Universidad de las Ciencias Informáticas

Facultad 10

Título: Propuesta de técnicas de estimación y métricas para la metodología ágil SXP

Trabajo de Diploma para optar por el Título de Ingeniero en Ciencias Informáticas

Autores: Ailema Sisinta Ayra
Adolis Daniel Rodriguez Reyes

Tutor: Ing. Yaumara Arce Ajo

Ciudad de la Habana, 17 de junio de 2009

“Codifica siempre como si la persona que finalmente mantendrá tu código fuera un psicópata violento que sabe dónde vives”

Martin Golding

Declaración de autoria

Declaramos ser autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

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

Ailema Sisinta Ayra	Adolis Daniel Rodriguez Reyes
Autor	Autor

Ing. Yaumara Arce Ajo

Tutor

Agradecimientos

Ailema

En primer lugar quiero agradecerle a mis padres y a mi hermanito, sin ellos nada de esto habría sido posible.

A mi familia, cuñada y amigos, por su apoyo incondicional, mi más sincera gratitud.

A mi bello compañero de tesis, muchas gracias.

A la Revolución por su confianza.

A nuestro Comandante en Jefe cuyo ejemplo siempre será mi guía.

A la Universidad de las Ciencias Informáticas por brindarme la posibilidad de formarme en ella.

A todos los implicados en el desarrollo de este trabajo de diploma, a nuestra tutora por el tiempo dedicado.

A mis amigos que durante estos años han sido mi familia y que siempre han estado presentes para mi.

A los profesores que ayudaron a nuestra formación personal.

A todas las personas que de alguna manera u otra me ayudaron a ser mejor persona cada día, a todos muchas gracias.

Adolis

A la mujer más importante de mi vida, esa, que ha sido guía, ejemplo y meta, mi mamá.

A mi hermano, que me hace sentir orgulloso de tenerlo a mi lado en todo momento

A mi papá,

A ella, que ha sabido lidiar con mis fallos, que siempre ha estado presente, que me ha hecho contar con todo el apoyo del mundo, que me conoce como nadie y es dueña de mi cariño a Maria Victoria, gracias.

A todos aquellos que me hacen dichoso contando con ellos como amigos, a los que me sacaron de problemas y me ayudaron a disfrutar los buenos y malos momentos, a esos que nunca dudaron, Jose, Leo, Alejandro, Eliexy, Ricardo, Obel, Yordanis.

A nuestro grupo en general, Ana, Osvanys, a Ailema, mi incansable compañera de tesis, a todos en general, a los que confiaron sin reservas....

Muchas gracias

Dedicatoria

Ailema:

A mi papito y a mi mamita.

A mi hermanito del alma

A mi abuelos, a mis tios, mis primos, a mi familia en general

Adolis:

Dedicado a mi mamá, mi hermano y mi papá, que hicieron posible la realización de este sueño.

A todos aquellos que sin dudar me apoyaron en todo momento, a los que no lo hicieron y me hicieron aprender de igual manera, a todos, va dedicado este final.

Resumen

En la actualidad las tecnologías han alcanzado un gran desarrollo trayendo consigo que los proyectos de software tiendan a ser cada vez más exquisitos. En consecuencia ha sido necesario aplicar una Gestión de proyectos más detallada en cada una de sus áreas, siendo la planicación una de las más importantes, por ser anfitriona principal de las actividades de estimación y medición, estas actividades, aunque son generalmente conocidas por la mayoría de los equipos de desarrollo de la Universidad de las Ciencias Informáticas (UCI), no es un área que sea considerada de suma importancia. El objetivo fundamental de este trabajo es realizar una propuesta de técnicas de estimación y métricas para la metodología ágil SXP, que se integre al proceso de desarrollo de software garantizando la gestión efectiva de proyectos empezando por la planificación y manteniéndose desde el inicio y hasta el fin de cada una de las etapas de la metodología. En la investigación, se realiza una sistematización de las técnicas de estimación, métricas y herramientas automáticas utilizadas en el proceso de producción de software, basada en la cual se procede a formular una propuesta de estas. Finalmente con la propuesta de dichas técnicas y herramientas, se espera que se puedan aplicar como complemento a la metodología en todos los proyectos productivos de la facultad, y mejorar así la esfera de producción de la misma.

Índice general

Introducción

Un proyecto es un instrumento utilizado para recopilar, crear y analizar datos de cualquier índole, de una manera sistemática y organizada con el fin de obtener resultados satisfactorios, es de suma importancia pues permite estructurar de manera general el entorno de trabajo. Particularmente, un proyecto de software es el proceso de gestión llevado a cabo para la creación de un sistema o software [1]. Consta de manera general con tres fases importantes, la fase de planificación, la fase de ejecución y la de entrega o puesta en marcha. La planificación se resume en el hecho de armonizar dos tipos de elementos: los objetivos, que pueden ser exigentes, múltiples, difíciles, y los recursos que pueden ser limitados, costosos, rígidos.

Como primera fase en el proceso de desarrollo de software, la planificación constituye una herramienta utilizada para la gestión y la toma de decisiones, de esta manera, no es recomendable imaginar una planificación desde el primer momento totalmente acertada, pues con el transcurso del proyecto se hará evidente la necesidad de someter el desarrollo del producto a toda una serie de ajustes como modificar tareas, reasignar recursos, entre otros, materializándose la importancia de esta fase en: ayudar a fijar prioridades, permitir enfoques correctos dirigidos a el fortalecimiento del equipo de desarrollo y ayudar al tratamiento de los problemas de cambio [2].De esta manera se puede hablar de una etapa de planificación siempre que se intente prever un comportamiento futuro a través de el seguimiento de medidas orientadas al mejoramiento del producto de la manera más apropiada, para esto, se hace de obligatorio cumplimiento la realización de diferentes actividades dentro de las cuales se encuentran: la elección de técnicas de estimación y el uso de las métricas, siendo estas el tema principal de la investigación.

Dentro de todo proyecto, como antes se mencionó, sobresalen actividades de marcada responsabilidad en sus inicios, la estimación, como una de estas actividades, es una mirada al futuro, que hace aceptar con alguna resignación cierto grado de incertidumbre en cuanto al desarrollo esperado. Aunque

la estimación es más un arte que una ciencia, es una actividad importante que no debe llevarse a cabo de forma descuidada puesto que es la base de todas las actividades de planificación del proyecto y sirve como guía para una buena Ingeniería de sistema o software.

De suma importancia también, son las mediciones a realizar de distintas variables, para esto, se utilizan las métricas, estas son medidas efectuadas sobre algunos aspectos del sistema en desarrollo que permiten obtener conclusiones sobre ellos, con el fin de tomar las decisiones necesarias, evaluar el rendimiento de una forma más práctica, y de manera permanente, cuantificar y medir las operaciones de seguridad, transformar las políticas en acciones y medir los resultados.

La UCI, es protagonista de la creciente revolución del desarrollo del software en nuestro país a partir del año 2003 y alberga en su seno la creación de numerosos proyectos pertenecientes a diferentes ramas, de ahí que se encuentre compuesta por facultades que responden a producciones de cierta manera independientes en materia de perfiles, de esta manera la Facultad 10 tributa con sus actividades al desarrollo de Software Libre, dirigiendo la gestación de varios proyectos entre los cuales se encuentra el proyecto Unicornios. Este proyecto, utiliza, para documentar el ciclo de vida de sus aplicaciones una metodología ágil denominada SXP que fue creada en años anteriores como un complemento entre las mejores funcionalidades de las metodologías Scrum y XP, esta ha sido aplicada a diferentes subsistemas y servicios ayudando esto a su mejoramiento y optimización, aunque, aún no cuenta con funcionalidades que evidencien un buen desarrollo, por esta razón, se decide añadirle de manera complementaria las herramientas necesarias que permitan realizar estimaciones de costo y esfuerzo y medición a través de métricas, de esta forma se desea desarrollar una propuesta de técnicas de estimación y métricas adaptables a los procesos de SXP.

Para darle solución a la situación problemática descrita anteriormente se ha decidido proponer el siguiente problema científico: ¿Cómo optimizar las actividades de estimación y métricas dentro de la metodología SXP?

Preguntas Científicas:

- ¿Una técnica definida de estimación puede garantizar el éxito del empleo de la metodología SXP?

- ¿La definición de un conjunto de métricas adecuadas a la metodología SXP podrá garantizar una medición eficaz para aquellos proyectos que la empleen?
- ¿Qué herramientas se pudieran utilizar para el desarrollo de las actividades de estimación y métricas con el objetivo de agilizar estos procesos?

Siendo el objeto de estudio de la investigación el proceso de producción de software y el campo de acción de la misma, el área de estimaciones y métricas en la metodología SXP.

Para la introducción de dichas técnicas en la metodología SXP se trazó el siguiente objetivo general: Realizar una propuesta de técnicas de estimación y métricas para la metodología ágil SXP.

Como objetivos específicos se definieron:

- Aportar una sistematización bibliográfica de las técnicas de estimación y métricas en metodologías ágiles y tradicionales.
- Proponer una selección de las herramientas automáticas factibles para llevar a cabo los procesos de estimación y medición en los proyectos productivos de la Facultad 10.
- Evidenciar la necesidad de la utilización del modelo de calidad CMMI, en conjunto con la metodología SXP.
- Elaborar una propuesta de técnicas de estimación y métricas para la metodología ágil SXP.

De esta manera y para la obtención de una propuesta de técnicas de estimación y métricas para la metodología ágil SXP se ha dispuesto cumplir con los objetivos trazados anteriormente por lo que se hace necesario desarrollar las siguientes tareas:

- Sistematizar los antecedentes y el estado actual de las técnicas de estimación y métricas en metodologías ágiles y tradicionales.
- Seleccionar las herramientas automáticas factibles para llevar a cabo los procesos de estimación y métricas en los proyectos productivos de la Facultad 10.
- Realizar un estudio del modelo de calidad CMMI, particularmente en las actividades de estimación y utilización de métricas.

- Fundamentar el conjunto de técnicas de estimación y métricas a proponer como resultado de la investigación.

En la investigación se evidencia el uso de dos métodos teóricos, el Analítico-Sintético, cuando se analizan todos los documentos y teorías concernientes al desarrollo de la etapa de planificación de un proyecto, con el objetivo de arribar a conclusiones que contribuirán a la propuesta final, y de la misma manera el método de Análisis Histórico-Lógico, pues se estudia de forma íntegra la evolución y el desarrollo de las actividades: estimación de costos y esfuerzo y métricas, dentro de esta etapa. La entrevista, como método empírico, se emplea, a manera de herramienta para profundizar la investigación a través de las opiniones de expertos en el tema obteniendo la mayor cantidad de información posible.

La investigación está desarrollada en 2 capítulos, cuya estructura se describe a continuación:

En el capítulo 1 se realiza el estado del arte de la investigación donde se plasman los conceptos necesarios sobre la metodología SXP, el modelo de calidad CMMI, estimación y métricas, así como de las herramientas encargadas de llevar a cabo estos procesos, se reflejan sus aspectos fundamentales en las metodologías ágiles y en las metodologías tradicionales de desarrollo de software.

En el capítulo 2, se realiza la propuesta de técnicas de estimación y métricas, y una recomendación sobre las herramientas automáticas más factibles que permiten medir dichos procesos para así ser utilizadas en los proyectos productivos de la facultad.

Capítulo 1

Fundamentación Teórica

1.1. Introducción

La gestión u organización de los proyectos de software no ha sido nunca una tarea fácil, se caracterizan por ser procesos extremadamente rigurosos, que desde hace algún tiempo son soportados por las metodologías de desarrollo. Las metodologías para el desarrollo de software son un proceso disciplinado sobre el desarrollo de software con el fin de hacerlo más predecible y eficiente. Lo hacen desarrollando un proceso detallado con un fuerte énfasis en planificar inspirado por otras disciplinas de la ingeniería.

Estas metodologías ingenieriles han estado presentes durante mucho tiempo. No se han distinguido precisamente por ser muy exitosas. Aún menos por su popularidad. La crítica más frecuente a estas metodologías es que son burocráticas. Hay tanto que hacer para seguir la metodología que el ritmo entero del desarrollo se retarda.

Hoy en día existen numerosas propuestas metodológicas que inciden en distintas dimensiones del proceso de desarrollo. Un ejemplo de ellas son las propuestas tradicionales centradas específicamente en el control del proceso Estas han demostrado ser efectivas y necesarias en un gran número de proyectos sobre todo aquellos proyectos de gran tamaño (respecto a tiempo y recursos). Sin embargo, la experiencia ha demostrado que las metodologías tradicionales no ofrecen una buena solución para proyectos donde el entorno es volátil y donde los requisitos no se conocen con exactitud, porque no están pensadas para trabajar con incertidumbre [3]. De aquí, que hallan surgido las metodologías ágiles para el desarrollo de software, que se consideran como ideales para pequeños equipos de desarrollo, evitando el exceso de robustez en las documentaciones y adaptándose perfectamente a lidiar con todo tipo de cambios enfrentados a lo largo del proceso de producción. Existen varias metodologías de tendencia ágil que han

alcanzado un gran nivel de popularidad dados sus resultados, como descendiente de dos de las más populares a nivel mundial, se crea en la Universidad de las Ciencias Informáticas la metodología SXP.

1.2. SXP

SXP es una metodología compuesta por las metodologías SCRUM y XP que ofrece una estrategia tecnológica, a partir de la introducción de procedimientos ágiles que permiten actualizar los procesos de software para el mejoramiento de la actividad productiva fomentando el desarrollo de la creatividad, aumentando el nivel de preocupación y responsabilidad de los miembros del equipo y ayudando al líder del proyecto a tener un mejor control del mismo. SCRUM es una forma de gestionar un equipo de manera que trabaje de forma eficiente y de tener siempre medidos los progresos, de forma que se sepa por dónde se anda. XP más bien es una metodología encaminada para el desarrollo; consiste en una programación rápida o extrema, cuya particularidad es tener como parte del equipo, al usuario final, pues es uno de los requisitos para llegar el éxito del proyecto. Consta de 4 fases principales: planificación-definición; desarrollo; entrega; y por último mantenimiento. De cada una de estas fases se realizan numerosas actividades de donde se generan artefactos para documentar todo el proceso.

Por cada fase se generan las siguientes actividades y artefactos:

Planificación-Definición: Se establece la visión, se fijan las expectativas y se realiza el aseguramiento del financiamiento del proyecto.

- Actividades:
  - Escribir la visión (Concepción del sistema), presupuesto y reserva del producto con estimaciones iniciales.
- Artefactos:
  - Concepción del sistema
  - Historia de usuario
  - Lista de reserva del producto (LRP)
  - Lista de riesgos
  - Modelo de diseño
  - Modelo de historia de usuario del negocio

Desarrollo: Se realiza la implementación del sistema hasta que esté listo para ser entregado.

- Actividades:
  - Reunión de planificación de la iteración.
  - Definir la reserva de la iteración.
  - Reuniones de coordinación.
  - Revisión de la iteración.
  - Realización de las pruebas de aceptación del sistema.
- Artefactos:
  - Caso de prueba de aceptación
  - Cronograma de producción
  - Estándar de código
  - Plan de release
  - Tarea de ingeniería

Entrega: Puesta en marcha.

- Actividades:
  - Documentación
  - Entrega
  - Marketing
- Artefactos:
  - Manual de desarrollo.
  - Manual de identidad.
  - Manual de usuario.

Mantenimiento: Donde se realiza el soporte para el cliente.

- Actividades:
  - Soporte
- Artefactos:

- Gestión de cambio.

Cada uno de estos artefactos es desarrollado y supervisado por los siguientes roles:

- Líder del Proyecto (Scrum Master).
- Gerente (Management).
- Especialistas.
- Consultor.
- Cliente (Customer).
- Miembros del Proyecto (Scrum Team).
  - Programadores (Programmers).
  - Analista (Analyst).
  - Diseñadores (Designers).
  - Encargado de Pruebas (Tester).
  - Arquitecto (Architect).

Las entregas son frecuentes, y existe una refactorización continua, lo que permite mejorar el diseño cada vez que se le añada una nueva funcionalidad.

SXP está especialmente indicada para proyectos de pequeños equipos de trabajo, rápido cambio de requisitos o requisitos imprecisos, donde existe un alto riesgo técnico y se orienta a una entrega rápida de resultados y una alta flexibilidad. Ayuda a que trabajen todos juntos, en la misma dirección, con un objetivo claro, permitiendo además seguir de forma clara el avance de las tareas a realizar, de forma que los jefes pueden ver día a día cómo progresa el trabajo.

1.3. Modelos que garantizan la calidad

Uno de los problemas que se afrontan actualmente en la esfera de los procesos de desarrollo, es la calidad del software. Cuando se habla de calidad del software se hace referencia al conjunto de cualidades que determinan su utilidad. En sí, la calidad, es el grado en que un software cumple con los requisitos especificados tales como eficiencia, flexibilidad, corrección, mantenimiento, seguridad e integridad [15].

Hoy en día las compañías industrializadas de todo el mundo reconocen que la calidad del producto se traduce en ahorro de costos y en una mejora general. La industria de desarrollo de software no es la excepción, por lo que en los últimos años se han realizado intensos trabajos para aplicar los conceptos de calidad en el ámbito del software. Hablar de calidad del software implica la necesidad de contar con parámetros que permitan establecer los niveles mínimos que un producto de este tipo debe alcanzar para que se considere de calidad.

El problema es que la mayoría de las características que definen al software no se pueden cuantificar fácilmente, generalmente, se establecen de forma cualitativa, lo que dificulta su medición, ya que se requiere establecer métricas que permitan evaluar cuantitativamente cada característica dependiendo del tipo de software que se pretende calificar.

La política establecida para lograr una buena calidad debe estar sustentada sobre tres principios básicos: tecnológico, administrativo y ergonómico.

- El principio tecnológico define las técnicas a utilizar en el proceso de desarrollo del software.
- El principio administrativo contempla las funciones de planificación y control del desarrollo del software, así como la organización del ambiente o centro de ingeniería de software.
- El principio ergonómico define la interfaz entre el usuario y el ambiente automatizado.

La adopción de una buena política contribuye en gran medida a lograr la calidad del software, pero no la asegura. Para el aseguramiento de la calidad es necesario su control o evaluación.

Todos los autores coinciden en que el software posee determinados índices medibles que son las bases para la calidad, el control y el perfeccionamiento de la productividad. Una vez seleccionados los índices de calidad, se debe establecer el proceso de control.

