Universidad de las Ciencias Informaticas

Facultad 8

TRABAJO DE DIPLOMA PARA OPTAR POR EL TÍTULO DE INGENIERÍA EN CIENCIAS INFORMÁTICAS

Métricas para la evaluación del proceso de desarrollo de Software Educativo

Autores:

Daismary Oliva Agüero

Olga Esther Cruz Pichardo

Tutor:

Ing. Risell Ramírez Ramos

Asesor:

Msc. Mayda A. Rodríguez González

Ciudad de la Habana, Junio 2008

"Año 50 de la Revolución"

DECLARACIÓN DE AUTORÍA

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 _____

Daismary Oliva Agüero	Olga Esther Cruz Pichardo	Risell Ramírez Ramos
_____	_____	_____
Firma del Autor	Firma del Autor	Firma del Tutor

"Sin motivación no hay amor, y sin amor por la tarea que se cumple, no hay resultados."

Raúl Castro

Agradecimientos

A nuestra tutora Risell por estar junto a nosotras desde el principio, por su dedicación permanente e incondicional, por aconsejarnos siempre de forma delicada y persuasiva y por brindarnos, a pesar de su juventud, todas sus experiencias.

Al profesor José Manuel Santos por su disposición y apoyo que fueron de gran importancia en el desarrollo de este trabajo.

A Mayda Rodríguez por sus recomendaciones y su preocupación.

A nuestras compañeras de cuarto, en especial a Heidy, Yinet y Daris por su compañía y buen humor.

Al inigualable grupo 8_06 por hacer de nuestro paso por la UCI una experiencia maravillosa.

A los profesores que nos brindaron sus conocimientos y a los que no, también gracias por hacer de nosotras mejores autodidactas.

A todo el que de una forma u otra colaboró en el desarrollo de este trabajo.

A la Revolución Cubana y al Comandante en Jefe Fidel Castro por haber inculcado tantos principios en nosotros los jóvenes y por la extraordinaria idea de haber creado esta Universidad.

Muchas Gracias

Es ahora que nos corresponde agradecerles con acciones, es ahora que retribuiremos con creces lo que con dedicación y espera lograron de nosotras.

Quiero agradecer:

A mis padres, Estrella y Rafael, por el amor, la comprensión y dedicación que han sido capaces de brindarme durante toda mi vida, sobre todo, en estos años de estudio que han sido tan importantes para mí.

A mis abuelos, Lucia y Agüero, por transmitirme su experiencia, su ayuda incondicional y por estar presentes siempre que los he necesitado.

A Yasser por formar parte de mi vida, por brindarme todo su apoyo en los buenos y malos momentos, y a su familia por acogerme con tanto amor y cariño.

A mis familiares por ayudarme, apoyarme, alentarme y preocuparse siempre por mí, en especial a mis tíos Pedro, Frank y Félix por ser como mis padres.

A mi compañera de tesis, Olguita, pues sin ella no hubiese sido posible realizar este trabajo con empeño, dedicación, esfuerzo y amor.

A mis amistades y profesores del preuniversitario que a pesar de la distancia los tengo siempre presentes por todos los años que compartimos juntos, en especial a Evelyn, Shirley y Cleidy.

A todas las amistades tan lindas que la UCI me ha dado la oportunidad de concer.

A los que de una manera u otra han contribuido a mi formación profesional.

A todos muchas gracias.

Daismary

Agradezco:

A mi mamá Elsa porque lo es todo para mí, por no faltar nunca cuando la necesito, por su paciencia, por escucharme y aconsejarme, por entregarme lo dulce de su vida, en fin, por dedicarse enteramente a ser madre.

A mis hermanos Robe y Yuyi por regalarme una infancia maravillosa y apoyarme en todo momento.

A mi novio Ariel por llegar e invadir mi vida de alegría, por el amor y la dedicación que me ha brindado, y a su familia por acogerme y apoyarme en estos cinco años.

A mis abuelas Olga y Esther por darme su nombre y a mis abuelitos por el gran cariño que me ofrecieron y me ofrecen día a día.

A mis tíos Rafael, Enrique y Victor por ser un ejemplo para mi formación profesional.

A mis amigos de toda la vida Marlien y Jorgito, por ser un ejemplo de la verdadera amistad.

A todas mis amigas del preuniversitario por los buenos momentos que pasamos juntas y en especial a Ismary, Yunelsys, Annelis, y Magdey porque a pesar de la lejanía siempre se mantuvieron cerca de mi corazón.

A Daismary por los días y noches interminables que pasamos juntas para hacer realidad este sueño, porque el éxito se alcanza sólo cuando se tiene con quien compartirlo.

A todas las personas que conocí en la UCI y que ahora forman parte de mi círculo de amigos.

Olguita

Dedicatoria

A mi mamá, por ser la estrella que ilumina mi camino.

A mi papá, por ser la alegría de todos mis días.

Daismary

A Riselda e Isidro por ser mi luz en la oscuridad.

A Ariel por convertirse en la razón de mi existir.

A mi mamá porque para ella será siempre todo lo bueno que alcance en mi vida.

Olguita

RESUMEN

La Universidad de las Ciencias Informáticas (UCI), como parte del programa nacional de Informatización de la Sociedad, se encuentra inmersa en una intensa tarea para el mejoramiento del proceso docente-educativo cubano. Con el fin de cumplir sus objetivos, en estos momentos, produce gran cantidad de Software Educativo (SWE) para los diferentes niveles de enseñanza del país. Pero, lograr buena calidad en estos productos se ha convertido en una preocupación para sus creadores debido, entre otros factores, a que la institución no cuenta con instrumentos capaces de garantizar la medición constante de la calidad de los procesos de desarrollo que se llevan a cabo para construir un buen software. La presente investigación tiene como principal objetivo proponer métricas que permitan evaluar cuantitativamente los procesos de desarrollo de SWE que se toman en cuenta en el Modelo de Evaluación del Proceso de Desarrollo del Software Educativo (MEPDSE). Para conformar la propuesta se seleccionaron las métricas que se adecuaban a los procesos de desarrollo, y además se elaboraron algunas al no existir las suficientes para cubrir la evaluación de todos los procesos. El conjunto de métricas cuenta con un procedimiento de evaluación que permitirá que el modelo MEPDSE ofrezca resultados objetivos sobre la calidad de cada uno de los procesos que se evalúen, lo que ayudará, en gran medida, a mejorar la calidad de los SWE que se crean en la UCI.

Palabras Claves

Procesos de desarrollo, software educativo, métricas.

ÍNDICE DE CONTENIDOS

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

La presencia de las Tecnologías de la Información (TI) en las aulas del mundo actual, es cada día mayor; sin embargo, es un reto garantizar que se haga un uso adecuado de las mismas. Parte de este reto es contar con un software de calidad. La calidad es un elemento multidimensional y contextual, por tanto, altamente cualitativo; sin embargo la tendencia mundial en cuanto a evaluación de la "calidad", se orienta a la "cuantificación" de los aspectos sometidos a consideración, con el fin de contribuir en la toma de decisiones dentro de un proceso de selección y adquisición de alternativas.(ESTRADA and ESTÉVEZ 2003)

Debido a la cantidad creciente de aplicaciones de Software Educativo (SWE) disponibles para crear y administrar contenidos de cursos basados en tecnología, se plantea la necesidad de contar con técnicas que permitan evaluar objetivamente los productos desarrollados.

En consecuencia, podemos afirmar que actualmente existe una *crisis del SWE*, caracterizada principalmente por; una calidad del proceso de desarrollo así como del producto final en muchos casos deficientes, y en cualquier caso difícil de estimar a priori y de controlar durante el proceso de desarrollo. (ROSELLÓ *et al.* 1994)

El SWE cubano, tiene diferentes características marcadas en comparación con los desarrollados en otros países del mundo. Todo esto producido por el avance y la experiencia acumulada en el área de la pedagogía en nuestro país, que lo ubica, en uno de los primeros lugares del planeta en este sentido. (RICARDO 2006)

El desarrollo del SWE se ha convertido en una necesidad para el sistema educativo nacional. Pero hoy la calidad de los procesos no se mide de forma correcta; debido a que en la mayoría de los productos no se lleva a cabo una revisión constante, sino que se evalúa al final de la producción y generalmente se debe regresar al inicio para erradicar los problemas y defectos encontrados. Esta situación provoca; pérdida de tiempo, de recursos y problemas a la hora de certificar los productos, por lo que Cuba no está exento de dicha crisis.

Cuando se habla de calidad de SWE, se requiere un producto que satisfaga tanto las expectativas de los docentes como de los usuarios, a un menor costo, libre de defectos y cumpliendo con ciertas especificaciones. (PRESSMAN 2002) Esta necesidad conlleva a incrementar el interés, de personas afines al tema, por construir un modelo que especifique la calidad de los SWE, enfocándolo no sólo como un producto sino considerando también los procesos para construirlo. Sin un buen proceso de desarrollo es casi imposible obtener un buen producto.

La Universidad de las Ciencias Informáticas (UCI), como parte de los centros productores de SWE en el país, está inmersa en una intensa tarea para mejorar la calidad de las aplicaciones de software que en ella se crean, producto de las exigencias cada vez mayores de los clientes, que obligan a elevar la calidad de las producciones.

Actualmente, en la UCI no se evalúa la calidad de forma correcta. Debido, entre otros factores, a que no se utiliza un modelo como guía para la evaluación de los proceso de desarrollo del SWE que incluya métricas para ofrecer resultados objetivos sobre estas evaluaciones.

Para dar solución a esta problemática se creó el Modelo de Evaluación del Proceso de Desarrollo del Software Educativo (MEPDSE), que permite gestionar la calidad del proceso de desarrollo del SWE, pero que necesita métricas para ser aplicable, pues en estos momentos depende del conocimiento de los evaluadores.

El presente trabajo surge como necesidad de dar solución a las situaciones antes expuestas; por lo que el **problema científico** que se debe solucionar es: La inexistencia de métricas en el modelo MEPDSE, dificulta la evaluación de forma eficiente de los procesos de desarrollo que se toman en cuenta en dicho modelo.

De aquí que el **objeto de estudio** esté constituido por la evaluación de la calidad de procesos de desarrollo del SWE y **el campo de acción** sean las métricas para la evaluación de la calidad de procesos de desarrollo del SWE.

Todo ello encaminado a cumplir el **objetivo general** de proponer métricas que permitan evaluar los procesos de desarrollo que utiliza el modelo de evaluación MEPDSE de forma eficiente y satisfactoria.

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

- Realizar el estudio del estado del arte para definir la posición teórica del investigador.
- Definir métricas para evaluar los procesos de desarrollo que utiliza el modelo de evaluación MEPDSE.

Para el desarrollo de la investigación se parte de la **idea a defender,** que si se utilizan métricas para evaluar los procesos de desarrollo que se tienen en cuenta en el modelo de evaluación MEPDSE, entonces será posible darle una calificación a cada uno de esos procesos que se ajuste a la realidad de los mismos.

Para el cumplimiento de los objetivos trazados y la idea a defender definida se proponen un conjunto de **tareas** que ayudarán a que la investigación se realice de forma eficiente, a continuación se especifican:

- Investigación sobre las métricas de evaluación más utilizadas en la actualidad.
- Revisión de los procesos que se pretenden evaluar en el modelo MEPDSE para determinar qué métricas se adecuan a este.
- Selección de las métricas que mejor se adaptan a los procesos que se tienen en cuenta para la evaluación en el modelo MEPDSE.
- Proposición y descripción del conjunto de métricas para evaluar los procesos que toma en consideración el modelo de evaluación MEPDSE.
- Proposición de un procedimiento para la evaluación de los procesos que utiliza el modelo MEPDSE empleando las métricas propuestas.
- Validación del conjunto de métricas mediante el método de expertos.

La propuesta de métricas permitirá que el modelo de evaluación MEPDSE ofrezca resultados objetivos sobre el estado de los procesos. Esto ayudará, en gran medida, a mejorar la calidad de los procesos de desarrollo de SWE en los proyectos que lo apliquen.

La importancia de que el resultado, obtenido en la presente investigación, sea aplicado en los proyectos productivos de SWE en la UCI, consiste en la necesidad de conseguir en las evaluaciones datos cuantitativos que sirvan de basamento para asegurar si los procesos de desarrollo en un SWE se han realizado con calidad.

La estructura del presente trabajo de diploma consta de tres capítulos:

Capítulo 1: Fundamentación Teórica. Para una mejor comprensión del tema en este capítulo se abordan conceptos que sirven de sustento teórico a la investigación desarrollada, como son: procesos de desarrollo de software, calidad de SWE, entre otros. Además, se presentan ejemplos de métricas que sirven de apoyo para el siguiente capítulo, organizadas por las categorías a las que pertenecen los procesos que se pretenden evaluar.

Capítulo 2: Descripción de la Propuesta de Solución. Este capítulo se inicia realizando el análisis y descripción de los métodos empíricos utilizados. Se describen, además, cada una de las métricas que conforman la propuesta especificando las variables que integran sus fórmulas, así como el procedimiento con que se evaluarán los procesos utilizando las métricas definidas.

Capítulo 3: Evaluación Técnica de la Propuesta de Solución. En este capítulo se determina la probabilidad de éxito que tiene la propuesta. Para ello se utiliza el método de experto, que permite tomar decisiones para aceptar o no la propuesta de acuerdo con diferentes criterios de evaluación.

1

CAPÍTULO1: FUNDAMENTACIÓN TEÓRICA

1.1. Introducción

Actualmente el término métricas de software está siendo ampliamente utilizado. La empresa cubana del software y la Universidad de las Ciencias Informáticas (UCI) están incursionando en la aplicación de métricas, como parte de su esfuerzo para desarrollar software con calidad.

En el presente capítulo se exponen una serie de conceptos de métricas, así como sus características y diferentes clasificaciones. Para una mejor comprensión del tema se inicia abordando conceptos que sirven de sustento teórico a la investigación desarrollada.

1.2. Proceso de desarrollo del software

El proceso de desarrollo del software es un marco donde se define un pequeño número de actividades del marco de trabajo en la construcción de un proyecto, con independencia de su tamaño o complejidad. Las evaluaciones de procesos del software se han establecido como instrumento objetivo para la determinación de la capacidad de los procesos de una organización. Es por ello que se ha hecho muy común, entre los profesionales de la Industria del Software, utilizar este término.

Algunos autores definen un proceso de desarrollo como:

Un conjunto de herramientas, métodos y prácticas que se emplean para producir software. (ROMÁN *et al.* 1998)

Un proceso de desarrollo de software es el conjunto de actividades necesarias para transformar los requisitos de un usuario en un sistema software. (IVAR JACOBSON 2000) (Ver Figura 1)

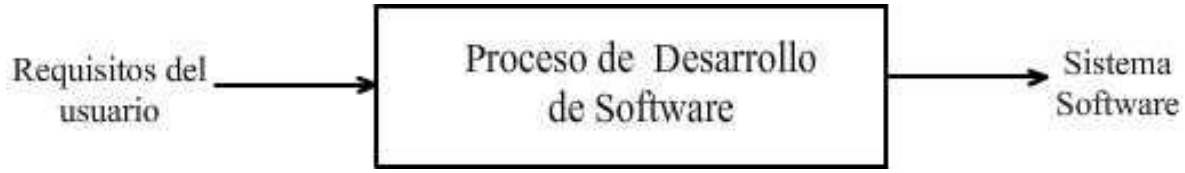

Figura 1 Proceso de Desarrollo de Software

Es un marco de trabajo de las tareas que se requieren para construir software de alta calidad. (Ver Figura 2)

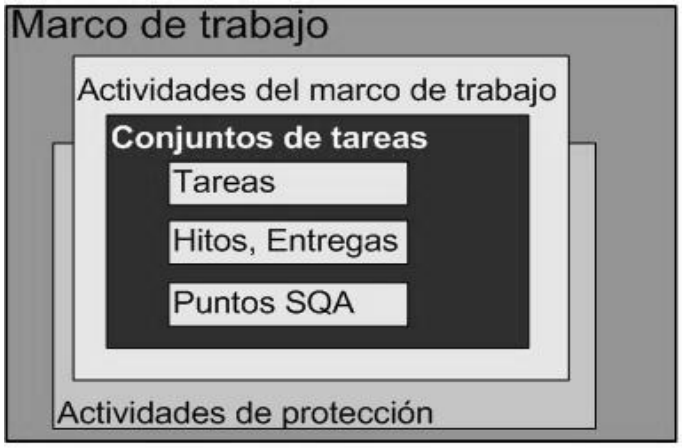

Figura 2 Proceso de Desarrollo de Software

El marco de trabajo es aplicable a todos los proyectos del software, como se muestra en la figura 2, contiene un conjunto de tareas. Cada una de ellas es una colección de tareas de trabajo de ingeniería de software, hitos de proyectos, productos de trabajo, y puntos de garantía de calidad que permiten la adaptación de las actividades a las características del proyecto del software y a los requisitos del equipo del proyecto. Finalmente, las actividades de protección son independientes de cualquier actividad del marco de trabajo y aparecen durante todo el proceso. (PRESSMAN 2002)

Se puede decir, que los procesos de desarrollo del software son el conjunto de todas aquellas actividades y tareas que se deben llevar a cabo para obtener un producto final.

1.3. Calidad del software

La calidad es el término que más preocupa hoy en día a las empresas desarrolladoras de software, producto de la exigencia de los clientes para que se realicen software cada vez más complejos en menor tiempo. Debido a esta situación la calidad se ha vuelto un reto para cualquier equipo de desarrollo y un factor determinante para alcanzar el éxito en esta industria, pues es fundamental para lograr la satisfacción de las necesidades y expectativas del cliente.

A continuación se exponen algunas definiciones de “calidad” que sirven de preámbulo al hablar de “calidad de software”.

La calidad, es sinónimo de garantía y seguridad en el momento de adquirir un producto o un servicio ya que da tranquilidad a los que lo adquieren. (FARFÁN 1997)

Es la medida en que las propiedades de un bien o servicio cumplen con los requisitos establecidos en la norma o especificaciones técnicas, así como con las exigencias del usuario de dicho bien o servicio en cuanto a su funcionalidad, durabilidad y costo.(SEI 2002)

La norma ISO 8402 define la calidad como la totalidad de características de un producto o servicio que le confieren su aptitud para satisfacer unas necesidades expresadas o implícitas. (CATALDI *et al.* 2003)

Refiriéndose ya a calidad de software, Callaos plantea que no es algo que depende de una sola característica en particular, sino que obedece al compromiso de todas sus partes. A la hora de definir la calidad del software se debe diferenciar entre la calidad del producto de software y la calidad del proceso de desarrollo de éste “calidad de diseño y fabricación”. No obstante, las metas que se establezcan para la calidad del producto van a determinar los objetivos del proceso de desarrollo, ya que la calidad del primero va a depender, entre otros aspectos, de estos últimos. (N. CALLAOS and CALLAOS 1993)

La ausencia de defectos, la aptitud para el uso, la seguridad, la confiabilidad y la reunión de especificaciones son elementos que están involucrados en el concepto de calidad del software. (ESTRADA and ESTÉVEZ 2003)

Pressman agrupa todos los aspectos que antes se mencionan para referirse a la calidad del SWE en un concepto único y realmente convincente; “es la concordancia con los requerimientos funcionales y de rendimiento explícitamente establecidos, con los estándares de desarrollo explícitamente documentados y con las características implícitas que se espera de todo software desarrollado profesionalmente.” (PRESSMAN 2002)

En síntesis, se considera que calidad, resume las características, propiedades, cualidades y en general atributos propios de un producto software. Determinando, sobre este, la ausencia de defectos y la conformidad de todo el personal que de una forma u otra se vinculan con él.

