se han completado las cuatro primeras columnas de la tabla de riesgo, la tabla es ordenada por probabilidad y por impacto. Los riesgos de alta probabilidad y de alto impacto pasan a lo alto de la tabla, y los riesgos de baja probabilidad caen a la parte de abajo. Esto consigue una priorización del riesgo de primer orden.

La supervisión del riesgo es una actividad de seguimiento del proyecto con tres objetivos principales:

- Valorar cuando un riesgo previsto ocurre de hecho.
- Asegurarse de que los procedimientos para evitar los riesgos se están aplicando apropiadamente.
- Recoger información que pueda emplearse en el futuro para analizar riesgos.

Los problemas que ocurren durante la estimación de software pueden estar relacionados por uno o varios riesgos. Lo fundamental es determinar “el origen” (qué riesgos ocasionan tal problema) a lo largo de todo el proyecto.

2.2 Descripción del método de estimación.

Los elementos que se tendrán en cuenta para la descripción del método de estimación serán los tiempos a emplear y el esfuerzo de cada rol, en dependencia de la complejidad de los casos de uso, así como una descripción de las principales actividades que se realizan dentro de las etapas de levantamiento de

requisitos y prototipo. La complejidad de los casos de uso se determinó de una manera empírica e intuitiva, pues en la no se cuenta con una basta experiencia en las estimaciones de proyectos.

2.2.1 Roles.

Los principales roles que intervienen son el Experto Funcional, el Analista Diseñador, Programador Junior. y el Realizador, cada uno de ellos tienen diferentes desempeño en todas las actividades a desarrollar dentro de las etapas (ARIAS 2006).

- Experto Funcional: Aclarará todas las dudas que surjan del negocio a automatizar durante el levantamiento de requisitos y el resto del desarrollo, participará en las pruebas de calidad que se realicen.
- Analista Diseñador: Participa en la definición del proyecto, interviene en la modelación del negocio, interactúa con el usuario final en la definición de los requisitos de la aplicación, crea el modelo de casos de uso del sistema, define el prototipo de interfaz de usuario elemental, es el responsable de traducir la comunicación entre usuarios finales y desarrolladores, gestionando los requisitos adicionales que aparezcan durante el desarrollo del software. Será el encargado además de interpretar la información como resultado del análisis y traducir al lenguaje de los programadores (interfaz, negocio y acceso a datos), definir los elementos de diseño a tener en cuenta para la implementación de los casos de uso, diseñar la implementación sobre la arquitectura definida, integrar los componentes de la solución y definir las interfaces, así como de dirigir el trabajo de los programadores.
- Programador Junior: A este rol se le asignan las tareas de codificación que poseen menor complejidad dentro del proyecto.
- Realizador: Construye estáticamente el prototipo de interfaz de usuario a partir de las pautas definidas para el diseño de la interfaz, las vistas estáticas que constituyen el punto de partida de los programadores de Interfaz de Usuario.

2.2.2 Estimación de Tiempos.

En el método de estimación se identifican las etapas por las que debe pasar el proceso de desarrollo de software, las actividades incluidas en cada una de ellas, así como la cantidad de horas empleadas en

dependencia de la complejidad Baja, Media o Alta por casos de uso identificados, teniendo en cuenta además los roles que intervienen según la etapa a medir, permitiendo otros cálculos más generales como:

- El Total de Horas empleadas en desarrollar un caso de uso durante el recorrido por las dos primeras etapas, teniendo en cuenta la complejidad.

Etapa	Actividades	Bajo	Medio	Alto
Levantamiento de Requisitos	Planificación, Obtención del PEN, Desarrollo PEN, Identificar las Salidas del Sistema, Elaboración del Modelo Conceptual, Elaboración de CU	12	24	36
Prototipo	Prototipo de Interfaz de Usuario, Mapa de navegación	8	16	22
Total Horas		**20**	**40**	**58**

Tabla 2: Etapas del Proceso de Desarrollo

2.2.3 Estimaciones en dependencia de la complejidad de los casos de uso.

Teniendo en cuenta la complejidad de los casos de uso y las horas que emplean los roles que intervienen en las etapas se mide:

- El total de horas que emplea cada rol durante las dos etapas que estamos tratando, levantamiento de requisitos, prototipo.
- El total de horas que emplean todos los roles que intervienen en cada etapa.
- El promedio Horas/Hombre por etapas.

Complejidad Baja.

Como se muestra en la **Tabla 3**, para un caso de uso de complejidad Baja en el levantamiento de requisitos, el Experto Funcional y el Analista Diseñador, deben emplear 4 y 8 horas de trabajo respectivamente, que hacen el total de las 12 horas empleadas para desarrollar este tipo de caso de uso.

En la realización del prototipo, los roles que intervienen son el Analista Diseñador que emplea 2 horas, el Programador Junior que emplea 2 horas y el Realizador que utiliza 4 horas, que hacen las 8 horas que se emplean para desarrollar estos casos de uso.

BAJO							
Etapa	**Hr**	**Fun**	**A/D**	**P. Sr**	**P.Jr**	**Real**	**Prob**
Levantamiento de Requisitos	**12**	4	**8**				
Prototipo	**8**		2		2	**4**	
		4	10	0	2	4	0

Tabla 3: Complejidad Baja

Complejidad Media.

Según los datos que se muestran en la **Tabla 4**, para un caso de uso de complejidad Media en el levantamiento de requisitos, el Experto Funcional y el Analista Diseñador deben emplear 8 y 16 horas de trabajo respectivamente, que hacen el total de las 24 horas empleadas para desarrollar este tipo de caso de uso.

En la realización del prototipo los roles que intervienen son el Analista Diseñador que emplea 4 horas, el Programador Junior que emplea 4 horas y el Realizador que utiliza 8 horas, que hacen las 16 horas que se emplean para desarrollar estos casos de uso.

MEDIO							
Etapa	**Hr**	**Fun**	**A/D**	**P. Sr**	**P.Jr**	**Real**	**Prob**
Levantamiento de Requisitos	**24**	8	**16**				
Prototipo	**16**		4		4	**8**	
		8	20	0	4	8	0

Tabla 4: Complejidad Media

Complejidad Alta.

Como muestra la **Tabla 5**, para un caso de uso de complejidad Alta en el levantamiento de requisitos, el Experto Funcional y el Analista Diseñador deben emplear 12 y 24 horas de trabajo respectivamente, que hacen el total de las 36 horas empleadas para desarrollar este tipo de caso de uso.

En la realización del prototipo los roles que intervienen son el Analista Diseñador que emplea 6 horas, el Programador Junior que emplea 6 horas y el Realizador que utiliza 10 horas, que hacen las 22 horas que se emplean para desarrollar estos casos de uso.

ALTO							
Etapa	**Hr**	**Fun**	**A/D**	**P. Sr**	**P.Jr**	**Real**	**Prob**
Levantamiento de Requisitos	**36**	12	**24**				
Prototipo	**22**		6		6	**10**	
		12	30	0	6	10	0

Tabla 5: Complejidad Alta

Esfuerzo estimado para cada rol dependiendo de la complejidad.

En el estudio preliminar que se realizó en el proyecto SIGEP, se identificaron 156 casos de uso de complejidad Baja, 144 de complejidad Media y 32 de complejidad Alta.

Con los valores obtenidos anteriormente, se pudo estimar además la productividad de cada rol, siempre teniendo en cuenta la complejidad de los casos de uso.

	Bajo	**Medio**	**Alto**
Roles	**Hr / Hom**	**Hr / Hom**	**Hr / Hom**
Funcionales	624	1.152	384
Analista Diseñador	1.560	2.880	960
Programadores Sr.	-	-	-
Programadores Jr.	312	576	192
Realizadores	624	1.152	320
Probadores	-	-	-
Total	**3.120**	**5.760**	**1.856**

Tabla 6: Esfuerzo estimado por Roles según la Complejidad

En la **Tabla 6** se muestra cuantitativamente las horas/hombre (esfuerzo) empleadas por cada rol, calculándose a través de la multiplicación de la cantidad total de casos de uso según la complejidad, con la cantidad de horas empleadas por cada rol. **(Ver Tablas 3, 4, 5).** La métrica utilizada fue:

*Esfuerzo de cada rol (Horas/Hombre) = Cant. Total de CU (Según complejidad) * Cant. Horas rol.*

Roles	Hr / Rol
Funcionales	2.160
Analista Diseñador	5.400
Programadores Sr.	-
Programadores Jr.	1.080
Probadores	2.096
Realizadores	-
Total	**10.736**

Tabla 7: Esfuerzo total estimado para cada rol

En la **Tabla 7** se muestra cuantitativamente las horas/hombre (esfuerzo) total para cada uno de los roles. Se calcula sumando la cantidad total de horas empleadas por un rol en los casos de uso de complejidad Alta, Media y Baja.