Para controlar la calidad del software es necesario, ante todo, definir los parámetros, indicadores o criterios de medición, ya que, como bien plantea Tom De Marco, usted no puede controlar lo que no se puede medir.

La calidad de un producto no es algo que se añade al final, es algo que se cuida a lo largo de todo el proyecto de construcción. En el software esto es especialmente cierto, ya que es un resultado básicamente intelectual dependiente del trabajo de los profesionales especializados.

La obtención de un software con calidad implica la utilización de modelos o procedimientos estándares para el análisis, diseño, desarrollo y prueba del software que permitan uniformar la filosofía de trabajo, para lograr una mayor confiabilidad, mantenibilidad y facilidad, a la vez que elevan la productividad, tanto para la labor de desarrollo como para el control de la calidad del software.

Particularmente, en el área de la tecnología, una norma o estándar, es una especificación que reglamenta procesos y productos para garantizar la interoperabilidad.

Por su parte una norma de calidad, es una regla o directriz para las actividades, diseñada con el fin de conseguir un grado óptimo de orden, en el contexto de la calidad.

Las normas son documentos técnicos con las siguientes características:

- Contienen especificaciones técnicas de aplicación voluntaria
- Son elaborados por consenso de las partes interesadas.
- Están basados en los resultados de la experiencia y el desarrollo tecnológico
- Son aprobados por un organismo nacional, regional o internacional de normalización reconocido
- Están disponibles al público

Las normas ofrecen un lenguaje de punto común de comunicación entre las empresas, la administración y los usuarios y consumidores, establecen un equilibrio socio económico entre los distintos agentes que participan en las transacciones comerciales, base de cualquier economía de mercado, y son un patrón necesario de confianza entre cliente y proveedor.

Por su parte un modelo constituye un arquetipo o punto de referencia para imitarlo o reproducirlo. Los datos indican que los problemas de los proyectos de software se van reduciendo progresivamente desde mediados de los noventa gracias a la introducción de métodos de trabajo más sistemáticos y fiables por ejemplo, CMMI, ISO, SPICE.

Normas Organización Internacional para Estandarización (ISO):

A través de los estándares internacionales ISO se establece una serie de pautas y patrones que las entidades deberán seguir con la finalidad de implementar un sistema de gestión y aseguramiento de la calidad en el desarrollo de sus procesos.

Dentro de los estándares internacionales voluntarios elaborados por dicha organización encontramos a los de la familia ISO 9000, referidos a la gestión y aseguramiento de la calidad, e ISO 14000, sobre la gestión ambiental.

La familia ISO 9000, a través de la cual se propone la implementación de sistemas de gestión y aseguramiento de la calidad, engloba varios estándares internacionales. Dentro de ellos se destacan los estándares ISO 9001, sobre diseño, producción, instalación y servicio post-venta; ISO 9002, referidos a la instalación y servicio post-venta; ISO 9003, inspecciones y ensayos finales, e ISO 9004-1, que se constituye en una guía para la gerencia en el desarrollo de un sistema de calidad.

ISO 14000, en cambio, es el término genérico utilizado para designar a la familia de estándares internacionales sobre gestión ambiental, que enfatiza la acción preventiva antes que correctiva y un desempeño de continua mejora de temas ambientales.

En las áreas contenidas en dicha familia se encuentran los sistemas de gestión ambiental (ISO 14001 y 14004); auditoría ambiental e investigación relacionada (ISO 14010, 14011 y 14012); evaluación de desempeño ambiental (14031); etiquetado ambiental (14022,14023); ciclo de vida (14040, 14041); términos y definiciones (14050) y estándares ambientales de productos (14060). Los certificado s ISO son otorgados por las denominadas entidades certificadoras, que pueden ser entidades nacionales o extranjeras, las cuales realizan una evaluación exhaustiva de los procesos de las empresas que pretenden obtener dicho documento.

La obtención de certificados que garanticen ciertos estándares de calidad o de preservación del medio ambiente ocasiona a las empresas una serie de ventajas competitivas. Entre las más importantes se tienen: reducción de costos, mayor rentabilidad, mejoras en la productividad, motivación y compromiso por parte del personal en una cultura de calidad y mejor posicionamiento en el mercado al constituir

una importante herramienta de marketing. De la misma manera en que aportan grandes beneficios en el sistema de calidad a las empresas, aunque estas normas están generalmente diseñadas para agregar valor en el sistema de calidad, no siempre se cumple el objetivo, no por causa de la misma norma [16].

El efecto negativo puede tener origen en diferentes aspectos, el más común, es que no todas las empresas adoptan la norma como un sistema de calidad, la motivación hacia el ISO es más un certificado necesario que otorga ventajas competitivas, alejándolos del propósito inicial de la misma norma, por lo que el interés está centrado más que en el mejoramiento, en la certificación y lo que ello significa, y la otra razón es que se inicia el proceso de implementación sin antes hacer un debido proceso de sensibilización que la facilite, pues todas las empresas no están en las condiciones ideales para iniciar un proceso de certificación en la norma ISO [17].

SPICE:

Software Process Improvement and Capability Determination (SPICE): Estándar ISO/IEC 15504 para la evaluación de los procesos de desarrollo de software. Su principal objetivo es que constituye un estándar de evaluación de procesos de software para evaluar la capacidad de los procesos, contribuir a la mejora continua y ser la base para el comercio internacional de software. Como características fundamentales que presenta se puede encontrar que:

- Establece un marco para métodos de evaluación.
- Comprende evaluación de procesos, mejora de procesos, determinación de capacidad.
- Está alineado con el estándar ISO/IEC 12207 que define los procesos del ciclo de vida del desarrollo, mantenimiento y operación de los sistemas de software.
- Equivalencia y compatibilidad con CMMI. ISO forma parte del panel elaborador del modelo CMMI y SEI mantiene la compatibilidad y equivalencia de ésta última con 15504. Sin embargo CMMI aún no es conforme a SPICE - ISO 15504.

Es un modelo de dos dimensiones:

Desde la dimensión de proceso agrupa a los procesos en tres grupos que contienen cinco categorías de acuerdo al tipo de actividad.

Procesos primarios

- CUS: Cliente - Proveedor
- ENG: Ingeniería

Procesos de soporte

- SUP: Soporte

Procesos organizacionales

- MAN: Gestión
- ORG: Organización

Para todos los procesos se definen los componentes: identificador, nombre, tipo, propósito, salidas y notas.

Desde la dimensión de capacidad el modelo define una escala de 6 niveles para determinar la capacidad de cualquier proceso:

Nivel 0: Incompleto

Nivel 1: Realizado

Nivel 2: Gestionado

Nivel 3: Establecido

Nivel 4: Predecible

Nivel 5: En optimización[18]

1.3.1. CMMI

Capability Maturity Model Integration (CMMI) es un modelo de mejora de procesos de desarrollo que provee orientación para diseñar procesos efectivos tales como tiempo y coste, en distintos dominios que pueden ser desarrollo de productos y servicios, adquisiciones y mantenimiento, etc., dentro del ámbito de una organización, cuya principal premisa es que la calidad de un producto es determinada en gran medida por la calidad del proceso utilizado para desarrollarlo y mantenerlo.

Las metas de este modelo son:

- Entregar software de mejor calidad, y cumplir con las expectativas y necesidades del cliente, o sea, con los requisitos del cliente.
- Incrementar la productividad, o sea, controlar el desarrollo con menos errores y detección de los mismos en etapas más tempranas.
- Mejorar el modelo de estimación y planificación del esfuerzo en coste y tiempo para lograr ser más eficientes en la entrega de productos y servicios, evitando los sobrecostes que conllevan a una mala estimación y planificación [19].

1.3.2. Componentes del Modelo CMMI

Se denomina componente a cualquiera de los elementos principales de la arquitectura que componen el modelo CMMI. Los componentes del modelo CMMI son: áreas de procesos, metas genéricas, metas específicas, prácticas genéricas, prácticas específicas y sub prácticas como se muestra en el Anexo A.

A continuación se definen cada uno de los elementos del CMMI:

- Áreas de Procesos (PA)

Un área de proceso es un conjunto de prácticas relacionadas, que ejecutadas colectivamente satisface un conjunto de metas consideradas importantes para hacer mejoras significativas en esa área.

- Metas Genéricas (GG)

Son llamadas genéricas porque la misma descripción aparece en múltiples áreas de procesos.

- Metas Específicas (SG)

Las metas específicas se aplican a áreas de procesos y direccionan características únicas que describen lo que debe ser implementado para satisfacer el área de proceso.

- Prácticas Genéricas (GP)

Las prácticas genéricas proveen institucionalización para asegurar que el proceso asociado con el área de proceso debe ser repetible y duradero. Son categorizadas por metas genéricas y características comunes.

- Prácticas Específicas (SP)

Una práctica específica es una actividad que es considerada importante en la meta específica asociada. Describe las actividades esperadas para conseguir las metas específicas de un área de proceso.

- Sub prácticas

Son descripciones detalladas que proveen guías para interpretar prácticas específicas o genéricas [20].

1.3.3. Niveles de CMMI

CMMI permite enfocar las evaluaciones y mejora de los procesos usando dos representaciones: la continua y la escalonada. Ver Anexos B y C.

La representación continua permite a una organización seleccionar un área de proceso o grupo de áreas de proceso y mejorar los procesos relacionados con ella. Esta representación utiliza los niveles de capacidad para caracterizar la mejora en relación con un área de proceso individual.

La representación escalonada utiliza un grupo predefinido de áreas de proceso para definir un camino de mejora para una organización. Este camino de mejora se caracteriza por niveles de madurez. Cada nivel de madurez proporciona un conjunto de áreas de proceso que caracterizan los diferentes comportamientos de la organización.

Los niveles son utilizados en CMMI para describir un camino evolutivo recomendado para toda organización que quiera mejorar los procesos que utiliza para desarrollar y mantener sus productos y servicios.

Los 6 niveles de capacidad definidos en CMMI son:

Nivel 0 Incompleto: el proceso no se realiza o es realizado parcialmente y no se logran sus objetivos.

Nivel 1 Ejecutado: el proceso se ejecuta y se logran sus objetivos; se obtienen como resultado importantes mejoras, pero las mismas pueden perderse con el transcurso del tiempo.

Nivel 2 Gestionado: además de ser ejecutado, el proceso se planifica, se revisa y se evalúa para comprobar que cumple los requisitos.

Nivel 3 Definido: además de ser un proceso gestionado se ajusta a la política de procesos que existe en la organización, alineada con las directivas de la empresa.

Nivel 4 Cuantitativamente gestionado: además de ser un proceso definido, se controla utilizando técnicas cuantitativas.

Nivel 5 Optimizado: además de ser un proceso cuantitativamente gestionado, de forma sistemática se revisa y modifica para adaptarlo a los objetivos.

De la mano con estos niveles, este modelo define para calificar la madurez de una organización cinco niveles, estos serán mencionados a continuación:

Nivel 1 Inicial: es el nivel en que se encuentran la mayoría de las empresas cuando se inician en el proceso de producción de software. Los procesos se caracterizan por su estado caótico. La organización no suele proporcionar un entorno estable para apoyar los procesos. Se definen pocos procesos y el éxito depende del esfuerzo individual. No hay control sobre el estado del proyecto, el desarrollo del proyecto es completamente opaco. Con frecuencia las organizaciones superan sus presupuestos, no cumplen sus horarios y tienden al abandono de los procesos en un momento de crisis. Incapacidad para repetir sus éxitos.

Nivel 2 Gestionado: la principal diferencia entre este nivel y el anterior es que el proyecto es gestionado y controlado durante el desarrollo del mismo. Los procesos se planifican y ejecutan conforme a la política del proyecto. El éxito de los resultados obtenidos se puede repetir. El desarrollo no es opaco y se puede saber el estado del proyecto en todo momento.

Nivel 3 Definido: los procesos se gestionan de manera más activa utilizando una comprensión de las interrelaciones de las actividades del proceso y medidas detalladas del proceso, sus productos y servicios.

Los procesos están bien caracterizados y comprendidos, y se describen las normas, procedimientos, herramientas y métodos. El proceso de gestión de las actividades del software y de ingeniería se documenta, se estandariza y se integra dentro de un proceso de software de toda una organización.

Nivel 4 Cuantitativamente Gestionado: se recopilan medidas detalladas del proceso del software y de la calidad del producto. Mediante la utilización de medidas detalladas, se comprenden y se controlan cuantitativamente tanto los productos como el proceso del software. En este nivel se incluyen todas las características definidas para el nivel 3.

Nivel 5 Optimizado: mejora continuamente sus procesos sobre la base de una comprensión cuantitativa de las causas comunes de la variación inherente a los procesos. Mediante una retroalimentación cuantitativa del proceso, ideas y tecnologías innovadoras se posibilita una mejora del proceso [21].

El nivel 1 de CMMI es el nivel en el que están todas las empresas u organizaciones, más bien se tendría que haber llamado nivel 0, ya que sólo por el mero hecho de existir como empresa u organización de software se está en el nivel 1.

Por lo tanto todas aquellas empresas que quieran implantar CMMI o tan sólo quieran mejorar su manera de trabajar para conseguir mejores resultados quieren avanzar hasta el nivel 2. Esta es la situación en la que se encuentra la Universidad de las Ciencias Informáticas, optando por alcanzar el segundo nivel.

El nivel 2 de CMMI pese al ser el primer nivel es muchas veces el más difícil de alcanzar y esto es porque requiere que se cambie la forma de trabajar de la empresa, lo que la mayoría de las veces implica un cambio cultural de la misma. Por este motivo es necesario un fuerte apoyo de la dirección para afrontar este cambio, ya que sin él no se tendrá suficiente autoridad en momentos difíciles.

Para alcanzar cada uno de estos niveles, es necesario tener conocimiento del objetivo que persigue cada uno de ellos, el nivel 2 de CMMI pretende conseguir que en los proyectos de la organización haya una gestión de los requisitos y que los procesos, que son las formas de hacer las cosas, estén planeados, ejecutados, medidos y controlados. Como se había mencionado antes, se debe dar cumplimiento para alcanzar dichos niveles a la materialización de ideas particulares de las áreas de procesos de cada uno de ellos.

1.3.4. Áreas de proceso del Nivel 2 de CMMI

Estas ideas se materializan en las siguientes áreas de proceso, ver Anexo D:

- Gestión de Requisitos.
- Planificación de proyectos.
- Monitorización y Control de proyectos.
- Medición y Análisis.
- Aseguramiento de la calidad.
- Gestión de la configuración.

Se pasará a explicar particularmente las áreas de proceso que comprenden las actividades de estimación y medición con un poco más de detalle aunque también se hará alusión al área de aseguramiento de la calidad por la relación que guarda con este tema.

Planificación de proyectos

La gestión de un proyecto de software comienza con un conjunto de actividades que globalmente se denominan planificación del proyecto.

La planificación comienza con los requisitos que definen el producto y el proyecto e incluye estimación de los atributos de las tareas y productos de trabajo, determinación de los recursos necesarios, negociación de los compromisos, realización de un calendario e identificación y análisis de los riesgos del proyecto.

La planificación del proyecto es considerada una de las 7 áreas de negocio que deben ser estandarizadas para llegar al nivel 2 de madurez en CMMI. El objetivo de la planificación de proyectos es establecer y mantener planes que definan las actividades del proyecto.

El área de proceso de CMMI, Planificación de proyecto, incluye las siguientes tareas:

- Desarrollar un plan inicial del proyecto.

- Establecer una relación adecuada con todas las personas involucradas en el proyecto.
- Obtener compromiso con el plan.
- Mantener el plan durante el desarrollo del proyecto.

Se basa en tres metas:

- Establecer las estimaciones.
- Desarrollar un plan del proyecto.
- Obtener el compromiso con el plan.

Dentro de la fase de la planificación. Establecimiento de estimaciones:

Las estimaciones deben sustentarse sobre una base sólida para infundir confianza de manera tal que cualquier plan sobre la base de estas estimaciones sea capaz de confirmar los objetivos del proyecto. Deben tenerse en cuenta a su vez un conjunto de factores para efectuar las estimaciones, entre los cuales se encuentran: requisitos del proyecto y el producto, alcance del proyecto, atributos de las tareas y productos de trabajo tales como complejidad y tamaño, y por último modelos o datos históricos que permitan convertir los atributos de las tareas y productos de trabajo en horas de trabajo y costo.

Para estimar todo proyecto de software, el siguiente proceso debe ser explicado:

- Primero hay que estimar el tamaño, o sea la cantidad de cosas que hay que hacer. En ocasiones esto se entiende como la complejidad. Existen varios tipos de métricas para ello, siendo los puntos función la que ofrece mejores resultados, aunque su utilización y actualización también requiere un esfuerzo mayor.
- Después de estimar el tamaño, estimar el esfuerzo, medido normalmente en días de trabajo.
- Sobre los días de trabajo previstos, estimar el coste.
- Por último se cierra al establecer un calendario de trabajo, teniendo en cuenta los recursos disponibles, toda la información anterior y las propias condiciones de contorno en que se realizará el proyecto.

Existe un problema con las estimaciones sobre el tamaño y es que este depende de muchos factores. La única manera de tenerlo todo en consideración es preguntar a los expertos, y refinar la lista más adelante.

También se pueden utilizar datos históricos, y existen bases de datos de proyectos, pero están adaptadas a los puntos de función, que es la forma más reconocida para comparar proyectos. Es una manera muy rigurosa y precisa, pero cuesta mucho.

Para dar cumplimiento a esta meta de establecimiento de estimaciones se implantan y mantienen estimaciones de los parámetros del proyecto.

En el modelo CMMI, estas metas u objetivos específicos, se subdividen en un conjunto de prácticas específicas, las mismas están definidas por:

Estimar el alcance del proyecto

Una vez definidos los requisitos, será posible determinar cuáles serán las características del producto, sus limitaciones, cuánto trabajo requerirá la realización del proyecto y cuáles serán los posibles resultados, además de proporcionar un esquema para la organización del trabajo de proyecto alrededor del producto y los componentes del producto que apoyan el trabajo.

Para realizar la estimación del alcance del proyecto, CMMI propone establecer un nivel superior de Estructura de Descomposición del Trabajo del proyecto (en inglés, WBS), el desarrollo de esta estructura divide el proyecto en un conjunto interconectado de componentes gestionables. El WBS es un producto que proporciona un plan para identificar y organizar las unidades lógicas de trabajo para ser gestionadas, que son llamadas paquetes de trabajo y proporciona una referencia y un mecanismo de organización para determinar esfuerzo, calendario y responsabilidades y es usado como una estructura para organizar y controlar el trabajo realizado en el proyecto.