De ahí la importancia de: (1) mantener los productos bajo un constante control, (2) desarrollar mediciones que den como resultado sistemas de alta calidad y (3) contar con métodos y herramientas efectivas que evalúan la calidad con objetividad, no con subjetividad.

Para la ejecución de los tres aspectos anteriores los encargados del aseguramiento de la calidad, en un proyecto, deben basarse en medidas que garanticen la calidad de los productos software, entre las que se pueden citar: (PRESSMAN 2002)

- **Corrección:** La corrección es el grado en el que el software lleva a cabo su función requerida. Un programa debe operar correctamente o proporcionará poco valor a sus usuarios.
- **Facilidad de mantenimiento:** La facilidad de mantenimiento es la facilidad con la que se puede corregir un programa si se encuentra un error, se puede adaptar si su entorno cambia, o mejorar si el cliente desea un cambio de requisitos.
- **Integridad:** Este atributo mide la capacidad de un sistema para resistir ataques (tanto accidentales como intencionados) contra su seguridad. El ataque se puede realizar en cualquiera de los tres componentes del software: programas, datos y documentos.
- **Facilidad de uso:** La facilidad de uso de un sistema se basa en lo amigable que puede ser este con el usuario.
- **Eficacia de la Eliminación de Defectos (EED):** La EED es una medida de la habilidad de filtrar las actividades de la garantía de calidad y de control al aplicarse a todas las actividades del marco de trabajo del proceso.

1.4. Calidad del SWE

El SWE proporciona múltiples ventajas en el proceso de enseñanza-aprendizaje como medio didáctico. Sin embargo, su uso adecuado, requiere una selección cuidadosa para garantizar un producto de calidad. Pero, antes de hablar de calidad de SWE, es necesario conocer que se entiende por este término.

Software Educativo (SWE)

Es una aplicación informática, que soportada sobre una estrategia pedagógica bien definida, apoya directamente el proceso de enseñanza-aprendizaje constituyendo un efectivo instrumento para el desarrollo educacional del hombre del próximo siglo. (LAMAS 2000)

Según Gros se considera SWE a cualquier producto basado en computadora con una finalidad educativa. (GROS, B *et al.* 1997)

Galvis expresa que en el campo educativo suele denominarse SWE a aquellos programas que permiten cumplir y apoyar funciones educativas. En esta categoría entran tanto los que dan soporte al proceso de enseñanza-aprendizaje (ej. un sistema para enseñar

matemáticas, ortografía, contenidos o ciertas habilidades cognitivas), como los que apoyan la administración de procesos educacionales o de investigación (ej. un sistema que permita manejar un banco de preguntas). (GALVIS 2000)

De las definiciones anteriores se puede considerar entonces como SWE, a los programas para computadoras creados con el propósito específico de ser utilizados como medio didáctico, es decir, para facilitar los procesos de enseñanza y de aprendizaje.

Características del SWE

Es conveniente que todo SWE, independientemente de la materia que trate, ofrezca un entorno de trabajo que resulte amigable y de fácil manejo, debido a ello debe adecuarse a las particularidades de cada profesor y estudiante. Para esto poseen cinco características esenciales: (MÁRQUEZ 2005)

- Son materiales elaborados con una **finalidad didáctica**.
- **Utilizan el ordenador** como soporte en el que los alumnos realizan las actividades que ellos proponen.
- **Son interactivos**, contestan inmediatamente las acciones de los estudiantes y permiten un diálogo y un intercambio de informaciones entre el ordenador y los estudiantes.
- **Individualizan el trabajo** de los estudiantes, ya que se adaptan al ritmo de trabajo de cada uno y pueden adaptar sus actividades según las actuaciones de los alumnos.
- **Son fáciles de usar**. Los conocimientos informáticos necesarios para utilizar la mayoría de estos programas son similares a los conocimientos de electrónica necesarios para usar un vídeo, es decir, son mínimos, aunque cada programa tiene unas reglas de funcionamiento que es necesario conocer.

Evaluación del SWE

El tema de evaluación del SWE ha sido estudiado y documentado por diversos autores del ámbito, proporcionando medidas de evaluación en el área educativa y técnica. Existen varios métodos o modelos propuestos para evaluar un SWE. Se pueden considerar los aspectos computacionales y los aspectos educativos o bien ambos a la vez.

La evaluación de SWE se centra tradicionalmente en dos momentos durante: (1) su utilización real por los usuarios, para juzgar su eficiencia y los resultados que con él se obtienen y (2) el proceso de diseño y desarrollo, con el fin de corregir y perfeccionar el programa. (ESTRADA and ESTÉVEZ 2003).

En toda fase de preparación de la evaluación para SWE se especifican tres pasos generales, a saber: (1) selección de las métricas de calidad, (2) definición de los niveles de puntaje y (3) definición de los criterios de valoración.

En el primer paso (o actividad) los evaluadores deben decidir y seleccionar las métricas directas (o indirectas). Las métricas seleccionadas, deben correlacionarse con las características respectivas, difieren del contexto y de la fase del proceso de desarrollo.

En el segundo paso, (Definición de los Niveles de Puntaje), se deben definir los niveles de satisfacción. Esto es necesario debido a que el valor de una métrica, es calculado en un rango de una escala, que no expresa el nivel de satisfacción. Por lo tanto, el estándar ISO 9126 indica que se deben definir grados o rangos de valores de satisfacción de los requerimientos. Indica además, que, debido a que la calidad se refiere a necesidades dadas y específicas, ningún nivel de puntaje general es posible sino que se debe definir con respecto a cada evaluación en particular.

En el tercer paso (Definición de los Criterios de Valoración), los evaluadores deben definir procedimientos para resumir los resultados de las diferentes características. Se pueden usar tablas de decisión o promedios pesados. El procedimiento puede incluir otros aspectos como tiempo y costos, en un contexto particular. (CRISTANCHO 2006)

La evaluación de calidad de un SWE debe estar presente desde el inicio de su desarrollo y, aún, cuando el cliente y el usuario lo estén utilizando. Se destacan cuatro espacios en los que es de suma importancia la evaluación: *el proyecto*, *los procesos de desarrollo*, centro de interés en el presente trabajo, *el producto* y *la utilización*.

Ciertamente en una evaluación integral de la calidad de un SWE, deben considerarse todos los aspectos posibles y los factores que implican. Este análisis se enfocó en los aspectos técnicos y no en los educativos, que aunque de gran importancia, se alejan del tema a tratar.

Calidad del SWE

Para Gros la calidad del SWE está determinada no sólo por los aspectos técnicos del producto sino por el diseño pedagógico y los materiales de soporte. Este último aspecto es uno de los más problemáticos ya que existen pocos programas que ofrezcan un soporte didáctico. (GROS, B. 2000)

Se puede decir que la calidad del SWE es como un organismo humano que funciona como un todo donde todas sus partes están interrelacionadas, entiéndase por parte: calidad técnica, calidad pedagógica y calidad funcional. Si una de estas partes no alcanzan su calidad óptima el producto SWE no tendrá la calidad total. (CINTRA and CISNEROS 2007)

Finalmente, se puede afirmar que un SWE tiene calidad cuando logra satisfacer las expectativas de los clientes y cumplir con los guiones técnico y pedagógico previamente especificados.

1.5. La medición en el Software

Medir es conocer, y este conocimiento permite avanzar sobre bases sólidas. Pero medir proporciona algo más, posibilita aplicar las poderosas herramientas que las matemáticas ofrecen, facilitando la manipulación de datos con objeto de alcanzar una visión de la entidad estudiada que de otra forma hubiera sido imposible obtener.

La realización de medidas aplicadas en un entorno propio del desarrollo de programas informáticos, necesita definiciones concisas que identifiquen claramente las entidades a estudiar, así como aquellos atributos que serán objeto de dichas medidas. Los *procesos*, *productos* y *recursos* son entidades importantes en este entorno.

Una vez definidas las entidades a las que se les presta atención se han de considerar los atributos a medir. Estos se pueden dividir en dos tipos fundamentales *atributos directos***,** como aquellos que pueden ser medidos en términos de las entidades a estudiar, y *atributos indirectos***,** definidos como aquellos que sólo pueden ser medidos por medio de cómo productos, procesos y recursos se relacionan con su entorno.

La tabla 1 muestra un resumen de atributos, recursos, procesos y entidades relacionadas con la medida del software. (MELIÁN and BALLESTEROS 2003)

Tabla 1 Ejemplos de atributos directos e indirectos

Atributo	Directo	Indirecto	Entidad
Tamaño	X		Producto (especificaciones, diseño, código…)
Tamaño	X		Recurso (software, equipos, instalaciones…)
Funcionalidad	X		Producto (especificaciones, diseño, código…)
Coste		x	Proceso (diseño, pruebas, captura de especificaciones…)
Esfuerzo	X		Proceso (diseño, pruebas, captura de especificaciones…)
Recuso	X		Producto (especificaciones, diseño, código…)
Productividad		x	Recurso (equipo de trabajador, programador, hardware…)
Calidad		x	Producto (prueba con datos, diseño, código…)

Calidad		x	Proceso (creación de especificaciones…)
Calidad		x	Recurso (software, hardware, personal…)
Complejidad		x	Producto (diseño)
Facilidad de Uso		x	Producto (código)
Fiabilidad		x	Producto (código), Recurso (software, hardware, personal…)

Razones para medir los procesos, producto y recursos

Dentro del amplio y extenso espectro de las actividades que se desarrollan para construir un software con calidad, no es extraño encontrar personas que aún no tienen una idea clara de la importancia de la necesidad de medir. El área de las mediciones de software, a pesar de ser una de las áreas en la ingeniería de software donde se ha investigado desde hace más de 30 años, todavía no ha sido bien comprendida.

Suele haber preocupación de que las mediciones señalarán problemas en un proyecto o en una organización que no eran visibles antes de que el proceso de medición fuera implementado. Por ejemplo, el análisis de las mediciones puede mostrar que los planes de desarrollo no son realistas, o que determinado programador está atrasado en sus tiempos de entrega. Estas preocupaciones son reales, y sobreponerse a ellas, requiere un entendimiento de las mediciones, así como saber usar sus resultados apropiadamente en todos los niveles de la organización. (MCGARRY and BAILEY 1998)

Hay cuatro razones para medir los procesos del software, los productos y los recursos: (PRESSMAN 2002)

- **Caracterizar:** Se caracteriza para comprender mejor los procesos, los productos, los recursos y los entornos y para establecer las líneas base para las comparaciones con evaluaciones futuras.
- **Evaluar:** Se evalúa para determinar el estado con respecto al diseño. Las medidas utilizadas permiten conocer cuando los proyectos y los procesos están fuera de control. También se evalúa para determinar el impacto de la tecnología y las mejoras del proceso en los productos y procesos.
- **Predecir:** Se predice para poder planificar. Esto se hace porque se quieren establecer objetivos alcanzables para el coste, planificación, y calidad (de manera que se puedan aplicar los recursos apropiados). Las medidas de predicción también son la base para la extrapolación de tendencias, con lo que la estimación para el coste, tiempo y calidad se puede actualizar basándose en la evidencia actual. Las proyecciones y las estimaciones

basadas en datos históricos también ayudan a analizar riesgos y a realizar intercambios diseño/coste.

- **Mejorar:** Se mide para mejorar cuando se recoge la información cuantitativa que ayuda a identificar obstáculos, problemas de raíz, ineficiencias y otras oportunidades para mejorar la calidad del producto y el rendimiento del proceso.

Para realizar un proceso de medición correcto existen cinco actividades: (PRESSMAN 2002)

- **Formulación:** Obtención de medidas y métricas del software apropiadas para la representación del software en cuestión.
- **Colección:** Mecanismo empleado para acumular datos necesarios para obtener las métricas formuladas.
- **Análisis:** Cálculo de las métricas y la aplicación de herramientas matemáticas.
- **Interpretación:** Evaluación de los resultados de las métricas en un esfuerzo por conseguir una visión interna de la calidad de la representación.
- **Realimentación:** Recomendaciones obtenidas de la interpretación de métricas técnicas transmitidas al equipo que construye el software.

En general, se puede afirmar que el proceso de medición en el software persigue tres objetivos fundamentales: (1) entender qué ocurre durante el desarrollo y el mantenimiento del software, (2) controlar qué ocurre en los proyectos y (3) mejorar procesos y productos. Pero lamentablemente, en ocasiones, los resultados de los procesos de medición no son interpretados de la mejor forma. En la actualidad no se ha adquirido una cultura adecuada sobre la medición, por desconocimiento del alcance de madurez y calidad que pudiera ganar el producto final.

1.6. Métricas del software

En la actualidad se ha hecho imprescindible el uso de las mediciones como parte de las actividades cotidianas de todas las organizaciones de software, pues estas brindan la información objetiva necesaria para la toma de decisiones, teniendo un impacto efectivo en el negocio y desempeño en la ingeniería.

Para poder asegurar que un proceso o sus productos resultantes son de calidad o poder compararlos, es necesario asignar valores, descriptores, indicadores o algún otro mecanismo mediante el cual se pueda llevar a cabo dicha comparación. (ROSELLÓ *et al.* 1994) Estas asignaciones son realizadas frecuentemente cuando se utilizan métricas.

Las métricas del software se refieren a un amplio elenco de mediciones para el software de computadora. La medición se puede aplicar al proceso del software con el intento de mejorarlo sobre una base continua. Se puede utilizar en el proyecto del software para ayudar en la estimación, el control de calidad, la evaluación de la productividad y el control de proyectos. Finalmente, el ingeniero de software puede utilizar la medición para ayudar a evaluar la calidad de los resultados de trabajos técnicos y para ayudar en la toma de decisiones tácticas a medida que el proyecto evoluciona. (PRESSMAN 2002)

Los términos métricas, medición y medida suelen utilizarse indistintamente, por lo que es necesario aclarar que no tienen el mismo significado. Una *medida* proporciona una indicación cuantitativa de la extensión, cantidad, dimensiones, capacidad o tamaño de algunos atributos de un proceso o producto; la *medición* es el acto de determinar una medida.

En cambio se le denomina *métrica* a:

La continua aplicación de técnicas basadas en la medición al proceso de desarrollo de software y a sus productos para proveer información administrativa, significativa y oportuna, junto con el uso de esas técnicas para mejorar el proceso y sus productos. (WESTFALL 1995)

Una medida cuantitativa del grado en que un sistema, componente o proceso posee un atributo dado. (PRESSMAN 2002)

Un buen medio para entender, monitorizar, controlar, predecir y probar el desarrollo del software y los proyectos de mantenimiento. (L.BRIAND *et al.* 1996)

Las métricas del software pueden ser utilizadas para que los profesionales e investigadores tomen las mejores decisiones. (PRESSMAN 2002) Proporcionan información objetiva que contribuye al mejoramiento de los procesos y productos de software lo cual, evidentemente, favorece al logro de la calidad, siempre que se haga un uso adecuado de las mismas.

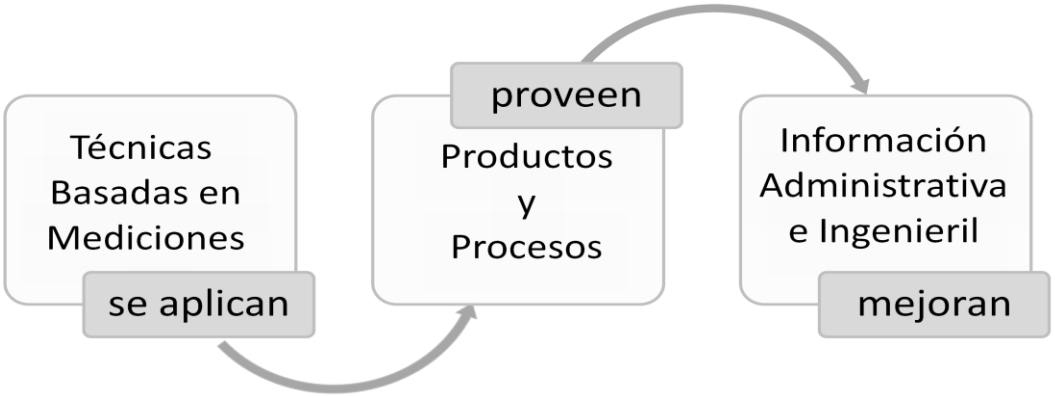

Figura 3 Métricas del Software

En resumen, una métrica es el término que describe muchos y muy variados casos de medición. Siendo 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, funcionalidad, documentación, métodos, proceso, usuario, entre otros.

1.6.1. Características de las métricas de software

Muchas han sido las métricas propuestas para el software, pero no todas proporcionan un soporte práctico para el desarrollador. Algunas demandan mediciones que son demasiado complejas y otras violan las nociones básicas intuitivas de lo que realmente es el software de alta calidad.

Para que sea útil en el contexto del mundo real, una métrica del software debe ser objetiva, simple y calculable, consistente en el empleo de unidades y tamaños, persuasiva, además debería ser independiente del lenguaje de programación y proporcionar una realimentación eficaz para el desarrollador de software. (PRESSMAN 2002)

Ejiogu define un conjunto de atributos que deberían acompañar a las métricas efectivas del software: (PRESSMAN 2002)

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

- **Consistentes y objetivas:** La métrica debería siempre producir resultados sin ambigüedad. Un tercer equipo debería ser capaz de obtener el mismo valor de métrica usando la misma información del software.
- **Consistentes en el empleo de unidades y tamaños:** El cálculo matemático de la métrica debería emplear medidas que no conduzcan a extrañas combinaciones de unidades. Por ejemplo, multiplicando el número de personas de un equipo por las variables del lenguaje de programación en el programa resulta una sospechosa mezcla de unidades que no son intuitivamente persuasivas.
- **Independientes del lenguaje de programación:** Las métricas deberían basarse en el modelo de análisis, modelo de diseño o en la propia estructura del programa. No deberían depender de los caprichos de la sintaxis o semántica del lenguaje de programación.
- **Un eficaz mecanismo para la realimentación de calidad:** La métrica debería proporcionar, al desarrollador de software, información que le lleve a un producto final de mayor calidad.

Aunque la mayoría de las métricas de software satisfacen las características anteriores, algunas de las empleadas comúnmente dejan de cumplir una o dos.

Las métricas deben ser una herramienta que ayude a mejorar el proceso, producto o proyecto de software. No es conveniente aplicar métricas que lejos de ayudar a los desarrolladores constituyan un freno; bien por ser demasiado complejas, porque no se entiendan correctamente los objetivos que persiguen o porque arrojen resultados imprecisos que no puedan ser interpretados por los ingenieros de software.

1.6.2. Clasificación de las métricas de software

Hay, en la actualidad, múltiples métricas con propósitos diferentes que muestran o describen la conducta del software. Estas pueden medir, entre otros aspectos; la competencia, calidad, desempeño y la complejidad del software. Contribuyendo a establecer de una manera metódica y objetiva una visión interna del trabajo, mejorando así la calidad del producto.

Las métricas del software han sido clasificadas de distintas formas por los estudiosos del tema.