Esfuerzo Total (Rol) = Esfuerzo Complejidad Baja (Rol) + Esfuerzo Complejidad Media (Rol) + Esfuerzo Complejidad Baja (Rol).

Según los datos que se obtuvieron anteriormente, se pudo determinar que la etapa de levantamiento de requisitos y de prototipo, necesitarían 2 semanas de trabajo cada una respectivamente, para completar todas sus actividades satisfactoriamente, dando como resultado final 1 mes para el desarrollo de las mismas.

2.3 Descripción de las actividades por etapas.

A continuación se describe las actividades identificadas en el método de estimación, teniendo en cuenta algunos aspectos para el desarrollo de estas.

2.3.1 Etapa Levantamiento de Requisitos.

El levantamiento de requisitos es el proceso mediante el cual los requisitos de software son identificados, documentados y revisados para garantizar que el software en construcción satisfaga los objetivos de los clientes y usuarios, añadiendo valor al negocio. El levantamiento de requisitos tiene como objetivo general determinar qué debe hacer el sistema de software a desarrollar (RUP 2003).

Planificación.

La planificación, tiene como objetivo principal, como su nombre lo indica, planificar las actividades para el levantamiento de requisito. Dentro de esta etapa se desarrolla un primer paso que tiene como objetivo identificar y evaluar las áreas de la organización para determinar cuáles están listas para iniciar el levantamiento de requisitos. Sus resultados son imprescindibles para el desarrollo exitoso de las restantes actividades concebidas en esta etapa (HERRERA 2006).

Obtención de los PEN.

Para identificar los casos de uso es necesario conocer primero cómo es el negocio que se va a modelar, identificar como funcionan los Procesos Elementales y describir la interrelación que existe entre ellos. Se identifica como Proceso a una secuencia de actividades interrelacionadas entre sí que tienen un inicio y un fin, cuyo producto crea un valor intrínseco para el usuario, utiliza recursos y se gestiona con el fin de que las entradas sean transformadas en salidas. Los procesos se clasifican en Procesos Claves y Procesos de Apoyo (DUEÑAS 2004)**.**

Los Procesos Claves tradicionalmente han sido objeto de medición y control, pues tienen una repercusión directa sobre la calidad del producto o servicio agregándole valor al mismo. Los Procesos de Apoyo aunque no influyen en la calidad directamente tienen un enorme potencial de mejora de la productividad y la exigencia, contribuyendo al desarrollo de los procesos.

Los elementos fundamentales que componen un proceso son: las entradas, recursos, salidas, control y límites.

Para identificar los procesos elementales del negocio, se debe llegar a un entendimiento común de los límites de la organización que se está describiendo y determinar los procesos que necesitarán ser descritos en más detalle. Este entendimiento debe efectuarse entre los clientes y los desarrolladores, pues los clientes son los responsables de ofrecer a los desarrolladores toda la información necesaria para que estos entiendan correctamente el funcionamiento de los procesos de la organización, logrando que el producto final cumpla con las expectativas de los clientes.

El objetivo de esta actividad es:

- Determinar la terminología.

- Identificar los objetivos de la organización que soportan la estrategia de negocio de la misma.
- Esbozar una idea general del modelo de casos de uso de la organización.
- Priorizar los casos de uso del negocio que deben ser descritos en más detalle.

Esta actividad comienza identificando los objetivos de la organización. Esto se hace paralelamente con el refinamiento de los Actores y Casos de Uso del Negocio. Dependiendo del alcance del (BAYONA 2007)esfuerzo en la modelación del negocio cuando se define la Visión del Negocio, los casos de uso y actores existentes en el negocio, podrían ser utilizados como punto de partida y refinados o reconsiderados completamente.

Los casos de uso del negocio son detallados lo suficiente como para comprender su impacto y priorizarlos. Los requisitos cualitativos o cuantitativos (como rendimiento de procesamiento o los estándares utilizados) que gobiernan el comportamiento de los casos de uso del negocio, deben ser documentados en las Especificaciones Suplementarias del Negocio.

Los términos comúnmente usados y las definiciones, deben ser reflejados en el Glosario del Negocio. Cualquier regla del negocio que sea descubierta en este proceso debe ser documentada en el documento Reglas del Negocio.

El rol que interviene en la identificación de los procesos elementales del negocio es el Analista de Procesos del Negocio.

Desarrollo de los PEN (PEN).

El objetivo de esta actividad es detallar los Procesos Elementales de Negocio incluyendo las personas involucradas y la información que manejan, para ayudar a la comprensión de los procesos de la organización por parte del equipo de desarrollo y a partir de esta descripción determinar los requerimientos del sistema.

Los Procesos Elementales de Negocio se describen a partir del Manual de Procesos, además los puntos débiles o ausentes en él son aclarados con el cliente obteniéndose la descripción detallada de los mismos.

Para esta descripción se utilizan de apoyo, cuando sea necesario, los diagramas de actividad (para representar flujos de trabajo), diagramas de transición de estados y diagramas de casos de uso del

negocio. Se genera además un glosario de términos para acordar un vocabulario común entre desarrolladores y clientes/usuarios y definir los principales términos usados en el proyecto.

Identificar las Salidas del Sistema.

Como se explicó anteriormente las salidas del sistema son elementos importantes a tener en cuenta a la hora de describir un proceso y estas a su vez son tomadas como punto de partida para obtener las salidas del sistema. Normalmente las salidas de un proceso van a formar parte de las entradas de los procesos subsecuentes. Frecuentemente las salidas de estos procesos pueden estar relacionadas tanto con los clientes internos como con los externos.

Cuando se analizan los procesos de la organización, a partir de ellos se obtienen los casos de uso y las salidas del sistema. Estas son el resultado que se obtiene de procesar las entradas. Al igual que las entradas, las salidas del sistema pueden adoptar la forma de productos, servicios e información. Las mismas son el resultado del funcionamiento del sistema o, alternativamente, el propósito para el cual existe el sistema. Las salidas de un sistema se convierten en entrada de otro, que la procesará para convertirla en otra salida, repitiéndose este ciclo indefinidamente.

Elaboración del Modelo Conceptual (o Dominio).

Para los casos de uso donde sea necesario, se incluye un prototipo de interfaz de usuario elemental con un mapa de navegación asociado y un modelo conceptual de datos, identificando las entidades y relaciones que forman parte del sistema de información objeto de análisis. La elaboración del modelo conceptual es opcional. Este se utiliza para un caso de uso que lo requiera, pues su funcionalidad no es evidente o no basta con una simple descripción narrativa.

El modelo conceptual representa el vocabulario del dominio: Ideas, conceptos, objetos. Un modelo conceptual permite establecer los requisitos funcionales de un sistema. La construcción de dicho modelo conceptual es un proceso de descubrimiento en el cual la interacción entre el cliente y el desarrollador debe ser estrecha. La construcción iterativa e incremental del modelo conceptual es la estrategia más adecuada. El proceso incluye básicamente cuatro actividades: captura de requisitos, modelado o especificación, verificación de criterios de calidad y consistencia, y finalmente validación. El grado de validación con el cliente es determinante para asegurar que el producto se ajuste a lo esperado (ÁNGEL ROCHE 2007).

Elaboración de Casos de Uso.

Los casos de uso no son algo aislado, sino que deben situarse en su contexto para poder comprenderlos con mayor exactitud. La elaboración de los casos de uso no es una actividad analítica, sino sintética, pues no se trata de analizar o desmenuzar algo que ya existe, sino de crear junto con los clientes, una concepción común del sistema a desarrollar, logrando que los casos de uso expresen el funcionamiento del sistema como un todo (no de sus partes).

En el momento de elaborar un caso de uso se aconseja una comunicación real entre los desarrolladores y los clientes, tratar de no complicar las cosas, tener siempre en cuenta el criterio de las partes interesadas (cliente). Una vez que se haya culminado su elaboración, se deben revisar cuidadosamente con el usuario, la interacción entre el actor y el software debe ser descrita sin ambigüedad, expresar tanto los requisitos funcionales como los no funcionales.

Los objetivos fundamentales que persigue la elaboración de casos de uso son:

- Definir el límite entre el sistema a desarrollar y los elementos externos a ese sistema (actores usuarios del sistema).
- Capturar el conjunto de funcionalidades y comportamiento del sistema a desarrollar.

Los casos de uso se deben elaborar en reuniones de captura de requisitos con los desarrolladores, clientes y usuarios finales, al principio de la iteración en formato extendido y esencial, se refinan a lo largo de las demás iteraciones.

2.3.2 Etapa Prototipo.

Prototipo de Interfaz de Usuario.

El prototipo de interfaz de usuario es un artefacto que se obtiene durante el levantamiento de requisitos. El objetivo esencial que se persigue con su construcción es ayudar a comprender y especificar las interacciones entre actores humanos y el sistema durante esta etapa (IVAR JACOBSON 1999), no solo apoyando el desarrollo de una mejor interfaz gráfica, sino también para una mejor comprensión de los casos de uso.