Establecer las tareas y productos de trabajo

Esta es una de las actividades esenciales de la planificación, para estimar esfuerzo y costo y para la realización del cronograma y el plan del proyecto. Las estimaciones de las tareas y los productos de trabajo deben establecerse y mantenerse.

El tamaño es la entrada principal a muchos modelos usados para estimar esfuerzo, costo y plan. Los modelos pueden también basarse en entradas tales como conectividad, complejidad y estructura. Las estimaciones deben ser consistentes con los requisitos del proyecto para determinar el esfuerzo, el coste y el calendario del proyecto. Un nivel relativo de dificultad o complejidad se debe asignar para el tamaño de cada atributo.

Definir el ciclo de vida del proyecto

La determinación de las fases del ciclo de vida de un proyecto permite la evaluación y la toma de decisiones para períodos planificados. Éstas se definen normalmente para apoyar los puntos de decisión lógicos en los cuales los acuerdos significativos se hacen respecto a los recursos y al acercamiento técnico. Tales puntos proporcionan los acontecimientos planeados en los cuales las correcciones del curso del proyecto y las determinaciones del alcance futuro y el coste pueden ser realizadas.

Las fases del ciclo de vida del proyecto necesitan ser definidas dependiendo del alcance de los requisitos, de las estimaciones para los recursos del proyecto y de la naturaleza del proyecto.

Proyectos grandes pueden contener fases múltiples, por ejemplo exploración, desarrollo, producción, operaciones y disposición. Dentro de estas fases, las sub-fases pueden ser necesarias. Una fase del desarrollo puede incluir sub-fases tales como análisis de requisitos, diseño, fabricación, integración y verificación. La determinación de las fases del proyecto incluye generalmente la selección y el refinamiento de uno o más modelos del desarrollo para tratar interdependencias y un orden apropiado de las actividades en las fases.

Dependiendo de la estrategia para el desarrollo, puede haber fases intermedias para la creación de prototipos de incrementos de la capacidad o de ciclos de modelo espiral.

Entender el ciclo de vida del proyecto es crucial en la determinación del alcance del esfuerzo de planificación y de la sincronización de la planificación inicial, tan bien como la sincronización y los criterios o hitos críticos para la re-planificación.

Determinar las estimaciones de esfuerzo y costo

Las estimaciones del esfuerzo se basan generalmente en los resultados del análisis usando los modelos o los datos históricos aplicados al tamaño, a las actividades y a otros parámetros de la planificación. La confianza en estas estimaciones está basada en el análisis razonado para el modelo seleccionado y la naturaleza de los datos.

Puede haber ocasiones, cuando los datos históricos disponibles no se aplican, que los esfuerzos no tengan precedentes, esto es debido a que, en cierto grado, un producto o componente similar nunca se ha construido; dichos esfuerzos son más riesgosos, requieren más investigación para desarrollar bases razonables de estimación y requieren más gestión de reserva.

Hoy en día se han desarrollado un conjunto de modelos para realizar estimaciones; el uso de estos modelos como la fuente única de estimación no es recomendado porque estos modelos se basan en los datos históricos del proyecto que pueden o no, ser pertinentes al proyecto. Es aconsejable utilizar junto a estos modelos, algunos métodos de estimación para asegurar un alto nivel de confianza.

Existe un grupo de elementos necesarios para la estimación de esfuerzo, entre los cuales se encuentran las estimaciones proporcionadas por un experto o conjunto de expertos, las tareas de trabajo, conocimientos, aptitudes y necesidades de formación, instalaciones necesarias, entre otros [21]. Las estimaciones deben realizarse siguiendo un modelo lógico y documentado, aunque eso va contra la lógica de la estimación. Es necesario empezar a definir y controlar ese proceso.

Medición y Análisis

El propósito de la medición y el análisis es desarrollar y sostener capacidades de medición para dar soporte a la gerencia de la información. Los datos tomados para la medición deben estar alineados con los objetivos de la organización para proporcionar información útil a la misma.

Se ha de implantar un mecanismo de recogida de datos, almacenamiento y análisis de los mismos de forma que las decisiones que se tomen puedan estar basadas en estos datos.

Este sistema tiene que permitir además:

- Planificación y estimación objetiva.
- Comparar el rendimiento actual contra el rendimiento esperado en el plan.
- Identificar y resolver problemas relacionados con los procesos.
- Proporcionar una base para añadir métricas en procesos futuros.

Dentro de esta área se den realizar las siguientes actividades:

- Alinear mediciones y actividades de análisis: donde se alinean los objetivos y actividades de medición con los objetivos y necesidades de información identificados.
- Proporcionar resultados de mediciones: donde se proporcionan los resultados de mediciones dirigidos a los objetivos y necesidades de información.

Cuando se planifica un proyecto se tiene que obtener estimaciones del costo y esfuerzo humano requerido por medio de las mediciones de software, que se utilizan para recolectar los datos cualitativos acerca del software y sus procesos para aumentar su calidad. Esta actividad no sólo es útil en las estimaciones y planificación del proyecto, sino que además se sustenta en un conjunto de métricas que permiten conocer el estado actual del proyecto, el producto y el proceso, por lo que constituye una actividad importante a tener en cuenta para el seguimiento y control de proyectos informáticos.

Aseguramiento de la calidad

El objetivo del aseguramiento de la calidad es proporcionar personas y gestión con el objetivo de que los procesos y los elementos de trabajo cumplan los procesos.

Esto se consigue mediante:

- Evaluar objetivamente la ejecución de los procesos, los elementos de trabajo y servicios contra las descripciones de procesos, estándares y procedimientos.

- Identificar y documentar los elementos no conformes.
- Proporcionar información a las personas que están usando los procesos y a los gestores, de los resultados de las actividades del aseguramiento de la calidad.
- Asegurar que los elementos con los cuales no se esté completamente conformes, son arreglados.

Esta es un área de proceso clave, que a veces no se le da la suficiente importancia, pero que sin ella no será posible implantar un modelo de calidad [22].

1.4. Estimación de software

La gestión de un proyecto de software comienza con un conjunto de actividades que globalmente se denominan planificación de proyectos. Antes de que el proyecto comience se debe realizar una estimación del trabajo a realizar, los recursos necesarios y el tiempo que transcurrirá desde el comienzo hasta el fin de su realización.

La acción de estimar, se ha convertido, por su importancia en el proceso de desarrollo del software, en una amplia y variada ciencia, mediante la cual tanto los equipos de desarrollo como gestores o encargados del negocio obtienen cada vez mayores beneficios.

Estimar: Es echar un vistazo al futuro y aceptar resignados cierto grado de incertidumbre [2].

La estimación es un tipo de análisis de medición empleado para establecer valores de referencia o expectativas numéricas para las futuras actividades del proyecto, basado en los datos actuales disponibles. La misma produce proyecciones de tamaño de producto, esfuerzo y cronograma requerido para completar el proyecto. Puede incluso producir proyecciones de calidad del producto. Estos estimados forman la base para los planes iniciales del proyecto y las siguientes re-planificaciones. Es importante completar los estimados para las medidas claves del proyecto tales como tamaño y esfuerzo en diferentes puntos del ciclo de vida [4].

La estimación de la duración de las actividades que conforman el desarrollo de software es un contenido que tiene mucha relación con la gestión y control de proyectos para asegurar el éxito. Aunque la

estimación, es más un arte que una ciencia, es una actividad importante que no debe llevarse a cabo de forma descuidada.

Al estimar se toma en cuenta no sólo del procedimiento técnico a utilizar en el proyecto, sino que se toman en cuenta los recursos, costos y planificación. El tamaño del proyecto es otro factor importante que puede afectar la precisión de las estimaciones. A medida que el tamaño aumenta, crece rápidamente la interdependencia entre varios elementos del software [2]. Como se puede observar se estima generalmente entorno a estas variables, recursos, costos tamaño del producto, entre otras, es válido decir que independientemente de que se puedan obtener estimaciones particulares de cada una de estas variables, todas tienen una estrecha relación influyendo cada una en los demás valores de estimación a obtener.

En la mayor parte de los proyectos la estimación es el primer análisis que se hace pero, estos estimados iniciales con frecuencia son imprecisos, ya puede ser porque a menudo se estima antes de escoger el equipo de proyecto o por cualquier otro motivo, por lo que se debe actualizar esa estimación durante todo el ciclo de vida del proyecto.

1.4.1. Para una buena práctica de la estimación

Pasos a seguir para realizar una estimación correcta:

Para iniciar este proceso, se debe proceder a una búsqueda exhaustiva de información no sólo del proyecto sino de el entorno en el que se va a desarrollar, lo cual enfoca el desarrollo hacia el mejor conocimiento de restricciones y objetivos a seguir, de esta forma generalmente se le da cumplimiento a la siguiente secuencia de pasos:

Seleccionar el Enfoque: seleccionar un enfoque de medición para el elemento. El enfoque puede emplear un modelo matemático o una técnica basada en relaciones de estimación simple.

Asociar y Calibrar: asociar el enfoque a la secuencia de actividades del ciclo de vida del proyecto, y calibrar los modelos de estimación asociados con los datos históricos de la organización.

Computar los estimados: computar los estimados de tamaño, esfuerzo, cronograma y calidad empleando el enfoque o modelo seleccionado.

Evaluar Estimados: comparar los resultados con las restricciones del proyecto y consideraciones para evaluar el estimado. Si el estimado no satisface las restricciones del proyecto, entonces se deben hacer los ajustes apropiados y rehacer los estimados [4].

La mayoría de las veces, las estimaciones se llevan a cabo valiéndose de la experiencia pasada como única guía. Si un nuevo proyecto es bastante similar en tamaño y función a un proyecto pasado, es muy probable que el nuevo proyecto demande aproximadamente la misma cantidad de esfuerzo, que dure aproximadamente el mismo tiempo y que cueste aproximadamente lo mismo que el producto anterior. Pero si el proyecto es totalmente distinto, la experiencia no será suficiente.

Es muy importante que antes de hacer una estimación, el planificador del proyecto comprenda el ámbito del software a construir y genere una estimación de su tamaño. Luego de estar ubicados exactamente en el terreno en que se desarrollara el software deben tenerse en cuenta varios aspectos que marquen la diferencia entre una buena estimación y una promedio.

Un gran error en la estimación del costo puede ser lo que marque la diferencia entre beneficios y perdidas, la estimación del costo y del esfuerzo del software nunca será una ciencia exacta, son demasiadas las variables: humanas, técnicas, de entorno, políticas, que pueden afectar el costo final del software y el esfuerzo aplicado para desarrollarlo.

De la misma manera, una estimación correcta debe medir los siguientes aspectos:

- Tamaño del proyecto:
  El tamaño de un proyecto es una medida que permite estimar y obtener métricas sobre un proyecto en ejecución o terminado, esta pueden ser analizando las líneas de código ó la cantidad de horas–hombre.

- Esfuerzo:
  El esfuerzo es una de las magnitudes fundamentales de la estimación. Lo que mide el esfuerzo es el costo, en horas-hombre o meses-hombre, o una unidad por el estilo, del proyecto. Se podría decir que es un costo sin valorizar. Permite calcular aproximadamente el costo del proyecto, pero no necesariamente es tan lineal.

- Costo de recursos:
  El costo es la medida, en pesos, u otra moneda, que se estima va a demandar el proyecto. Habitualmente surge del esfuerzo pero, deberá reestimarse antes de comenzar el proyecto, si los supuestos con que se presupuestó el mismo son de imposible cumplimiento.

- Plazo en que se puede terminar el proyecto:
  Otro valor derivado del esfuerzo es el plazo de ejecución. Para obtener el plazo se debe basar en el esfuerzo, la cantidad de gente con la que se va a trabajar y sus perfiles, una evaluación de costos y beneficios de utilizar más recursos para terminar antes, si esto es posible.

- Alcance que es posible lograr con un costo y plazo dados:
  No suele considerarse entre las variables a estimar, su omisión es un error bastante serio. Cualquier proyecto, de software u otro, admite que se estime el alcance, la necesidad de estimarlo, viene dada por la solicitud de los clientes acerca de cuánta funcionalidad vamos a poder desarrollar con un determinado costo o un determinado plazo, es de extremada importancia, pues evita que en otros casos, dificultades durante el desarrollo o cambios de requisitos surgidos durante el mismo, hagan que cambiemos el alcance sin modificar costo ni plazo [6].

De esta manera, queda claro que el resultado de una estimación no es uno o varios valores que se calculen una vez y nunca más se toquen. Lo habitual es hacer una primera estimación gruesa la primera vez que necesitamos conocer la pre factibilidad de un proyecto. Luego de un primer relevamiento del proyecto, se debe estimar con mayor precisión, basándose en el mayor conocimiento que se tiene del producto a construir y del marco temporal y de recursos del proyecto.

1.4.2. Modelos y Técnicas de estimación

Existen diferentes Modelos y Técnicas de estimación, de la gran variedad conocida por los equipos de desarrollo en la UCI, se mencionan algunos a continuación:

Opinión de Expertos: son estimaciones basadas en conocimientos o experiencias acumuladas como ingenieros de software. Se consultan varios expertos en las técnicas de desarrollo de software propuestas y en el dominio de aplicación. Cada uno de ellos estima el costo del proyecto. Estas estimaciones se comparan y discuten. El proceso de estimación se itera hasta que se acuerda una estimación.

Analogías: estas técnicas son aplicables cuando otros proyectos en el mismo dominio de aplicación se han completado. Se estima el costo de un nuevo proyecto por analogía con estos proyectos completados anteriormente. Consiste en emitir un juicio acerca del proyecto a estimar donde la estimación inicial estará basándose en la diferencia entre el proyecto desarrollado con anterioridad y el actual, aunque sí es necesario cuantificar la similitud o diferencia entre los proyectos.

Descomposición: se divide el producto en sus componentes elementales y de igual manera un proyecto en sus tareas elementales, sumando las estimaciones para cada componente elemental para obtener el total estimado.

Modelo Empírico: deben ser subjetivamente un resultado de la experiencia acumulada, que relacione atributos de gran interés con otros atributos mensurables, este método es la base para cualquier proceso de estimación.

Modelado del algoritmo de costos: se desarrolla un modelo utilizando información histórica de costos que relaciona alguna métrica de software por lo general, su tamaño con el costo del proyecto. Se hace una estimación de esa métrica y el modelo predice el esfuerzo requerido.

Ley de Parkinson: la Ley de Parkinson establece que el trabajo se extiende para llenar el tiempo disponible. El costo se determina por los recursos disponibles más que por los objetivos logrados. Si el software se tiene que entregar en 12 meses y se dispone de cinco personas, el esfuerzo requerido se estima en 60 personas-mes.

Asignar costos para ganar: el costo del software se estima dependiendo de lo que el cliente esté dispuesto a pagar por el proyecto. El esfuerzo estimado depende del presupuesto del cliente y no de la funcionalidad del software.

Modelos PERT: el esfuerzo requerido es estimado tomando en cuenta las las estimaciones, la más baja, la más alta, y la más probable. Se calcula Esfuerzo $=$ más baja $+ 4 *$ más probable $+$ más alta$/6$. Las estimaciones individuales se hacen usando otros métodos.

Modelos Matemáticos: correlacionan algunas medidas con el esfuerzo total requerido. Estos modelos son bastante utilizados por metodologías tradicionales, dentro de ellos se encuentran técnicas de un alto ranking de utilización, o sea son preferidas por muchos equipos de desarrollo, entre ellas se encuentran:

- Puntos de función:

  Se Mide lo que el usuario pide y lo que recibe, así como los 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. El análisis de los puntos de función se desarrolla considerando cinco parámetros básicos externos del sistema: entrada, salida, consultas, ficheros lógicos internos, ficheros lógicos externos, y luego se utiliza COCOMO para calcular el esfuerzo de desarrollo, a partir de la conversión de los puntos de función sin ajustar en líneas de código fuente.

- COCOMO:

  El Modelo Constructivo de Costos (COnstructive COst MOdel) es una jerarquía de modelos de estimación para el software. Esta jerarquía está constituida por los siguientes modelos:

  COCOMO básico: es un modelo univariable estático que calcula el esfuerzo y el costo del desarrollo de software en función del tamaño del programa expresando en Líneas De Código (LDC) estimadas.

  COCOMO intermedio: calcula el esfuerzo del desarrollo de software en función del tamaño del programa y de un conjunto de conductores de costo, que incluyen la evaluación subjetiva del producto, del hardware, del personal y de los atributos del proyecto.

  COCOMO avanzado: incorpora todas las características de la versión intermedia y lleva a cabo una evaluación de impacto de los conductores de costo en cada fase análisis, diseño, etc. del proceso de ingeniería de software.

  Los modelos COCOMO están definidos para tres tipos de proyecto de software:

  Modelo Orgánico: proyectos de software relativamente pequeños y sencillos en los que trabajan pequeños equipos, con buena experiencia en la aplicación, sobre el conjunto de requisitos poco rígidos.

  Modelo Semiacoplado: proyectos de software intermedio en tamaño y complejidad donde los equipos, con variados niveles de experiencia, deben satisfacer requisitos poco o medio rígido.

  Modelo Empotrado: proyectos de software que deben ser desarrollados en un conjunto de hardware, software y restricciones operativas muy restringidas.

- 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.

1.4.3. Estimaciones ágiles

Los modelos y técnicas mencionadas anteriormente son mayormente tradicionales, aunque no se debe olvidar que plantean aspectos básicos que son vistos en todo tipo de herramienta o técnica de estimación independientemente del enfoque que estas evidencien, a continuación haremos énfasis en técnicas o modelos de estimación adaptables a metodologías ágiles.

Según Rodrigo Corral, la estimación y la planificación plantean un montón de problemas de gran envergadura, tanta que numerosos equipos ni siquiera se plantean el tratar de resolverlos. Sin duda, la solución que tradicionalmente se ha dado, es tan pesada que hacía que muchos proyectos muriesen en la fase de planificación, tampoco ayudaba a que los equipos de desarrollo se lanzasen a realizar una planificación basada en estimaciones. Y los equipos que se atrevían a menudo sucumbían ante el coste que una planificación detallada basada en descomposición de tareas y la identificación de hitos requiere [7].

En estos momentos la mayoría de los expertos en el tema se desvían hacía un proceder ágil, significando esto el auge del uso de técnicas como el Planning Poker ó el Wideband Delphi, es objetivo de nuestra investigación escoger una de estas técnicas para su posterior utilización en la metodología SXP.