Pressman clasifica el campo de las métricas en seis categorías o grupos de métricas distintos: (PRESSMAN 2002)

- **Métricas técnicas:** Miden la estructura del sistema, el cómo esta hecho, es decir, están centradas en las características del software más que en su proceso de desarrollo.
- **Métricas de calidad:** Proporcionan una indicación de cómo se ajusta el software a los requisitos implícitos y explícitos del cliente.
- **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 al tamaño:** Es para saber en qué tiempo se va a terminar el software y cuantas personas se van a necesitar, son medidas directas al software y al proceso por el cual se desarrolla.
- **Métricas orientadas a la función:** Son medidas indirectas del software y del proceso por el cual se desarrolla, se centran en la funcionalidad o utilidad del programa.
- **Métricas orientadas a la persona:** Proporcionan medidas e información sobre la forma en que las personas desarrollan el software, son las medidas que se van a hacer del personal que hará el sistema.

Otras formas de clasificaciones:

- **Directas o Indirectas** (MUÑOZ 2006)
    - Directa: Una métrica de la cual se pueden realizar mediciones sin depender de ninguna otra métrica y cuya forma de medir es un método de medición.
    - Indirecta: Una métrica cuya forma de medir es una función de cálculo, es decir, las mediciones de dicha métrica utilizan las medidas obtenidas en mediciones de otras métricas directas o indirectas.
- **Internas o Externas** (AUTORES 2004)
    - Interna: Puede ser aplicada a un producto de software no ejecutable (como una especificación o código fuente) durante el diseño y la codificación.
    - Externa: Usa medidas de un producto de software, derivadas del comportamiento del mismo, a través de la prueba, operación y observación del software.
- **Dinámicas o Estáticas** (ALONSO 2004)
    - Dinámicas: Son aquellas recogidas por las mediciones hechas en un programa en ejecución en relación directa con los atributos de calidad del software.
    - Estáticas: Son aquellas recogidas por las mediciones hechas en las representaciones del sistema como el diseño, el código y la documentación.

- **De Proyecto, del Producto o de Procesos**

Se hará énfasis en esta última clasificación por ser de gran importancia para la investigación en curso.

Métricas del Proyecto

Las métricas del proyecto se utilizan para minimizar la planificación del desarrollo haciendo los ajustes necesarios que eviten retrasos, reduzcan problemas y riesgos potenciales. Son manejadas para evaluar la calidad de los productos en el momento actual y cuando sea necesario, modificando el enfoque técnico que mejore la calidad. (PRESSMAN 2002)

Métricas del Producto

Los productos están compuestos por artefactos (documentos, modelos, módulos, o componentes), por tanto, las métricas del producto deben hacerse sobre la base de medir cada uno de los artefactos. Las métricas del producto describen características como el tamaño, la complejidad, los rasgos del diseño, el rendimiento y el nivel de calidad. (ESTRADA and ESTÉVEZ 2003)

Métricas de Proceso

La recolección de métricas del proceso es esencial para la mejora del mismo, incluso en proyectos a pequeña escala. Se utilizan para evaluar la eficiencia de un proceso o si este ha mejorado con los cambios realizados. Los indicadores del proceso permiten al gestor, evaluar lo que funciona y lo que no; y a la organización, tener una visión profunda de la eficacia de un proceso ya existente**.** (ILINA and GONZÁLEZ 2007)

Las métricas de procesos constituyen el eje central de la presente investigación, de ellas se abordará en el siguiente epígrafe.

1.7. Métricas para el proceso

La medición del proceso implica las mediciones de las actividades relacionadas con el software, siendo algunos de sus atributos típicos: el esfuerzo, el coste y los defectos encontrados.

Las métricas permiten tener una visión profunda del proceso de software que ayudará a tomar decisiones más fundamentadas. Permiten analizar el trabajo desarrollado, conocer si se ha mejorado o no con respecto a los anteriores, ayudan a detectar áreas con problemas para poder remediarlos a tiempo y a realizar mejores estimaciones.

Para mejorar un proceso se deben medir los atributos del mismo, desarrollar métricas de acuerdo a estos atributos y utilizarlas para proporcionar indicadores que conduzcan a la mejora del proceso. Los errores detectados antes de la entrega del software, la productividad, recursos y tiempo consumido, y ajuste con la planificación son algunos de los resultados que pueden medirse en el proceso, así como las tareas específicas de la ingeniería del software. (HERNÁNDEZ and NÚÑEZ 2007)

Las métricas de procesos del software son algo más que una simple medida de algún "atributo" del proceso de software; una métrica es información que sirve para planificar, predecir y evaluar el estado de un proyecto. (VARGAS 2005)

Las métricas pueden proporcionar beneficios significativos a medida que una organización trabaja por mejorar su nivel global de madurez del proceso. Sin embargo, al igual que todas las métricas, estas pueden ser mal utilizadas, originando más problemas de los que pueden solucionar. Grady sugiere una "etiqueta de métricas del software" adecuada para gestores al tiempo que instituyen un programa de métricas de proceso: (GRADY 1992)

- Utilice el sentido común y una sensibilidad organizativa al interpretar datos de métricas.
- Proporcione una retroalimentación regular para particulares y equipos que hayan trabajado en la recopilación de medidas y métricas.
- No utilice métricas para evaluar a particulares.
- Trabaje con profesionales y equipos para establecer objetivos claros y métricas que se vayan a utilizar para alcanzarlos.
- No utilice nunca métricas que amenacen a particulares o equipos.
- Los datos de métricas que indican un área de problemas no se deberían considerar "negativos". Estos datos son meramente un indicador de mejora de proceso.
- No se obsesione con una sola métrica y excluya otras métricas importantes.

A continuación se especificarán por categorías distintos tipos de métricas de procesos.

1.7.1. Categoría del Proceso del Proyecto

Los procesos que pertenecen a esta categoría son utilizados en la gestión de cualquier tipo de proyecto.

1.7.1.1. Error de la Planificación

Un método para medir la calidad de la planificación es durante el desarrollo de un trabajo o cuando se termine, tomar las mediciones reales de tiempo, costo o tamaño, y calcular el error entre las horas reales y las planificadas, el costo real y el planificado; o entre las líneas de código real y estimadas, respectivamente.

El error de la planificación se puede calcular como:

$$\text{Error } \% = \frac{(\text{Real} - \text{Estimado}) * 100}{\text{Estimado}}$$

Basándose en que:

$$\text{Error} = \text{Real} - \text{Estimado}$$

Será mejor la calidad de la planificación cuando el error esté oscilando en valores cercanos a cero. (ESTRADA and ESTÉVEZ 2003)

1.7.1.2. Razón de Costo de Planificación

La razón de costo de planificación (RCP) es una de las formas de medir la calidad de la planificación; se puede analizar por tiempo, dinero o líneas de código.

$$\text{RCP} = \frac{\text{CP}}{\text{CR}}$$

donde:

CP: costo planificado.

CR: costo real

En caso de que dicha razón se iguale a 1, significa que se gasta en planificar lo mismo que se gasta en desarrollar, o sea el costo no varía.

En caso de que la RCP sea menor que 1, se evidencia pobre desempeño del proyecto, se gasta más de lo que se gana.

Idealmente el RCP debe ser ligeramente mayor que 1, lo cual demostraría un buen desempeño del proyecto, o sea se gasta menos de lo que se gana. No obstante habría que analizar diferencias abismales, las cuales también pueden significar que los planes están siendo muy conservadores. Un valor más o menos adecuado de RCP es 0.8. (ESTRADA and ESTÉVEZ 2003)

1.7.2. Categoría del Proceso de Soporte

Los procesos que dan soporte al resto de los procesos, en distintos puntos del ciclo de vida del software pertenecen a la categoría del Proceso de Soporte.

1.7.2.1. Métricas del Mantenimiento

El estándar IEEE sugiere un índice de madurez del software (IMS) que proporciona una indicación de la estabilidad de un producto software (basada en los cambios que ocurren con cada versión del producto). Se determina la siguiente información:

M_t= número de módulos en la versión actual

F_c = número de módulos en la versión actual que se han cambiado

F_a = número de módulos en la versión actual que se han añadido

F_d = número de módulos de la versión anterior que se han borrado en la versión actual

El índice de madurez del software se calcula de la siguiente manera:

$$\mathrm{IMS} = [\mathrm{M_t} - (\mathrm{F_a} + \mathrm{F_c} + \mathrm{F_d})]/\mathrm{M_t}$$

A medida que el IMS se aproxima a 1.0 el producto se empieza a estabilizar. EL IMS puede emplearse también como métrica para la planificación de las actividades de mantenimiento del software. El tiempo medio para producir una versión de un producto software puede correlacionarse con el IMS desarrollándose modelos empíricos para el mantenimiento. (PRESSMAN 2002)

1.7.3. Categoría del Proceso de Producción

En la categoría de los Procesos de Producción se encuentran todos aquellos procesos que directamente especifican, implementan o mantienen el producto de software.

1.7.3.1. Métricas del Diseño a Nivel de Componentes

Las métricas de diseño a nivel de componentes se concentran en las características internas de los componentes del software e incluyen medidas de las “3Cs” (1) la cohesión, (2) el acoplamiento y (3) la complejidad del módulo. Estas tres medidas pueden ayudar al desarrollador de software a juzgar la calidad de un diseño a nivel de los componentes.

Las métricas presentadas en esta sección son de caja blanca en el sentido de que requieren conocimiento del trabajo interno del módulo en cuestión. Las métricas de diseño de los

componentes se pueden aplicar una vez que se ha desarrollado un diseño procedimental. También se pueden retrasar hasta tener disponible el código fuente.

1.7.3.1.1. Métricas de Cohesión

Bieman y Ott definen una colección de métricas que proporcionan una indicación de la cohesión de un módulo. Las métricas se definen con cinco conceptos y medidas: (PRESSMAN 2002)

- **Porción de datos:** Dicho simplemente, una porción de datos es una marcha atrás a través de un módulo que busca valores de datos que afectan a la localización de módulo en el que empezó la marcha atrás. Debería resaltarse que se pueden definir tanto porciones de programas (que se centran en enunciados y condiciones) como porciones de datos.
- **Muestras (tokens) de datos:** Las variables definidas para un módulo pueden definirse como muestras de datos para el módulo.
- **Señales de unión:** El conjunto de muestras de datos que se encuentran en una o más porciones de datos.
- **Señales de superunión:** Las muestras de datos comunes a todas las porciones de datos de un módulo.
- **Pegajosidad:** La pegajosidad relativa de una muestra de unión es directamente proporcional al número de porciones de datos que liga.

Bieman y Ott desarrollan métricas para cohesiones funcionales fuertes (CFF), cohesiones funcionales débiles (CFD) y pegajosidad (el grado relativo con el que las señales de unión ligan juntas porciones de datos). Estas métricas se pueden interpretar de la siguiente manera. (PRESSMAN 2002)

Las métricas de cohesión tienen valores que van de 0 a 1. Tienen un valor de 0 cuando un procedimiento tiene más de una salida y no muestra ningún atributo de cohesión indicado por una métrica particular. Un procedimiento sin muestras de superunión (sin muestras comunes a todas las porciones de datos), no tiene una cohesión funcional fuerte (no hay señales de datos que contribuyan a todas las salidas). Un procedimiento sin muestras de unión, es decir, sin muestras comunes a más de una porción de datos (en procedimientos con más de una porción de datos), no muestra una cohesión funcional débil y ninguna adhesividad (no hay señales de datos que contribuyan a más de una salida).

La cohesión funcional fuerte y la pegajosidad se obtienen cuando las métricas de Bieman y Ott toman un valor máximo de 1.

Para ilustrar el carácter de estas métricas, se considera la métrica para la cohesión funcional fuerte:

$$\mathrm{CFF(i)} = \mathrm{SU(SA(i))/muestra(i)}$$

donde SU (SA(i)) denota muestra de superunión (el conjunto de señales de datos que se encuentran en todas las porciones de datos de un módulo i). Como la relación de muestras de superunión con respecto al número total de muestras en un módulo i aumenta hasta un valor máximo de 1, la cohesión funcional del módulo también aumenta.

1.7.3.1.2. Métricas de Acoplamiento

El acoplamiento de módulo proporciona una indicación de la "conectividad" de un módulo con otros módulos, datos globales y el entorno exterior.

Dhama ha propuesto una métrica para el acoplamiento del módulo que combina el acoplamiento de flujo de datos y de control, acoplamiento global y acoplamiento de entorno. Las medidas necesarias para calcular el acoplamiento de módulo se definen en términos de cada uno de los tres tipos de acoplamiento apuntados anteriormente. (PRESSMAN 2002)

Para el acoplamiento de flujo de datos y de control:

d_i = número de parámetros de datos de entrada

c_i = número de parámetros de control de entrada

d_0 = número de parámetros de datos de salida

c_0 = número de parámetros de control de salida

Para el acoplamiento global:

g_d = número de variables globales usadas como datos

g_c = número de variables globales usadas como control

Para el acoplamiento de entorno:

w = número de módulos llamados (expansión)

r = número de módulos que llaman al módulo en cuestión (concentración)

Utilizándose estas medidas, se define un indicador de acoplamiento de módulo, m_c, de la siguiente manera:

$$m_c = k/m$$

donde k=1 es una constante de proporcionalidad.

$$M = d_i + a * c_i + d_0 + b * c_0 + g_d + c * g_c + w + r$$

donde a=b=c=2 .

Cuando mayor sea el valor de m_c, menor es el acoplamiento de módulo. Por ejemplo, si un módulo tiene un solo parámetro de entrada y salida de datos, no accede a datos globales y es llamado por un solo módulo:

$$m_c = \frac{1}{1+0+1+0+0+0+1+0} = \frac{1}{3} = 0.33$$

Se debería esperar que un módulo como este presentara un acoplamiento bajo. De ahí que, un valor de m_c=0.33 implica un acoplamiento bajo. Por el contrario, si un módulo tiene 5 parámetros de salida y 5 parámetros de entrada de datos, un número igual de parámetros de control, accede a 10 elementos de datos globales y tiene una concentración de 3 y una expansión de 4,

$$m_c = \frac{1}{5+2*5+5+2*5+10+0+3+4} = 0.02$$

el acoplamiento implicado es grande.

Para que aumente la métrica de acoplamiento a medida que aumente el grado de acoplamiento, se puede definir una métrica de acoplamiento revisada como:

$$C = 1 - m_c$$

donde el grado de acoplamiento no aumente linealmente entre un valor mínimo en el rango de 0.66 hasta un máximo que se aproxima a 1.0.

1.7.3.1.3. Métricas de Complejidad

Se pueden calcular una variedad de métricas del software para determinar la complejidad del flujo de control del programa. Muchas de estas se basan en una representación denominada grafo de flujo. Un grafo es una representación compuesta de nodos y enlaces (también denominados aristas). Cuando se dirigen los enlaces (aristas), el grafo de flujo es un grafo dirigido.

McCabe identifica un número importante de usos para las métricas de complejidad: (PRESSMAN 2002)

Las métricas de complejidad pueden emplearse para predecir la información crítica sobre la fiabilidad y mantenimiento de sistemas software de análisis automáticos de código fuente (o información de diseño procedimental). Las métricas de complejidad también realimentan la información durante el proyecto de software para ayudar a controlar la "actividad del diseño". Durante las pruebas y el mantenimiento, proporcionan una detallada información sobre los módulos software para ayudar a resaltar las áreas de inestabilidad potencial.

La métrica de complejidad más ampliamente usada (y debatida) para el software es la *complejidad ciclomática,* originalmente desarrollada por Thomas McCabe. (PRESSMAN 2002)

La métrica de McCabe proporciona una medida cuantitativa para probar la dificultad y una indicación de la fiabilidad última. Estudios experimentales indican una fuerte correlación entre la métrica de McCabe y el número de errores que existen en el código fuente, así como el tiempo requerido para encontrar y corregir dichos errores.

McCabe también defiende que la complejidad ciclomática puede emplearse para proporcionar una indicación cuantitativa del tamaño máximo del módulo. Recogiendo datos de varios proyectos de programación reales, ha averiguado que una complejidad ciclomática de diez parece ser el límite práctico superior para el tamaño de un módulo. Cuando la complejidad ciclomática de los módulos excedía ese valor, resultaba extremadamente difícil probar adecuadamente el módulo.

1.7.3.2. Métrica de la Complejidad Ciclomática

La complejidad ciclomática es una métrica del software que proporciona una medición cuantitativa de la complejidad lógica de un programa. Cuando se usa en el contexto del método de prueba del camino básico, el valor calculado como complejidad ciclomática define el número de caminos independientes del conjunto básico de un programa y da un límite superior para el número de pruebas que se deben realizar para asegurar que se ejecuta cada sentencia al menos una vez.

La complejidad ciclomática está basada en la teoría de grafos y da una métrica del software extremadamente útil. La complejidad se puede calcular de tres formas:

1. El número de regiones del grafo de flujo coincide con la complejidad ciclomática.
2. La complejidad ciclomática, V(G), de un grafo de flujo G se define como:

$$V(G) = A - N + 2$$

donde A es el número de aristas del grafo de flujo y N es el número de nodos del mismo.

3. La complejidad ciclomática, V(G), de un grafo de flujo G también se define como:

$$V(G) = P + 1$$

donde P es el número de nodos predicado contenidos en el grafo de flujo G.

1.7.3.3. Métricas de la Eficacia de la Eliminación de Defectos (EED)

La eficacia de la eliminación de defectos (EED) es una métrica de la calidad que proporciona beneficios tanto a nivel del proyecto como del proceso. En esencia, es una medida de la habilidad de filtrar las actividades de la garantía de calidad y de control al aplicarse a todas las actividades del marco de trabajo del proceso.

Cuando un proyecto se toma en consideración globalmente, EED se define de la forma siguiente:

$$EED = E/(E + D)$$

donde E es el número de errores encontrados antes de la entrega del software al usuario final y D es el número de defectos encontrados después de la entrega.

El valor ideal de EED es 1. Esto es, no se han encontrado defectos en el software. De forma realista, D será mayor que 0, pero el valor de EED todavía se puede aproximar a 1. Cuando E aumenta (para un valor de D dado), el valor total de EED empieza a aproximarse a 1. De hecho, a medida que E aumenta, es probable que el valor final de D disminuya (los errores se filtran antes de que se conviertan en defectos). Si se utilizan como una métrica que proporciona un indicador de la habilidad de filtrar las actividades de la garantía de la calidad y del control, EED anima a que el equipo del proyecto de software instituya técnicas para encontrar todos los errores posibles antes de su entrega. EED también se puede utilizar dentro del proyecto para evaluar la habilidad de un equipo en encontrar errores antes de que pasen a la siguiente actividad estructural o tarea de ingeniería del software. Por ejemplo, la tarea del análisis de los requisitos produce un modelo de análisis que se puede revisar para encontrar y corregir errores.

Esos errores que no se encuentren durante la revisión del modelo de análisis se pasan a la tarea de diseño (en donde se pueden encontrar o no). Cuando se utilizan en este contexto, EED se vuelve a definir como:

$$EED_i = E_i/(E_i + E_{i+1})$$

en donde E_i es el número de errores encontrado durante la actividad de ingeniería del software i y E_{i+1} , es el número de errores encontrado durante la actividad de ingeniería del software i + I que se puede seguir para llegar a errores que no se detectaron en la actividad de la ingeniería del software i. (PRESSMAN 2002)

1.7.3.4. Métricas del Ciclo de Tiempo

El ciclo de tiempo es una medida que proporciona el tiempo que se emplea para llevar a cabo un proceso. Hay dos medidas de tiempo:

1. Ciclo de tiempo estático: Se utiliza el tiempo promedio actual que se emplea para ejecutar el proceso. Por ejemplo qué tiempo emplea un módulo en corregir una falla y ejecutar un caso de prueba.
2. Ciclo de tiempo dinámico: Es calculado al dividir el número de elementos en progreso (elementos que únicamente han completado el proceso parcialmente) entre los nuevos elementos comenzados y nuevos elementos terminados durante el período.