El rol que interviene en el diseño del prototipo de interfaz de usuario es el Diseñador de Interfaz de Usuario. Los principales artefactos de entrada para su elaboración son, el modelo de casos de uso, una lista de requisitos adicionales, la descripción de los casos de uso y un glosario de términos.

El prototipo va a incluir las pautas para el diseño de la interfaz y el mapa de navegación. Constituye un punto de partida para la implementación. Se trata de prototipos que permiten al usuario hacerse una idea más o menos precisa de las interfaces que proveerá el sistema y así, conseguir retroalimentación de su parte respecto a los requisitos del sistema. Estos prototipos se realizarán como: dibujos a mano en papel, dibujos con alguna herramienta gráfica o prototipos ejecutables interactivos, siguiendo ese orden de acuerdo al avance del proyecto. Sólo los de este último tipo serán entregados al final de la fase de Elaboración, los otros serán desechados. Así mismo, este artefacto, será desechado en la fase de Construcción en la medida que el resultado de las iteraciones vayan desarrollando el producto final.

Mapa de navegación.

El mapa de navegación expresa la estructura de los elementos de interfaz de usuario en el sistema, al mismo tiempo que los posibles caminos potenciales de navegación. Es útil aclarar que existe un mapa de navegación por sistema.

El propósito de este artefacto es mostrar las principales rutas de la interfaz de usuario a través del sistema, siendo los senderos principales a través de las pantallas del sistema. El mapa de navegación no contempla necesariamente todas las rutas posibles, aunque si una descripción de como el usuario se mueve a través de los elementos de interfaz de usuario para probar la características del sistema, siendo el mapa de navegación el que define cuáles son las rutas válidas. (RUP 2003)

El rol que interviene en la elaboración de este artefacto es el diseñador de interfaz de usuario y sus principales responsabilidades son:

- Identificar qué senderos de navegación deben existir entre los elementos de interfaz de usuario.
- Determinar si los senderos de navegación son demasiado largos y esto pueda afectar la usabilidad del sistema.
- Asegurar que el Mapa de Navegación permanezca compatible con la tabla de historia.

2.4 Análisis teórico del método de estimación.

Al analizar el método de estimación anteriormente descrito, se evidencia cierto desajuste en las actividades a realizar en las dos primeras etapas: Levantamiento de Requisitos y Prototipo.

En la **Tabla 2,** se muestran las etapas definidas en el método de estimación de la UCI. La primera etapa a la que se hace referencia es el levantamiento de requisitos, con las actividades:

- Planificación.
- Obtención del PEN.
- Desarrollo PEN.
- Identificar las Salidas del Sistema.
- Elaboración del Modelo Conceptual.
- Elaboración de CU.

El método no se refleja una estructura lógica y consecuente de las etapas necesarias para desarrollar un producto software. Lo primero que muestra, es una incorrecta ubicación de las actividades por etapas, pues no se define una etapa para la modelación del negocio, sino que las actividades de esta etapa, se encuentran implícitas dentro del levantamiento de requisitos y se debe analizar que algunas actividades de la modelación del negocio, como la obtención y desarrollo de los PEN, constituyen hitos a cumplir para luego proceder al levantamiento de requisitos.

Otro elemento que repercute negativamente en los resultados que se obtienen a partir del método, provocando retrasos que influyen negativamente en la planificación, es que no toma en cuenta la influencia de los riesgos que pueden presentarse durante el proceso de desarrollo. El método no prevé cómo vincular los riesgos una vez identificados, de manera que se pueda conocer el impacto que ocasionan sobre las estimaciones, por lo cual la planificación que se obtiene a partir de estos datos, no tendrá la capacidad de adaptarse ante cualquier cambio que surja en el entorno de desarrollo.

Se consideró una etapa para obtener el prototipo de interfaz de usuario que contando o no con el nivel de complejidad, es un artefacto que se obtiene durante el levantamiento de requisitos. Las actividades que se definieron en el método aplicado para esta etapa fueron:

- Prototipo de interfaz de usuario.
- Mapa de navegación.

Una consecuencia de mezclar las actividades en el método de estimación elaborado, es que no se ven claramente delimitadas las etapas en que los especialistas funcionales intervienen con mayor peso. La participación de los especialistas funcionales es mayor cuando se está modelando el negocio, pues intervienen activamente durante la definición de los procesos elementales. Para el levantamiento de requisitos la participación de estos, va siendo menos activa en comparación con los analistas que tienen mayor responsabilidad en la definición del sistema, aunque los especialistas son los que validan el resultado final de esta etapa. Esta situación provoca pérdidas en costo y recursos empleados.

A la hora de estimar para hacer una planificación eficiente a partir de los datos obtenidos, es necesario tener en cuenta todas las posibles actividades a ejecutar dentro del proceso de desarrollo y precisamente una de estas actividades es la validación, que debe ser aplicada en diferentes momentos en todas las etapas del proyecto. En el desarrollo de este proceso, luego de haber culminado una actividad y obtenido cierto producto, este debe ser validado tanto por los especialistas funcionales como por los clientes y demás interesados. Es en esta actividad donde se van a identificar las posibles inconformidades de los clientes, para a partir de ahí, trabajar en los reajustes que necesite el producto entregado, y pasar a realizar la siguiente actividad planificada. Este proceso debe repetirse cada vez que se obtenga un entregable para los clientes, el cual se utiliza como entrada para el desarrollo de la actividad siguiente. Precisamente la validación es una actividad de suma importancia que debe tenerse en cuenta a la hora de planificar, pues como se explicó en el objetivo de su realización, este es un proceso que puede tener varias iteraciones, teniendo implícito en su realización, el consumo de tiempo, costo y recursos.

En el método de estimación explicado anteriormente, la actividad de “validación” no se tiene en cuenta como una de las tareas estimadas en la planificación del proyecto. Para hacer entrega de la descripción oficial de los procesos elementales del negocio, estos tienen que ser validados, por los especialistas funcionales, los clientes y demás interesados, para después trabajar en los reajustes que necesite la información entregada. Es entonces que se puede proceder a la siguiente actividad de levantamiento de requisitos. Esta actividad no se planifica en el método de estimación, lo que puede provocar un retraso con respecto a la planificación realizada para un proyecto.

Por otra parte las revisiones internas se hacen con el objetivo de medir el avance y el rendimiento de las actividades que se deben llevar a cabo en el proyecto, así como para prevenir que los defectos continúen a través de etapas posteriores en el ciclo de vida, eliminándolos en una etapa temprana y de una forma eficiente. Estas pueden ser formales o informales de acuerdo al estilo de trabajo que se desarrolle en el proyecto y deben realizarse periódicamente por algún integrante del equipo de desarrollo, el cual valorará la calidad del trabajo realizado. El desarrollo de esta actividad no se considera en el método de estimación, el cual omite por tanto, los tiempos que se deberían emplear para la planificación de esta actividad, afectando otros atributos del proceso sobre los que influye.

Los retrabajos o arreglos, tampoco son tomados en cuenta para las estimaciones que se realizan a partir del método. Esta actividad se lleva a cabo una vez que los clientes y los desarrolladores consideran necesario realizar algún cambio en un producto obtenido, de manera que alcance la calidad requerida o cumpla con las expectativas del cliente. Evidentemente situaciones como estas pueden surgir en el transcurso del proceso de desarrollo de software, pero es fundamental aclarar la repercusión que tiene la acción de repetir un trabajo que sólo estaba pronosticado en un período determinado, cuando presenta conflictos en tiempo y esfuerzo al comprobarlo con la estimación prevista en el método que se plantea.

El esfuerzo empleado en la obtención de los PEN, tampoco es tenido en cuenta dentro del método, pues toma la cantidad de casos de uso identificados en el estudio preliminar como base para realizar sus cálculos y estos no tienen una relación directa con los PEN, lo cual significa que si identifican y describe un cantidad determinada de PEN, no se tiene que obtener precisamente a partir de ellos esa misma cantidad de casos de uso.

Ciertamente la gran mayoría de las instituciones cuando comienzan a planificar, estiman de inicio a fin el proyecto, considerando las mejores condiciones, o sea, el escenario del “mejor caso”. Estimar no resulta una tarea fácil, pero tener en cuenta las posibles alternativas, hace más elevado el nivel de certeza, ante un asombro posterior a lo previsto, donde las posibilidades de corrección ya sean poco probables por las estimaciones que se mantienen corriendo en tiempo, costo y recursos.