1.4.4. Técnicas de estimaciones ágiles

Wideband Delphi

Es una técnica de estimación estructurada en grupo. Se deriva de un método creado en los años de la década de 1940. Se basa fundamentalmente en que varios expertos, tras crear estimaciones individuales, se reúnen para ponerse de acuerdo en una estimación. Un estudio del método original decía que no se obtenían mejores resultados que con estimaciones individuales debido a las presiones políticas que se podían ejercer sobre el grupo. Así que Boehm y Farquhar crean en 1970 el Wideband Delphi como mejora del original.

El proceso mejorado es el siguiente:

1. El coordinador presenta a cada experto la especificación y un formulario de estimación.
2. Los estimadores trabajan individualmente. (Se puede hacer esto tras el paso 3).
3. Se hace una reunión en la que los expertos hablan de los posibles problemas de estimación.
4. Los expertos rellenan las estimaciones y se las dan al coordinador de manera anónima.
5. El coordinador prepara un resumen de las estimaciones y la reparte a todos los expertos.
6. Se reúnen tanto el coordinador como los expertos para ver variaciones en las estimaciones.
7. Los expertos votan anónimamente si aceptan la estimación media. Si alguien vota que No, se vuelve al paso 3.
8. La estimación final es una estimación única (single-point estimate) [8].

Planning Poker

El Planning Poker es muy sencillo, se presentan los requisitos a estimar uno por uno haciendo una descripción de los mismos, donde siempre se discuten los detalles de las estimaciones que han quedado claros.

Con esta técnica es fácil estimar los requisitos de una manera democrática, ágil y rápida y que lleve a estimaciones respaldadas por todos los involucrados y basadas en consensos, en definitiva estimaciones que todo el mundo puede asumir como suyas y que todo el mundo respetará [9]. Se presentan los requisitos a estimar uno por uno haciendo una descripción de los mismos. Luego se procede a discutir aquellos detalles más relevantes o que no hayan quedado claros.

Las estimaciones son privadas, hasta que los que participan en esta técnica tienen su estimación, que luego se hacen públicas y se analizan nuevamente, si es necesario, el requisito.

La tendencia a utilizar estimaciones ágiles, propone que debe ser necesario que todos los desarrolladores, expertos o no, puedan asumir las estimaciones como suyas, poniendo en alta estima la motivación, si un desarrollador no asume una estimación como realista no trata de cumplirla. Steve McConnell habla de las estimaciones percibidas de no realistas como uno de los principales factores que dañan la motivación. La manera de asegurarse que las estimaciones se perciben como realista es basarlas en el consenso y construirlas con la participación de todos los implicados. De la misma manera en que se lleva a cabo la estimación, también es necesario medir a través de ciertas reglas ya sea procesos, producto, o recursos, y medir con el mayor grado de eficiencia posible, también estas reglas se ajustan o no a procesos ágiles. A continuación comenzaremos un estudio de dichas reglas o métricas como término utilizado anteriormente.

1.5. Métricas para la medición del software

Todo software contiene determinados parámetros que son medibles y que constituyen las bases para lograr un producto con la calidad requerida y evaluar la productividad de su personal conociendo así su grado de efectividad. La medición de software se ha llegado ha convertir en una pieza fundamental para realizar las estimaciones del esfuerzo, coste y tiempo necesarios para el desarrollo de productos software. Para llevar a cabo esta medición se hace necesario el apoyo en las métricas de software.

Desde hace mucho tiempo y de manera errónea las palabras métrica, medición y medida se han usado con gran confusión para expresar la misma cosa pero realmente no significan lo mismo.

Métrica: es el término que describe muchos y muy variados casos de medición. Siendo una métrica una medida estadística que se aplica a todos los aspectos de calidad de software los cuales deben ser medidos desde diferentes puntos de vista como el análisis, construcción, documentación, métodos, proceso, usuario, entre otros [4].

Medición: es el acto de determinar una medida, un valor numérico para un atributo cuya magnitud se desea valorar en función de una escala concreta.

Medida: proporciona una indicación cuantitativa de la extensión, cantidad, dimensiones, capacidad o tamaño de algunos atributos de un proceso o producto [4]. Apoyándose en la definición de métrica se puede decir, que una métrica es una medida, no se aplica la misma situación para el caso de una medida. Como cada actividad a desarrollar en el proceso de creación de software la utilización de métricas está sustentada en varias necesidades a la hora de construir un producto.

1.5.1. Razones por las cuales medir

Existen cuatro razones por las cuales medir, ellas son:

- Caracterizar: para comprender mejor los recursos, productos, procesos y entornos del software y para establecer las líneas bases para las comparaciones con evaluaciones futuras.
- Evaluar: para determinar el estado con respecto al diseño Se evalúa para valorar la consecución de los objetivos de calidad del software, el impacto de las tecnologías y las mejoras del proceso.
- Predecir: para poder planificar. Se realizan mediciones para la predicción. Las medidas de predicción son la base para la extrapolación de tendencias, con lo que la estimación para el coste, planificación y calidad se pueden actualizar basándose en la evidencia actual.
- Mejorar: se lleva a cabo para mejorar la información cuantitativa que ayuda a identificar obstáculos problemas de raíz, ineficiencia y otras oportunidades para mejorar la calidad del producto y el rendimiento del proceso.

Las mediciones de software brindan al administrador del proyecto la visión que necesita para la toma de decisiones críticas para el éxito del mismo. Al integrarse en la totalidad de los procesos de administración de proyectos permiten al líder del proyecto poder identificar los riesgos, llevar un seguimiento de dificultades específicas y evaluar su impacto en el costo del proyecto y en el desempeño de las actividades con él relacionadas [4].

Desafortunadamente la medición se aleja de lo común en el mundo de la Ingeniería del software. Se encuentran dificultades en ponerse de acuerdo sobre qué medir y cómo se van evaluar las medidas. Por ello existen varias razones para medir un producto:

- Para indicar la calidad del producto.

- Para evaluar la productividad de la gente que desarrolla el producto.
- Para evaluar los beneficios en términos de productividad y de calidad derivados del uso de nuevos métodos y herramientas de la ingeniería de software.
- Para establecer una línea de base para la estimación.
- Para ayudar a justificar el uso de nuevas herramientas o de formación adicional [10].

Expuestas las razones por las cuales se efectúan las medidas, debemos enfocarnos en las diferentes maneras que existen de hacerlo en cuanto a proceso y software se refiere.

1.5.2. Tipos de medidas

Las mediciones del mundo físico pueden englobarse en dos categorías: medidas directas y medidas indirectas.

Medidas Directas: son aquellas que no dependen de ninguna métrica de otro atributo. En el proceso de ingeniería se encuentran el costo y el esfuerzo aplicado, las líneas de código producidas, velocidad de ejecución, el tamaño de memoria y los defectos observados en un determinado período de tiempo.

Medidas Indirectas: son aquellas que se obtienen a partir de métricas directas. Se encuentra la funcionalidad, calidad, complejidad, eficiencia, fiabilidad, facilidad de mantenimiento, etc.

Las métricas del software se refieren a un amplio elenco de mediciones. Un ingeniero de software recopila medidas y desarrolla métricas las cuales proporcionan una visión más profunda llevando a una toma de decisiones más fundamentada. Son utilizadas principalmente para a partir de ellas obtener las bases para la estimación, determinar la complejidad relativa de los proyectos y seguir sus progresos, darse cuenta y analizar los defectos del producto y entre otras validar las mejores prácticas. Responden a dos objetivos fundamentales: la valoración y la estimación. La primera tiene como principales magnitudes la calidad, fiabilidad y productividad mientras que a la segunda corresponden el esfuerzo y el tiempo.

De manera general, las métricas son utilizadas para que los profesionales e investigadores puedan tomar las mejores decisiones, son un medio para asegurar la calidad en los procesos y productos o proyectos de software. Son un buen medio para entender, monitorizar, controlar, predecir y probar el desarrollo de software y los proyectos de mantenimiento [4].

Es importante conocer que las métricas en su totalidad deben cumplir con una serie de características que garantizaran su correcto funcionamiento. En nuestra investigación exponemos algunos de los estándares que debe cumplir la implementación de una métrica para su óptima utilización.

1.5.3. Características de las métricas

Antes de comenzar con las características de las métricas en general, es necesario acotar que estas pueden seguir varios enfoques, pueden estar orientadas al proceso, o al producto o proyecto.

Métricas sobre el proceso: Las métricas del proceso se recopilan en el curso de todos los proyectos y durante largos períodos, permiten, evaluar lo que funciona y lo que no, además de obtener un conjunto de indicadores de proceso que conduzcan a la mejora de los procesos en sí, a largo plazo. También permite adoptar una visión estratégica a la organización.

Métricas sobre el producto y el proyecto: Las métricas sobre el producto están orientadas a estimar las características del mismo antes de su desarrollo. Estas estimaciones se basan en el conocimiento que los desarrolladores adquieren a partir de datos obtenidos de proyectos anteriores. Las métricas del proyecto se emplean básicamente con dos fines: minimizar el tiempo de desarrollo y valorar la calidad del producto sobre una base actual, permiten: valorar el estado de un proyecto en curso, rastrear riesgos potenciales y. adoptar una visión táctica a la organización.

De manera independiente a su enfoque las métricas deben cumplir con características generales:

- Deben ser sencillas y fáciles de calcular.
- Empíricas e intuitivamente persuasivas: deben satisfacer las nociones intuitivas del desarrollador sobre el atributo del producto en evaluación.
- Consistentes y objetivas: deben presentar resultados sin ambigüedad.
- Consistentes en el empleo de unidades y tamaños: deben emplearse medidas que no conduzcan a extrañas combinaciones de unidades.
- Independientes del lenguaje de programación: no deben depender de la sintaxis ó semántica del lenguaje de programación.

- Un eficaz mecanismo para retroalimentación de calidad: deben proporcionar al desarrollador de software información que le lleve a un producto final de mayor calidad [4].

1.5.4. Tipos de métricas

En un proyecto de software, las métricas son analizadas y evaluadas por los administradores del software, pero a menudo las medidas son reunidas por los ingenieros de software.

Existen diferentes tipos de métricas, entre ellas se pueden encontrar:

- Métricas del Software: son las métricas que están relacionadas con el desarrollo del software como funcionalidad, complejidad, eficiencia.
- Métricas técnicas: estas métricas poseen un modelo de valoración entre cero (0) y cinco (5) y por decisión del equipo de trabajo se puede asumir una valoración en porcentajes. Se centran en las características del software por ejemplo: la complejidad lógica y el grado de modularidad. Se derivan de una relación empírica según las medidas contables del dominio de información del software y de evaluaciones de complejidad y miden la estructura del sistema, o sea, el cómo esta hecho.
- Métricas de productividad: se centran en el rendimiento del proceso de la Ingeniería del software. Es decir, qué tan productivo va a ser el software que se va a diseñar.
- Métricas orientadas a la persona: proporcionan medidas e información sobre la forma en que la persona desarrolla el software de computadoras y sobre todo el punto de vista humano de la efectividad de las herramientas y métodos. Son las medidas que se van a hacer del personal que hará el sistema.
- Métricas orientadas a la función: son medidas indirectas del software y del proceso en el cual se desarrollan. En lugar de calcular las LDC, las métricas orientadas a la función se centran en la funcionalidad o utilidad del programa. Emplean como un valor de normalización una medida de la funcionalidad que entrega la aplicación.
- Métricas orientadas a objetos: no proporcionan suficiente granularidad para la planificación y los ajustes de esfuerzo. Las siguientes son métricas sugeridas para proyectos desarrollados mayormente sobre bases orientadas a objeto: número de guiones de escenario, número de clases clave,

número de clases de apoyo, número promedio de clases de apoyo por clase clave y por ultimo el número de subsistemas.

- Métricas orientadas a casos de uso: el caso de uso se define en etapas tempranas del proceso de software lo que permite emplearlo en la estimación antes de iniciar las actividades significativas de modelado y construcción.

- Métricas para calidad del software: las métricas de calidad de software se enfocan sobre el proceso, el proyecto y el producto. Estas métricas las podemos dividir en dos grupos; el primero recolecta antes de la entrega del producto y el otro luego de haberlo entregado. Dentro de estas métricas es necesario destacar que se miden según las siguientes variables

  - Corrección: es el grado en que el software desempeña la función para la que fue creado. Su medida es número de defectos por KLDC.
  - Facilidad de Mantenimiento: es la facilidad para corregir un error, adaptar un programa a cambios, o mejorarlo si el cliente desea un cambio. Su medida es Tiempo Medio de Cambio (TMC).
  - Integridad: es la capacidad para resistir ataques, provocados o no, contra su seguridad, ya sea sobre programas, datos y documentos.
  - Facilidad de uso: se refiere a habilidad intelectual o física requerida para aprender a utilizar el sistema; es decir, amistad con el usuario.

- Métricas Bang: estas ayudan a evaluar el análisis y diseño, proporcionan medidas de la complejidad y ayudan a diseñar pruebas más efectivas. Estas métricas se dividen en: medidas del análisis, medidas de especificación y medidas de diseño.

  - Medidas del software según el análisis: el propósito es evaluar el conjunto de primitivas, es decir los elementos más bajos del análisis que no se pueden subdividir más, estos son:
    - Primitivas funcionales (PFu) evaluan el Nivel inferior.
    - Elementos de datos (ED) evaluan los Atributos de objetos.
    - Objetos (OB) evaluan Objetos de datos.
    - Relaciones (RE) evaluan Conexiones entre Objetos de Datos.
    - Estados (ES) evaluan el Estado y Diagrama de estado.
    - Transiciones (TR) evaluan el Número de transacciones y Diagrama de transición.

- Métricas de calidad de especificación: valoran y modelan los requisitos, en estas se definen varias variables, nr – número requisitos en una especificación, nf – número de requisitos funcionales y rnf – número de requisitos no funcionales (rendimiento). Estos se relacionan mediante la ecuación $nr = nf + rnf$.
- Métricas de diseño: Se dividen en los siguientes tres aspectos en materia de complejidad:
  - Complejidad estructural: se calcula mediante la función, $S(i) = Fout(i)$ donde $Fout(i)$ es la expansión del módulo i define la expansión como la cantidad de módulos inmediatamente subordinados al módulo i o como el número de módulos invocados por el módulo i.
  - Complejidad de los datos: se calcula de la siguiente manera, $D(i) = V(i)/[Fout(i) + 1]$ siendo $V(i)$ el número de variables de entrada y salida del módulo i.
  - Complejidad del sistema: calculada mediante la ecuación, $C(i) = S(i) + D(i)$.

- Métricas de punto de función de Albrecht: 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.

- Métricas del diseño arquitectónico: Las métricas de diseño de alto nivel se concentran en las características del programa especialmente en la estructura arquitectónica y en la eficiencia de los módulos, estas métricas no requieren de ningún conocimiento del trabajo interno de un módulo en particular, mientras que Card y Glass definen tres medidas de la complejidad del diseño, complejidad estructural, complejidad de los datos y complejidad del sistema, Henry y Kafura definen la métrica de complejidad MHK y Fenton sugiere las métricas de tamaño, profundidad y anchura.

- Métricas de diseño a nivel de componentes: Se concentran en las características internas de los componentes del software, incluyen medidas de cohesión, acoplamiento y complejidad del módulo que pueden ayudar a valorar la calidad de un diseño a nivel de componentes, estas métricas requieren conocimiento del trabajo interno del módulo

- Métricas de cohesión: Bieman y Ott desarrollaron un conjunto de métricas que proporcionan una

indicación de la cohesión de un módulo para cohesiones funcionales fuertes, cohesiones funcionales débiles y pegajosidad la cual constituye el grado relativo con el que las señales de unión juntan porciones de datos, estas métricas se definen con las medidas: porción de datos, muestras de datos, señales de unión, señales de superunión y pegajosidad.

- Métricas de acoplamiento: Dhama propuso una métrica para el acoplamiento de un módulo que combina el acoplamiento del flujo de datos y de control, acoplamiento global y acoplamiento de entorno proporcionando una indicación de la conectividad de un módulo con otros módulos, datos globales y el entorno exterior.

- Métricas de complejidad: Pueden emplearse para predecir la información crítica sobre la fiabilidad y mantenimiento de sistemas de software, realimentan la información durante el proyecto para ayudar a controlar la actividad del diseño y proporcionan una detallada información sobre los módulos para ayudar a resaltar las áreas de inestabilidad potencial.

- Complejidad ciclomática: Definida por McCabe proporciona una medida cuantitativa para probar la dificultad y una indicación de la fiabilidad última, esta métrica también proporciona una indicación cuantitativa del tamaño máximo del módulo.

- Métricas de diseño de interfaz: Métricas que proporcionan un visión interna de la calidad y facilidad de empleo de la interfaz, estas métricas son muy necesarias para asegurar que a los usuarios finales les gusta la interfaz y están satisfechos con las interacciones requeridas, Sears sugiere la conveniencia de la representación como una valiosa métrica de diseño para interfaces de usuario la cual se emplea para valorar diferentes distribuciones propuestas de interfaces y la sensibilidad de una representación en particular a los cambios en las descripciones de tareas.

- Métricas del código fuente: Usan un conjunto de primitivas que se pueden obtener una vez que se ha generado o estimado el código después de completar el diseño, estas medidas son usadas para desarrollar expresiones para la longitud global del programa, volumen mínimo potencial para un algoritmo, volumen real, nivel del programa, nivel del lenguaje así como el esfuerzo de desarrollo, tiempo de desarrollo y número esperado de fallos en el software.

- Métricas para pruebas: Se usan para descubrir errores en el módulo antes de integrarlo en un sistema, usando métricas obtenidas de medidas de Halstead se puede estimar el esfuerzo de las

pruebas, a medida que se van haciendo las pruebas se puede obtener una indicación de la compleción de las mismas a través de la amplitud de las pruebas, la profundidad y los perfiles de fallos.

- Métricas del mantenimiento: El estándar IEEE propone una métrica diseñada explícitamente para actividades de mantenimiento, el índice de madurez del software proporciona una indicación de la estabilidad de un producto basada en los cambios que ocurren con cada versión y puede emplearse como métrica para la planificación de las actividades de mantenimiento del software.

1.6. Herramientas que soportan la estimación y métricas

La Ingeniería del software es un área de estudios muy reciente, que ha devenido de la necesidad de organizar la joven Industria del software, que se potencia diariamente.