Conociendo el ciclo de tiempo para los subprocesos en el proceso de desarrollo de software se podrá hacer una mejor estimación del plan y de los recursos requeridos, también permite monitorear el impacto de las actividades del proceso en el mejoramiento del ciclo de tiempo para este proceso. (DORIA 2001)

El siguiente ejemplo ilustra el cálculo de métricas del ciclo de tiempo estático y del ciclo de tiempo dinámico:

Ciclo de tiempo estático:

Calendario de tiempo para completar el proceso:

Módulo A: 5 días

Módulo B: 10 días

Módulo C: 7 días

Módulo D: 8 días

(5+10+7+8)/4 = 7.5 días

Ciclo de tiempo dinámico:

Total de elementos en progreso / (elementos iniciados + elementos completados)/ 2

52 módulos iniciados en este mes

68 módulos terminados en este mes

12 módulos en progreso en este mes

(12/ ((52+68) /2)) * 30 días en el mes = 6 días

1.7.3.5. Métricas del PSP (Proceso de Software Personal)

1.7.3.5.1. La relación Valoración/Fallos (V/F)

Según el PSP, es sencillo calcular el valor V/F como el tiempo de revisión de código dividido por el total del tiempo de compilación y pruebas.

$$\mathrm{VF} = \frac{\text{Tiempo de revision de código}}{\text{Total del tiempo de compilación y pruebas}}$$

El valor de V/F mide la cantidad relativa de tiempo dedicado a encontrar defectos antes de la primera compilación, es un buen indicador de la probabilidad de encontrar defectos en las pruebas.

Cuando el valor de V/F es menor que 1, las pruebas del programa generalmente encuentran muchos defectos. Aunque esto no está garantizado, muchos programas en este rango tendrán un valor bastante alto de defectos/KLOC (1000 líneas de código) en las pruebas. (HUMPHREY 2001)

1.7.3.6. Métricas de Defectos

Durante el desarrollo del proceso de control de configuración, muchos de los cambios solicitados son producto de defectos en el software o el prototipo entregado al usuario final. Por esta razón es importante controlar los defectos que aparecen de esta manera en el producto.

Tabla 2 Resumen de Defectos

Etapa	Insertados	Eliminados	Acumulativo Insertados	Acumulativo Eliminados	Escapes Netos	Rendimiento
Planificación	1	0	1	0	1	
Diseño	5	0	6	0	6	
Inspección Diseño	0	3	6	3	3	50%
Codificación	15	1	21	4	17	
Inspección Código	0	8	21	12	9	47.1%
Prueba	0	6	21	18	3	
Control de	0	3	21	21	0	

Cambio						
Total	21	21				57.1%

Nótese que con la tabla de resumen de defectos, no se observa el avance del rendimiento, solamente su valor final. Sin embargo el método de calcularlo es mucho más fácil, el rendimiento por cada etapa se podrá calcular por:

$$\text{Rendimiento} = \frac{\text{Defectos Eliminados} * 100}{\text{Defectos Eliminados} + \text{Escapes Netos}}$$

El rendimiento total del proceso se calcula como:

$$\text{RTP} = \frac{\text{Defectos Eliminados antes prueba} * 100}{\text{Defectos Eliminados antes prueba} + \text{Escapes antes de Prueba}}$$

donde RTP es el Rendimiento total del proceso

Debe quedar claro que un valor de rendimiento alto es mucho mejor que un valor bajo, el cual es un resultado pobre. El objetivo o meta es un rendimiento de 100%, o sea que se traten de eliminar los defectos tan pronto como sea posible. (ESTRADA and ESTÉVEZ 2003)

1.7.3.6.1. Defectos por hora

Los defectos encontrados por hora en una fase específica indican la efectividad del tiempo dedicado a las inspecciones. En la medida que el rendimiento incrementa, es natural una disminución de los defectos por unidad de tiempo. (ESTRADA and ESTÉVEZ 2003)

Los defectos eliminados por unidad de tiempo en una etapa específica se pueden calcular como:

$$\text{Defectos encontrados por hora} = \frac{\text{Defectos eliminados en la etapa} * 60}{\text{Minutos dedicados a esa etapa}}$$

(Cualquier otra conversión de unidad de tiempo es válida)

1.7.3.7. Métricas relacionadas con la Calidad

Muy relacionadas con la definición de densidad de defectos del software se encuentran otras medidas que se agrupan con el nombre de "tasas". Son indicadores de la calidad del proceso, algunas de las medidas más utilizadas son:

$$\text{TPD} = \frac{\text{Número defectos ocasionados al corregir defectos}}{\text{Número de defectos corregidos}}$$

donde TPD es la tasa de propagación de defectos.

$$\text{TEP} = \frac{\text{Número de objetos probados al menos una vez} * 100}{\text{Número de objetos totales}}$$

donde TEP es la tasa de efectividad de las pruebas.

Con las ecuaciones anteriores se pueden realizar una política de pruebas y captura de medidas muy valiosas para el desarrollo de aplicaciones y su control de calidad. (MELIÁN and BALLESTEROS 2003)

1.8. Conclusiones

Luego de realizar un estudio de los conceptos teóricos relacionados con las métricas del software, se puede aseverar la importancia que tiene la aplicación de métricas en la evaluación de la calidad del software, principalmente si se utilizan orientadas a los procesos de desarrollo.

Para realizar una buena medición de los procesos es necesario centrarse en los atributos que se muestran en la siguiente porción de la tabla 1(Ver epígrafe 1.5), pues los restantes se refieren a medidas del producto, y tenerlos en cuenta entorpecería el desarrollo de la presente investigación.

Tabla 3 Resumen de atributos a evaluar

Atributo	Directo	Indirecto	Entidad
Coste		X	Proceso (diseño, pruebas, captura de especificaciones…)
Esfuerzo	x		Proceso (diseño, pruebas, captura de especificaciones…)
Productividad		X	Recurso (equipo de trabajador, programador, hardware…)
Calidad		X	Proceso (creación de especificaciones…)

Por ello las métricas que se tomarán en la propuesta de solución, estarán orientadas a la medición de estos atributos para la evaluación de procesos.

De la selección realizada en el epígrafe 1.7 se utilizarán todas aquellas métricas que se adapten a los procesos de desarrollo de SWE propuestos en la tesis de maestría “Propuesta del proceso para el desarrollo de proyectos de Software Educativo en la Universidad de las Ciencias Informáticas”, que se tienen en cuenta en el modelo de evaluación MEPDSE.

La siguiente tabla muestra las métricas que se utilizarán en la elaboración de la propuesta.

Tabla 4 Selección de métricas para la Propuesta

Métrica	Selección		Justificación de la Selección
	Si	No	
Error de la Planificación	X		Puede ser utilizada para obtener información sobre los errores cometidos durante el proceso de Planificación.
Razón de Costo de Planificación	X		Muestra la calidad con la que se ha desarrollado el proceso de planificación en un proyecto.
Métricas del Mantenimiento	X		Indica la estabilidad de un producto software basada en los cambios que ocurren con cada versión.
Métricas de Cohesión		X	
Métricas de Acoplamiento	X		Muestra si se mantuvo un bajo acoplamiento en el diseño.
Métricas de Complejidad		X	
Métrica de la Complejidad Ciclomática	X		Da un valor límite superior para el número de pruebas que se deben diseñar y ejecutar para garantizar que se cubren todas las sentencias del programa.
Métricas de la Eficacia de la Eliminación de Defectos	X		Es una medida de la habilidad de filtrar las actividades de la garantía de calidad y de control al aplicarse a todas las actividades del marco de trabajo del proceso.
Métricas del Ciclo del Tiempo		X	
La relación Valoración/Fallos		X	
Defectos por hora		X	
Tasa de Propagación de Defectos	X		Indica el comportamiento de la propagación de defectos al intentar corregirlos durante el proceso de Prueba.
Tasa de Efectividad de las Pruebas	X		Informa sobre el porciento de objetos que no han sido probados en el software.

Una vez analizada y seleccionadas las métricas existentes se concluye que la propuesta de solución quedará conformada por:

- El procedimiento para la evaluación de los procesos que utiliza el modelo de evaluación MEPDSE usando las métricas propuestas.
- El conjunto de métricas que incluirá las seleccionadas y otras elaboradas que ayuden a cubrir la evaluación de los procesos de desarrollo.

2

CAPÍTULO 2: DESCRIPCIÓN DE LA PROPUESTA DE SOLUCIÓN

2.1. Introducción

En este capítulo se describe detalladamente el conjunto de métricas para evaluar los procesos de desarrollo de SWE, que se tienen en cuenta en el modelo de evaluación MEPDSE. El conjunto propuesto quedará conformado por las métricas seleccionadas en el capítulo anterior que se adecuan a los procesos que se pretenden evaluar, algunas se toman al igual que las describen los autores y otras se modifican. Para completar la evaluación de todos los procesos fue necesaria la elaboración de nuevas métricas, así como la utilización de un procedimiento de evaluación que permitiera dar una calificación a los procesos empleando las métricas propuestas.

2.2. Descripción de la Encuesta

Para la asignación de pesos, a las actividades o características de cada uno de los procesos que se van a evaluar con las métricas propuestas, fue necesario aplicar una encuesta (Ver Anexo 1). Dicha encuesta permite conocer el nivel de importancia que poseen las diferentes actividades o características en la evaluación de la calidad de un SWE.

Las preguntas están diseñadas para conocer la importancia que le conceden los especialistas de calidad de la UCI a los aspectos que se quieren medir dentro de los procesos que toma en consideración el modelo MEPDSE para realizar la evaluación.

En la encuesta se utilizó una combinación de preguntas abiertas y cerradas, con el objetivo de que los encuestados contaran con una guía que les permitiera seguir la línea de la investigación; pero siempre se dejaron opciones para que pudieran expresar sus criterios.

2.2.1. Análisis de la Encuesta

La encuesta se le aplicó a los 8 especialistas de la dirección de calidad de la UCI, quienes representan el 100% de la población seleccionada. Una vez procesada se obtuvieron

diversos resultados, en dependencia de la importancia que cada encuestado le concedió a la medición en los procesos de desarrollo de SWE.

Pregunta 1 y 2

El 100% de las personas responden afirmativamente sobre, la importancia de la medición y su experiencia en asignaciones de pesos.

Pregunta 3

La pregunta arrojó que el 87.5% de los encuestados le conceden el mismo nivel de importancia a los aspectos que se miden con las métricas planteadas para el proceso de Gestión de Riesgo.

Pregunta 4

Para el proceso de Planificación y Seguimiento el 75% de los especialistas coinciden en que medir el desarrollo realista de la planificación y el control de las tareas retrasadas, en un proyecto, tienen igual nivel de importancia.

Pregunta 5

El 62% de los especialistas concuerdan en que la medición del control de los datos del personal, en el proceso de Gestión de los Recursos, tiene menor importancia en el logro de la calidad de este proceso, que la medición de las actividades, aprovechamiento de los componentes reutilizables y la estimación realista de los recursos.

Pregunta 6

El 62% de los encuestados señalaron un nivel mayor de importancia a la medición de, la eficiencia en la obtención de los requisitos y la agilidad en la gestión de cambio de los mismos que, a la productividad en la corrección de errores en el proceso de Realización del Flujo de Trabajo Gestión de los Requisitos.

Pregunta 7

En cuanto al proceso de Gestión de Configuración y Cambios el 87.5% de los encuestados le asignaron menor importancia a la medición de la eficacia en la información de los cambios al personal implicado, en un proyecto, que a la medición de, el índice de madurez del SW y el estado de los pedidos de cambio.

Pregunta 8

El 100% de los especialistas opinan que las tres actividades que se miden para evaluar el proceso de Aseguramiento de la Calidad tienen el mismo nivel de importancia.

Pregunta 9

El 75% de los encuestados coinciden en que medir la habilidad de un equipo en encontrar errores (antes de que pasen a la siguiente actividad o tarea de Ingeniería del SW) y controlar los casos de usos (CU) diseñados (para alcanzar la calidad en el proceso de Realización del Flujo de Trabajo de Análisis y Diseño) es más importante que controlar el bajo acoplamiento en los módulos.

Pregunta 10

En el proceso de Realización del Flujo de Trabajo de Implementación el 87.5% le concede mayor nivel de importancia al control de los CU implementados que al control de las pruebas de unidad. Siendo mínima la diferencia entre los niveles.

Pregunta 11

El 75% piensa que para alcanzar gran calidad en el proceso de Realización del Flujo de Trabajo de Prueba no es tan importante la medición del porciento de pruebas diseñadas con respecto al valor de la complejidad ciclomática.

2.3. Procedimiento para la Evaluación de los Procesos de Desarrollo SWE

El procedimiento para la evaluación está basado en una fórmula perteneciente al campo de la lógica difusa llamada Coeficiente de Adecuación (CA).

El CA representa el grado de competencia de acuerdo con unos mínimos exigibles en perfiles idóneos sin importar el hecho de superarlos. Un grado de competencia del 100% se alcanzaría siempre que todas las componentes del perfil fueran iguales o superiores al perfil idóneo.(GARCÉS and VILLAR 2007)

Para utilizar el CA es necesario expresar el resultado de las métricas, aproximadamente, en un rango de 0 a 1. Además se debe establecer un valor ideal en cada una de las métricas. Este será el valor a partir del cual la métrica arrojará resultados deseables sobre lo que se está evaluando.

A continuación se expresa la fórmula del Coeficiente de Adecuación.

$$\mathbf{CA_j = 1 \wedge (1 - u_{kj} + u_{ij})}$$

donde:

u_{kj}: es el valor ideal especificado a la métrica j.

u_{ij}: valor real obtenido al calcular la métrica j.

∧: representa el mínimo entre dos valores.

Se debe registrar los datos de todos los procesos que se pretendan evaluar en una tabla como la que se muestra. (Ver tabla 5)

Tabla 5 Registro de datos para la evaluación del proceso

Proceso X	Métrica 1	Métrica 2	Métrica 3
Valor Ideal (K)	0.75	0.85	0.75
Valor Real (i)	0.65	0.90	1

En la tabla se debe especificar el nombre del proceso, así como cada una de las métricas que se proponen para evaluarlo.

Para el empleo del procedimiento es imprescindible el cumplimiento de los siguientes pasos. Para un mayor entendimiento se explican a través de un ejemplo en el cual se utilizarán los valores que se especificaron en la tabla 5.

Primer paso:

Cálculo del CA en cada una de las métricas que se plantean para un proceso X.

Para la métrica 1 se calcula:

$$\mathbf{CA_j = 1 \wedge (1 - u_{kj} + u_{ij})}$$

$$\mathbf{CA_j = 1 \wedge (1 - 0.75 + 0.65)}$$

$$\mathbf{CA_j = 1 \wedge (0.90)}$$

$$\mathbf{CA_j = 0.90}$$

De la misma forma se calcula para las demás métricas del proceso en cuestión.

Segundo paso:

Multiplicar cada uno de los coeficientes de adecuación por el peso relativo que se le haya asignado a la métrica correspondiente.

Métrica 1:

$$\mathrm{CA}_1 * \mathrm{Peso} = 0.90 * 0.35 = 0.315$$

Tercer paso:

Realizar una sumatoria de los valores obtenidos en cada una de las métricas

Métrica 1: 0.315

Métrica 2: 0.35

Métrica 3: 0.30

$$S = 0.315 + 0.35 + 0.30 = 0.965$$

Cuarto paso:

Comparar el valor resultante con la siguiente escala para obtener una evaluación final.

Tabla 6 Escala para la evaluación final de procesos

Escala	Evaluación
$0 \leq S < 0.60$	M
$0.60 \leq S < 0.75$	R
$0.75 \leq S \leq 1$	B

Finalmente, como el valor del ejemplo resultó estar en el rango entre 0.75 y 1 el proceso queda evaluado de bien.

2.4. Métricas para la Evaluación de los Procesos de Desarrollo de SWE

Las métricas definidas para la evolución de los procesos de desarrollo de SWE se organizan por las diferentes categorías a las que pertenecen estos procesos. Cada una de ellas indica la calidad con que se realizaron los mismos y se detallan a continuación.

2.4.1. Categoría del Proceso del Proyecto

2.4.1.1. Gestión de Riesgo

La Gestión de Riesgo es un proceso en el cual se desarrollan una serie de pasos que ayudan al equipo de software a gestionar la incertidumbre.

Un riesgo es un problema potencial que puede o no ocurrir. Por lo que es realmente importante identificarlo, evaluar su probabilidad de aparición, estimar su impacto y establecer un plan de contingencia por si ocurre el problema.

A continuación se definen las métricas que se utilizarán para la evaluación del proceso Gestión de Riesgo.

2.4.1.1.1. Agilidad en la Preparación para el Manejo de Riesgos

En el proceso de Gestión de Riesgo uno de los primeros pasos lo constituye la preparación para el manejo de riesgos. Durante esta actividad se definen las principales fuentes y parámetros de riesgos y se establecen las estrategias necesarias para su manejo.

La métrica para determinar la Agilidad en la Preparación para el Manejo de Riesgos (AMR) brinda información sobre la relación que existe entre las fuentes de riesgos y las estrategias que se establecen para manejarlos.

La AMR se define:

$$\mathrm{AMR} = (\frac{\mathrm{Est_i}}{\mathrm{F_i}})/10$$

donde:

F_i: fuentes de riesgos (siempre va a ser 1).

Est_i: número de estrategias para el manejo de riesgos de la fuente i.

La AMR puede resultar cualquier valor entre 0 y 1, ello va a depender de la cantidad de estrategias que se tracen para cada fuente de riesgo. Los valores deseados son los mayores que 0.3, indican que por cada fuente se crearon 3 o más estrategias para el manejo de riesgos. Para esta métrica se establece un valor límite de 10, pues se considera que en un proyecto no se establecen, para una fuente de riesgo, más de 10 estrategias.

Tabla 7 Métrica para la agilidad en la preparación para el manejo de riesgo

Agilidad en la Preparación para el Manejo de Riesgo $0 \leq \mathrm{AMR} \leq 1$ Peso 0.33 Valor ideal 0.3	Valores	Evaluación
	$\mathrm{AMR} = 0$	M
	$\mathrm{AMR} = 0.1$ ó $\mathrm{AMR} = 0.2$	R
	$0.3 \leq \mathrm{AMR} \leq 1$	B

2.4.1.1.2. Métrica de la Identificación de los Riesgos Genéricos

La identificación de riesgo se debe basar en dos categorías (1) riesgos específicos, se identifican examinando el plan del proyecto y la declaración del ámbito del software y (2) riesgos genéricos, que son comunes a todos los proyectos de software. Sería imperdonable no identificar estos últimos; puesto que para realizar un proyecto, los datos históricos deben ser lo primero a tenerse en cuenta.

La métrica de la Identificación de los Riesgos Genéricos (IDRG) indica la efectividad con que se realizó la identificación de riesgo, perteneciente a la categoría de Riesgos Genéricos.

Se calcula la IDRG como:

$$\mathrm{IDRG} = 1 - \frac{\mathrm{RGONI}}{\mathrm{RONI}}$$

siendo $\mathrm{RONI} = \mathrm{RO} - \mathrm{ROI}$ y $\mathrm{RGONI}: \mathrm{RONI} \in \mathrm{RG}$

donde:

RO: número de riesgos ocurridos en un proyecto.