En el caso de la UCI, el método estimado refleja claramente que sólo se tuvo en cuenta el escenario del “mejor caso”, pues si se retoma lo anteriormente analizado al inicio de este capítulo, se evidencia que toda estimación estuvo concebida bajo condiciones ideales, donde el resultado de la mayoría de los factores

para realizar un buen método tomaron valor nominal, porque no se analizaron debidamente, ni se gestionaron los riesgos que todo proyecto siempre tendrá que combatir con un buen plan.

El análisis práctico a este método de estimación se podrá ver bien fundamentado en el siguiente capítulo mediante un ejemplo real: Proyecto SIGEP, donde se analizará con datos obtenidos a partir de la experiencia, el desajuste entre lo estimado y los resultados de la práctica.

Capítulo 3: Análisis del Método para el caso real: SIGEP.

Se enfocará el presente capítulo en analizar los resultados de la aplicación del método de estimación en la UCI tomando el caso real: proyecto SIGEP, luego de estudiar la situación de las actividades realizadas con los tiempos y el personal involucrado, durante el estudio preliminar, la modelación del negocio y el levantamiento de requisitos. Se comparan datos históricos de otros proyectos de exportación que aplicaron el método, se evalúa la precisión de la estimación del método y al final se proponen algunos elementos a ser considerados para perfeccionar el método de estimación.

Para la primera etapa del proyecto SIGEP, se realizó la modelación del negocio y captura de requisito para los módulos Control Penal, Clasificación y Tratamiento, Salud Integral, Sala Situacional y Custodia y seguridad; con el objetivo de proveer una solución que permitiera ejercer un control mínimo del nivel operativo del sistema penitenciario.

3.1 Descripción de las actividades realizadas por etapas.

Para el desarrollo del SIGEP se ha considerado RUP (Proceso Unificado de Desarrollo) como metodología de referencia (ARIAS 2006). Según lo estimado en el método aplicado en el proyecto SIGEP, se contemplaban las etapas de levantamiento de requisitos y prototipo. A continuación se presenta la situación de las etapas realizadas con sus objetivos, salidas y actividades relacionadas.

Los datos necesarios para la estimación se obtuvieron a partir de un estudio preliminar, cuyo objetivo era obtener una visión de la organización y de las necesidades de informatización.

Artefactos de Salida: Proyecto Técnico, Evaluación de áreas de la organización y Planificación.

Actividades:

- Definición de alcance del sistema: Se definió con el cliente y el equipo de desarrollo lo que el sistema debía hacer, así como la interfaz gráfica del usuario, centrada en las necesidades y metas de los usuarios y clientes.
- Evaluación de la organización: Mediante una lista de chequeo se verificaron los procesos del negocio que estaban listos para iniciar el levantamiento de requisitos. Se determinó que más del 50 % de ellos, no estaban listos para proceder con la siguiente etapa.

- Planificación: Con los datos obtenidos en el estudio preliminar se planificaron las actividades del levantamiento de requisitos y la definición del sistema.

3.1.1 Etapa Modelación del Negocio.

Objetivo: Comprender y representar el funcionamiento de la institución.

Artefacto de Salida: PEN, Glosario de términos, reglas de negocio.

Actividades:

- Identificación de puntos débiles y ausentes: Se identificaron los puntos débiles y ausentes, con el objetivo de aclararlos con el cliente, garantizando que la descripción de los Procesos Elementales del Negocio estuviera completa y correcta.
- Entrevistas y talleres: Esta actividad se inició con los talleres de Control Penal con la participación de los asesores, analistas de software cubanos y los clientes. La documentación que se entregó en esta etapa sirvió como guía para la correcta modelación del negocio.
- Identificación de los procesos claves: Se identificaron aquellos procesos que inciden de manera significativa en los objetivos estratégicos y son críticos para el éxito del negocio.
- Identificación de subprocesos: Se identificaron los subprocesos pertenecientes a cada proceso, detectándose algunos problemas de definición.
- Identificación de las tareas que se ejecutan en cada subproceso: Se identificaron y describieron cada una de las tareas que se relacionaban con los subprocesos.
- Organización de la secuencia en que ocurren las tareas (diagrama de actividad): Se organizó el orden y la prioridad en que debían ocurrir las tareas que pertenecían a cada proceso.
- Identificación de flujos alternos y casos críticos: Se identificaron otros posibles caminos a tomar en el desarrollo de los casos de uso y se determinó cuales serían más importantes para desarrollar inicialmente.
- Descripción de las actividades: Se describieron cada una de las actividades que se debían realizar en los procesos.

- Conformar documento de PEN: Con los PEN descritos correctamente, se procedió a confección del documento que contemplaría todos estos procesos descritos.
- Validación con especialistas funcionales: Los artefactos que se obtuvieron se validaron con los especialistas funcionales, detectándose las inconformidades de los clientes.
- Actualización y corrección del documento de PEN: Se actualizó el artefacto documento PEN a partir de las inconformidades del cliente, detectadas en la validación.

El proceso de modelado de negocio comenzó para la primera etapa desarrollo de SIGEP, con la realización de los talleres de Control Penal el 21 y 22 de noviembre, con la participación de los asesores y analistas de software cubanos y especialistas venezolanos.

La documentación entregada sirvió como guía durante el modelado de negocio, no obstante existían puntos débiles que fueron detectados durante el trabajo. De los procesos, se modificaron sus descripciones, se unificaron algunos, aparecieron nuevos procesos, se profundizó en la descripción de las operaciones de cada proceso, se identificaron flujos alternos, nuevas entradas y salidas de información, así como se elaboraron propuestas de formatos de documentos.

En las áreas de Control de Visitas, Observación y Clasificación y Salud Integral se contaba con una descripción general de estos subprocesos pero no existían definiciones acerca de cómo proceder en cada una de sus actividades. El trabajo fue de creación conjunta con el cliente hasta llegar a un consenso. En muchas ocasiones se utilizó la técnica de proponer y validar.

De forma general el trabajo no solo fue de modelar un negocio existente, sino que se trabajó en la propia definición de un modelo de negocio inexistente en la práctica y como resultado se obtuvo la descripción de un grupo de procesos con sus respectivos subprocesos. Esto representó un riesgo que trajo como consecuencia un incumplimiento con lo inicialmente estimado.

3.1.1.1 Estadísticas del trabajo realizado en la Etapa de Modelación del Negocio.

En la etapa de modelación del negocio, se contó con la participación de 5 analistas para las actividades efectuadas, se trabajaron 6 días a la semana, lo que sumaría 24 días al mes.

La **Tabla 8** muestra la cantidad, el tiempo en horas y el por ciento que representan algunos aspectos estadísticos de la descripción de los PEN, las entrevistas y reuniones de validación.

Aspectos	Cantidad	Tiempo (Horas)	Por ciento
Talleres de Control Penal (iniciales).	2	14	2
Entrevistas a especialistas funcionales (sin contar las reuniones de validación).	29	101	18
Talleres de validación.	9	21	4
Esfuerzo en descripción de procesos.		281	49
Esfuerzo para incluir modificaciones en los procesos.		151	27
Total	**40**	**568**	**100**
Total de horas promedio empleadas por analista.		414	
Promedio de horas diarias por analista.		7	

Tabla 8: Estadísticas del trabajo realizado en la Modelación del Negocio

Los datos obtenidos del esfuerzo en descripción de procesos, no consideraron el esfuerzo empleado en reflejar las modificaciones. Para el promedio de horas diarias por analista, no se contó el tiempo empleado en transportación, atrasos en el inicio de las reuniones, entrevistas o talleres.

3.1.2 Etapa Levantamiento de Requisitos

Objetivo: Definir un sistema que se adecue a las necesidades de la institución.

Artefacto de Salida: Especificación de requisitos del software, modelo lógico de datos

Actividades:

- Estructuración de alto nivel del sistema: En esta actividad se estructuró lo que sería el sistema, los subsistemas y los módulos del proyecto.
- Identificación de los objetivos de alto nivel de subsistemas y módulos: Se obtuvieron los objetivos esenciales de cada uno de los subsistemas y los módulos.
- Identificación de casos de uso de cada módulo: Se obtuvieron los casos de uso críticos, de cada uno de los módulos que se estructuraron.
- Breve descripción de la funcionalidad de cada caso de uso: Se describió las características esenciales de las funcionalidades de los casos de uso.
- Validación de las funcionalidades: Entre el cliente y el equipo de desarrollo se validaron las funcionalidades de los casos de uso, donde hubo una retroalimentación entre ambas partes.

- Definición de un modelo lógico de datos. Se definió la base de datos y los atributos de cada una de las tablas.
- Construcción de un prototipo navegable y no funcional. Se obtuvo un prototipo navegable y poco funcional que mostraba esencialmente las necesidades y exigencias de los clientes.
- Validación: Se realizaron talleres de validación entre el cliente y el equipo de desarrollo con la presencia de los especialistas funcionales, donde se obtuvieron las inconformidades o la aceptación de los clientes. Hubo una retroalimentación entre ambas partes.
- Descripción del prototipo: Se elaboró el documento donde se describieron las características esenciales de prototipo definido.