En la actualidad existen muchos procesos de desarrollo de software. Con el pasar de los años, la Ingeniería de software ha introducido y popularizado una serie de estándares para medir y certificar la calidad, tanto del sistema a desarrollar, como del proceso de desarrollo en sí. Se han publicado muchos libros y artículos relacionados con este tema, con el modelado de procesos del negocio y la reingeniería. Ha surgido un número creciente de herramientas automatizadas para ayudar a definir y aplicar un proceso de desarrollo de software efectivo.

Para continuar primeramente se debe comprender el significado de qué es una herramienta:
Herramienta: instrumento que ayuda a realizar un trabajo.
Más concretamente enfocado al área de la informática una herramienta representa:
Funciones que ofrece un programa a través de una barra con íconos, en ocasiones que representan los distintos recursos del software para realizar una tarea determinada.

Las herramientas de ayuda al desarrollo de sistemas de información, surgieron para intentar dar solución a los problemas inherentes a los proyectos de generación de aplicaciones informáticas: plazos y presupuestos incumplidos, insatisfacción del usuario, escasa productividad y baja calidad de los desarrollos. Algunas de estas herramientas se dirigen principalmente a mejorar la calidad, como es el caso de las herramientas *Computer Aided Software Engineering* (CASE). Otras van dirigidas a mejorar la productividad durante la fase de construcción, como es el caso de los lenguajes de cuarta generación (4GL-Fourth Generation Language) [12].

Se pudiese decir que todas las herramientas tienen al compararse unas con otras marcadas diferencias, algo que es muy cierto, pero no sólo se deben ver como rivales sino más bien como medios de ayuda, por la posibilidad de combinar funcionalidades de unas con otras.

1.6.1. Funcionalidades básicas

Existen ciertas funcionalidades que deben cumplir las herramientas de ayuda al desarrollo, a continuación se menciona la mayoría de estas:

- Ayuda de la herramienta: es una ayuda incorporada al programa, brindando información sobre el uso de los componentes de la propia herramienta, de fácil acceso y con utilidades de búsqueda de temas o palabras claves. Una ayuda interactiva evita el manejo de manuales.
- Diccionario de datos: descripción lógica de los datos para el usuario. Reúne la información sobre los datos almacenados en una base de datos, tales como descripción, significado, estructura, consideraciones de seguridad y uso de aplicaciones, etc.
- Sistema de información–SI: conjunto de elementos físicos, lógicos, de comunicación, datos y personal que, interrelacionados, permiten el almacenamiento, transmisión y proceso de la información.
- Workbench: es una interfaz gráfica que permite modelar procesos y datos. Está basada en el mismo principio de la programación visual: no se emplea lenguajes, sino íconos, los cuales no son dibujos del tipo flow, sino objetos que se almacenan en el repositorio.

Basadas en sus funcionalidades y características específicas, de cada una de ellas se ha llegado a determinar una clasificación bien marcada en grupos diferentes, refiriéndose a los mismos bajo el nombre de tipos de herramientas, los cuales son mostrados a continuación.

1.6.2. Tipos de herramientas

- Herramientas de gestión de proyectos: facilitan la labor de planificación y seguimiento de tareas y recursos, para conseguir que el proyecto logre sus objetivos en plazo y presupuesto.
- Herramientas de gestión de la configuración: identifican y definen los elementos de un sistema, controlan los cambios y las versiones de dichos elementos.

- Herramientas de ayuda en las pruebas: facilitan la tarea de probar el equipo lógico desarrollado, para asegurar que cumple las especificaciones del diseño.
- Herramientas de control de calidad: dentro de este apartado podrían englobarse la gran mayoría de las herramientas citadas aquí, ya que de una u otra forma todas van dirigidas a una mejora de la calidad de las aplicaciones. No obstante, se hace referencia bajo esta denominación a las herramientas que se centran en la fase de análisis, diseño y construcción.
- Herramientas de revamping: sirven para maquillar una aplicación existente en modo carácter, mediante una interfaz gráfica de usuario sobre PC.

Dentro de la etapa de gestión de proyectos se llevan a cabo las actividades de estimación y medición, las mismas son procesos que también pueden ser realizados de manera automática, apoyándose en la utilización de diversas herramientas que garanticen un trabajo óptimo y a la vez ágil.

1.6.3. Herramientas de estimación

Permiten estimar costo y esfuerzos. Aunque existen muchas herramientas automáticas de estimación, todas exhiben las mismas características generales y todas requieren de una o más de las siguientes clases de datos:

- Una estimación cuantitativa del tamaño del proyecto o de la funcionalidad.
- Característica cualitativa del proyecto tales como la complejidad, la fiabilidad requerida o el grado crítico del negocio.
- Alguna descripción del personal y entorno de desarrollo.

A partir de estos datos el modelo implementado por la herramienta automática de estimación proporcionará estimaciones del esfuerzo requerido para llevar a cabo el proyecto, los costos, la carga del personal, la duración y en algunos casos la planificación temporal del desarrollo y riesgos asociados.

El planificador del proyecto de software tiene que estimar 3 cosas antes de que comience el proyecto: cuánto durara, cuánto esfuerzo requerirá y cuánta gente estará implicada. Además el planificador deberá predecir los recursos de hardware y software que va a requerir y el riesgo implicado [8].

Existen herramientas automáticas de estimación que implementan técnicas de descomposición o modelos empíricos y están provistas de entornos gráficos, interfaz interactiva y uso estandarizado que hacen de ellas una buena opción. Además permiten describir características de la organización tales como experiencia, entorno, etc. y el software a desarrollar para que a partir de estos datos, sea posible obtener estimaciones de costo, tiempo y esfuerzo. Las más recomendadas son las de código abierto por las facilidades de reutilización del mismo y las que realizan el trabajo de forma ágil.

Herramientas de estimación libres

En la investigación, se hará énfasis en el estudio de las herramientas libres, por el gran nivel de aceptación que han alcanzado a nivel mundial, y su perfecta adaptabilidad a las condiciones de trabajo, no propietarias y orientadas al software libre existentes en la facultad 10 de la Universidad de las Ciencias Informáticas.

- Construx Estimate: Ayuda a mejorar las capacidades de estimación de software, aprovecha una mezcla de modelos de estimación probados para predecir el esfuerzo, presupuesto y programa para proyectos basados en estimaciones de tamaño. Esta herramienta viene calibrada con datos de industria, pero es más poderosa cuando es calibrada con datos de la organización en particular. Como parte de la misión por adelantar el arte y la ciencia de la ingeniería de software comercial, se proporciona Construx Estimate™ versión 2.0 con descarga libre y licencia limitada.
- USC Cocomo II: Programa gratuito que implementa el modelo Cocomo II de estimación de tamaño y esfuerzo para proyectos de software, aunque es una herramienta poco amigable consta con una interfaz gráfica sencilla y práctica, presenta una ayuda personalizada que explica paso a paso cómo trabajar con ella.

Herramientas de estimación ágil

- Hoja Excel para Scrum: Hoja de cálculo para gestionar el trabajo en cada sprint: tareas, asignación, estado y tiempos. Genera de forma automática los gráficos para el seguimiento de esfuerzo y tareas. Herramienta creada para proyectos simples y pequeños.
- iceScrum: Con la misma interfaz para todos los roles, ofrece las opciones de operación, consulta, estimación de historias de usuario activas o no, según el usuario sea propietario del producto, gestor, equipo o interesado. Incluye listas de historias de usuario o backlog, de asuntos, de problemas

y de pruebas; un chat en línea y un juego de cartas con el que el equipo puede hacer estimación de poker de las historias propuestas en el backlog.

De igual manera a que existen herramientas que ayudan a calcular costo, esfuerzo, tiempo, etc. existen otras dedicadas a la medición, las cuales se enfocan fundamentalmente en obtener un vistazo de la calidad que va obteniendo el producto.

1.6.4. Herramientas para la medición de métricas

Determinar la calidad de un producto de software es algo complicado, más aún si su proceso de desarrollo es desconocido para quien lo evalúa o si las entregas son muy numerosas.

La manera más rápida y fiable de tener visibilidad sobre el producto es recogiendo métricas del mismo, es decir, midiendo el producto más que los procesos que se han seguido para desarrollarlo, y que dicho así parece una trivialidad pero que es la base desde la que muchos argumentan que apoyarse sólo en modelos de calidad de procesos como CMMI no da plena seguridad sobre la calidad del producto, que es en realidad la importante en situaciones como esta.

Pero realizar mediciones software de manera periódica sin una infraestructura adecuada puede ser tan costoso en tiempo que puede hacer que no tenga sentido medir.

Hoy en día es posible automatizar gran parte de la medición de la calidad del producto, tales como el análisis estático de código, mantenibilidad, código duplicado, complejidad ciclomática, cobertura de las pruebas, la compilación del proyecto, el lanzamiento de las pruebas, el control de incidencias, el estilo de programación, las excepciones no controladas, etc. Además se han creado herramientas de software libre muy fiables, posibilitando infraestructuras altamente productivas que permiten evaluar la calidad del software de manera diaria y totalmente automatizada. Pero no todo es color de rosas, la automatización genera a veces tanta información en tan poco tiempo que en ocasiones esto puede ser peor que no tener información.

A partir de todo esto se puede afirmar que en la actualidad es realista y necesario, disponer de una infraestructura automática que implemente un modelo de medición de la calidad, en un período razonable y proporcionando un gran beneficio [9].

Nunca se debe perder de vista el objetivo final, que será, alinear objetivos de negocio con planes de mejora, incrementar la productividad y rentabilidad del desarrollo, aumentar la calidad del software y la satisfacción del usuario y crear una ventaja competitiva respecto a la competencia.

- METRE C Software Metrics Tool: Es una utilidad de línea de comandos que determina métricas de software para módulos C, programas y proyectos. Puede brindarle métricas útiles tales como: complejidad ciclomática de McCabe (cyc), complejidad extendida de Myers (exc), longitud de programa de Halstead (pln), vocabulario de programa (pvc), volumen de programa (pvl), nivel de programa (plv), esfuerzo de programación (pef), contenido de inteligencia (inc), tiempo de programación en horas (ptm), nivel de lenguaje (llv), número de fuente (scl), código (cdl), comentarios (cml), líneas en blanco (bll), número de ejecutables (exs), declaraciones (dcs), declaraciones de preprocesador (pps), puntos de función disparados por atrás (fnp), máximo control de profundidad (mcd), contador identificador (idc), longitud (idl), número de funciones (fnc) y módulos (mdc) Específicamente, Metre reporta estas métricas para cada función, módulo y proyecto.

- DataDrill: Este software permite que las empresas desplieguen rápidamente las métricas de un proyecto y el sistema de administración que entrega información crítica a los administradores cada vez que lo necesiten. Para sistemas, software e IT administradores, DataDrill entrega las mejores prácticas fuera de lo común, así como una buena configuración para apoyar las necesidades propias de una compañía.

Herramientas ágiles para la medición de métricas

- ScrumDesk: Herramienta intuitiva de administración de proyectos para el desempeño Scrum de equipos de cualquier tamaño distribuidos en todo el mundo. Este software no es sólo para administradores de proyecto, pues conecta equipos de proyecto y miembros del equipo con los clientes y los gestores. Cualquiera puede identificar fácilmente el estado del proyecto usando informes que muestran las métricas usadas típicamente en Scrum. ScrumDesk proporciona en cualquier momento un acceso sencillo a herramientas de colaboración como mensajería, llamadas por Internet, correos electrónicos, páginas web y sistemas de seguimiento de bugs.

- Sprintometer: Programa gratuito, contenido en un único fichero ejecutable, que no necesita instalación, para gestión, medición y seguimiento de programas con modelos ágiles tipo XP o Scrum.

1.7. Conclusiones

En el presente capítulo se realizó una pequeña descripción de la metodología SXP, haciendo énfasis en cada una de sus fases, actividades, roles y artefactos, se hizo referencia a los diferentes modelos de calidad, centrándose en el modelo CMMI y se llevó a cabo un estudio detallado de las diferentes técnicas de estimación y métricas más utilizadas a nivel mundial en la actualidad, así como un levantamiento de algunas de las herramientas capaces de sustentar dichas actividades adaptables al proceso de desarrollo de software de la facultad.

Capítulo 2

Propuesta de Técnicas de Estimación y Métricas

2.1. Introducción

El desarrollo tecnológico, especialmente de las Tecnologías de la Información y las Comunicaciones, entre las que se destaca Internet ha propiciado el auge de la llamada Sociedad de la Información o Sociedad del Conocimiento, el mismo va imponiéndose a la Sociedad Industrial y cambiando poco a poco las reglas del juego, ya que la incorporación de estas tecnologías en todos los procesos productivos, ciertamente facilita la inserción a los mercados globales, donde la intensa competencia obliga a reducir costes y a ajustarse de manera casi inmediata a las cambiantes condiciones del mercado.

De aquí, que las empresas u organizaciones tienen que ser más eficaces y más eficientes para poder seguir compitiendo en esta nueva sociedad de la información. La eficacia tiene que ver con los resultados y está relacionada con el logro de los objetivos, mientras que la eficiencia, en cambio, se enfoca a los recursos y a su utilización de la mejor manera posible. Los procesos de mejora de procesos, enfocan todo su esfuerzo en crear grandes expectativas concernientes al tópico, en el presente capítulo, se brinda en general un estudio del modelo de calidad CMMI, y sus principales etapas o fases, y la propuesta a realizar en cuanto a las técnicas de estimación y métricas que sean adaptables a los procesos de SXP. Se precisan los elementos principales sobre dicha propuesta, y los aspectos fundamentales tanto de la técnica de estimación seleccionada como de las métricas aplicables.

2.2. Estado actual del uso de métricas y estimación en el grupo Unicornios.

El grupo Unicornios, ha utilizado para la documentación de la ingeniería de software la metologia ágil SXP a partir de su creación en anos anteriores, dicha metodología ha sido utilizada de manera que ha aportado notables beneficios al grupo y a la comunidad en general, mas no posee, como se explica anteriormente en la investigación, alguna técnica para la estimacion de costo y esfuerzo, ni métricas que se pudieran utilizar para medir el software. Con la falta de alguna técnica concreta definida para la estimacion, la dirección del proyectos opto por aplicar, temèporalmente la opinión de expertos para esta actividad, mientras que en la actividad de la medicion a través de métricas no han existido hasta ahora, ninguna propuesta o solución.

2.3. Situación de la estimación y las métricas

En estudios realizados por el Centro de Estudios de Ingeniería de Sistemas del Instituto Superior Politécnico José Antonio Echeverria (ISPJAE) en la industria de software cubana se detectaron varios problemas tales como la ausencia de la definición de los procesos de la empresa siguiendo los principios de la Ingeniería y la gestión de software, la necesidad de incrementar el número de encuentros predefinidos para hacer pedidos de cambios, por citar otro ejemplo, el hecho de no planificar correctamente, lo cual impide que exista algún tipo de seguimiento, por lo que es imposible ver el estado del progreso del proyecto. La Universidad de las Ciencias Informáticas no está exenta de esta situación, hoy, la mayoría de los desarrolladores de software en la UCI, díganse líderes de proyectos, programadores, diseñadores, etc. no planean ni registran su trabajo y no miden y administran la calidad de los productos.

El desconocimiento de las técnicas de estimación y las métricas en general ha sido un factor de peso en la evaluación del software. Se han utilizado, en materia de estimación, técnicas basadas en algoritmos matemáticos, y como métricas se le ha dado prioridad a aquellas que presentan de manera general una tendencia tradicional.

Con el auge del desarrollo de software de pequeño y mediano formato en el país y la necesidad de orientar los procesos de producción hacia soluciones y documentaciones ágiles, estos métodos que han

sido utilizados hasta el momento, quedan de cierta manera obsoletos, por lo que se hace fehaciente la importancia de definiciones de nuevas técnicas de estimación y métricas aplicables a procederes ágiles en general.

Como se menciona en el primer capítulo de la investigación realizada, la estimación atraviesa una de sus mejores etapas en la actualidad, pues independientemente de haberse convertido en una de las actividades más importantes de la etapa de planificación, se ha creado en la comunidad del software internacional una conciencia basada en los numerosos resultados alcanzados con una buena práctica en este proceso.

Tema de debate continuado, ha sido la controversia entre los enfoques a seguir a la hora de realizar una buena estimación, las metodologías tradicionales y las ágiles, como se ha evidenciado a lo largo de la investigación difieren de manera radical en cuanto a maneras tanto de ver la estimación como de cómo tratarla al realizarla.

Los equipos numerosos, dígase de grandes compañías productoras de software a lo largo de la revolución que ha habido en este campo, han optado por estimaciones tradicionales basadas en modelos matemáticos, mientras que las empresas de menor envergadura prefieren los métodos ágiles. Actualmente se ha generalizado la tendencia a utilizar los últimos, debido a su gran rapidez, su robustez y sus grandes y seguros resultados alcanzados, esto ha sido en gran medida debido a la popularidad que han alcanzado metodologías ágiles como SCRUM y XP. A continuación, la propuesta de una técnica de estimación a integrarse en la metodología SXP [23].

2.4. Propuesta de técnica de estimación aplicable a SXP

Lo habitual, cuando se estima en un proyecto, es hacer una primera estimación gruesa la primera vez que se necesita conocer la pre-factibilidad de un proyecto. Luego de un primer relevamiento del proyecto, se debe estimar con mayor precisión, basados en el mayor conocimiento que se tiene del producto a construir y del marco temporal y de recursos del proyecto. Cuando se comienza a ejecutar el proyecto, el responsable del mismo ante la dirección, deberá volver a estimar, dado que pueden haber cambiado el marco temporal y los recursos disponibles.

A medida que el proyecto avanza, es deseable seguir estimando, para poder corregir desvíos o prever escenarios de salida del proyecto distintos a los previstos en el inicio, si así fuese necesario. Finalmente, al final del proyecto, se debe hacer un análisis de las estimaciones en cada etapa, de modo tal que este análisis sirva para futuros proyectos de características similares.

De manera general en un proyecto se estima para conocer costos y plazos, mas dicho de esta manera se le pueden restar importancia a las diferentes variables existentes en esta importante actividad. Por eso, se puede decir a grandes rasgos que se estima, el tamaño, para comparar proyectos y medir productividad, el esfuerzo, para comparar proyectos y tener una medida ideal del costo, el costo, para el costeo en general y, a veces, para establecer el precio, el plazo, para poder informar al cliente cuándo podrá contar con el producto, y finalmente el alcance, para que el cliente sepa qué esperar en un plazo y costo dados. Por lo tanto, las estimaciones no se hacen para consumo exclusivo del área comercial, sino que se utilizan constantemente y en diferentes situaciones.