ROI: número de riesgos ocurridos en un proyecto que fueron identificados.

RONI: número de riesgos ocurridos en un proyecto que no fueron identificados.

RG: riesgos que pertenecen a la categoría riesgos genéricos.

RGONI: número de riesgos ocurridos en un proyecto que no fueron identificados pertenecientes a la categoría riesgos genéricos.

Si el valor de RONI es 0, la identificación de riesgos se realizó de forma satisfactoria, pues no hubo ocurrencia de riesgos no planificados y no es necesario continuar calculando la IDRG.

El valor de IDRG se encuentra entre 0 y 1, mientras más se acerque a 1 mejores serán los resultados, significa que se identificó el mayor número de riesgos genéricos posibles.

Tabla 8 Métrica de la identificación de Riesgos Genéricos

Métrica de la Identificación de Riesgos Genéricos $0 \leq \mathrm{IDRG} \leq 1$ Peso 0.33 Valor Ideal 0.85	Valores	Evaluación
	$0 \leq \mathrm{IDRG} < 0.65$	M
	$0.65 \leq \mathrm{IDRG} < 0.85$	R
	$0.85 \leq \mathrm{IDRG} \leq 1$	B

2.4.1.1.3. Eficacia de la Mitigación de Riesgos

La mitigación de riesgos consiste en el conjunto de acciones dirigidas a reducir o atenuar los efectos generados por la ocurrencia de un riesgo. Busca implementar acciones que disminuyan la magnitud del riesgo y, por ende, disminuir al máximo los daños. Con la prevención y la mitigación se trata de evitar que se produzcan grandes desastres.

La métrica de la Eficacia de la Mitigación de Riesgos (EMR) proporciona una medida de la efectividad con la que se realizaron las actividades propuestas en el Plan de Mitigación de Riesgos.

La EMR se puede calcular de la siguiente forma:

$$\text{EMR} = \frac{\text{RM}}{\text{R}}$$

donde:

R: número de riesgos identificados que se pueden mitigar.

RM: número de riesgos mitigados

El valor máximo de la EMR es 1, esto ocurre cuando se logran mitigar todos los riesgos identificados.

Tabla 9 Métrica para la eficacia de la mitigación de riesgos

Eficacia de la Mitigación de Riesgos $0 \leq \text{EMR} \leq 1$ Peso 0.33 Valor Ideal 0.75	Valores	Evaluación
	$0 \leq \text{EMR} < 0.5$	M
	$0.5 \leq \text{EMR} < 0.75$	R
	$0.75 \leq \text{EMR} \leq 1$	B

Con las tres métricas anteriormente definidas se realizará la evaluación del proceso de Gestión de Riesgo utilizando el procedimiento que se define en el Capitulo 2 (Ver epígrafe 2.3).

2.4.1.2. Planificación y Seguimiento del Proyecto

El objetivo de la planificación del proyecto de software es proporcionar un marco de trabajo que permita al gestor hacer estimaciones razonables de recursos, coste y planificación temporal. Estas estimaciones se hacen dentro de un marco de tiempo limitado al comienzo de un proyecto de software, y deberían actualizarse regularmente a medida que progresa el proyecto.

Seguidamente se muestran métricas para evaluar el proceso de planificación y seguimiento.

2.4.1.2.1. Error de Planificación

La planificación implica la estimación, su intento por determinar cuánto dinero, esfuerzo, recursos, y tiempo supondrá construir un sistema o producto especifico de software.

El Error de Planificación (EP) (Ver epígrafe 1.7.1.1) es una métrica con la cual se obtiene en porciento el error cometido durante la planificación.

$$\text{Error } \% = \frac{(|\text{Real} - \text{Estimado}|) * 100}{\text{Estimado}}$$

Para mayor facilidad en el trabajo con esta métrica se prefiere utilizar la fórmula de manera que el 100% sea el mejor valor.

$$\mathrm{EP} = 100 - \mathrm{Error}\%$$

Si el Error% alcanza un valor cercano a 100% significa que la planificación no se realizó con la calidad requerida.

Tabla 10 Métrica para el error de planificación

Error de Planificación $0 \leq \mathrm{EP} \leq 100\%$ Peso 0.3 Valor Ideal 0.75	Valores	Evaluación
	$0 \leq \mathrm{EP} < 50\%$	M
	$50\% \leq \mathrm{EP} < 75\%$	R
	$75\% \leq \mathrm{EP} \leq 100\%$	B

Como se puede apreciar el valor ideal siempre se encuentra entre 0 y 1, por lo que en el caso de las métricas, que como la anterior, expresan su resultado en porciento (%) es necesario dividirlo entre 100 para poder utilizar el procedimiento planteado. (Ver epígrafe 2.3)

2.4.1.2.2. Razón de Costo de Planificación

La métrica para calcular la Razón de Costo de Planificación (RCP) (Ver epígrafe 1.7.1.2) es una muestra de la calidad con la que se ha desarrollado el proceso de planificación en un proyecto.

RCP se puede calcular:

$$\mathrm{RCP} = \frac{\mathrm{CP}}{\mathrm{CR}}$$

donde:

CP: costo planificado.

CR: costo real

Los valores ideales de RCP son los cercanos a 1, ya sean mayores o menores, ellos indican que la planificación se ha desarrollo de la forma más realista posible.

Cuando la métrica resulte cualquier valor entre 1.2 y 1.4 para aplicar el procedimiento propuesto (Ver epígrafe 2.3) será necesario que se tome un valor entre 0.6 y 0.8 y si se obtiene un valor mayor que 1.4 se tomará como si la métrica hubiese resultado 0.

Tabla 11 Métrica para la razón de costo de planificación

	Valores	Evaluación
Razón de Costo de la Planificación $0 \leq \text{RCP}$ Peso 0.35 Valor Ideal 0.8	$0 \leq \text{RCP} < 0.6$ $1.4 < \text{RCP}$	M
	$0.6 \leq \text{RCP} < 0.8$ $1.2 < \text{RCP} \leq 1.4$	R
	$0.8 \leq \text{RCP} \leq 1.2$	B

2.4.1.2.3. Métrica para el Control de las Tareas Retrasadas

El proceso de seguimiento se desarrolla de varias formas en los proyectos de software. La evaluación de los resultados de todas las revisiones realizadas a lo largo del proceso de ingeniería, la comparación de la fecha real de inicio con las previstas para cada tarea del proyecto y el control sobre el cumplimiento de los hitos formales del proyecto en la fecha programada son algunas de las responsabilidades que deben cumplirse durante el seguimiento.

La Métrica para el Control de las Tareas Retrasadas (CTR) permite tener conocimiento sobre el retraso en el inicio de las actividades planificadas en una etapa determinada.

$$\text{CTR} = \frac{\text{TR} * 100}{\text{TPla}}$$

Para mayor facilidad en el trabajo con esta métrica se prefiere utilizar la fórmula de manera que el 100% sea el mejor valor.

$$\text{CTR} = 100 - \frac{\text{TR} * 100}{\text{TPla}}$$

donde:

TR: tareas con inicio retrasado.

TPla: total de tareas planificadas para la etapa.

CTR se expresa en porciento y un bajo valor de este (cercano a 0) puede ocasionar, un gran retraso en la etapa y la necesidad de una replanificación del proyecto.

Tabla 12 Métrica para el Control de las Tareas Retrasadas

Métrica para el Control de las Tareas Retrasadas $0 \leq \text{CTR} \leq 100\%$ Peso 0.35 Valor Ideal 0.75	Valores	Evaluación
	$0 \leq \text{CTR} < 50\%$	M
	$50\% \leq \text{CTR} < 75\%$	R
	$75\% \leq \text{CTR} \leq 100\%$	B

2.4.1.3. Gestión de los Recursos

En el proceso de Gestión de los Recursos se deben realizar un conjunto de actividades, que permitan mantener un control de todos aquellos recursos que se pretenden utilizar en el desarrollo del proyecto.

Según Pressman, para construir un software, existen tres tipos de recursos; las herramientas de hardware y software, los componentes de software reutilizables y el recurso primario, que es el personal.

Figura 4 Recursos (PRESSMAN 2002)

Para evaluar el proceso de Gestión de los Recursos se proponen las métricas siguientes.

2.4.1.3.1. Métrica para el Control de los Datos del Personal

En el proceso de Gestión de los Recursos es de gran importancia mantener el control de los recursos humanos. Conocer al personal con que se cuenta para desarrollar el software, es uno de los aspectos principales a tener en cuenta para una buena asignación de recursos. Por ello se deben tener archivado todos los documentos que puedan ayudar a una buena distribución de tareas y roles.

La Métrica para el Control de los Datos del Personal (CDP) muestra el grado de eficiencia con el que se ha controlado la información necesaria sobre el personal.

El CDP se puede calcular como:

$$\mathrm{CDP} = \frac{\mathrm{TPe} - \mathrm{PI}}{\mathrm{TPe}}$$

donde:

TPe: total de personal que participa en el proyecto.

PI: personal indocumentado.

Esta métrica, CDP, tiene su mejor y máximo valor en 1.

Tabla 13 Métrica para el control de los datos del personal

Métrica para el Control de los Datos del Personal $0 \leq \mathrm{CDP} \leq 1$ Peso 0.2 Valor Ideal 0.85	Valores	Evaluación
	$0 \leq \mathrm{CDP} < 0.6$	M
	$0.6 \leq \mathrm{CDP} < 0.85$	R
	$0.85 \leq \mathrm{CDP} \leq 1$	B

2.4.1.3.2. Métrica para el Aprovechamiento de los Componentes Reutilizables

Entre los recursos con más valor en el desarrollo de un proyecto se encuentran los componentes reutilizables. Su uso permite, entre otros elementos, el ahorro de tiempo, de suma importancia para cualquier desarrollador. Varios autores coinciden en que no es necesario crear algo que ya ha sido creado por otros.

La Métrica para el Aprovechamiento de los Componentes Reutilizables (ACR) da el porciento de los componentes que pudieron haber sido reutilizables del total de todos los componentes nuevos que se realizaron.

Se calcula como:

$$\mathrm{ACR} = \frac{\mathrm{CNNR} * 100}{\mathrm{CN}}$$

Para mayor facilidad en el trabajo con esta métrica se prefiere utilizar la fórmula de manera que el 100% sea el mejor valor.

$$\mathrm{ACR} = 100 - \frac{\mathrm{CNNR} * 100}{\mathrm{CN}}$$

donde:

CNNR: componentes nuevos que pueden ser sustituidos por componentes reutilizables.

CN: componentes nuevos.

Mientras más se acerque al 100% el ACR, significa que se hizo mejor uso de componentes reutilizables.

Tabla 14 Métrica para el aprovechamiento de los Componentes Reutilizables

Métrica para el Aprovechamiento de los Componentes Reutilizables $0 \leq \text{ACR} \leq 100\%$ Peso 0.4 Valor Ideal 0.75	Valores	Evaluación
	$0 \leq \text{ACR} < 50\%$	M
	$50\% \leq \text{ACR} < 75\%$	R
	$75\% \leq \text{ACR} \leq 100\%$	B

Para evaluar el proceso de Gestión de Recursos, además de las dos métricas expresadas anteriormente, se utilizara la métrica de Razón de Costo de Planificación (Ver epígrafe 2.4.1.2.2). En este caso a la métrica se le asigna un peso de 0.4 para evaluar el proceso.

2.4.2. Categoría del Proceso de Soporte

2.4.2.1. Gestión de Configuración y Cambios

La Gestión de Configuración y Cambio del software, es una actividad de autoprotección que se aplica durante el proceso del software. Como el cambio se puede producir en cualquier momento, las actividades de gestión de configuración y cambio sirven para (1) identificar el cambio, (2) controlar el cambio, (3) garantizar que el cambio se implemente adecuadamente e (4) informar del cambio a todos aquellos que puedan estar interesados. (PRESSMAN 2002)

Las métricas que se proponen a continuación serán utilizadas para evaluar la calidad del proceso de Gestión de Configuración y Cambio.

2.4.2.1.1. Índice de Madurez del Software

El Índice de Madurez del Software (IMS) (Ver epígrafe 1.7.2.1) proporciona una indicación de la estabilidad de un producto software basada en los cambios que ocurren con cada versión del producto.

IMS se calcula:

$$\text{IMS} = \frac{[\text{M}_\text{t} - (\text{F}_\text{a} + \text{F}_\text{c} + \text{F}_\text{d})]}{\text{M}_\text{t}}$$

donde:

M_t: número de módulos en la versión actual

F_c: número de módulos en la versión actual que se han cambiado

F_a: número de módulos en la versión actual que se han añadido

F_d: número de módulos de la versión anterior que se han borrado en la versión actual

Mientras más se acerca a 1 el IMS, mayor será la estabilidad del software.

Tabla 15 Métrica para el Índice de Madurez del Software

Índice de Madurez del Software $0 \leq \text{IMS} \leq 1$ Peso 0.35 Valor Ideal 0.75	Valores	Evaluación
	$0 \leq \text{IMS} < 0.5$	M
	$0.5 \leq \text{IMS} < 0.75$	R
	$0.75 \leq \text{IMS} \leq 1$	B

2.4.2.1.2. Métrica para el Estado de los Pedidos de Cambio

El estado de los pedidos de cambio permite llegar a la conclusión clara de en qué estado están estancados los pedidos de cambio y por tanto darle una advertencia al jefe del proyecto de donde tiene que insistir, e incluso, si es necesaria una reunión de la Junta de Control de Cambios.

En el proceso están definidos 5 estados de las peticiones de cambio: (ESTRADA and ESTÉVEZ 2003)

- **Aprobada:** Peticiones de cambio que ya han sido aprobadas por la junta de control de cambios pero no se han comenzado a desarrollar por un especialista
- **Cerrada:** Petición de cambio que ya fue resuelta
- **En cola:** Petición de cambio pendiente de análisis por la junta de control de cambios.
- **En desarrollo:** Petición de cambio aprobada por la junta que está siendo resuelta por un desarrollador
- **Posible rechazada:** Solicitudes de cambio que la junta tiene pendiente por tener incompleto los datos.

La Métrica para el Estado de Pedidos de Cambio (EPC) indica el porciento de los estados de peticiones que más pueden afectar en el desarrollo del proceso de Gestión de los Pedidos de Cambio.

$$\text{EPC} = \frac{(\text{PAp} + \text{PCo} + \text{PRe}) * 100}{\text{TP}}$$

Para mayor facilidad en el trabajo con esta métrica se prefiere utilizar la fórmula de manera que el 100% sea el mejor valor.

$$\text{EPC} = 100 - \frac{(\text{PAp} + \text{PCo} + \text{PRe}) * 100}{\text{TP}}$$

donde :

TP: total de peticiones.

PAp: peticiones aprobadas.

PCo: peticiones en cola.

Pre: peticiones posibles de rechazar.

Si EPC se encuentra próximo a 0 significa que hay un gran retraso en los estados de peticiones.

Tabla 16 Métrica para el Estado de Pedidos de Cambio

Métrica para el Estado de Pedidos de Cambio $0 \leq \text{EPC} \leq 100\%$ Peso 0.35 Valor Ideal 0.75	Valores	Evaluación
	$0 \leq \text{EPC} < 50\%$	M
	$50\% \leq \text{EPC} < 75\%$	R
	$75\% \leq \text{EPC} \leq 100\%$	B

2.4.2.1.3. Competencia en la Información de los Cambios

Entre las tareas más importantes en el proceso de Gestión de Configuración y Cambio está la de informar, sobre todos los cambios que se efectúen, al personal relacionado con los elementos que han sido modificados.

La Competencia en la Información de los Cambios (CIC) se puede calcular a través de una métrica que indica la medida en que han sido informados los cambios que se han realizado.

Se define:

$$\text{CIC} = 1 - \frac{\text{CNI}}{\text{CReg}}$$

donde:

CNI: cambios registrados que se han efectuado y no han sido informados al personal relacionado.

CReg: cambios registrados que se han efectuado.

Si se informaron todos los cambios realizados la CIC será 1.

Tabla 17 Métrica para la competencia en la información de los cambios

Competencia en la Información de los Cambios $0 \leq \text{CIC} \leq 1$ Peso 0.3 Valor Ideal 0.75	Valores	Evaluación
	$0 \leq \text{CIC} < 0.5$	M
	$0.5 \leq \text{CIC} < 0.75$	R
	$0.75 \leq \text{CIC} \leq 1$	B

2.4.2.2. Aseguramiento de la Calidad

El Aseguramiento de la Calidad es una actividad que se aplica a nivel de proyecto durante todo el proceso de desarrollo de software. Cada persona, involucrada en esta actividad, tiene un impacto en la calidad del software resultante.

El Aseguramiento de la Calidad como actividad de protección está presente en los métodos y herramientas de análisis, diseño, programación, prueba, inspecciones técnicas formales, control de la documentación del software y de los cambios realizados durante todo el proceso de desarrollo. (AGÜERO 2007)

Las métricas que seguidamente se describen, podrán utilizarse para evaluar la calidad con la que se desarrolló el proceso de Aseguramiento de la Calidad.

2.4.2.2.1. Destreza en la Resolución de No Conformidades

No conformidad es una terminología utilizada por la familia de normas UNE-EN ISO 9000, para referirse a cualquier incidencia a la hora de cumplir con un requisito especificado o establecido, pudiendo ser el motivo por un incumplimiento de un requisito contractual de los productos o servicios, así como una desviación de lo establecido en el Sistema de Gestión de la Calidad. (GANDOLFO 2004)

Durante el proceso de Aseguramiento de la Calidad una de las actividades fundamentales es identificar no conformidades. Cada una de ellas debe ser comunicada a los responsables para así asegurar su resolución.

La Destreza en la Resolución de No Conformidades (DRNC) es una métrica que indica en qué medida el equipo de desarrollo del software fue capaz de resolver las no conformidades identificadas en una etapa.

DRNC se calcula como:

$$\text{DRNC} = \frac{\text{NCR}}{\text{NC}}$$

donde:

NCR: cantidad de no conformidades que han sido resueltas.

NC: cantidad de no conformidades que se han identificado en una etapa.

Si el valor de DRNC se encuentra próximo a 1, representa que han sido resuelta la mayor cantidad posible de no conformidades.

Tabla 18 Métrica para la destreza en la resolución de No Conformidades

Destreza en la Resolución de No Conformidades $0 \leq \text{DRNC} \leq 1$ Peso 0.33 Valor Ideal 0.75	Valores	Evaluación
	$0 \leq \text{DRNC} < 0.5$	M
	$0.5 \leq \text{DRNC} < 0.75$	R
	$0.75 \leq \text{DRNC} \leq 1$	B

2.4.2.2.2. Métrica para la Violación de las Metodologías

Al comienzo de un proyecto, una de las prioridades a cumplir es la selección de la metodología de desarrollo de software que se va a utilizar. Esta actividad se incluye dentro de las responsabilidades del proceso de Aseguramiento de la Calidad, que también vela por el posterior cumplimiento de las pautas que se plantean en dicha metodología.

Para calcular en qué medida se viola la metodología seleccionada en el proyecto durante el desarrollo de los procesos se define la Métrica para la Violación de las Metodologías (MVM).

Se calcula:

$$\text{MVM} = 1 - \frac{\text{PVM}}{\text{PA}}$$

donde:

PVM: número de procesos analizados que violan la metodología seleccionada a utilizar en el proyecto.