3.2 Análisis del caso real.

La estimación inicial para planificar el desarrollo de software en el proyecto SIGEP, se hizo sobre una base optimista. Teóricamente las condiciones estaban previstas y estimadas en tiempo, costo y esfuerzo, pero la experiencia demostró que sólo es la práctica, el criterio de la verdad.

Como en la elaboración del método de estimación no se previó el modo de vincular los riesgos, una vez identificados estos en el proyecto SIGEP durante el estudio preliminar, no hubo forma de prevenir el impacto de algunos, sobre las estimaciones ya previstas en el método.

La experiencia de los desarrolladores fue un factor de riesgo que influyó negativamente sobre las estimaciones, ya que no se tuvo en cuenta que la mayoría son estudiantes que aún no poseen suficiente experiencia en la producción de software.

Otro factor que influyó negativamente sobre las estimaciones fue la incorrecta definición de los procesos. El método suponía precondiciones ideales, con procesos bien definidos, pero esto no se dio en el caso del proyecto SIGEP cuando al evaluar la áreas, mediante una Lista de Chequeo **Anexo 2**, se demostró que más del 50% (HERRERA 2006) de los procesos no reunían las condiciones mínimas para proceder al levantamiento de requisitos.

Según el análisis se determinó que los procesos identificados y evaluados, fueran considerados medianamente satisfactorios al observar los grupos de procesos considerados por la institución y los subprocesos que incluían (HERRERA 2006).

Se definieron los procesos en: procesos claves y de apoyo. La clasificación de los procesos no estaba bien delimitada, pues los procesos de gestión, que son aquellos que ofrecen salidas que regulan o determinan lineamientos para otros procesos de la organización, no fueron identificados. Este tipo de proceso estuvo incluido erróneamente dentro de otro, provocando la pérdida de su razón de existencia dentro de la organización. Desde la hora en punto que se decide informatizar una institución, la clasificación apropiada de los procesos permite establecer relaciones adecuadas dentro del sistema informático.

La descripción de cada uno de los procesos evaluados como no satisfactorios, fue observada tanto en su estructura como en el contenido que presentaban (HERRERA 2006). La estructura que se definió para describir los procesos no aportaba información valiosa, dejándose de documentar en la descripción otros aspectos para comenzar el levantamiento de requisitos. Aunque actualmente se acepta dentro del enfoque basado en procesos, que la información entre estos pueda entrecruzarse en más de un área, no significaba que en la descripción de cada uno de los procesos en particular, coincidieran las actividades, fuesen similares o incluso existieran actividades propias de un subproceso dentro de otro subproceso. La información que presentaba el listado de normativa legal para cada proceso contenía información legalizada no válida en la actualidad.

Otra de las causas que provocó atrasos para proceder al levantamiento de requisitos estuvo dada por la ineficiente documentación para conocer qué flujo de información circulaba desde cada uno de los procesos y hacia dónde se dirigía. Se precisaba emplear esfuerzo, en el menor tiempo posible jugando con el estimado y reunir toda la información para continuar con las etapas restantes.

Todo esto provoca que el trabajo concebido para la primera etapa sobrepase las expectativas planteadas en el método de estimación con respecto al tamaño y tiempo estimado.

Las **Tabla 9** y **Tabla 10** muestran el tiempo estimado y real del proyecto SIGEP, con las etapas y sus principales actividades durante el proceso de desarrollo.

Levantamiento de Requisitos	
Actividades	**Tiempo (Semanas)**
Planificación	2
Obtención de los PEN	
Desarrollo de los PEN	
Identificar las Salidas del Sistema	
Elaboración del Modelo Conceptual	
Elaboración de CU	
Prototipo	
Prototipo de Interfaz de Usuario	2
Mapa de navegación	

Tabla 9: Tiempo Estimado

Estudio Preliminar	
Actividades	**Tiempo (Semanas)**
Identificación de alcance del sistema	12
Evaluación de áreas de la organización	
Estructuración de alto nivel del sistema	
Planificación	
Modelación del Negocio	
Identificación de los PEN	5 + 3
Descripción de los PEN	
Reglas del Negocio	
Validación y Arreglos	
Definición del Sistema	
Identificación de los Casos de Uso y Salidas del Sistema	12
Elaboración del Prototipo de Interfaz de Usuarios	
Mapa de Nevegación	
Modelo Conceptual	
Validación y Arreglos	

Tabla 10: Tiempo real

Figura 4: Tiempo estimado y real.

Los datos estimados muestran sólo 2 semanas para levantamiento de requisitos y prototipo respectivamente, sumando 4 semanas en total para desarrollar todas las actividades, mientras que los datos reales muestran para las etapas identificadas (Modelación del Negocio y Definición del Sistema) una variación considerable de los tiempos para un total de 17 semanas en desarrollar cada una de estas actividades.

La **Tabla 11** muestra la variación entre lo estimado y real del tamaño del software, dado en casos de uso en dependencia de la complejidad: Baja, Media y Alta.

	Complejidad de Casos de Uso			
	Baja	**Media**	**Alta**	**Total**
Estimado	59	67	19	145
Real	122	44	30	196

Tabla 11: Tamaño estimado y real.

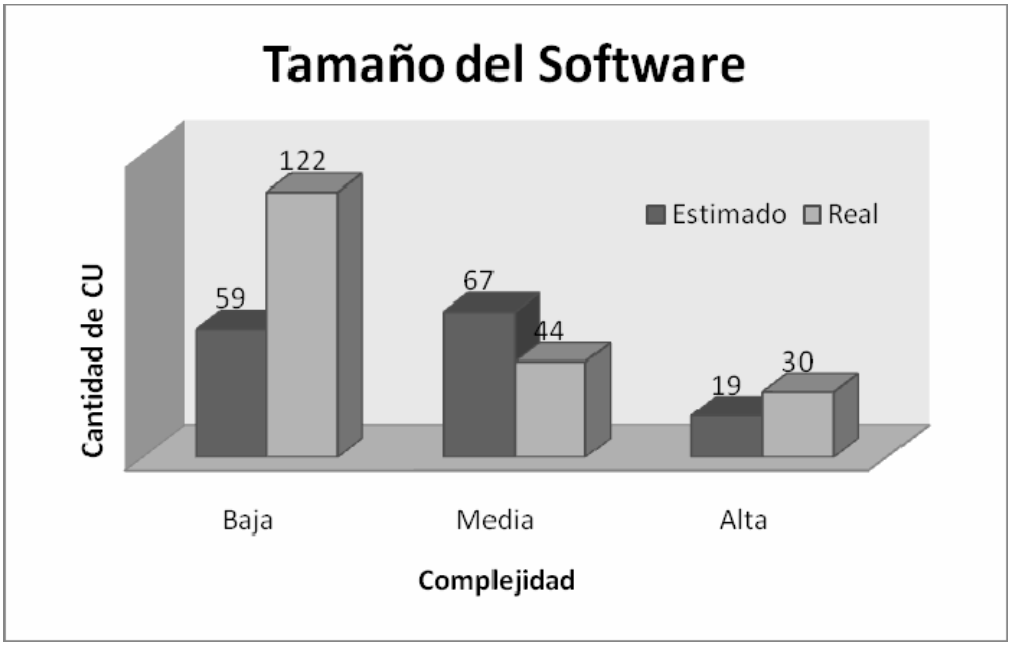

Figura 5: Tamaño estimado y real

Para los casos de uso de Complejidad Baja la diferencia fue de 63 casos de uso por debajo, para los casos de uso de Complejidad Media la diferencia fue de 23 casos de uso por encima y para los casos de uso de Complejidad Alta la diferencia fue de 11 casos de uso por debajo, con un total en la diferencia de 51 casos de uso, sobreasando las estimaciones iniciales para esta etapa de desarrollo.

La cifra de casos de uso puede variar durante el levantamiento de requisitos, pues aparecen nuevos, se unifican los existentes y otros dejan de existir. Sin embargo la deferencia total entre las funcionalidades estimadas no muestra mucha aproximación a las identificadas.

Cuando se elaboró el método tampoco se contó con la participación de especialistas de calidad, arquitecto para requisitos no funcionales y diseñador de bases de datos; roles necesarios, porque había que chequear las actividades realizadas, avalar las funcionalidades del sistema y definir la base de datos con las posibles entidades involucradas, lo que provocó también una variación de los recursos estimados.

La práctica demostró la necesidad de efectuar revisiones internas y talleres de validación para dejar claras las inconformidades o la aceptación de los clientes. Lógicamente esta tarea precisaba de tiempo y recursos que no se estimaron en el método tampoco.

3.3 Análisis del método en otros proyectos.

Mediante las tablas que aparecen a continuación se muestra un registro histórico de los datos obtenidos por proyectos que aplicaron el mismo método de estimación.