2.4.1. Cómo estimar

En la presente investigación, se persigue proponer una técnica de estimación para la metodología tratada, de manera que permita gestionar u organizar las estimaciones obtenidas por el equipo para llegar finalmente a un consenso, en este epígrafe se mencionarán, algunas de las cuestiones a tener en cuenta para realizar las estimaciones particulares, que servirán de base para la utilización de la técnica de estimación propuesta.

La estimación y planificación temporal de un proyecto de software requiere: experiencia, buena información histórica, y coraje de confiar en las métricas, para obtener buenos resultados, debido a que cada proyecto es diferente, cada empresa es diferente y el contexto de los sistemas que se desarrollan cambian constantemente, no existe un método que se adapte completamente a cualquier proyecto, así la estimación debe ajustarse localmente.

Hay cuatro factores que influyen significativamente en las estimaciones:

- La complejidad del proyecto.
- El tamaño del proyecto.

- El grado de incertidumbre estructural.
- Disponibilidad de información histórica.

Para el mejor trato de estos factores es de suma importancia manejar con claridad el ámbito del software y su factibilidad, estos, no son más que las funciones y características, datos de entrada, salida y el contenido que se presentan a los usuarios finales.

El ámbito se define mediante una descripción narrativa, o el desarrollo por parte de los usuarios de un conjunto de casos de uso, mientras que la factibilidad del software debe estar orientada en base a, tecnología, finanzas, tiempo y recursos. La Figura 2.1 muestra un mapa conceptual, que explica a grandes rasgos los pasos a seguir para lograr una buena estimación.

Figura 2.1: Mapa conceptual

2.4.2. Técnica de estimación a incluir en la metodología SXP

La tendencia ágil dentro de los procesos de desarrollo de software evidencia la necesidad de sencillez en todas las etapas del proceso, la estimación, por supuesto, debe demostrar una perfecta armonía entre eficiencia y rapidez, Planning Poker, a nuestra manera de ver y como propuesta definitiva para su integración a la metodología SXP, es el mejor ejemplo de lo antes planteado.

El anclaje, la redundancia, y a partir de estos las estimaciones demoradas y costosas, son los principales problemas a los cuales se enfrenta la estimación en general en estos momentos. La técnica elegida es un compendio entre tres pilares muy importantes a la hora de estimar, la colaboración a través de todo el equipo, la creación de estimaciones por consenso en vez de tener a un único individuo estimando y la exposición de los problemas de manera temprana, a través de discusiones sobre cada historia de usuario.

El Planning Poker es muy sencillo, se presentan los requisitos a estimar uno por uno haciendo una descripción de los mismos. Luego se procede a discutir aquellos detalles más relevantes o que no hayan quedado claros. Tras este período de discusión cada uno de las personas implicadas en el proceso de estimación elije de su mazo de cartas, numeradas según la serie de Fibonacci, la carta que representa su estimación del trabajo que implica implementar el requisito que se está discutiendo.

Las estimaciones son privadas hasta que todo el mundo cuenta con una estimación. En ese momento, todas las estimaciones se hacen públicas, haciéndose de esta manera para evitar que las estimaciones de unas personas arrastren o influyan en las de otras. Si existe dispersión entre las estimaciones se vuelve al discutir el requisito y se vuelve a realizar el proceso de estimación.

Generalmente la dispersión entre las estimaciones es síntoma de que la información que manejan parte de los involucrados en el proceso de estimación no es completa o exacta. La segunda ronda de discusión permite aclarar aquellos puntos oscuros, diferencias de criterio y desvelar información que pueda no ser obvia sobre el requisito, de tal manera que en la siguiente ronda de estimación, la dispersión de las estimaciones será mucho menor y se podrá llegar fácilmente a un consenso.

De esta manera es fácil estimar los requisitos de una manera democrática, ágil y rápida y que lleva a estimaciones respaldadas por todos los involucrados y basadas en consensos, en definitiva, estimaciones que todo el mundo puede asumir como suyas y que todo el mundo respetará. El funcionamiento es muy simple: cada participante dispone de un juego de cartas, y en la estimación de cada tarea, todos vuelven boca arriba la combinación que suma el esfuerzo estimado.

Cuando se considera que éste es mayor de 10 horas, o del tamaño máximo considerado por el equipo para una tarea, se levanta la carta con el valor infinito.

Cada equipo u organización puede utilizar un juego de cartas con las numeraciones adecuadas a la unidad de esfuerzo con la que trabajan, y el tamaño máximo de tarea que se va a estimar.

Lo relevante no es el número de cartas, la unidad de medida empleada, o si el tamaño máximo de tarea debe ser 5, 7 ó 10 días, sino trabajar con el modelo más apropiado al equipo, respetando los principios siguientes:

- No definir tareas demasiado grandes, porque dificulta la precisión de las estimaciones y la identificación de riesgos. Un criterio válido, si el equipo no dispone de experiencia previa, es descomponer en otras menores las que tengan una duración mayor de 7 días ideales de trabajo.
- No definir tareas con duraciones inferiores a medio día ideal de trabajo.
- Utilizar la misma unidad de medida: story points, días, horas, en todos los gráficos y backlogs.

Basado en el hecho de que al aumentar el tamaño de las tareas, aumenta también el margen de error, se ha desarrollado una variante que consiste en emplear sólo números de la sucesión de Fibonacci para realizar las estimaciones, de forma que:

- El juego de cartas está compuesto por números de la sucesión de Fibonacci.
- La estimación no se realiza levantando varias cartas para componer la cifra exacta, sino poniendo boca arriba la carta con la cifra más aproximada a la estimación.

De la mano con esta técnica de estimación, y como complemento a la metodología SXP, se propondrán a continuación un conjunto de métricas para la medición de software.

2.5. Propuesta de métricas de software aplicables a SXP

Muchos investigadores han intentado desarrollar una sola métrica que proporcione una medida completa de la complejidad del software. Aunque se han propuesto docenas de métricas o medidas, no todas proporcionan suficiente soporte práctico para su desarrollo. Algunas demandan mediciones que son demasiado complejas, otras son tan esotéricas que pocos profesionales tienen la esperanza de entenderlas, y otras violan las nociones básicas intuitivas de lo que realmente es el software de alta calidad. Es por eso que se han definido una serie de atributos que deben acompañar a las métricas efectivas de software.

Por otra parte, la mayoría de los desarrolladores de software todavía no miden a pesar de que las mediciones pueden ayudar a lograr que su trabajo cada día sea mejor y con una mayor calidad. El desconocimiento de las métricas por parte del equipo de trabajo hace que establezcan un rechazo hacia ellas. Casi todos los desarrolladores piensan que sólo realizando las pruebas de caja negra y caja blanca se obtiene un producto de calidad emitiendo observaciones y sugerencias en cada uno de los errores encontrados hasta llegar al punto donde ocurre el error dando poca importancia a las métricas que dan una medida cuantitativa de la calidad que presenta el producto.

Con la utilización de las métricas se puede determinar si el producto que se está desarrollando está mejorando y si posee o no la calidad requerida. En un intento por esclarecer la importancia de las métricas muchos desarrolladores oponen resistencia, no entienden para qué sirven y se preguntan por qué es tan necesario medir el proceso de desarrollo de software y el producto que produce. Los rigores del trabajo diario de un proyecto del software no dejan mucho tiempo para pensar en estrategias. Los líderes del proyecto de software están más preocupados por otros aspectos como: desarrollo de estimaciones significativas del proyecto, terminar el producto a tiempo y dejan a un lado la calidad del producto. Mediante el uso de la medición para establecer una línea base del proyecto cada uno de estos asuntos se hace más fácil de manejar. La línea base sirve como un lineamiento para la estimación. Las métricas proporcionan beneficios a nivel de proyecto, estratégico y técnico.

La propuesta de un conjunto de métricas para la metodología SXP es uno de los principales objetivos de la investigación, las métricas en general, como se ha podido comprobar a lo largo de este trabajo, constituyen incluso cuando existen estándares utilizados por numerosos equipos, implementaciones particulares de los diferentes equipos de desarrollo, esto independientemente de si son o no, adaptables a procesos ágiles, pues en materia de métricas sólo se pueden seguir algunos patrones que evidencien un comportamiento ágil, ya que en esencia son todas implementaciones matemáticas.

Como ha quedado demostrado en el anterior capitulo, las métricas son aquellas funciones que reciben datos del software, y muestran como salida un valor único que debe ser interpretado como el grado que posee el software de un atributo dado que afecta a la calidad. Es necesario no desfallecer en los intentos de implementarlas y usarlas correctamente, si de esta manera se hace, estas deben permitir:

- Definir cuantitativamente el grado de éxito o fracaso para un producto, de un proceso o las tareas

realizadas por una persona.

- Identificar y cuantificar las mejoras, su ausencia o la degradación del producto, del proceso o de las tareas realizadas por las personas.
- Hacer entendibles y útiles las decisiones técnicas y las de gestión.
- Identificar tendencias.
- Realizar estimaciones cuantificables y con sentido.

2.5.1. Métricas a incluir en la metodología SXP

El uso de métricas en los productos de software proporciona un control sobre el proceso de monitoreo de proyectos, dando estimaciones sobre el estado de las diferentes actividades y del proceso en general. Da la posibilidad de realizar estudios comparativos entre diferentes proyectos y muestra el estado de la mejora de procesos en la institución. Como propuesta de la investigación se mencionarán algunas de las métricas utilizadas por el modelo de calidad CMMI, entre otras escogidas, principalmente porque evidencian un enfoque ágil, fácilmente adaptables a la metodología tratada.

- Las métricas de productividad serán usadas para medir el esfuerzo real de un proyecto. Con estas métricas se determina si es correcto el número de trabajadores asignados a esa tarea y el tiempo:
  Productividad (E)
  $E =$ esfuerzo
  $P =$ número de personas
  $T =$ cantidad de tiempo
  Se calcula $E = P * T$
  Con esta métrica se puede definir que es lo mismo dos personas trabajando tres meses, que tres personas trabajando dos meses. Utilizada en el monitoreo de estado.

- Para definir con precisión la duración del proyecto:
  Estimación de plazo (EP).
  $EP =$ precisión en la estimación de plazo
  $DR =$ duración real
  $DE =$ duración estimada

Se calcula $EP = DR/DE$

Utilizada en el monitoreo de estado.

- Usada para determinar la desviación existente del tiempo real con respecto al planificado. Con este resultado se decide si se necesita replanificar:

  Desviación del tiempo (DT).

  $DT =$ desviación del tiempo total de desarrollo respecto al tiempo total planificado

  $TT =$ tiempo total de desarrollo

  $TD =$ tiempo usado de desarrollo del proyecto hasta el momento

  $TP =$ tiempo planificado para las actividades por realizar en el proyecto

  $TPI =$ tiempo total planificado inicialmente

  Se calcula $TT = TD + TP DT = TT - TPI$

  Si se obtiene $DT = 0$ entonces el proyecto está cumpliendo todo en su plazo de tiempo planificado.

  Si $DT > 0$ entonces es necesario replanificar las actividades del proyecto porque hay un retraso en el tiempo.

  Si $DT < 0$ significa que el tiempo en que se cumplieron las actividades fue menor al planificado, entonces se dice que hay una holgura de tiempo y esta puede ser usada más adelante en caso de que exista algún retraso. Utilizada en el monitoreo de estado.

- Usada para darle un nivel de prioridad al riesgo y determinar cuánto problema puede traer:

  Riegos (R).

  $R =$ exposición al riesgo

  $I =$ impacto sobre el proyecto en una escala de 1 a 5 donde:

  $P =$ probabilidad de que ocurra

  Se calcula $R = I * P$

  El riesgo y la prioridad van a ser directamente proporcional si uno aumenta el otro también. Según la prioridad que presente cada riesgo será el orden de la mitigación de este. Utilizada en el monitoreo de riesgo.

- Para determinar la posibilidad de que el proyecto cumpla con los compromisos requeridos:

  Cumplimiento de compromisos (CC).

  En el caso de que el cumplimiento con el compromiso dependa de actividades a realizar se determinará de la siguiente forma:

$CAi =$ cantidad de actividades necesarias incumplidas hasta el momento de la revisión

$CAn =$ cantidad de actividades necesarias para cumplir con el compromiso hasta la revisión

$CAr =$ cantidad de actividades necesarias realizadas hasta el momento de la revisión

Se calcula $CAi = CAn–Car$

Si $CAi = 0$ entonces todas las actividades necesarias para que se cumpla el compromiso fueron realizadas y éste tiene una gran probabilidad de cumplirse.

Si $CAi > 0$ hubo un incumplimiento en las actividades y es posible que no se cumpla con el compromiso.

Si el cumplimiento del compromiso depende de la asignación de recursos se determinará de la siguiente forma:

$X =$ recursos no disponibles de los necesarios para cumplir el compromiso

$XN =$ recursos necesarios para cumplir el compromiso

$XD =$ recursos disponibles de los necesarios para cumplir el compromiso

Si $RN = 0$ entonces es posible que se cumpla el compromiso ya que existen todos los recursos que requiere.

Si $RN > 0$ es posible que el compromiso no se cumpla porque no se cuenta con todos los recursos necesarios.

Se calcula $X = XN–XD$

Utilizada en el monitoreo de compromisos.

- Para determinar la desviación existente del costo con lo que se ha panificado realmente:

  Desviación de Costo (CD).

  $IC =$ índice de actuación de costo

  $PP =$ presupuesto planificado

  $CR =$ costo real

  $CTP =$ costo total planificado

  Se calcula $IC = PP + CRCD = IC - CTP$

  Si $CD = 0$ esto significa que se está cumpliendo con el presupuesto planificado.

  Si $CD > 0$ se necesita replanificar porque hubo una desviación de costo al planificado y mientras mayor sea éste mayor será la desviación.

  Si $CD < 0$ entones se cuenta con una reserva monetaria que puede ser usada en otros momentos.

Utilizada en el monitoreo de costo.

- Para conseguir una estimación exacta de lo que se ha gastado en el proyecto, con ella se pueden comparar los resultados de cada fase para detectar desviaciones significativas:
  Porciento gastado del presupuesto general (PG).
  HF : costo hasta la fecha
  PPT : presupuesto total
  Se calcula $PG = (HF * 100)/PPT$
  Esta métrica se utiliza en el monitoreo de costo.

Además de las métricas de CMMI escogidas y con el objetivo de ampliar el marco de la medición del software se proponen las siguientes métricas expuestas por Pressman para la medición de diferentes aspectos de gran importancia, no solo en la etapa de planificación sino en todas las etapas del proceso en general.

- Para predecir el tamaño de un sistema que se va a obtener de un modelo de análisis:
  Métricas basadas en la función (PF)
  Cuenta-total: suma de todas las entradas PF obtenidas
  $Fi(i = 1ai = 14)$: valores de ajuste de complejidad.
  Se calcula PF = $cuenta - total * (0,65 + 0,1 * Sum[Fi])$

- Utilizadas para valorar la calidad del modelo de análisis y la especificación de requisitos:
  Calidad de la especificación (CE)
  Nr = requisitos en una especificación
  Nf = requisitos funcionales
  Nnf = requisitos no funcionales
  Se calcula $Nr = Nf + Nnf$ Especifidad (ausencia de ambigüedad) de los requisitos (ER):
  Nui = número de requisitos para los que todos los revisores tuvieron interpretaciones idénticas
  Se calcula $ER = Nui/Nr$
  Mientras más cerca de 1 esté el valor de ER menor será la ambigüedad. compleción de los requisitos funcionales (CRF):
  Un = número de requisitos únicos de función

Ni = número de entradas definidas o implicadas por la especificación

Ns = número de estados especificados

Se calcula $CRF = Un/(Ni * Ns)$

Esta relación mide el porcentaje de funciones necesarias que se han especificado para un sistema.

grado de validación de los requisitos (GVR):

Nc = número de requisitos validados como correctos

Nnv = número de requisitos que no se han validado

Se calcula $GVR = Nc/(Nc + Nnv)$

Esta relación trata los requisitos no funcionales.

- Utilizadas para conocer la estructura arquitectónica y la eficiencia de los módulos:

  Diseño arquitectónico: métrica de Henry y Kafura (MHK)

  $Longitud(i)$ = número de sentencias en lenguaje de programación en el modulo i

  $Fin(i)$ = concentración del modulo i

  $Fout(i)$ expansión del modulo i

  Se calcula $MHK = longitud(i) * [Fin(i) + Fout(i)]2$

  Un aumento en la MHK conduce a una mayor probabilidad de que aumente el esfuerzo de integración y pruebas del módulo.

- Para la Visión interna de la calidad y facilidad de empleo de la interfaz:

  Diseño de interfaz: Conveniencia de la representación (CR)

  K = transición específica de una entidad de representación a la siguiente al realizar una tarea específica

  CRO = costo de la representación óptima

  CRP = costo de la representación propuesta. Para asignar costos a cada secuencia de acciones:

  C = costos

  FT = frecuencia de transición

  CT = costo de transición

  C = $Sum[FT(k) * CT(k)]$

  Esta relación se da con todas las transiciones de una tarea en particular o conjunto de tareas requeridas para conseguir alguna función de la aplicación.

  Se calcula $CR = 100 * [(CRO)/(CRP)]$

donde: $CR = 100$ para una representación óptima. Para calcular la representación óptima de una interfaz de usuario:

N = posiciones

K = entidades de representación

NPD = número posible de distribuciones

Se calcula $NPD = [N!/(K! * (N - K)!] * K!$

A medida que crece el número de posiciones de representación, el número de distribuciones posibles se hace muy grande.

- Utilizadas en la etapa de pruebas:

  Pruebas

  Para descubrir errores en el módulo antes de integrarlo en un sistema

  V = volumen de un programa

  NP = nivel del programa

  e = esfuerzo de la ciencia del software

  $NP = 1/[(n1/2) * (N2/n2)]$

  Dando lugar a: $e = V/NP$

- Utilizadas para a partir del número de operadores y operandos presentes en el código valorar la calidad del programa:

  Código fuente

  Para determinar la longitud global del programa:

  $n1$ = número de operadores diferentes que aparecen en el programa

  $n2$ = número de operandos diferentes que aparecen en el programa

  $N1$ = número total de veces que aparece el operador

  $N2$ = número total de veces que aparece el operando

  $N = n1log2n1 + n2log2n2$

  Para determinar el volumen de programa.