PA: número de procesos analizados.

Mientras menos procesos de los analizados violen la metodología seleccionada en el proyecto, más cerca de 1 estará el valor de MVM.

Tabla 19 Métrica para la violación de las metodologías

Métrica para la Violación de las Metodologías $0 \leq \text{MVM} \leq 1$ Peso 0.33 Valor Ideal 0.75	Valores	Evaluación
	$0 \leq \text{MVM} < 0.5$	M
	$0.5 \leq \text{MVM} < 0.75$	R
	$0.75 \leq \text{MVM} \leq 1$	B

2.4.2.2.3. Métrica para el Control del Ajuste a los Estándares

Para lograr un buen proceso de Aseguramiento de la Calidad se requiere, entre otros factores, de un procedimiento que garantice los ajustes a los estándares en el proceso de desarrollo de software, siempre que esto sea posible.

La Métrica para el Control del Ajuste a los Estándares (CAE) surge para proveer a los evaluadores de información cuantitativa sobre los elementos que se ajustan a los estándares preestablecidos para el desarrollo del proyecto.

Calculándose:

$$\mathrm{CAE} = \frac{\mathrm{ERE}}{\mathrm{ER}}$$

donde:

ERE: número de elementos revisados que cumplen con los estándares preestablecidos.

ER: número de elementos que se revisaron en la verificación de ajuste a los estándares.

Si la mayoría de los elementos analizados durante la verificación de ajuste a los estándares cumplen con lo establecido en ellos el valor de CAE estará próximo a 1.

Tabla 20 Métrica para el control del ajuste a los estándares

Métrica para el Control del Ajuste a los Estándares $0 \leq \mathrm{CAE} \leq 1$ Peso 0.33 Valor Ideal 0.75	Valores	Evaluación
	$0 \leq \mathrm{CAE} < 0.5$	M
	$0.5 \leq \mathrm{CAE} < 0.75$	R
	$0.75 \leq \mathrm{CAE} \leq 1$	B

2.4.3. Categoría del Proceso de Producción

2.4.3.1. Realización del flujo de trabajo de Gestión de Requisitos

Los requisitos, en un proyecto software, son el conjunto de especificaciones clasificadas, definidas y aprobadas por los clientes que tienden a cumplir necesidades dadas. Los requisitos existen debido a los deseos del cliente y a algunas funcionalidades y cualidades que el producto exige.

Durante el proceso de gestión de requisitos se realiza el levantamiento de requisitos, donde se especifican tanto, los requerimientos del cliente como los del producto.

2.4.3.1.1. Eficiencia en la Obtención de los Requisitos

La obtención de los requisitos sirve como fundamento para la ingeniería del hardware, ingeniería del software, la ingeniería de bases de datos y la ingeniería humana. Describe las funciones y características de los sistemas de computación y las restricciones que gobiernan su desarrollo.

Luego de la obtención de los requisitos es necesario tomar un acuerdo sobre estos. Para ello se tienen en cuenta los criterios de los implicados en el proyecto, ya sean desarrolladores o clientes, ofreciendo como resultado final la especificación de requisitos (incluye todos los requisitos aceptados).

La Eficiencia en la Obtención de los Requisitos (EOR) es la métrica que brinda una idea de lo bien que se realizó la obtención de los requisitos.

La EOR se calcula como:

$$\mathrm{EOR} = \frac{\mathrm{EEA}}{\mathrm{EEO}}$$

donde:

EEA: cantidad de elementos de la especificación que fueron aceptados.

EEO: cantidad de elementos de la especificación obtenidos.

EOR es 1 si todos los elementos de la especificación obtenida fueron aceptados. Esto demuestra la eficiencia con la que trabajaron los involucrados en la obtención de los requisitos.

Tabla 21 Métrica para la eficiencia en la obtención de los requisitos

Eficiencia en la Obtención de los Requisitos $0 \leq \mathrm{EOR} \leq 1$ Peso 0.4 Valor Ideal 0.75	Valores	Evaluación
	$0 \leq \mathrm{EOR} < 0.5$	M
	$0.5 \leq \mathrm{EOR} < 0.75$	R
	$0.75 \leq \mathrm{EOR} \leq 1$	B

2.4.3.1.2. Agilidad en la Gestión de Cambio de los Requisitos

Dentro del proceso de Gestión de Requisitos se realizan actividades relacionadas con la gestión de cambio, que permiten resolver todos los problemas que ocasionan los cambios en los requisitos. Estas actividades se deben realizar con la mayor agilidad posible para no afectar el tiempo planificado para el proceso en general.

La Agilidad en la Gestión de Cambio de los Requisitos (AGCR) proporciona información sobre la relación entre el tiempo dedicado a la a la Gestión de Cambio de los Requisitos y el utilizado en el proceso en general.

Se puede calcular AGCR:

$$\mathrm{AGCR} = 1 - \frac{\mathrm{TGCR}}{\mathrm{TPGR}}$$

donde:

TGCR: tiempo dedicado a la Gestión de Cambio de los Requisitos.

TPGR: tiempo total dedicado al proceso de Gestión de Requisitos.

Si el valor obtenido, al calcular AGCR, se acerca a 1 la Gestión de Cambio de los Requisitos se habrá realizado de la forma más rápida posible.

Tabla 22 Métrica para la agilidad en la gestión de cambio de los requisitos

Agilidad en la Gestión de Cambio de los Requisitos $0 \leq \mathrm{AGCR} \leq 1$ Peso 0.4 Valor Ideal 0.75	Valores	Evaluación
	$0 \leq \mathrm{AGCR} < 0.5$	M
	$0.5 \leq \mathrm{AGCR} < 0.75$	R
	$0.75 \leq \mathrm{AGCR} \leq 1$	B

2.4.3.1.3. Productividad en la Corrección de Errores

En el proceso de Gestión de Requisitos, como en todos los procesos de desarrollo, es de gran importancia la realización de una o varias revisiones. Los errores que se encuentran en cada una de ellas deben ser analizados y corregidos para que no pasen a la siguiente etapa de desarrollo del software.

La métrica para calcular la Productividad de la Corrección de los Errores (PCE), encontrados durante las revisiones, permite conocer la eficiencia con la que se realizaron las correcciones de errores durante la Gestión de Requisitos.

$$\mathrm{PCE} = \frac{(\mathrm{E_a} - \mathrm{E_{(a+1)}}) * 100}{\mathrm{E_a}}$$

donde:

E_a: número de errores encontrados durante la revisión a.

$E_{(a+1)}$: número de errores encontrados durante la revisión siguiente a la revisión a.

Mientras más cerca esté la PCE de 100% significa que se corrigieron la mayor cantidad de errores.

Tabla 23 Métrica para la productividad de la corrección de los errores

Productividad de la Corrección de los Errores $0 \leq \text{PCE} \leq 100\%$ Peso 0.2 Valor Ideal 0.75	Valores	Evaluación
	$0 \leq \text{PCE} < 50\%$	M
	$50\% \leq \text{PCE} < 75\%$	R
	$75\% \leq \text{PCE} \leq 100\%$	B

Esta métrica puede ser utilizada para evaluar la productividad de la corrección de errores durante cualquier proceso de desarrollo.

2.4.3.2. Realización del flujo de trabajo del Análisis y Diseño

Durante el flujo de Análisis y Diseño, primeramente, se analizan los requerimientos descritos en la etapa de Requisitos, refinándolos y estructurándolos. Su propósito es conseguir una comprensión más precisa de los requisitos y una descripción de los mismos que sea fácil de mantener y que ayude a estructurar el sistema completo, incluyendo la arquitectura.

En este proceso se aplican técnicas y principios con la intención de definir un producto con los suficientes detalles como para permitir su realización física. Con él se pretende construir un sistema que; satisfaga determinadas especificaciones y se ajuste a las limitaciones impuestas por el medio de destino.

Seguidamente se definen métricas que darán una evaluación de la calidad con que se desarrolló este flujo de trabajo.

2.4.3.2.1. Métrica de la Eficacia de la Eliminación de Defectos

La Métrica de la Eficacia de la Eliminación de Defectos (EED) (Ver epígrafe 1.7.5) fue definida por Pressman de dos formas diferentes. Una de estas formas se puede utilizar para evaluar la habilidad de un equipo en encontrar errores antes de que pasen a la siguiente actividad o tarea de ingeniería del software.

EED se define como:

$$\text{EED}_\text{i} = \text{E}_\text{i}/(\text{E}_\text{i} + \text{E}_{\text{i}+1})$$

donde

E_i: número de errores encontrado durante la actividad de ingeniería del software i.

E_{i+1}: número de errores encontrado durante la actividad de ingeniería del software i + 1.

Tabla 24 Métrica de la eficacia de la eliminación de defectos

Eficacia de la Eliminación de Defectos $0 \leq \text{EED} \leq 1$ Peso 0.35 Valor Ideal 0.75	Valores	Evaluación
	$0 \leq \text{EED} < 0.5$	M
	$0.5 \leq \text{EED} < 0.75$	R
	$0.75 \leq \text{EED} \leq 1$	B

2.4.3.2.2. Métrica para el Control de las Casos de Uso Diseñados

En el proceso de Análisis y Diseño deben quedar diseñados cada uno de los Casos de Uso (CU) que se definieron en el proyecto, cumpliendo con las funcionalidades descritas en el proceso de Gestión de Requisitos.

La métrica para el Control de los Casos de Uso Diseñados (CCUD) proporciona información sobre la cantidad de CU que se diseñaron con respecto a los que se precisaron para el proyecto.

Se calcula como:

$$\text{CCUD} = \frac{\text{CUD}}{\text{CUDef}}$$

donde:

CUD: número de CU que fueron diseñados.

CUDef: número de CU que fueron definidos.

El único valor para el que la métrica CCUD evalúa el proceso de B es 1.

Tabla 25 Métrica para el control de los Casos de Uso diseñados

Métrica para el Control de los Casos de Uso Diseñados $0 \leq \text{CCUD} \leq 1$ Peso 0.35 Valor Ideal 1	Valores	Evaluación
	$0 \leq \text{CCUD} < 0.75$	M
	$0.75 \leq \text{CCUD} < 1$	R
	$\text{CCUD} = 1$	B

2.4.3.2.3. Métrica de Acoplamiento de Módulos

Uno de los factores más importante en el proceso de Análisis y Diseño es mantener un bajo acoplamiento. El acoplamiento es una medida de la fuerza, con que una clase esta conectada a otras clases, con la que se conoce y con que recurre a ellas. Una clase con bajo acoplamiento no depende de muchas otras.

La Métrica para el Acoplamiento de Módulos (m_C) (Ver epígrafe 1.7.3.1.2) muestra la conectividad de un módulo con otros módulos.

La fórmula general para esta métrica es:

$$m_C = \frac{K}{M}$$

Las variables implicadas se especifican en el epígrafe 1.7.3.1.2.

Cuando mayor sea el valor de m_c, menor es el acoplamiento de módulo. Los valores de m_C son convenientes cuando alcanzan valores superiores a 0.15.

Tabla 26 Métrica para el acoplamiento de módulos

Métrica para el Acoplamiento de Módulos $0 < m_C \leq 1$ Peso 0.3 Valor Ideal 0.15	Valores	Evaluación
	$0 < m_C < 0.05$	M
	$0.05 \leq m_C < 0.15$	R
	$0.15 \leq m_C \leq 1$	B

2.4.3.3. Realización del flujo de trabajo de Implementación

La etapa de Implementación se comienza con el resultado de la etapa de Diseño y se implementa el sistema en términos de componentes, es decir, ficheros de código fuente, scripts, ficheros de código binario, ejecutables y similares.

El objetivo principal del proceso de implementación es desarrollar la arquitectura y el sistema como un todo. De forma más específica, los propósitos de la Implementación son:

- Definir la organización del código.
- Planificar las integraciones de sistema necesarias en cada iteración.
- Implementar las clases y subsistemas encontrados durante el Diseño.

Las siguientes métricas servirán para evaluar el proceso de implementación.

2.4.3.3.1. Métrica para el Control de las Casos de Uso Implementados

En el proceso de Implementación, se deben implementar las clases y subsistemas encontrados durante el diseño. Las clases, en particular, se implementan como componentes de ficheros que contienen código fuente. Esto se hace con el propósito de que queden implementados todos los Casos de Uso (CU) que se incluyeron en el diseño.

La Métrica para el Control de los CU Implementados (CCUI) muestra en qué medida fueron implementados los CU que fueron diseñados.

Se calcula:

$$\mathrm{CCUI} = \frac{\mathrm{CUI}}{\mathrm{CUD}}$$

donde:

CUI: número de CU que fueron implementados.

CUD: número de CU que fueron diseñados.

El mejor valor que puede tomar CCUI es 1.

Tabla 27 Métrica para el control de los CU implementados

Métrica para el Control de los Casos de Uso Implementados $0 \leq \mathrm{CCUI} \leq 1$ Peso 0.55 Valor Ideal 0.75	Valores	Evaluación
	$0 \leq \mathrm{CCUI} < 0.5$	M
	$0.5 \leq \mathrm{CCUI} < 0.75$	R
	$0.75 \leq \mathrm{CCUI} \leq 1$	B

2.4.3.3.2. Métrica para el Control de las Pruebas de Unidad

En el proceso de Implementación se deben probar los componentes individualmente y a continuación integrarlos, compilándolos y enlazándolos en uno o más ejecutables, antes de ser enviados para ser integrados y llevar a cabo las comprobaciones del sistema. A estas pruebas individuales se les llama Pruebas de Unidad.

La Métrica para el Control de las Pruebas de Unidad (CPU) brinda una medida de los componentes que han sido probados individualmente antes de la integración con respecto al total de componentes que fueron implementados.

Se puede calcular:

$$\mathrm{CPU} = \frac{\mathrm{CPI}}{\mathrm{CImp}}$$

donde :

CPI: número total de componentes implementados.

CImp: número de componentes que fueron probados individualmente antes de la integración.

Si CPU es igual a 1 significa que todos los componentes implementados fueron probados individualmente antes de la integración.

Tabla 28 Métrica para el control de las Pruebas de Unidad

Métrica para el Control de las Pruebas de Unidad $0 \leq \text{CPU} \leq 1$ Peso 0.45 Valor Ideal 0.75	Valores	Evaluación
	$0 \leq \text{CPU} < 0.5$	M
	$0.5 \leq \text{CPU} < 0.75$	R
	$0.75 \leq \text{CPU} \leq 1$	B

2.4.3.4. Realización del flujo de trabajo de Prueba

La prueba del software es un elemento crítico para la garantía de la calidad del software. Su objetivo es identificar y asegurar que los defectos encontrados se corrijan antes de entregar el software al cliente.

Se deben diseñar pruebas que sistemáticamente saquen a la luz diferentes clases de errores, haciéndolo con la menor cantidad de tiempo y esfuerzo posible.

La prueba no puede asegurar la ausencia de defectos; sólo puede demostrar que existen defectos en el software.

Las siguientes métricas permitirán evaluar el proceso de Prueba.

2.4.3.4.1. Métrica de la Eficacia de la Eliminación de Defectos

La métrica de la eficacia de la eliminación de defectos (EED) (Ver epígrafe 1.7.5) proporciona una medida de la habilidad, que poseen el personal implicado en un proyecto, para encontrar errores.

Esta métrica se utilizará de la misma manera que la define Pressman para cuando un proyecto se toma en consideración globalmente:

$$\text{EED} = \text{E}/(\text{E} + \text{D})$$

donde

E: número de errores encontrados antes de la entrega del software al usuario final.

D: número de defectos encontrados después de la entrega.

Tabla 29 Métrica de la eficacia de la eliminación de defectos

Eficacia de la Eliminación de Defectos $0 \leq \text{EED} \leq 1$ Peso 0.3 Valor Ideal 0.75	Valores	Evaluación
	$0 \leq \text{EED} < 0.5$	M
	$0.5 \leq \text{EED} < 0.75$	R
	$0.75 \leq \text{EED} \leq 1$	B

2.4.3.4.2. Tasa de Propagación de Defectos

Durante el proceso de prueba pueden surgir defectos ocasionados al corregir otros defectos existentes. A este fenómeno se le denomina propagación de defectos.

La tasa de propagación de defectos (TPD) (Ver epígrafe 1.7.8) indica el comportamiento de la propagación de defectos durante un proceso dado.

Calculándose:

$$\text{TPD} = 1 - \frac{\text{Número defectos ocasionados al corregir defectos}}{\text{Número de defectos corregidos}}$$

El valor de TPD ofrece mejores resultados cuando está más cerca de 1. Al calcular esta métrica se pueden obtener valores negativos (cuando el número de defectos ocasionados al corregir defectos es mayor que el número de defectos corregidos), estos valores no se tendrán en cuenta a la hora de trabajar con los resultados de las métricas; sino que se tomarán como 0.

Tabla 30 Tasa de propagación de defectos

Tasa de Propagación de Defectos $0 \leq \text{TPD}$ Peso 0.3 Valor Ideal 0.75	Valores	Evaluación
	$0 \leq \text{EED} < 0.5$	M
	$0.5 \leq \text{EED} < 0.75$	R
	$0.75 \leq \text{EED} \leq 1$	B

2.4.3.4.3. Tasa de Efectividad de las Pruebas

Las pruebas de software deben ser aplicadas a todos los componentes del software, para que se desarrolle una etapa de prueba efectiva.

La tasa de efectividad de las pruebas (TEP) (Ver epígrafe 1.7.8) informa sobre el porciento de objetos que no han sido probados en el software.

TEP se calcula:

$$\text{TEP} = \frac{\text{Número de objetos probados al menos una vez} * 100}{\text{Número de objetos totales}}$$

Tabla 31 Tasa de efectividad de las pruebas

Tasa de Efectividad de las Pruebas $0 \leq \text{TEP} \leq 100\%$ Peso 0.3 Valor Ideal 0.75	Valores	Evaluación
	$0 \leq \text{TEP} < 50\%$	M
	$50\% \leq \text{TEP} < 75\%$	R
	$75\% \leq \text{TEP} \leq 100\%$	B

2.4.3.4.4. Métrica para Pruebas de Camino Básico

La prueba del camino básico es una técnica de prueba de caja blanca. Permite al diseñador de casos de prueba obtener una medida de la complejidad lógica de un diseño procedimental y usar esa medida como guía para la definición de un conjunto básico de caminos de ejecución.

La complejidad ciclomática (Ver epígrafe 1.7.4) da un límite superior para el número de caminos independientes que componen el conjunto básico y, consecuentemente, un valor límite superior para el número de pruebas que se deben diseñar y ejecutar para garantizar que se cubren todas las sentencias del programa.

La métrica para pruebas de camino básico (PCB) se basa en el valor de la complejidad ciclomática y muestra el porciento de pruebas diseñadas con respecto a dicho valor.

Se calcula de la siguiente forma:

$$\text{PCB} = \frac{\text{P}}{\text{V(G)}} * 100$$

donde:

P: número de pruebas diseñadas.

V(G):complejidad ciclomática.

Si el valor de PCB no está cerca del 100%, el número de pruebas diseñadas no es suficiente para asegurar que se ejecutan todas las sentencias del programa.