3.3.1 Tiempos estimados y reales en otros proyectos.

Para los proyectos SIGEP, CICPC y CTAISC se realiza un análisis de las variaciones de los tiempos estimados y reales, como se muestra en la **Tabla 12**. En el método de estimación las etapas definidas fueron: levantamiento de requisitos y prototipo. En la práctica se realizó una modelación del negocio y levantamiento de requisitos.

	SIGEP	CICPC	CTAISC
Etapas Estimadas			
Levantamiento de Requisitos	2	2	4
Prototipo	2	1	3
Total	**4**	**3**	**7**
Etapas Reales			
Modelación del Negocio	5+3(modificaciones)	6+2	7
Levantamiento de Requisitos	12	5	9
Total	**17**	**11**	**16**

Tabla 12: Tiempos estimados y reales de proyectos en la UCI

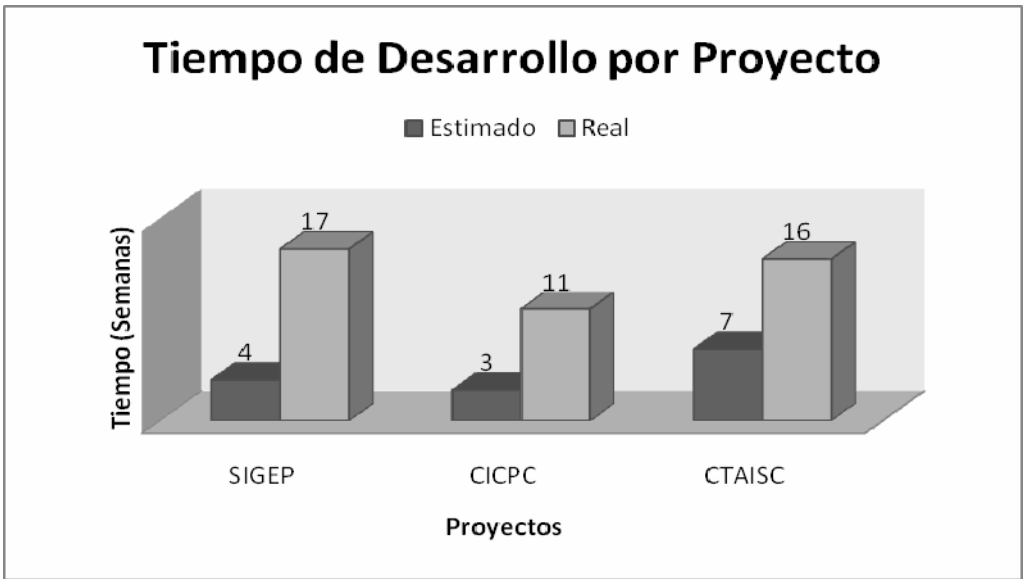

Figura 6: Comparación del tiempo de desarrollo entre proyectos

3.3.3 Evaluación de la precisión de la estimación.

Algunos expertos en el estudio de procesos de desarrollo de software (Conte, Dunsmore, De Marco y Shen) sugieren que un método es aceptable si la media del error relativo (MER) es menor o igual que el 25%, y si al menos el 75% de los valores predichos (*pred*) están dentro del 25% de sus correspondientes valores actuales. En términos de evaluación de la certeza de un método dado, se considera un buen método aquel que consiga que MER<=0,25 y *pred* (0,25)>=0,75 (CERRILLO 1999).

La siguiente métrica calcula el error relativo (ER) en la estimación de un proyecto, siendo Ve el valor estimado y Vr el valor real (CERRILLO 1999)

$$ER = |Vr - Ve| / Vr$$

Luego, para conocer la precisión de las estimaciones se necesita obtener la media del error relativo para un conjunto de estimadores de n proyectos (CERRILLO 1999):

$$\overline{ER} = 1/n \sum_{i=1}^{n} ER_i$$

La magnitud del error relativo será (CERRILLO 1999):

$$\overline{MER} = 1/n \sum_{i=1}^{n} ER_i$$

Se considera el MER = |ER|. La métrica *pred* proporciona una indicación del grado de ajuste para un conjunto de datos, basado en el valor de ER obtenido para cada dato.

Esta noción se usa para definir la métrica calidad de predicción. Para un conjunto de n proyectos, i es el número de ellos cuya magnitud media del error relativo es menor o igual que l (l<=0,25) y n es la cantidad de proyectos a evaluar.

$$pred(l) = i/n$$

Basado en este estudio se evalúa la precisión de la estimación del método para el tiempo estimado y real, en los proyectos: SIGEP, CICPC, CTAISC.

Evaluación.

SIGEP	**CICPC**	**CTAISC**
ER=|Vr- Ve | / Vr	ER=|Vr- Ve| / Vr	ER=|Vr- Ve| / Vr
ER= |17- 4| / 17	ER= |11 - 3| /11	ER= |16- 7| / 16
ER= 0, 76	ER= 0, 72	ER= 0, 56

$$\overline{MER} = 1/3 \sum_{i=1}^{3} MER_i$$

$$\overline{MER} = 1/3 \sum_{i=1}^{3} (0.76 + 0.72 + 0,56)$$

$$\overline{MER} = 0,68$$

pred(0, 25) = i/n

pred(0, 25) = 0/3

pred(0, 25) = 0

Los resultados obtenidos de la evaluación de la precisión de los tiempos para la muestra de los tres proyectos, demuestran que el método no es aceptable ya que MRE>0,25 y *pred* (0,25)<0,75.

Es conveniente aclarar que estas evaluaciones aún no muestran la certeza de que el método no sea totalmente aceptable, porque en realidad para afirmar la precisión de estos cálculos se necesita una población de proyectos más grande, aunque confirman la tendencia de estimaciones erróneas.

3.4 Elementos para perfeccionar el método.

Luego del análisis teórico y práctico del método de estimación aplicado en varios proyectos de gran envergadura en la UCI y a partir de los resultados obtenidos, se proponen elementos para perfeccionar el método, con el objetivo de mejorar las futuras estimaciones.

1. Contemplar en el método la forma de vincular los riesgos identificados en los proyectos. Para la identificación de los riesgos se podrá apoyar en una Lista de Comprobación de Riesgos.

2. Desglosar las etapas de levantamiento de requisitos y prototipo estimadas en el método en tres etapas lógicas con sus actividades y tiempos **Tabla 13**. Incluir las actividades no identificadas en el método utilizado como: Identificación de alcance del sistema, evaluación de áreas de la organización, estructuración de alto nivel del sistema, validaciones y arreglos. Para los tiempos, se propone promediar los tiempos obtenidos en los proyectos analizados, para dar un aproximado según las experiencias.

Etapa	Actividades	Tiempo estimado (semanas)
Estudio Preliminar	Identificación de alcance del sistema, Evaluación de áreas de la organización, estructuración de alto nivel del sistema, planificación	12
Modelado de Negocio	Identificación de PEN, Descripción de los PEN, reglas de negocio. Validación y arreglos	6
Definición de sistema	Identificación de casos de uso y salidas del sistema, elaboración de prototipo de interfaz, mapa de navegación y modelo conceptual. Validación y arreglos	9

Tabla 13: Tiempos propuestos

3. El equipo de desarrollo involucrado en esta etapa, debe estar conformado por:

- 6 analistas (aproximadamente).
- 1 Diseñador de Base de Datos.
- 1 Asegurador de Calidad.
- 1 Analista Principal.
- 1 Arquitecto.
- 1 Especialista en Procesos.

Estos datos propuestos, son válidos para proyectos con un tamaño similar, entre 150 a 200 Casos de uso.

Conclusiones

Una vez estudiado y analizado el método de estimación elaborado en la UCI, para los proyectos de exportación de gran envergadura y basado en los resultados obtenidos en la práctica del proyecto SIGEP, se arriba a las siguientes conclusiones:

- El método de estimación elaborado en la UCI tiene deficiencias en las estimaciones.
- El método, no se hizo partiendo de un análisis estadístico y solo tuvo en cuenta la opinión de especialistas con poca experiencia en el desarrollo de software de grandes proporciones.
- El método de estimación, no tomó en cuenta factores de riesgos para la Modelación del Negocio y Levantamiento de Requisitos, obteniéndose una estimación muy optimista, puesto que en la práctica los tiempos dedicados a esta actividad fueron muy superiores.

Recomendaciones

Se recomienda para próximas aplicaciones del método de estimación:

- Actualizar el método de estimación de modo que cuantifique los factores de riesgo, partiendo de un estudio preliminar de los proyectos a desarrollar.
- Mantener actualizado un registro estadístico que permita llevar el histórico de los esfuerzos empleados para desarrollar software, de modo que se cuente con una base real para futuras estimaciones.
- Determinar otros factores de riesgos externos e internos, relacionados con el entorno propio de la UCI.
- Comunicar estos resultados a la dirección de IP.