  $V = Nlog2(n1 + n2)$

  V variará con el lenguaje de programación y representa el volumen de información en bits necesarios para especificar un programa

  Para determinar el volumen mínimo para un algoritmo:

$L = 2/n1 * n2/N2$

L debe ser siempre menor de 1

- Para obtener una indicación de la estabilidad de un producto:
  Índice de madurez del software (IMS)
  Mt = número de módulos en la versión actual
  Fc = número de módulos en la versión actual que se han cambiado
  Fa = número de módulos en la versión actual que se han añadido
  Fd = número de módulos en la versión anterior que se han borrado en la versión actual
  $IMS = [Mt–(Fa + Fc + Fd)]/Mt$
  A medida que el IMS se aproxima a 1 el producto se empieza a estabilizar.

A lo largo de la última década, todas las corrientes de opinión, referentes a los estándares de Gestión y excelencia empresarial, han defendido la necesidad de situar la satisfacción del cliente en el centro de las operaciones de toda organización, con el objetivo de proporcionar a quienes las usen, desde el punto de vista de la medición del software a través de métricas, un enfoque ágil. A través de la información que se obtiene de estas métricas se conocerá el nivel de satisfacción de los clientes, pudiéndose utilizar este dato por los desarrolladores para tratar de elevar ese nivel, lo que trae consigo una serie de grandes beneficios, entre ellos se pueden destacar:

- Los costos de mantenimiento, diseño y rediseño de la web, se reducen.
- La empresa o institución obtiene como beneficio la lealtad de sus clientes y por ende, la posibilidad de venderle el mismo u otros productos adicionales en el futuro.
- Se contaría con una difusión gratuita que el cliente satisfecho realiza a sus familiares, amistades y conocidos.
- Mejora de la imagen de la organización conquistando un determinado lugar en el mercado.
- Aumento del número de visitantes que se convierten en clientes.
- De aquí la propuesta de métricas para medir la satisfacción del cliente basada en dichas características aunque la misma no se encuentra implementada aún.

- Proporciona un indicador de cómo de bien una aplicación cumplió las expectativas del cliente o usuario.

  Tasa de Estabilidad (TE)

  Se calcula $TE = 1 - (NC/CCU)$ donde:

  NC: número de cambios solicitados durante el primer trimestre, es decir durante los primeros 90 días después de la implementación, o sea, cambios efectuados en el software por parte de los desarrolladores debido a que no pueden por problemas técnicos o lógicos acoplarse en un 100 % a los requisitos que exige el cliente.

  CCU: cantidad de Casos de Uso del tamaño de la aplicación, es decir los casos de uso del sistema obtenidos de los requisitos funcionales. El rango de la tasa de estabilidad está entre $[-\infty; 1]$, mientras más cerca esté el valor de 1 más estable será la aplicación, debido a que el número de cambios solicitados ha de ser mucho menor que la cantidad de casos de uso con que cuenta la aplicación

- Es de suma importancia por su aporte en datos cuantitativos a los distintos miembros de los equipos de desarrollo

  Tasa de Defectos (TD)

  Se calcula $TD = ND/CCU$

  ND: número de defectos, es el total de incidencias en las que la aplicación no cumplió las especificaciones o expectativas del cliente, estos pudieron haberse ocasionados por algún descuido de los miembros del equipo o bien por un mal entendimiento de los requisitos.

  CCU: cantidad de Casos de Uso, del tamaño de la aplicación, es decir, los casos de uso del sistema obtenidos de los requisitos funcionales.

  El intervalo está entre $[0; \infty]$, mientras más cercano este de cero menor es la tasa de defectos, ya que en este caso el número de defectos va ha ser menor que el de casos de uso.

- Para considerar los compromisos de entrega del producto final o por etapas cuantificando la desviación en los plazos de entrega en tiempo.

  Cumplimiento de la Planificación de Entrega (CPE)

  Se calcula $CPE = (TR * 100)/TE$

  TR: tiempo real, es el que se dedicó realmente a la entrega del producto. Se expresa en minutos.

  TE: tiempo estimado o planificado, el que se pactó con el cliente desde su inicio. Se expresa en

minutos.

El resultado de esta métrica indica que si es mayor que 100 %: hay retrasos en el plazo de entrega y de ser contrario el resultado, y si es menor o igual que 100, entonces se están consiguiendo mejor los resultados que lo planificado. Este cálculo debe realizarse mensual o trimestralmente.

Las métricas de estabilidad y fiabilidad fueron formuladas con la condición de que su rango estuviera ente $[-\infty; 1]$ y se encontraría el software más estable o fiable al acercarse a uno, debido a que para que algún producto cumpla con estas, no se puede poner en el rango de $[0; \infty]$, que mientras más tienda a cero mejor, o sea, que cuando esté en el valor uno, las dos métricas darán idea de que existen estas características para el software.

Se recomienda antes de comenzar el uso de las métricas propuestas, identificar el costo de aplicar las métricas. Estos se desglosan de la siguiente manera:

- Costos de utilización de la métrica en los que se incurre durante la recolección de datos, cálculos automáticos de métricas, aplicación, interpretación y reporte de resultados.
- Costos de cambios en el software provocados por cambios en el proceso de desarrollo.
- Costos de los cambios en la estructura organizacional provocada por la producción de software.
- Equipamiento especial para implementar el plan de métricas.
- Entrenamiento para implementar el plan de métricas.

Finalmente se debe tener en cuenta que pueden existir problemas detectados y disfuncionalidades de la organización que ya estaban ahí antes de utilizar una gestión ágil de proyectos como SXP, o sea, que no son el resultado de aplicarla. La transparencia que proporciona SXP, los hace más visibles lo cual ayuda a priorizar y tomar decisiones para lo cual es necesario disponer del máximo apoyo para alinear comportamientos, recursos y resultados con la estrategia de la organización.

2.6. Propuesta de herramientas que soportan la estimación y medición a través de métricas para la metodología SXP.

En la universidad no es muy común el uso de las herramientas automáticas para llevar a cabo la gestión de proyectos, actividades como la estimación y medición se realizan de forma manual, o sea,

son los mismos integrantes del proyecto los que llevan a cabo estos procesos sin el apoyo de los software destinados a esos fines. Luego cada unos de los resultados son registrados en los documentos disponibles.

No obstante no se niega que en algunos proyectos productivos además de realizarse este trabajo de forma manual, para confirmar que la obtención de los resultados es la correcta se apoyen en determinas herramientas, fundamentalmente las que son impartidas durante la asignatura de Ingeniería de software puesto que son las más comunes entre el estudiantado a manera de estándar y las que se enseñan a trabajar con ellas durante las clases prácticas en los laboratorios.

Es muy común utilizar herramientas como el Planner a la hora de realizar la gestión y seguimiento de un proyecto, mediante la representación en diagramas de Gantt.

Específicamente en el proyecto Unicornios es utilizado el Dotproject ofreciendo un marco completo para la planificación, gestión y seguimiento de múltiples proyectos para clientes diferentes, quienes pueden disponer también de acceso para monitorizar la evolución del desarrollo.

Es conocido y utilizado también el Gforge debido a que al presentar un entorno colaborativo para sus proyectos es adaptable a la planificación y gestiona dependencias entre tareas las cuales están organizadas en subproyectos.

2.6.1. Selección de las herramientas a proponer

Después de conocer la gran variedad de herramientas automáticas existentes actualmente en el mercado mundial, para realizar la selección de las que serían factibles para su aplicación en los proyectos productivos de la Facultad, empezando por el proyecto Unicornios, se procede a realizar la propuesta.

Atendiendo a las diversas características de cada uno de los sistemas seleccionados y a que nuestra facultad es la principal impulsora del Software Libre en la universidad, se ha decidido descartar las herramientas que corren en sistemas operativos propietarios, no se desechan las ventajas que pudiesen tener pero no son compatibles con nuestros principios de colaboratividad.

Se ha enfocado el análisis en aquellas herramientas que son libres, o sea, de código abierto y que pueden ser usadas sin ningún tipo de restricciones.

Por ello para la gestión de proyectos en general se propone la consideración por parte de los usuarios del uso de las siguientes herramientas, las cuales fueron escogidas por su capacidad de ser ágiles además y de ser creadas para ser utilizadas por proyectos o equipos de trabajo que utilicen Scrum o XP, principales pilares de SXP.

Herramienta de estimación:

- iceScrum 2:

  Es seleccionada puesto que a pesar de ser una nueva versión del iceScrum original constituye una herramienta web bastante completa la cual muestra en el navegador las columnas llenas de postits, implementada para Linux, Mac y Windows, constituye una herramienta libre y gratuita que corre bajo licencia GPL. Está disponible en varios idiomas entre ellos se destacan Inglés, Francés y Español. Su configuración puede estar dada mediante su instalación o mediante la conexión a su página online a través del navegador.

  Con la misma interfaz para todos los roles, ofrece las opciones de operación, consulta y estimación de historias de usuario, según el usuario sea propietario del producto, Scrum Master, equipo o invitado.Incluye listas de historias de usuario o backlog, de asuntos, de problemas y de pruebas; un chat en línea y un juego de cartas con el que el equipo puede hacer estimación de poker de las historias propuestas en el backlog.

  iceScrum2 permite a los usuarios administrar los proyectos usando Scrum. Con ella se puede had historias al product backlog, dividir el tiempo en sprints y adicionar historias desde el backlog al sprint. Los usuarios pueden pick up historias en el sprint backlog, estimarlas y completarlas. También permite adicionarle pruebas a las historias. Es bastante sencilla de instalar, presentando una interfaz gráfica, cómoda y vistosa, permite realizar el registro de usuarios y elegir el rol, desde propietario de producto a invitado, pasando por Scrum Master y miembro del equipo.

En función del papel elegido, se pueden desarrollar determinadas actividades, por ejemplo, el propietario de producto es el único que puede acceder al product backlog, no se puede poner una historia en un sprint hasta que ha sido planificada, esto, entre otras. La herramienta no es muy intuitiva inicialmente, y cuesta un poco ver como utilizar algunas de las funcionalidades, mas, luego de varias iteraciones con la misma, se hace confortable y en opinión de algunos usuarios divertido su uso.

. Tiene una funcionalidad realmente curiosa e interesante y es el juego de cartas para realizar la estimación del poker on-line estimando las historias. Cada uno de los miembros en su máquina y en su rol dentro del Scrum se conecta, el propietario de producto comienza la partida poniendo la primera historia del product bakclog a la vista de todos, cada miembro da su estimación y en el momento de darla, puede ver las estimaciones de los que hayan jugado antes que él, el último puede ver las de los anteriores. Además cuenta con un chat para poder realizar comentarios en línea. Del lado técnico, está comformada usando Spring, Hibernate y IceFaces y puede desplegarse en Tomcat usando MySQL. En cuanto a la interfaz, hace uso intensivo de las capacidades Ajax de IceFaces entre otras funcionalidades que la hacen cómoda, por ejemplo, el propietario del producto puede ordenar las historias simplemente arrastrándolas con el mouse, se pueden colorear según temas, los postit se pueden mover entre columnas arrastrándolos con el mouse, cada posit puede tener una persona asignada y requiere de un tiempo estimado, crea el gráfico burn down chart automáticamente, cada historia se puede dividir en muchas tareas, entre otras.

El código fuente está disponible en un Subversion de SourceForge y la aplicación se puede descargar como un war. Según criterios emitidos es la mejor herramienta en el mercado para Scrum, pues no es sólo Scrum, sino que toma lo mejor de cada método ágil.

Herramientas para la medición de métricas:

- Sprintometer:
  Es seleccionada puesto que es un programa gratuito, contenido en un único fichero ejecutable, que no necesita instalación, para gestión, medición y seguimiento de programas con modelos ágiles tipo Xp o Scrum. Registra las historias de usuario o funcionalidades, las tareas, estimaciones y su asignación a las personas del equipo. Se puede llevar el seguimiento diario, generar gráficos burn down,

exportar los datos a Excel, etc. Los datos de cada proyecto se pueden guardar en un fichero local, o en un servidor de base de datos que los autores del programa han puesto en sprintometer.com. Desarrollada por un equipo de programadores rusos, para emplearla en sus proyectos, la han liberado como freeware. En el zip de descarga, junto con el programa y un pdf de instrucciones van dos ficheros con proyectos de prueba. Cargarlos y echarles un vistazo, es una forma rápida de ver lo que puede hacer.

Sprintometer es una herramienta gratis, útil y flexible para gestionar proyectos SCRUM y XP, que son procesos de desarrollo basado en la metodología para desarrollar softwares. Sprintometer fue creado en un principio para que lo usaran los programadores de proyectos Agile, pero ahora es un producto para cualquier usuario y además gratis. El programa planifica y realiza sprints (iteraciones) con composición de equipo variable.

El sprint sirve para ofrecer una versión parcial del producto. Sprintometer está preparado para exportar los resultados a Microsoft Excel, y toda la información es almacenada en un archivo local o compartida con otros usuarios, que pueden acceder a ella desde un servidor. . Puede exportar/importar datos del proyecto al formato XML, todo ello en una interfaz moderna que recuerda al estilo de Microsoft Office 2007.

Independientemente de ser una herramienta implementada sobre software libre, el sprinttometer, solo se encuentra hasta el momento, disponible para su ejecución sobre sistemas operativos propietarios, por lo que se recomienda, directamente al equipo de desarrollo que utilizará la propuesta formulada, la implementación de esta herramienta sobre linnux, para esta tarea se deben tener en cuenta, de manera general, las principales funcionalidades que brinda, a grandes rasgos estas son:

- Seguimiento de proyectos SCRUM y XP
- Capacidad de quemado rápido
- Soporte de tamaños de equipos variables
- Predicción de la desviación del plan de sprint/iteración
- Capacidad de exportar a Microsoft Excel para todos los gráficos y hojas de cálculo
- conservar datos en archivos locales o en Bases de Datos compartidas en el servidor

- Interfaz de usuario amigable y moderna
- Soporte de formato XML
- Opción de impresión rápida
- Informes numerosos con todos los parámetros ágiles importantes
- Cálculo del alcance diario para los desarrolladores y probadores de ir por buen camino
- Conexión https segura a los servidores públicos
- Rápido tiempo de respuesta de aplicación

Esta implementación, debe ir estrechamente relacionada con la busqueda de los módulos que se evidencien necesarios para su adaptación a las métricas propuestas anteriormente para la metodología.

En general existe una gran variedad de herramientas disponibles comercialmente para la gestión de proyectos, sin embargo, las tendencias tecnológicas se encaminan hacia la agilidad de los sistemas. Este modelo mejora el desempeño de los procesos orientados a la gestión de proyectos y resuelve el problema de comunicación de la información generada en los proyectos, con los otros procesos de la organización, por lo que una visión ágil de los proyectos, mejora la capacidad de la organización para adaptarse a los cambios dinámicos que exige la competencia.

2.7. Conclusiones

Con la culminación de este capítulo, se ha llegado a un consenso que tiene como producto la propuesta de una técnica de estimación a incluir en la metodología SXP, así como las métricas que se han de utilizar en las mediciones del software. Se ha hecho un análisis profundo antes de la elaboración de esta propuesta tocando tópicos como los aspectos a tener en cuenta para una correcta utilización de la técnica de estimación escogida y de igual manera con las métricas.

Conclusiones

En la actualidad el desarrollo de software ha llegado a constituir la principal fuente de ingresos de muchos países, donde buena parte de la economía depende del mismo. La mayoría de los sistemas de automatización dependen o son controlados por software. Esto obliga a utilizar técnicas y procedimientos que permitan obtener un software que sea funcional, confiable y de calidad, para satisfacer las necesidades de las empresas, industrias, instituciones, profesionales, etc.

Debido a esto si una organización aspira a permanecer sana debe plantearse objetivos realistas. La planificación está comprometida en la fijación de los objetivos de la organización y en las formas generales para alcanzarlos. Con la expectativa de darle solución a los problemas de estimación y métricas presentes en la etapa de planificación de la mayoría de los proyectos productivos de la facultad se ha llegado al final de este trabajo.

En el mismo se abordó acerca de las técnicas de estimación, métricas y herramientas más conocidas y utilizadas actualmente en el mundo y se dio un recorrido panorámico hasta ver cómo se comportan las mismas en la universidad.

Uno de los aspectos que se debe tener muy en cuenta es que es necesario orientar a los equipos de desarrollo de software en comenzar la Gestión de proyectos con su etapa de planificación, los resultados de la misma se reflejarán en la calidad de los productos y procesos al lograr darle satisfacción a los requisitos de cada cliente.

Posteriormente, haciendo un estudio selectivo de cada tema se realizó la propuesta de dichas técnicas y herramientas para la etapa de planificación de la metodología SXP del proyecto Unicornios de la facultad.

Se pretende que este trabajo pueda servir de base para abrir una puerta poco abierta en cuanto a actividades tan importantes como las estimaciones y las mediciones en los diferentes proyectos de software, las cuales son realizadas en muchas ocasiones de manera muy superficial, incitándose a desarrollar un trabajo en equipo, sencillo y rápido pero que satisfaga las condiciones impuestas por los clientes, principales protagonistas en los procesos ágiles.

Recomendaciones

Al realizar todo trabajo no se puede llevar a cabo una sistematización completa de todos los aspectos tocantes a cada tema, siempre por un motivo u otro quedan algunas cosas que se deberían tratar o tener en cuenta en desarrollos futuros, como resultado de la investigación y como elementos a tener en cuenta, se hacen las siguientes recomendaciones, las cuales pueden ser desarrolladas en posteriores tesis para cursos venideros:

- La realización de talleres y encuentros donde se aborde la importancia de la planificación en los estudiantes, sobre todo la necesidad de estimar y de utilizar métricas que permitan evaluar la calidad del producto.
- Realizar una capacitación antes de poner en práctica la técnica de estimación, métricas y herramientas propuestas con el objetivo de preparar a los miembros de cada equipo para el proceso y propiciar el desenvolvimiento satisfactorio de su aplicación.
- Poner en práctica estas actividades como parte del proceso de la metodología ágil SXP en los diferentes proyectos productivos, puesto que debido a ciertos inconvenientes sólo se pudieron proponer, no pudiéndose verificar su efectividad.
- Continuar el estudio del tema para actualizar las técnicas y herramientas propuestas para obtener mejoras en futuras versiones, logrando adecuarlas cada vez más a las necesidades de la facultad.
- Que las técnicas y herramientas propuestas se retroalimenten de los resultados obtenidos de la puesta en práctica de las mismas en los proyectos de desarrollo de software.
- Realizar a modo de evaluación para la propuesta una encuesta de especialistas en materia de estimación y métricas en metodologías ágiles.