Tabla 32 Métrica para pruebas de Camino Básico

Métrica para Pruebas de Camino Básico $0 \leq \text{PCB} \leq 100\%$ Peso 0.1 Valor Ideal 0.75	Valores	Evaluación
	$0 \leq \text{PCB} < 50\%$	M
	$50\% \leq \text{PCB} < 75\%$	R
	$75\% \leq \text{PCB} \leq 100\%$	B

2.5. Conclusiones

En el desarrollo del presente capítulo se describe una encuesta aplicada como uno de los métodos utilizados para llevar a cabo la investigación y se analizan los resultados obtenidos en la misma. Además se explica el procedimiento que servirá para ofrecer una evaluación de la calidad de los procesos de desarrollo de SWE que utiliza el modelo MEDPSE empleando las métricas propuestas, que de igual forma quedaron bien detalladas. Para todas las métricas se hace una breve descripción de la actividad o característica que se pretende medir, se expresa la fórmula explicando cada una de las variables que la componen y mediante una tabla se delimita el rango de valores que puede tomar, así como el peso y el valor ideal de la métrica.

3

CAPÍTULO 3: EVALUACIÓN TÉCNICA DE LA PROPUESTA DE SOLUCIÓN

3.1 Introducción

En el presente capítulo se realiza la evaluación técnica de la propuesta descrita en el capítulo anterior. Para ello se utiliza el método multicriterio, el cual se basa en la evaluación cuantitativa de criterios previamente definidos por parte de expertos en el tema, que permite determinar si se acepta o no la propuesta analizada.

3.2 Guía para la evaluación técnica

A continuación se describen los pasos que se efectuaron para llevar a cabo la evaluación utilizando el método multicriterio:

1. Se elaboran los criterios que fueron utilizados en la evaluación y se agrupan de acuerdo a las características de la propuesta.

Grupo No 1: Criterios de mérito científico.

1. Valor científico de la propuesta.
2. Calidad de la investigación.
3. Contribución científica.

Grupo No 2: Criterios de implantación

4. Necesidad de empleo de la propuesta.
5. Obtención de productos finales con calidad.
6. Posibilidades de aplicación.

Grupo No 3: Criterios de flexibilidad.

7. Adaptabilidad a proyectos productivos de SWE independientemente del tipo de SWE que desarrollen.

8. Capacidad de las métricas propuestas para adecuarse a la evaluación de las diferentes áreas de procesos de SWE.
9. Capacidad de las métricas propuestas para adecuarse a la evaluación de procesos de desarrollo de proyectos productivos que no desarrollen SWE.

Grupo No 4. Criterios de impacto.

10. Efectos en la mejora de la evaluación del modelo MEPDSE.
11. Repercusión en la calidad de los procesos de desarrollo en los proyectos productivos de SWE.

2. Se le asigna un peso relativo a cada grupo de criterios de acuerdo al porciento que representa cada grupo del total y los intereses a evaluar.

   Grupo No. 1…….25

   Grupo No. 2…….30

   Grupo No.3……..25

   Grupo No.4……..20

3. Se realiza una selección de 7 expertos en la cual se tiene en cuenta su especialidad, grado científico y currículo.
4. Se hace entrega de la propuesta que se desea validar a todos los expertos para que se documenten sobre el tema de la investigación y luego expresen sus criterios en sendos modelos. En el primero (Ver Anexo 4), los expertos conceden pesos a cada uno de los criterios establecidos, teniendo en cuenta que la suma de los valores dados para un grupo no exceda del peso relativo asignado a este.

   El segundo modelo (Ver Anexo 5) permite realizar una evaluación cuantitativa de cada criterio con una escala de 1 a 5 y la apreciación cualitativa con una clasificación final del proyecto en excelente, bueno, aceptable, cuestionable y malo. También se ofrece la posibilidad de dar su opinión haciendo una valoración final del proyecto, emitiendo todas aquellas consideraciones que estimen convenientes.

5. Después de recibir los valores del peso relativo de cada criterio se construye la Tabla 33.

Tabla 33 Peso otorgado por los expertos a los criterios

G	C/E	E_1	E_2	E_3	E_4	E_5	E_6	E_7	Ep
25	C_1	8	10	5	7	8	8	9	7.85714286
	C_2	10	8	15	12	10	10	10	10.7142857
	C_3	7	7	5	6	7	7	6	6.4285714
30	C_4	15	10	12	13	12	15	10	12.4285714
	C_5	6	10	10	10	8	9	10	9
	C_6	9	10	8	7	10	6	10	8.5714285
25	C_7	9	10	12	8	10	9	9	9.5714285
	C_8	9	7	8	10	8	9	8	8.4285714
	C_9	7	8	5	7	7	7	8	7
20	C_{10}	10	10	10	10	10	10	10	10
	C_{11}	10	10	10	10	10	10	10	10
T		100	100	100	100	100	100	100	100

6. Se verifica la consistencia en el trabajo de los expertos, para lo que se utiliza el coeficiente de concordancia de Kendall y el estadígrafo Chi cuadrado (X^2). Siguiendo el procedimiento que se muestra a continuación.

   - Sea C el número de criterios que van a evaluarse y E el número de expertos que realizan la evaluación.
   - Para cada criterio se determina la ∑E que representa la sumatoria del peso dado por cada experto, Ep que es la puntuación promedio de los pesos correspondientes a cada criterio, M∑E es la media de la ∑E y ΔC que se obtiene calculando la diferencia entre ∑E y M∑E.
   - Se determina la desviación de la media, que posteriormente se eleva al cuadrado para obtener la dispersión S por la expresión:

$$S = \Sigma(\Sigma E - \Sigma\Sigma E/C)^2$$

Tabla 34 Cálculo de la Dispersión (S) para hallar la concordancia entre los expertos

	∑E	∑E/C	∑E-∑∑E/C	(∑E-∑∑E/C)^2
C_1	55	5	-8.636359	74.586696
C_2	75	6.818181	11.3641	129.142768
C_3	45	4.090909	-18.6359	347.296768
C_4	87	7.909090	23.3641	545.881168
C_5	63	5.727272	-0.6359	0.404368
C_6	60	5.454545	-3.6359	13.219768
C_7	67	6.090909	3.3641	11.317168
C_8	59	5.363636	-4.6359	21.491568
C_9	49	4.454545	-14.6359	214.209568
C_{10}	70	6.363636	6.3641	40.501768
C_{11}	70	6.363636	6.3641	40.50176881
∑∑E/C	-	63.636359	-	-
S=∑(∑E-∑∑E/C)^2	-	-	-	1438.55338

- Conociendo la dispersión se puede calcular el coeficiente de concordancia de Kendall W:

$$W = S/E^2(C^3 - C)/12$$

- El coeficiente de concordancia de Kendall permite calcular el Chi cuadrado real:

$$X^2 = E * (C - 1) * W$$

Los valores obtenidos se muestran en la tabla 35.

Tabla 35 Tabla para el cálculo de concordancia

Expertos/Criterios	E_1	E_2	E_3	E_4	E_5	E_6	E_7	∑E	Ep	ΔC	ΔC^2
C_1	8	10	5	7	8	8	9	55	7.8571428	-8	64
C_2	10	8	15	12	10	10	10	75	10.714285	12	144
C_3	7	7	5	6	7	7	6	45	6.428571	-18	324
C_4	15	10	12	13	12	15	10	87	12.428571	24	576
C_5	6	10	10	10	8	9	10	63	9	0	0
C_6	9	10	8	7	10	6	10	60	8.571428	-3	9
C_7	9	10	12	8	10	9	9	67	9.571428	4	16
C_8	9	7	8	10	8	9	8	59	8.428571	-4	16
C_9	7	8	5	7	7	7	8	49	7	-14	196
C_{10}	10	10	10	10	10	10	10	70	10	7	49
C_{11}	10	10	10	10	10	10	10	70	10	7	49
DC	100	100	100	100	100	100	100	100	90.428571	7	1443
M ∑E	63										
W	0.3493819										
X^2	18.682511										
$X^2_{(\alpha,\ c-1)}$	23.2093										

- El Chi cuadrado calculado se compara con el obtenido de las tablas estadísticas, si se cumple que el $X^2_{real} < X^2_{(\alpha,c-1)}$ se puede decir que existe concordancia en el trabajo de los expertos.

  En el presente caso se pude afirmar que existe concordancia en el trabajo de los expertos:

$$18.682511 < 23.2093$$

Si no existe concordancia se hace necesario repetir el trabajo de los expertos.

7. Posteriormente se identifica el peso relativo de cada criterio P y se calcula el Índice de Aceptación (IA) de la propuesta.

   Para esto se utiliza el procedimiento siguiente (Ver Tabla 36).

   - Conociendo el número de experto que realizan la evaluación E y la sumatoria de las puntuaciones de cada criterio C se puede calcular el peso de cada criterio P.
   - Conociendo el peso de cada criterio P y la calificación dada por los evaluadores c en una escala de 1 a 5 que se recogieron en el Modelo 2 (Ver Anexo 5) se puede calcular el valor de P × c.
   - Con el valor anterior se calcula el Índice de Aceptación del proyecto (IA).

$$\mathrm{IA} = \mathrm{P} * \mathrm{c}/5$$

Tabla 36 Calificación de cada criterio

Criterios	**Calificación (c)**					**P**	**P × c**
	1	**2**	**3**	**4**	**5**		
C1					X	0.0785714	0.392857
C2					X	0.107142	0.53571
C3					X	0.0642857	0.3214285
C4					X	0.124285	0.621425
C5					X	0.09	0.45
C6				X		0.085714	0.342856
C7					X	0.09571428	0.4785714
C8				X		0.08428571	0.33714284
C9			X			0.07	0.21

C10					X	0.10	0.5
C11					X	0.10	0.5
Total							4.689990
IA	0.9379						

8. Por último se determina la probabilidad de existo de la propuesta, ubicando el IA calculado anteriormente en rangos que están predefinidos (Ver tabla 37), en dependencia de donde se ubique será la probabilidad de éxito que tenga la propuesta.

Tabla 37 Rangos predefinidos de Índice de Aceptación

$0.7 < IA$	Existe alta probabilidad de éxito
$0.5 < IA < 0.7$	Existe probabilidad media de éxito
$0.3 < IA < 0.5$	Probabilidad de éxito baja
$\mathrm{IA} < 0.3$	Fracaso seguro

El IA calculado es 0.9379 lo que significa que existe alta probabilidad de éxito.

3.3. Conclusiones

Para la validación técnica de la propuesta se utilizó el método multicriterio. Con él se alcanzaron resultados favorables, obteniéndose una probabilidad de éxito alta demostrando que lo planteado hasta el momento se adapta valiosamente a las condiciones existentes y aporta novedosos elementos que resultan imprescindibles para la práctica de la aplicación de algo tan complejo y en ocasiones confuso como las métricas.

CONCLUSIONES

En la actualidad es cada vez más frecuente la consideración de métricas. El uso de éstas se ha adoptado con éxito en el amplio mercado del desarrollo de software, estableciendo la necesidad de un enfoque más disciplinado y de alta calidad. Las métricas para la evaluación de los procesos de desarrollo de SWE han constituido el eje central de la presente investigación, en la cual se ha arribado a las siguientes conclusiones:

- La medición ayuda a entender qué ocurre durante el desarrollo y el mantenimiento, permite controlar los proyectos y puede mejorar procesos y productos.
- A partir de las métricas propuestas, para los procesos de SWE seleccionados, es posible hacer una valoración objetiva de la calidad con que se desarrollaron dichos procesos.
- Contando con un procedimiento para el uso de las métricas planteadas, será más fácil la calificación de los procesos que se pretenden evaluar.
- Luego de realizar la evaluación técnica de la propuesta, mediante el método multicriterio, se obtuvo una alta probabilidad de éxito, lo que implica desde el punto de vista teórico, el cumplimiento de la idea a defender planteada.

RECOMENDACIONES

Para perfeccionar el contenido del presente trabajo se recomienda:

- Proponer una plantilla de recolección de datos para ser usados en los cálculos de las métricas.
- Hacer un estudio de las métricas que puedan aplicarse en las restantes áreas del proceso de desarrollo de SWE para incluirlas en la propuesta actual.
- Comenzar a aplicar las métricas propuestas en proyectos de SWE para probar su utilidad.

REFERENCIAS BIBLIOGRÁFICAS

1. AGÜERO, D. N. Áreas del aseguramiento de la calidad. *Universidad de Ciencia Informáticas*, 2007.
2. ALONSO, E. B. *Medición y métricas del software*, [Artículo]. Universidade de Vigo, 2004. [2008]. Disponible en: http://trevinca.ei.uvigo.es/~ebalonso/asignaturas/esx/guiones
3. AUTORES, C. D. *Guía Técnica sobre Evaluación de Software en la Administración Pública*, [Resolución Ministerial]. Publicado en el Diario Oficial "El Peruano", 2004. [2008]. Disponible en: http://www.ongei.gob.pe/Bancos/Banco_Normas/Archivos/Guia-evaluacion-sw.pdf
4. CATALDI, Z.; F. LAGE*, et al. Ingenieria de Software Educativo*, [Artículo]. Centro de Ingeniería del Software e Ingeniería del Conocimento (CAPIS) ITBA., 2003. [2008]. Disponible en: http://www.itba.edu.ar/capis/webcapis/RGMITBA/comunicacionesrgm/c-icie99-ingenieriasoftwareeducativo.pdf
5. CINTRA, N. A. and Y. V. CISNEROS Principios para la evaluación y certificación de la calidad de los productos de Software Educativo en la Universidad de las Ciencias Informáticas., 2007.
6. CRISTANCHO, J. A. *Evaluación de la calidad del software educativo bajo el estándar ISO 9126*, [Artículo]. Instituto Universitario de Tecnología Región los Andes IUT-Táchira, 2006. [2008]. Disponible en: http://www.saber.ula.ve/cgi-win/be_alex.exe?Acceso=T016300004104/5&Nombrebd=SSABER
7. DORIA, H. G. *Las Métricas de Software y su Uso en la Región*, 2001. [2008]. Disponible en: http://catarina.udlap.mx/u_dl_a/tales/documentos/lis/gonzalez_d_h/
8. ESTRADA, A. F. and I. P. ESTÉVEZ. *Medir el proceso de control de configuración, ¿una utopía para la Industria Nacional de Software?*, Instituto Superior Politécnico “José Antonio Echeverría”, 2003. [2008]. Disponible en: http://www.inf.udec.cl/revista/ediciones/edicion9/febles.pdf
9. FARFÁN, K. V. M. *Glosario de comunicación*, [Documento]. 1997. [2008]. Disponible en: http://www.monografias.com/trabajos16/diccionario-comunicacion/diccionario-comunicacion.shtml

10. GALVIS, A. *Ingeniería de software educativo* Uniandes Colombia., 2000. p.

11. GANDOLFO, A. *Aseguramiento de la Calidad*, 2004. [2008]. Disponible en: http://www.andima.es/revista/34/04.pdf

12. GARCÉS, J. R. and M. E. V. VILLAR. *Perfiles idóneos con matemática borrosa: Una utilidad para la orientación en Secundaria.*, 2007. [2008]. Disponible en: http://www.upv.es/cies/documentos/U0367544.pdf

13. GRADY, R. G. *Practical Software Metrics for Project Management and Process Improvement, Prentice-Hall,*. 1992. p.

14. GROS, B. *Del software educativo a educar con software*, 2000. [Disponible en: http://www.quadernsdigitals.net/articuloquaderns.asp?IdArticle=3743

15. GROS, B.; A. BERNARDO*, et al. Diseños y programas educativos, pautas pedagógicas para la elaboración de software.* . Editorial Ariel, 1997. p.

16. HERNÁNDEZ, L. L. T. and M. C. NÚÑEZ. *Propuesta de métricas para perfeccionar la calidad en los procesos de desarrollo del software*, Universidad de las Ciencias Informáticas, 2007. p.

17. HUMPHREY, W. S. *Introducción al Proceso Software Personal* 2001. p.

18. ILINA, L. T. and R. P. GONZÁLEZ. *Propuesta de un sistema de métricas para la evaluación de los proyectos de gestión de la UCI* Universidad de las Ciencias Informáticas, 2007. p.

19. IVAR JACOBSON, G. B., JAMES RUMBAUGH. *El Proceso Unificado de Desarrollo de Software*. 2000. 4 p.

20. L.BRIAND; C. B*, et al. An experimental comparison of the maintainability of objectoriented and structured design documents*. 1996. p.

21. LAMAS, R. R. Introducción a la Informática Educativa *La Habana*, 2000.

22. MÁRQUEZ, P. *El software educativo*, 2005. [2008]. Disponible en: http://www.lmi.ub.es/te/any96/marques_software

23. MCGARRY, J. and E. BAILEY. *Practical Software Measurement: A Foundation for Objective Project Management*. 1998. p.

24. MELIÁN, J. M. M. and J. F. H. BALLESTEROS. *La calidad del software y su medida*. Editorial Centro de Estudio Ramón Araces, S.A, 2003. p.

25. MUÑOZ, C. C. *Métricas del Software:Conceptos básicos, definición y formalización*, [Artículo]. Universidad de Castilla-La Mancha, 2006. [2008]. Disponible en: http://eisc.univalle.edu.co/materias/Material_Desarrollo_Software

26. N. CALLAOS and B. CALLAOS. *Designing with Systemic Total Quality*, en International Conference on Information Systems, 1993. p.

27. PRESSMAN, R. S. *Ingeniería del Software. Un enfoque práctico*. Quinta Edición. 2002. p.

28. RICARDO, F. Á. C. *Utilización del Patrón Modelo – Vista – Controlador (MVC) en el diseño de software educativos*, [Articulo]. 2006. [2007]. Disponible en: http://www.informaticahabana.com/evento_virtual/files/MUL053.pdf.

29. ROMÁN, I. R.; M. R. CARREIRA*, et al. Los Modelos Dinámicos y la Ingeniería del Software*, [Recurso Electrónico]. 1998. [2008]. Disponible en: http://www.sc.ehu.es/jiwdocoj/remis/docs/modelos.html

30. ROSELLÓ, E. G.; J. G. DACOSTA*, et al. ¿Existe una situación de crisis del software educativo?*, [Documento]. 1994. [2007]. Disponible en: http://lsm.dei.uc.pt/ribie/docfiles/txt2003729185619paper-144.pdf.

31. SEI. *Capability Maturity Model Integration (CMMISM)*. Carnegie Mellon University, 2002. p.

32. VARGAS, J. M. T. Temas selectos de programación 2. CIMAT métricas del proceso de software, 2005.

33. WESTFALL, L. L. *Software metrics that meet your information needs*. 1995. p.

BIBLIOGRAFÍA

ÁGUILA, L. M. A. *La evaluación del aprendizaje desde una perspectiva de la subjetividad y la incertidumbre. Una propuesta de modelo de autoevaluación a partir de competencias.*, Instituto Superior Politécnico José Antonio Echeverría "CUJAE", 2006. [2008]. Disponible en: http://www.rieoei.org/deloslectores/1392Alonso.pdf

ALUJA, J. G. *Elementos para una teoría de la decisión en la incertidumbre.*, Milladorios 1999. p.