Referencias Bibliográficas

ÁNGEL ROCHE, P. L., ELENA NAVARRO Y MANUEL LLAVADOR. *"VALIDACIÓN DE MODELOS USANDO ESCENARIOS Y PROTOTIPADO AUTOMÁTICO"*, 2007.

ARIAS, O. Y. A. *"Proyecto Técnico de Asesoría Especializada, Colaboración Médica Odontológica, Comunicación Institucional y Solución Tecnológica para apoyar la modernización del Sistema Penitenciario de la República Bolivariana de Venezuela"*, 2006.

BASTERRA, R. V. D. *"Administrando la inseguridad "*, 2006.

BAYONA, S., CALVO MANZANO, J., CUEVAS, G., SAN FELIU. *"TEAM SOFTWARE PROCESS (TSP): MEJORAS EN LA ESTIMACIÓN, CALIDAD Y PRODUCTIVIDAD DE LOS EQUIPOS EN LA GESTIÓN DEL SOFTWARE"*, 2007.

CAPUCHINO, A. M. M. S. *"Control y gestión de proyectos software, Unidad 4: Estimación de Proyectos Software"*, 1996.

CERRILLO, D. *"ESTIMACIÓN DEL SOFTWARE"*, 1999.

DUEÑAS, L. M. *"Caracterización de un Sistema de Gestión de Información Científico Tecnológica con enfoque a procesos: garantía para la mejora continua. Estudio de caso"*, 2004.

ESCORIAL, J. S. *"Calidad de Software: Medidas del Proceso"*, 2006.

HERNÁNDEZ, S. E. B. *"Métricas de estimación de tamaño: Puntos de Caso de Uso"*, 2002.

HERRERA, S. M. *"Instrumento para la identificación y evaluación de las áreas de la organización"*, 2006.

HURTADO, L. L. *"La primera comunidad libre donde aprender y compartir"*, 2007.

IVAR JACOBSON, G. B. Y. J. R. *"El proceso Unificado de Desarrollo de Software"*. 1999. p.

KAN., S. H. *"Metrics and Models in Software Quality Engineering"*. 2000. p.

KELVIN, L. *"Capacitación de Vanguardia para Desarrolladores"* 2000.

MARCELO, J. *"Técnicas de estimación y seguimiento"*. , 2006.

MARÍA, M. S. C. A. *"Estimación de Proyectos Software"*, 1998. p.

MENÉNDEZ, R. *"Gestión de riesgos en ingeniería del software"*, 2004.

OVEJERO, J. D. *"Estimación de Proyectos para sistemas basados en Conocimiento."* 2006.

PRESSMAN, R. S. *"Ingeniería del Software. Un Enfoque Práctico"*. 1998. p.

RUP. *"Rational Unified Process"*. 2003.

VARAS, M. P. *"Modelo de Gestión de Proyectos Software: Estimación del Esfuerzo de Desarrollo"*, 1995.

WIKIPEDIA1. *"The Free Enciclopedia"*, 2006.

ZUSE, H. *"History of Software Measurement"*. 1995.

Bibliografía Consultada

ARIAS, O. Y. A. *"Proyecto Técnico de Asesoría Especializada, Colaboración Médica Odontológica, Comunicación Institucional y Solución Tecnológica para apoyar la modernización del Sistema Penitenciario de la República Bolivariana de Venezuela"*, 2006.

BAYONA, S., CALVO MANZANO, J., CUEVAS, G., SAN FELIU. *"TEAM SOFTWARE PROCESS (TSP): MEJORAS EN LA ESTIMACIÓN, CALIDAD Y PRODUCTIVIDAD DE LOS EQUIPOS EN LA GESTIÓN DEL SOFTWARE"*, 2007.

C, M. P. V. *"Modelo de Gestión de Proyectos Software: Estimación del Esfuerzo de Desarrollo"*.

CAPUCHINO, A. M. M. S. *"Control y gestión de proyectos software, Unidad 4: Estimación de Proyectos Software"*, 1996.

CERRILLO, D. *"ESTIMACIÓN DEL SOFTWARE"*, 1999.

HERRERA, S. M. *"Instrumento para la identificación y evaluación de las áreas de la organización"*, 2006.

KAN., S. H. *"Metrics and Models in Software Quality Engineering"*. 2000. p.

LABDELAOUI, H. *"Métodos de Estimación de Tamaño Funcional Software Aplicación a Enfoques de desarrollo"*

LAVINTMAN, A. *"Introduccion al analisis y gestion del riesgo"*.

MANIASI, S. D. *"Identificación de riesgos de proyectos de software en base a taxonomías"*.

MARÍA, M. S. C. A. *"Estimación de Proyectos Software"*, 1998. p.

MENÉNDEZ, R. *"Gestión de riesgos en ingeniería del software"*, 2004.

MIGUEL GARRE RUBIO, M. C. C. *"ESTIMACIÓN DEL ESFUERZO DE UN PROYECTO SOFTWARE UTILIZANDO EL CRITERIO MDL-EM Y COMPONENTES NORMALES N-DIMENSIONALES. APLICACIÓN A UN CASO PRÁCTICO"*

OVEJERO, J. D. *"Estimación de Proyectos para sistemas basados en Conocimiento."* 2006.

PRESSMAN, R. S. *"Ingeniería del Software. Un Enfoque Práctico".* 1998. p.

RUP. *"Rational Unified Process".* 2003.

VERGAS, E. *"PLAN DE MÉTRICAS EN OCHO PASOS"*.

WIKIPEDIA1. *"The Free Enciclopedia"*, 2006.

Anexo 1: Necesidad de la medida

Existe una clara tendencia hacia la obtención de medidas rigurosas en el software.

"*No podemos controlar aquello que no se puede medir*". De Marco

"*La producción de Sw está fuera de control porque no se puede medir*". Fenton

Consideraremos pues, a la medida, como un elemento de mejora metodológica en la producción de software de calidad.

"Proceso que permite asociar números o símbolos a atributos de entidades de un dominio del mundo real, de forma que los describe de acuerdo a reglas claramente especificadas" Fenton y Pfleeger.

Problema	**Medir Ayuda**
Incorrecciones	Proporcionar requerimientos verificables, expresados en términos medibles.
Toma de Decisiones	Proporcionar evidencia cuantificable para apoyar las decisiones.
Falta de Control	Hacer más visible el desarrollo e identificar problemas anticipadamente.
Exceso de Gasto	Producir predicciones de coste y plazo justificables.
Costes de Mantenimiento	Recomendar determinadas estrategias de prueba e identificar los módulos problemáticos.
Evaluación de nuevos Métodos	Valorar los efectos en la productividad y calidad.

Anexo 2: Clasificación de las medidas

		Atributos	
Entidad	**Definición**	**Directos**	**Indirectos**
Proceso	Actividades relacionadas con el software y su desarrollo que normalmente poseen el parámetro "tiempo" como factor. Por ello serían actividades que se realizan durante una parte del tiempo total en cualquier proyecto.	Tiempo (duración del proceso), esfuerzo, incidencias ocurridas durante cierto proceso definido.	Calidad, coste, estabilidad, controlabilidad.
Producto	Es cualquier herramienta, servicio o resultado derivado de los procesos ejecutados.	Para documentación: longitud, modularidad, redundancia. Para código o diseño formal: estructuración, acoplamiento.	Mantenimiento, movilidad (portabilidad), reutilización.
Recurso	Cualquier entidad considerada como entrada para la producción de software. Pueden ser de muy diversa naturaleza. Herramientas, métodos, materiales, son ejemplos de recursos.	Edad del personal, salario, velocidad del equipo hardware.	Productividad, experiencia, fiabilidad, capacidad de reutilización.

Anexo 3: Impacto según las categorías de los componentes.

El impacto de cada controlador de riesgo en el componente de riesgo se divide en cuatro categorías de impacto: despreciable, marginal, crítico y catastrófico. La figura indica las consecuencias potenciales de errores (filas etiquetadas con 1) o la imposibilidad de conseguir el producto deseado (filas etiquetadas con 2) La categoría de impacto es elegida basándose en la caracterización que mejor encaja con la descripción de la tabla.