- Realizar una revisión detallada del código de las herramientas propuestas con el objetivo de encontrar posibles mejoras que optimicen su utilización.

Referencias

[1] Lian Lisett Hurtado. Planificacion de proyectos [en línea]. Febrero 2006. Disponible en: http://www.wikilearning.com/articulo/planificacion_de_proyectos-planeacion_y_estimacion_de_proyectos_informaticos/9597-1 [visitada 15 de mayo de 2009].

[2] Pedro Concepción. Planificación de proyectos de software [en línea]. Disponible en: http://www.getec.etsit.upm.es/articulos/gproyectos/art4.htm [visitada 15 de mayo de 2009].

[3] Metodologías de desarrollo de software. ¿cuál es el camino? Disponible en: http://www.monografias.com/trabajos60/metodologias-desarrollo-software/metodologias-desarrollo-software.shtml.

[4] Creación de un repositorio de datos para establecer líneas bases de estimaciones. Julio 2008.

[5] Sistema integrado de gestión estadística (sige). rol de planificador. Julio 2007.

[6] Carlos Fontela. Estimaciones: aclarando conceptos [en línea]. Disponible en: http://cysingsoft.wordpress.com/2008/09/15/estimaciones-aclarando-conceptos-carlos-fontela/ [visitada 15 de mayo de 2009].

[7] Rodrigo Corral. Estimación ágil [en línea]. Enero 2008. Disponible en: http://geeks.ms/blogs/rcorral/archive/2008/01/30/he-le-237-do-agile-estimating-and-planning-de-mike-cohn.aspx [visitada 15 de mayo de 2009].

[8] Pablo. Wideband delphi [en línea]. Enero 2007. Disponible en: http://pgpubu.blogspot.com/2007/01/tcnica-de-estimacin-wideband-delphi.html [visitada 15 de mayo de 2009].

[9] Rodrigo Corral. Planning poker [en línea]. Marzo 2008. Disponible en: http://geeks.ms/blogs/rcorral/archive/2008/03/16/planning-poker-distribuido.aspx [visitada 15 de mayo de 2009].

[10] Métricas, estimación y planificación de proyectos de software [en línea]. Disponible en: http://74.125.47.132/search?q=cache:eH3auKiR5_YJ:www.willydev.net/InsiteCreation/v1.0/descargas/willydev_planeasoftware.pdf+planificacion+de+proyectos\&hl=es\&ct=clnk&cd=16\&gl=cu\&lr=lang_es [visitada 15 de mayo de 2009].

[11] Disponible en: http://www.umanizales.edu.co/programs/ingenieria/Ventana/13/13-12-1.htm.

[12] Herramientas para el desarrollo de sistemas de información [en línea]. Disponible en: http://www.alipso.com/monografias/desarrollo_de_sistemas_de_informacion/ [visitada 15 de mayo de 2009].

[13] Herramientas automáticas de estimación [en línea]. Disponible en: http://74.125.47.132/search?q=cache:hR3kt36gwzYJ:www.lasnieves.edu.ar/indocs/p%255CAnalisisYDise%C3%B1oDeSistemas.doc+proyectos+de+software%2Bestimacion%2Bherramientas+utilizadas\&hl=es\&ct=clnk\&cd=14\&gl=cu\&lr=lang_es [visitada 15 de mayo de 2009].

[14] Javier Garzas. Algunas consideraciones sobre la evaluación de la calidad del producto software [en línea]. Agosto 2007. Disponible en: http://kybeleconsulting.blogspot.com/2007/08/algunas-consideraciones-sobre-la.html [visitada 15 de mayo de 2009].

[15] García León Delba Beltrán Benavides Alfa Fernández Carrasco, Oscar M. Un enfoque actual sobre la calidad del software [en línea]. 1995. Disponible en: http://bvs.sld.cu/revistas/aci/vol3_3_95/aci05395.htm [visitada 15 de mayo de 2009].

[16] Certificaciones de calidad-normas iso. Mayo 2005. Disponible en: http://www.monografias.com/trabajos25/normas-iso/normas-iso.shtml [visitada 20 de mayo de 2009].

[17] Dificultades en la certificación de calidad normas iso. Disponible en: http://www.monografias.com/trabajos14/dificultades-iso/dificultades-iso.shtml [visitada 20 de mayo de 2009].

[18] Isoiec 1504. Mayo 2009. Disponible en: http://es.wikipedia.org/wiki/ISO/IEC_15504 [visitada 20 de mayo de 2009].

[19] Demostración a medida de una evaluación cmmi clase c [en línea]. Disponible en: http://www.als-es.com/home.php?location=noticias_y_eventos/demo-evaluacion-cmmi [visitada 15 de mayo de 2009].

[20] Propuesta para aplicar el modelo cmmi en el proceso productivo de la uci. Junio 2007.

[21] Propuesta de una guía metodológica para la planeación, monitoreo y control de proyectos de realidad virtual.

[22] Joaquín Gracia. Cmm_cmmi nivel 2 [en línea]. Noviembre 2005. Disponible en: http://www.ingenierosoftware.com/calidad/cmm-cmmi-nivel-2.php [visitada 15 de mayo de 2009].

[23] Simetse – sistema de métricas para evaluar el software educativo. Julio 2007.

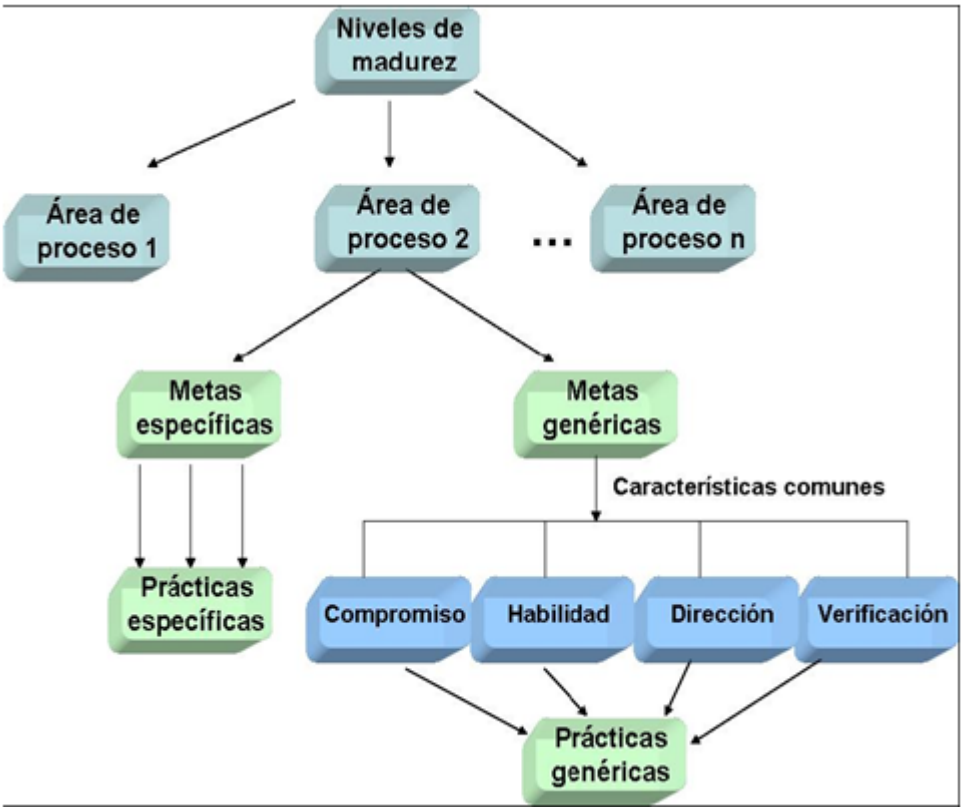
Bibliografía

- Humphrey, Watts S., Introducción al proceso de Software Personal.
- Pressman, Roger S., Ingeniería del Software, Un enfoque práctico.
- Llamosa Villalba, Ricardo, CMMISM for Systems Engineering/Software Engineering/Integrated Product and Process Development/Supplier Sourcing, Version 1.1 (CMMI-SE/SW/IPPD/SS, V1.1) Staged Representation.
- Ruiz Francisco, García Félix, Gestión de Proyectos Software en el marco de PMBOK.
- Carvajal Riola, Juan Carlos, Metodologías ágiles: Herramientas y modelo de desarrollo para aplicaciones Java EE como metodología empresarial, septiembre del 2008.
- Armas Andrade, Rolando, Chamorro Gómez Artuto, Montes Beobide Maite, Gutierrez de Mesa, José Antonio, Desde ISO 9001 hasta CMMI, pasos para la mejora de procesos y métricas, enero del 2007.
- Goncalves, Matias, Desarrollo de un nuevo modelo de estimación basado en metodología ágil de desarrollo y generadores de aplicaciones, segundo semestre del 2005.
- Pérez Sánchez, Jesús, Metodologías ágiles: la ventaja competitiva de estar preparado para tomar decisiones lo más tarde posible y cambiarlas en cualquier momento.
- Fuentes Arderiu, X., Antoja Ribó, F., Castiñeiras Lacambra, M.J., Manual de estilo para la redacción de textos científicos y profesionales.
- Gabardini, Juan, Scrum-Product owner y planificación, primer trimestre del 2008.

- Arregui, J.J.O. Revisión Sistemática de Métricas de Diseño Orientado a Objetos. 2005. Disponible en:
  <http://is.ls.fi.upm.es/doctorado/Trabajos20042005/Olmedilla.pdf
- Lovelle, J.M.C. Calidad del Software. 1999. Disponible en:
  http://gidis.ing.unlpam.edu.ar/downloads/pdfs/Calidad_software.PDF
- Gracia, J. CMM - CMMI Nivel 2. 2005. Disponible en :
  http://www.ingenierosoftware.com/calidad/cmm-cmmi-nivel-2.php
- Mendoza, G.M. Métricas Internas de la Calidad del producto de Software. 2006. Disponible en:
  http://mena.com.mx/gonzalo/maestria/calidad/presenta/iso_9126-3/
- Ávila, L.G. Procedimiento para el desarrollo del proceso de ingeniería de requisitos en un proyecto software (PROCIR). 2007. Disponible en:
  http://www.informaticahabana.com/evento_virtual/?q=node/153&;ev=III%20Taller%20Internacio
  20de%20Calidad%20en%20las%20TICs
- Vigil, A.C. CMM Y la Gerencia de Procesos. 2003. Disponible en: http://www.acis.org.co/memorias/
  JornadasGerencia/IJNGP/Presentacion%20CMM%20y%20PMI.pdf
- Joaquín Ataz López. Guia casi completa de BiBTeX.
- Tobias Oetiker. An acronym environment for LaTeX.
- Irene Hyna y Elisabeth Schlegl Tobias Oetiker, Hubert Partl. The not so short introduction to LaTeX.
- Raúl Mata Botana. Tablas en LaTeX.
- George Gratzer. Math into LaTeX.

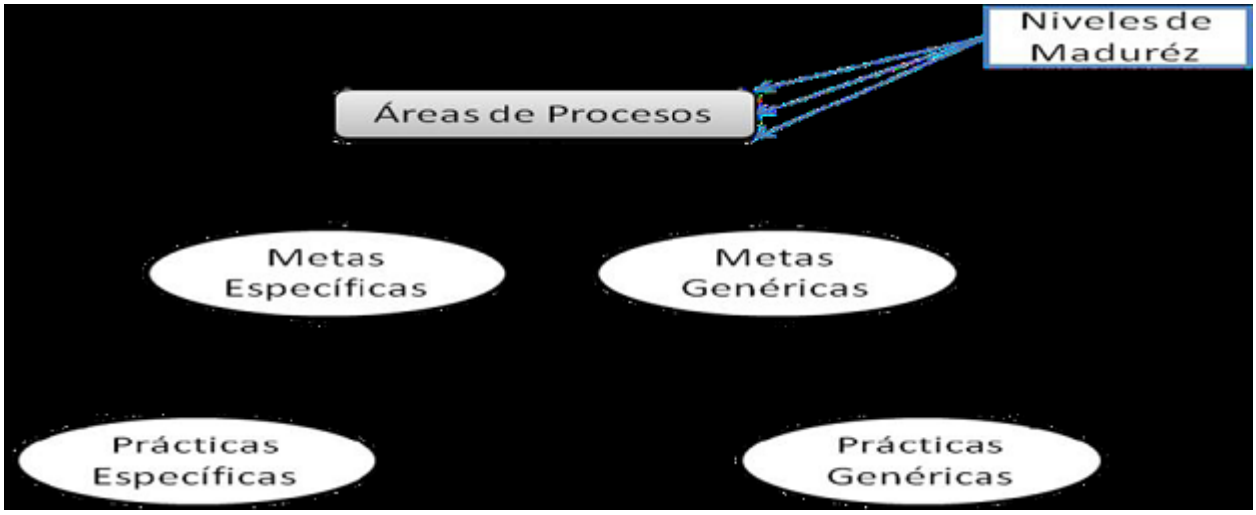
Anexo A

Componentes de CMMI



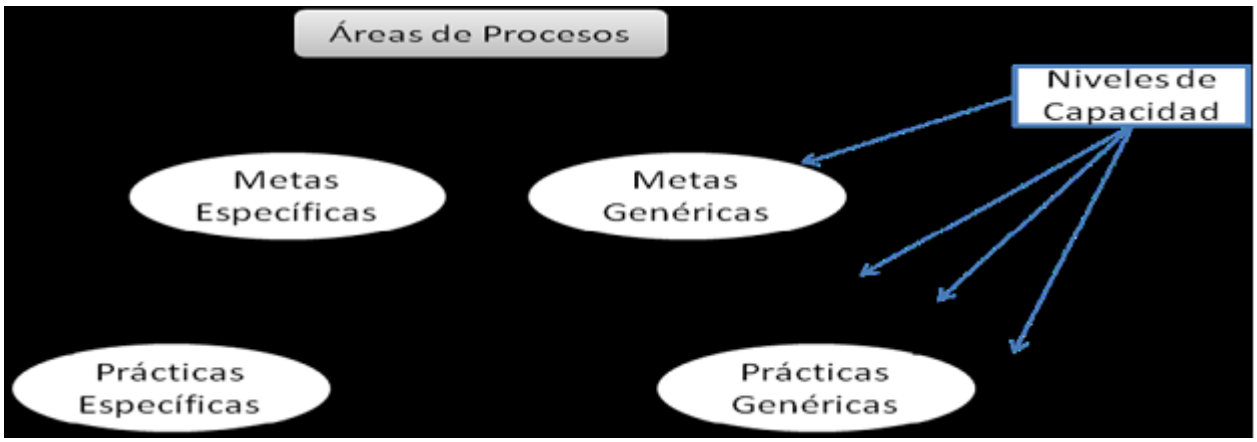
Anexo B

Representación escalonada



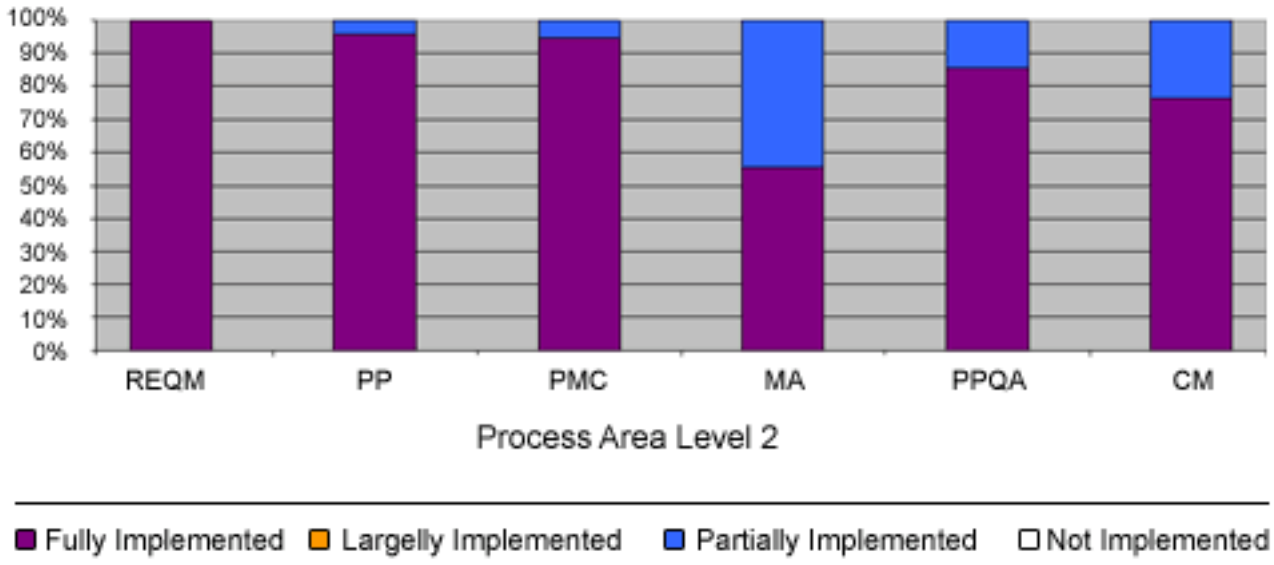
Anexo C

Representación continua



Anexo D

Área de proceso de nivel 2



Glosario de términos

UCI Universidad de las Ciencias Informáticas

CMMI *Capability Maturity Model Integration*:
Es un modelo de calidad del software que clasifica las empresas en niveles de madurez. Estos niveles sirven para conocer la madurez de los procesos que se realizan para producir software.

software Conjunto de programas, instrucciones y reglas informáticas para ejecutar ciertas tareas en una computadora.

CPM *Critical Path Method*

MCE *Minimum Cost Expediten*

GERT *Graphical Evaluation & Review Technique*

LDC Líneas De Código

TMC Tiempo Medio de Cambio

ISBSG *International Software Benchmarking Standards Group*

BFPUG *Brazilian Function Point Users Group*

FPA Function Point Análisis

AEMES Asociación Española de Métricas del Software

CASE *Computer Aided Software Engineering*:
Ingeniería de Software Asistida por Ordenador

PC *Personal Computer*

PSM *Practical Software and Systems Measurement*

ISPJAE Instituto Superior Politécnico José Antonio Echeverria

ISO Organización Internacional para Estandarización

SPICE *Software Process Improvement and Capability Determination*