ÁLVAREZ, J. L. G. *Análisis de los aspectos y factores a considerar para la Evaluación de la Calidad de un Software Educativo.*2003. [2007]. Disponible en: http://www.sepbcs.gob.mx/comunicacion/comunicados/Ponencia_Somece.pdf

ANDRADE, R. A.; A. C. GÓMEZ*, et al. Desde ISO 9001 hacia CMMI, pasos para la mejora de los procesos y métricas.*, Departamento de Ciencias de la Computación. Escuela Superior de Informática. Universidad Alcalá, 2007. [Disponible en: http://www.aemes.org/rpm/descargar.php?volumen=4&numero=1&articulo=3

BERNÁRDEZ, B.; M. T. A. DURÁN*, et al. Una Propuesta para la verificación de requisitos basada en métricas.*, [Revista de Procesos y Métricas de las Tecnologías de la Información]. Universidad de Sevilla, Depto. Lenguajes y Sistemas Informáticos, 2004. [2008]. Disponible en: http://www.aemes.org/rpm/contenidos/articulos.php

CARRERAS, C. *Introducción al diseño software*, Universidad Politécnica de Madrid, 2005. [2008]. Disponible en:http://www-lsi.die.upm.es/~carreras/ISSE/diseno_software.x2.pdf

CINTRA, N. A. and Y. V. CISNERO. *Principios para la evaluación y certificación de la calidad de los productos de Software Educativo en la Universidad de las Ciencias Informáticas.* La Habana, Universidad de las Ciencias Informáticas, 2007. p.

ESTÉVEZ, I. P. *Métricas para el control de proyectos de software.* La Habana, Instituto Superior Politécnico "José Antonio Echeverría", 2002. p.

FUENTES, J. M. *Optimización del proceso de Gestión de Requisitos en el desarrollo de aplicaciones software.*, 2007. [Disponible en: http://www.reusecompany.com/docs/Optimizaci%C3%B3n%20del%20proceso%20de%20Gesti%C3%B3n%20de%20Requisitos.pdf

BIBLIOGRAFÍA

GOLIATH, K. D. and Y. R. MARTÍNEZ. *Documentación imprescindible para los flujos de trabajo de diseño e implementación de software de gestión*. La Habana, Universidad de las Ciencias Informáticas, 2007. p.

LEMUS, Y. P. and Y. H. DÍAZ. *Sistema de Métricas para evaluar el Software Educativo.* La Habana Universidad de las Ciencias Informáticas, 2007. p.

MOJENA, B. G. *Procedimiento Propuesto para medir la Calidad en la Gestión de Requisitos*, Universidad de las Ciencias Informáticas, 2007. p.

MOMPIÉ, L. G. Ideas para la concepción de un modelo de evaluación de calidad de Software Educativo., 2006.

MOYA, O. P. *Entregables que garantizan la calidad del producto en el proceso de producción de software educativo.*, 2007. p.

ORTIZ, Y. V. *Análisis y diseño del sistema de información y gestión para la calidad del Software Educativo*. La Habana, Universidad de las Ciencias Informáticas, 2007. p.

PERALTA, M. L. *Asistente para la Evaluación de CMMI-SW*. Buenos Aires, 2004. p.

PÉREZ, M.; G. A. DÍAZ*, et al. Propuesta de una metodología de desarrollo de Software Educativo bajo un enfoque de calidad Sistémica.*, [Artículo]. 2007. [2007]. Disponible en: http://www.academia-interactiva.com/ise.pdf

ROLANDO ALFREDO HERNÁNDEZ LEÓN, S. C. G. El paradigma cuantitativo de la Investigación Científica, 2002.

GLOSARIO DE TÉRMINOS Y SIGLAS

Atributo: Característica propia de un producto, proceso o servicio.

Cliente: Persona, organización o grupo de personas que encarga la construcción de un sistema.

Cohesión: Es una medida de cuán relacionadas y enfocadas están las responsabilidades de un módulo determinado.

Defecto: Fisura descubierta después de la entrega al usuario final.

Docentes: Persona responsable de impartir conocimientos.

Entidad: Es la representación de un objeto.

Entorno: Es la fuente de información usada por todas las aplicaciones, conocida también como entorno general, donde están almacenadas las rutas y la definición de variables de usuario.

Error: Fisura en un producto de trabajo de la ingeniería de software o en la entrega descubierta por los ingenieros antes de que el software sea entregado al usuario final.

Evaluación: Es un proceso de información, interpretación y valoración para la toma de decisiones y para la mejora

Grafo: Representación compuesta de nodos y enlaces (también denominados aristas). Cuando se dirigen los enlaces (aristas), el grafo de flujo es un grafo dirigido.

Indicador: Magnitud utilizada para medir o comparar los resultados efectivamente obtenidos en la ejecución de un trabajo. Es, por tanto, un resultado cuantitativo que se mide en porcentaje, tasas y razones para permitir comparaciones.

Ingeniería del software: Aplicación de un enfoque sistemático, disciplinado y cuantificable al desarrollo, operación y mantenimiento del software.

Interfaz: Una interfaz define el límite de comunicación entre dos elementos, tales como software, hardware o un usuario.

Módulo: Un módulo es un componente autocontrolado de un sistema, el cual posee una interfaz bien definida hacia otros componentes.

Peso: nivel de importancia asignado a cada una de las actividades que se pretenden medir.

Producto: Resultado concreto, observable y medible que surge como consecuencia del proceso, proyecto o experiencia desarrollada.

Proceso: Conjunto de actividades, realizadas en forma secuencial, que realiza una organización, para crear, producir y entregar productos, de tal manera que satisfagan las necesidades de sus clientes.

Proyecto: Es un elemento organizativo a través del cual se gestiona el desarrollo de software, su resultado es un producto.

Recursos: Conjunto de personas, bienes materiales, financieros y técnicos con que cuenta en proyecto.

Requisito: Condición a capacidad que debe cumplir un sistema.

Software: Se refiere a los programas y datos almacenados en un ordenador.

Usuario: Persona que utiliza o trabaja con algún objeto o que es destinataria de algún servicio público o privado, empresarial o profesional.

Siglas

CFF: Cohesiones Funcionales Fuertes

CFD: Cohesiones Funcionales Débiles

CR: Conveniencia de la Representación

CU: Casos de Uso

EED: Eficacia de la Eliminación de Defectos

IA: Índice de Aceptación

IEEE: Instituto de Ingenieros Eléctricos y Electrónicos

IGU: Interfaz Grafica de Usuario

ISO: Organización Internacional para la Estandarización

KLOC: 1000 líneas de código

MEPDSE: Modelo de Evaluación del Proceso de Desarrollo del Software Educativo

PSP*:* Proceso de Software Personal

RTP: Rendimiento total de proceso

SWE: Software Educativo

GLOSARIO DE TÉRMINOS Y SIGLAS

TI: Tecnología de la Información

UCI: Universidad de las Ciencias Informáticas

VF: Valoración de Fallos

ANEXOS

<u>Anexo 1:</u> Modelo de la encuesta aplicada a los especialistas de la dirección de calidad de la UCI para la asignación de pesos en los aspectos a medir.

Compañero(a):

La presente encuesta tiene como propósito conocer el nivel de importancia que usted le concede a la medición de determinadas características o actividades en los procesos de desarrollo de Software Educativo. Esperamos que colabore y para ello le pedimos que al responder las preguntas lo haga con la mayor seriedad posible, ya que de ello dependerá en gran medida el resultado de nuestro trabajo, muchas gracias de antemano por su ayuda.

1. ¿Considera usted que es importante la medición para evaluar la calidad de los procesos de desarrollo del software?

   Si__ No__

2. ¿Ha realizado antes asignaciones de **Peso**?

   Si__ No__

En las siguientes preguntas marque con una **X** en el peso que usted considere justo.

Ejemplo:

1__ 2_**X**_ 3__

Siendo 1: alta relevancia, 2: media relevancia y 3: baja relevancia.

Para el caso de los **Pesos Relativos** marque con una **X** en la combinación de pesos que considere mejor. Si no esta de acuerdo con ninguna de las expresadas pude plantear la suya en la casilla **Otras** (los pesos relativos deben sumar 1).

3. Para el proceso de Gestión de Riesgos

Actividad a medir	Peso	Pesos Relativos			Otras
Agilidad en la Preparación para el Manejo de Riesgos	1__ 2__ 3__	0.2	0.33	0.2	
Identificación de Riesgos	1__ 2__ 3__	0.4	0.33	0.5	
Eficacia en la Mitigación de Riesgo	1__ 2__ 3__	0.4	0.33	0.3	

4. Para el proceso de Planificación y seguimiento del proyecto

Actividad a medir	Peso	Pesos Relativos			Otras
Porciento de errores cometidos en la Estimación	1__ 2__ 3__	0.3	0.33	0.3	
Desarrollo realista de la Planificación	1__ 2__ 3__	0.35	0.33	0.5	
Control de las tareas retrasadas	1__ 2__ 3__	0.35	0.33	0.2	

5. Para el proceso de Gestión de los Recursos

Actividad a medir	**Peso**	**Pesos Relativos**			**Otras**
Control de los Datos del Personal (documentos que lo acrediten)	1__ 2__ 3__	0.2	0.2	0.1	
Aprovechamiento de los Componentes Reutilizables	1__ 2__ 3__	0.2	0.3	0.2	
Estimación realista de los Recursos	1__ 2__ 3__	0.6	0.5	0.7	

6. Para el proceso de Gestión de requisitos

Actividad a medir	**Peso**	**Pesos Relativos**			**Otras**
Eficiencia en la Obtención de los Requisitos	1__ 2__ 3__	0.6	0.5	0.4	
Agilidad en la Gestión de Cambio de los Requisitos	1__ 2__ 3__	0.2	0.2	0.4	
Productividad en la Corrección de Errores	1__ 2__ 3__	0.*2*	0.3	0.2	

7. Para el proceso de Gestión de configuración y cambios

Actividad a medir	Peso	Pesos Relativos			Otras
Índice de Madurez del Software (basada en los cambios que ocurren con cada versión del producto)	1__ 2__ 3__	0.4	0.35	0.3	
Estado de los Pedidos de Cambio (estados de peticiones que más pueden afectar en el desarrollo del proceso)	1__ 2__ 3__	0.4	0.35	0.5	
Eficacia en la Información de los Cambios al personal	1__ 2__ 3__	0.2	0.3	0.2	

8. Para el proceso de Aseguramiento de la calidad

Actividad a medir	Peso	Pesos Relativos			Otras
Agilidad en la Resolución de No Conformidades	1__ 2__ 3__	0.3	0.33	0.35	
Control en la Violación de las Metodologías	1__ 2__ 3__	0.35	0.33	0.35	
Control del Ajuste a los Estándares	1__ 2__ 3__	0.35	0.33	0.3	

9. Para el Flujo de Trabajo de Análisis y Diseño

Actividad a medir	**Peso**	**Pesos Relativos**			**Otras**
Habilidad de un equipo en encontrar errores antes de que pasen a la siguiente actividad o tarea de ingeniería del software	1__ 2__ 3__	0.4	0.3	0.2	
Control de las Casos de Uso Diseñados	1__ 2__ 3__	0.4	0.3	0.5	
Control del bajo acoplamiento en los módulos	1__ 2__ 3__	0.2	0.3	0.3	

10. Para el Flujo de Trabajo de Implementación

Actividad a medir	**Peso**	**Pesos Relativos**			**Otras**
Control de los Casos de Uso Implementados	1__ 2__ 3__	0.6	0.5	0.7	
Control de las Pruebas de Unidad	1__ 2__ 3__	0.4	0.5	0.3	

11. Para el Flujo de Trabajo de Prueba

Actividad a medir	**Peso**	**Pesos Relativos**			**Otras**
Eficacia de la Eliminación de Defectos	1__ 2__ 3__	0.3	0.25	0.3	
Propagación de defectos (defectos ocasionados al corregir otros)	1__ 2__ 3__	0.3	0.25	0.3	
Efectividad de las Pruebas (con respecto a los objetos que no han sido probados en el software)	1__ 2__ 3__	0.2	0.25	0.3	
Porciento de pruebas diseñadas con respecto al valor de la complejidad ciclomática	1__ 2__ 3__	0.*2*	0.25	0.1	

Anexo 2: Primitivas de cada una de las métricas incluidas en la propuesta de solución.

CImp: número de componentes que fueron probados individualmente antes de la integración.

CN: componentes nuevos.

CNI: cambios registrados que se han efectuado y no han sido informados al personal relacionado.

CNNR: componentes nuevos que pueden ser sustituidos por componentes reutilizables.

CP: costo planificado.

CPI: número total de componentes implementados.

CR: costo real.

CReg: cambios registrados que se han efectuado.

CUD: número de CU que fueron diseñados.

CUDef: número de CU que fueron definidos.

CUI: número de Caso de Uso que fueron implementados.

E: número de errores encontrados antes de la entrega del software al usuario final.

E_a: número de errores encontrados durante la revisión a.

$E_{(a+1)}$: número de errores encontrados durante la revisión siguiente a la revisión a.

EEA: cantidad de elementos de la especificación que fueron aceptados.

EEO: cantidad de elementos de la especificación obtenidos.

E_i: número de errores encontrado durante la actividad de ingeniería del software i.

E_{i+1}: número de errores encontrado durante la actividad de ingeniería del software i + I.

ER: número de elementos que se revisaron en la verificación de ajuste a los estándares.

ERE: número de elementos revisados que cumplen con los estándares preestablecidos.

Est_i: número de estrategias para el manejo de riesgos de la fuente i.

D: número de defectos encontrados después de la entrega.

F_a: número de módulos en la versión actual que se han añadido.

F_c: número de módulos en la versión actual que se han cambiado.

F_d: número de módulos de la versión anterior que se han borrado en la versión actual.

F_i: fuentes de riesgos.

M_t: número de módulos en la versión actual.

NC: cantidad de no conformidades que se han identificado en una etapa.

NCR: cantidad de no conformidades que han sido resueltas.

P: número de pruebas diseñadas.

PA: número de procesos analizados.

PAp: peticiones aprobadas.

PCo: peticiones en cola.

PI: personal indocumentado.

Pre: peticiones posibles de rechazar.

PVM: número de procesos analizados que violan la metodología seleccionada a utilizar en el proyecto.

R: número de riesgos identificados que se pueden mitigar.

RG: riesgos que pertenecen a la categoría riesgos genéricos.

RGONI: número de riesgos ocurridos en un proyecto que no fueron identificados pertenecientes a la categoría riesgos genéricos.

RM: número de riesgos mitigados.

RO: número de riesgos ocurridos en un proyecto.

ROI: número de riesgos ocurridos en un proyecto que fueron identificados.

RONI: número de riesgos ocurridos en un proyecto que no fueron identificados.

TGCR: tiempo dedicado a la Gestión de Cambio de los Requisitos.

TP: total de peticiones.

TPe: total de personal que participa en el proyecto.

TPGR: tiempo total dedicado al proceso de Gestión de Requisitos.

TPla: total de tareas planificadas para la etapa.

TR: tareas con inicio retrasado.

V(G):complejidad ciclomática.

<u>Anexo 3:</u> Siglas que identifican a cada una de las métricas incluidas en la propuesta de solución.

(ACR): Aprovechamiento de los Componentes Reutilizables

(AGCR): Agilidad en la Gestión de Cambio de los Requisitos

(AMR): Agilidad en la Preparación para el Manejo de Riesgos

(CAE): Control del Ajuste a los Estándares

(CCUD): Control de los Casos de Uso Diseñados

(CCUI): Control de los CU Implementados

(CDP): Control de los Datos del Personal

(CIC): Competencia en la Información de los Cambios

(CPU): Control de las Pruebas de Unidad

(CTR): Control de las Tareas Retrasadas

(DRNC): Destreza en la Resolución de No Conformidades

(EED): Métrica de la Eficacia de la Eliminación de Defectos

(EMR): Eficacia de la Mitigación de Riesgos

(EOR): Eficiencia en la Obtención de los Requisitos

(EP): Error de Planificación

(EPC): Estado de Pedidos de Cambio

(IDRG): Identificación de los Riesgos Genéricos

(IMS): Índice de Madurez del Software

(MVM): Métrica para la Violación de las Metodologías

(m_C): Acoplamiento de Módulos

(PCB): pruebas de camino básico

(PCE): Productividad de la Corrección de los Errores

(RCP): Razón de Costo de Planificación

(TEP): tasa de efectividad de las pruebas

(TPD): tasa de propagación de defectos

<u>Anexo 4:</u> Modelo 1 de la encuesta para la validación técnica de la propuesta.

Guía para informar el peso de los criterios.

Fecha de recepción _____

Fecha de entrega _____

Nombre y Apellidos del evaluador _____

Pesos Relativos

Grupo No. 1…….25

Grupo No. 2…….30

Grupo No.3……..25

Grupo No.4……..20

Usted como experto debe conceder pesos a cada uno de los criterios establecidos, teniendo en cuenta que la suma de los valores dados, para un grupo, no exceda del peso relativo asignado a este.

Grupo No 1: Criterios de mérito científico.

1. Valor científico de la propuesta.

   Peso __

2. Calidad de la investigación.

   Peso __

3. Contribución científica.

   Peso __

Grupo No 2: Criterios de implantación

4. Necesidad de empleo de la propuesta.

   Peso __

5. Obtención de productos finales con calidad.

   Peso __

6. Posibilidades de aplicación.

   Peso __

Grupo No 3: Criterios de flexibilidad.

7. Adaptabilidad a proyectos productivos de SWE independientemente del tipo de SWE que desarrollen.

   Peso __

8. Capacidad de las métricas propuestas para adecuarse a la evaluación de las diferentes áreas de procesos de SWE.

   Peso __

9. Capacidad de las métricas propuestas para adecuarse a la evaluación de procesos de desarrollo de proyectos productivos que no desarrollen SWE.

   Peso __

Grupo No 4. Criterios de impacto.

10. Efectos en la mejora de la evaluación del modelo MEPDSE.

    Peso __

11. Repercusión en la calidad de los procesos de desarrollo en los proyectos productivos de SWE.

    Peso __

<u>Anexo 5:</u> Modelo 2 de la encuesta para la validación técnica de la propuesta.

Guía para la evaluación.

Fecha de recepción _____

Fecha de entrega _____

Nombre y Apellidos del evaluador_____

Usted como experto debe conceder una evaluación cuantitativa de cada criterio con una escala de 1 a 5.

Grupo No 1: Criterios de mérito científico.

1. Valor científico de la propuesta.

   Peso __

2. Calidad de la investigación.

   Peso __

3. Contribución científica.

   Peso __

Grupo No 2: Criterios de implantación

4. Necesidad de empleo de la propuesta.

   Peso __

5. Obtención de productos finales con calidad.

   Peso __

6. Posibilidades de aplicación.

   Peso __

Grupo No 3: Criterios de flexibilidad.

7. Adaptabilidad a proyectos productivos de SWE independientemente del tipo de SWE que desarrollen.

   Peso __

8. Capacidad de las métricas propuestas para adecuarse a la evaluación de las diferentes áreas de procesos de SWE.

   Peso __

9. Capacidad de las métricas propuestas para adecuarse a la evaluación de procesos de desarrollo de proyectos productivos que no desarrollen SWE.

   Peso __

Grupo No 4. Criterios de impacto.

10. Efectos en la mejora de la evaluación del modelo MEPDSE.

    Peso __

11. Repercusión en la calidad de los procesos de desarrollo en los proyectos productivos de SWE.

    Peso __

Exprese su apreciación cualitativa con una clasificación final del proyecto en excelente, bueno, aceptable, cuestionable o malo.

___ Excelente: Alta novedad científica, con aplicabilidad y resultados relevantes.

___ Bueno: Novedad científica, resultados destacados.

___ Aceptable: Suficientemente bueno con reservas.

___ Cuestionable: No tiene relevancia científica y los resultados son malos.

___ Malo: No aplicable.

Formule su opinión, haciendo una valoración final de la propuesta y sugiera todas las consideraciones que estimen convenientes, así como todos los elementos críticos que deben mejorarse.