Componentes / Categorías		Rendimiento	Soporte	Costo	Planificación Temporal
Catastrófico	1	Dejar de cumplir los requerimientos provocaría fracaso de la misión.		Malos resultados en un aumento de costos y retrasos de la planificación temporal.	
	2	Degradación significativa para no poder alcanzar el rendimiento técnico.	El software no responde o no admite soporte.	Recortes financieros significativos, presupuestos excedidos.	Fecha de entrega inalcanzable.
Crítico	1	Dejar de cumplir los requerimientos degradaría el rendimiento del sistema hasta un punto donde el éxito de la misión es cuestionable.		Malos resultados en retrasos operativos y/o aumento de costo.	
	2	Alguna disminución en el rendimiento técnico.	Pequeños retrasos en modificaciones de software.	Algunos recortes de los recursos financieros, posibles excesos del presupuesto.	Posibles retrasos de la fecha de entrega.
Marginal	1	Dejar de cumplir los requerimientos provocaría la degradación de la misión secundaria.		Los costos, impactos y/o retrasos recuperables de la planificación temporal.	
	2	De mínima a pequeña reducción en el rendimiento técnico.	EL soporte del software no responde.	Recursos financieros suficientes.	Planificación temporal realista, alcanzable.
Despreciable	1	Dejar de cumplir los requerimientos provocaría inconvenientes o impactos no operativos.		Los errores provocan impactos mínimos en el costo y/o planificación temporal.	
	2	No hay reducción en el rendimiento técnico.	Software fácil de dar soporte.	Posible superávit de presupuesto.	Fecha de entrega fácilmente

Anexo 4: Resultados de la Identificación y Evaluación de las Áreas

Mediante la Lista de Chequeo, se valora entre otros elementos, el estado en que se encontraban los procesos, durante la identificación y evaluación de las Áreas, para proceder al levantamiento de requisitos.

No	Elementos	No existe	¿Si existe ?			Recomendaciones
			Satisfactorio	Medianamente satisfactorio	No satisfactorio	
1	Organigrama.			X		
2	Plantilla de Trabajadores.		X			
3	Funciones por unidades organizativas.			X		
4	Identificación de los procesos.			X		
5	Clasificación de los procesos en ¨procesos claves, procesos de gestión y procesos de apoyo¨.			X		
6	Mapa de relaciones externas de los procesos con las instituciones jurídicas, de seguridad y organismos internacionales.				X	
7	Mapa de relaciones internas entre los procesos.				X	
8	Identificación de los productos (servicios) que van a satisfacer las necesidades de los usuarios.				X	
9	Manual de Calidad que cumpla con el grupo de normas ISO 9000 y contemple la fichas de los procesos e instrucciones en la Gestión de la Calidad.	X				

No	Elementos	No existe	¿Si existe ?			Recomendaciones
			Satisfactorio	Medianamente satisfactorio	No satisfactorio	
10	Fichas e instrucciones de los procesos restantes.				X	
11	Sistema de Gestión de los Recursos Humanos por Competencias.			X		
12	Reglamento Interno General de la Institución.	-	-	-	-	
13	Manual de perfiles de cargo de los puestos de trabajo con sus competencias.			X		
14	Sistema de Información de cada proceso	X				
15	Participación de los especialistas en la definición de sus procesos.				X	

Glosario de Términos

Ámbito del proyecto: Se definen los objetivos del proyecto, identifica funciones primordiales que debe llevar a cabo el software e intenta limitar esas funciones de manera cuantitativa.

Analogía: Semejanza existente entre las cosas que se comparan.

Atributo: Cualidad de ser.

Calidad de software: Concordancia con los requisitos funcionales y de rendimiento explícitamente establecidos, con los estándares de desarrollo explícitamente documentados, y con las características implícitas que se espera de todo software desarrollado profesionalmente.

Componente: Que forma parte de alguna cosa o de su composición.

Cuadro de Historia: Es una descripción lógica y conceptual de la funcionalidad del sistema para un escenario específico, incluyendo la interacción requerida entre los usuarios del sistema y el sistema.

Diagramas de actividad: Muestra el flujo de actividades dentro de un sistema.

Diagramas de casos de uso del negocio: Muestra un conjunto de casos de uso y actores, con sus relaciones.

Diagramas de transición de estados: Consta de estados, transiciones, eventos y actividades.

Elementos de interfaz de usuario: Comprende las pantallas, formulario, páginas web, etc. Con las que el usuario interactúa. Representan la interfaz de usuario del sistema.

Estándar: Que posee el tamaño, la forma o cualquier otra característica que sigue al modelo. Se aplica a lo que se produce en serie. Que sigue una tendencia muy extendida. Aquello que se considera modelo.

Experto: Persona que aporta conocimientos o experiencia específica con respecto a una organización, proceso, actividad o materia que se vaya a auditar.

Fiabilidad: Probabilidad de que algo funcione bien o sea seguro.

Gestión: Actividades coordinadas para dirigir y controlar una organización.

Hito: Suceso o acontecimiento que sirve como punto de referencia.

Indicadores de proyecto: Permiten al gestor de proyecto evaluar el estado del proyecto en curso, seguir la pista de los riesgos potenciales, detectar las áreas de problemas antes de que se conviertan en críticas, evaluar la habilidad del equipo de proyecto en encontrar la calidad de los productos de trabajo de software.

Método heurístico: Método o procedimiento mediante el cual se puede deducir o inducir la verdad.

Modelo empírico: Modelo de regresión que relaciona esfuerzo con tamaño o funcionalidad.

Modelo: Cosa que ha de servir de objeto de imitación. Objeto, construcción u otra cosa con un diseño del que se reproduce más iguales. Esquema teórico de un sistema o de una realidad compleja que se elabora para facilitar su comprensión y estudio.

Normas: Modelo, patrón o regla de obligado cumplimiento

Planificación: Es la actividad fundamental del gestor de proyecto que comprende la formulación de lo que hay que realizar para obtener una finalidad que será precisamente la del sistema que estamos planificando (Decidir + Hacer).Control: Es inspección, fiscalización, intervención. Esta actividad no se centra solamente en realizar planes, sino controlar su ejecución y puesta en práctica.

Proceso: Cualquier actividad, o conjunto de actividades, que utiliza recursos para transformar entradas en salidas.

Procesos Claves: Son aquellos procesos que inciden de manera significativa en los objetivos estratégicos y son críticos para el éxito del negocio.

Procesos de Apoyo: Aquellos que no siendo fundamental para la satisfacción de las necesidades de los elementos externos, podría ser necesario para viabilizarla.

Procesos de Gestión: Aquellos que ofrecen salidas que regulan o determinan lineamientos para otros procesos de la organización.

Productividad: Relación entre la cantidad de bienes y servicios producidos y la cantidad de recursos utilizados

Proyecto: Proceso único consistente en un conjunto de actividades coordinadas y controladas con fechas de inicio y de finalización, llevadas a cabo para lograr un objetivo conforme con requisitos específicos.

Puntos de caso de uso no ajustados: Representa una suma ponderada del número de actores y el número de casos de uso de la especificación.

Puntos de función sin ajustar: La sumatoria, del número de puntos por función basándose en el tipo de componente y su complejidad, que se le asigna a los componentes del sistema informático en términos de transacciones

Puntos de función: Miden la aplicación desde una perspectiva del usuario, dejando de lado los detalles de codificación. Se define como una función comercial del usuario final.

Recurso: Procedimiento o medio del que se dispone para satisfacer una necesidad, llevar a cabo una tarea o conseguir algo. Pueden ser personas o recursos materiales.

Requisito: Condición necesaria para que algo se cumpla.

Retrabajo: Definición utilizado por la parte venezolana cuando se refieren a volver a trabajar sobre un elemento determinado.

Revisión: Actividad emprendida para asegurar la idoneidad, la adecuación y eficacia de la materia objeto de la revisión, para alcanzar unos objetivos establecidos.

Riesgo (software): Proximidad a un daño o peligro.

Rol: Define el comportamiento y responsabilidad de un individuo o grupo de individuos..

Senderos de Navegación: Los senderos entre los elementos de interfaz de usuario, donde el usuario puede navegar mientras interactúa con la interfaz de usuario del sistema.

Sistema: Conjunto de elementos mutuamente relacionados o que actúan entre sí.

Transacción: Es una secuencia de actividades atómica, es decir, se efectúa la secuencia de actividades completamente, o no se efectúa ninguna de las actividades de la secuencia.

Transformación organizacional: Proceso a través del cual se dinamiza el poder individual y colectivo de la organización para generar y mantener la renovación, innovación y aprendizaje a tono con las características y las situaciones que se requieran y presenten como resultado o impacto de las relaciones internas y externas. El personal es el encargado de transformar sus propios roles. Este proceso involucra el estudio y contextualización de las perspectivas psicológicas y socio-políticas que son propias de la organización y las que determinan el entorno global.

Usabilidad: Es la característica de un sistema que pretende ser utilizado por el tipo o tipos específicos de usuario/s, la tarea o tareas que para las cuales el sistema se ha hecho y el contexto en el que se da la interacción. El "grado de usabilidad" de un sistema es, por su parte, una medida empírica y relativa de la usabilidad del mismo.

Validación: Confirmación mediante el suministro de evidencia objetiva de que se han cumplido los requisitos para una utilización o aplicación especifica prevista.

Variable: Magnitud que puede tener un valor cualquiera de los comprendidos en un conjunto.