Universidad de las Ciencias Informáticas

Facultad 8

Propuesta de métricas para la evaluación de Requisitos No Funcionales.

TRABAJO DE DIPLOMA PARA OPTAR POR EL TÍTULO DE

INGENIERO EN CIENCIAS INFORMÁTICAS

Autoras: Annie García Ramírez.

Adriana María Almaguer Castro.

Tutora: Ing. Yasirys Terry González.

Cotutora: Msc. Ing. Karina Pérez Teruel.

Ciudad de La Habana, junio del 2008

"Año 50 de la Revolución"

DECLARACIÓN DE AUTORÍA

Declaramos ser autoras 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 _____.

_____ _____

Adriana María Almaguer Castro

Autora

Annie García Ramírez

Autora

_____ _____

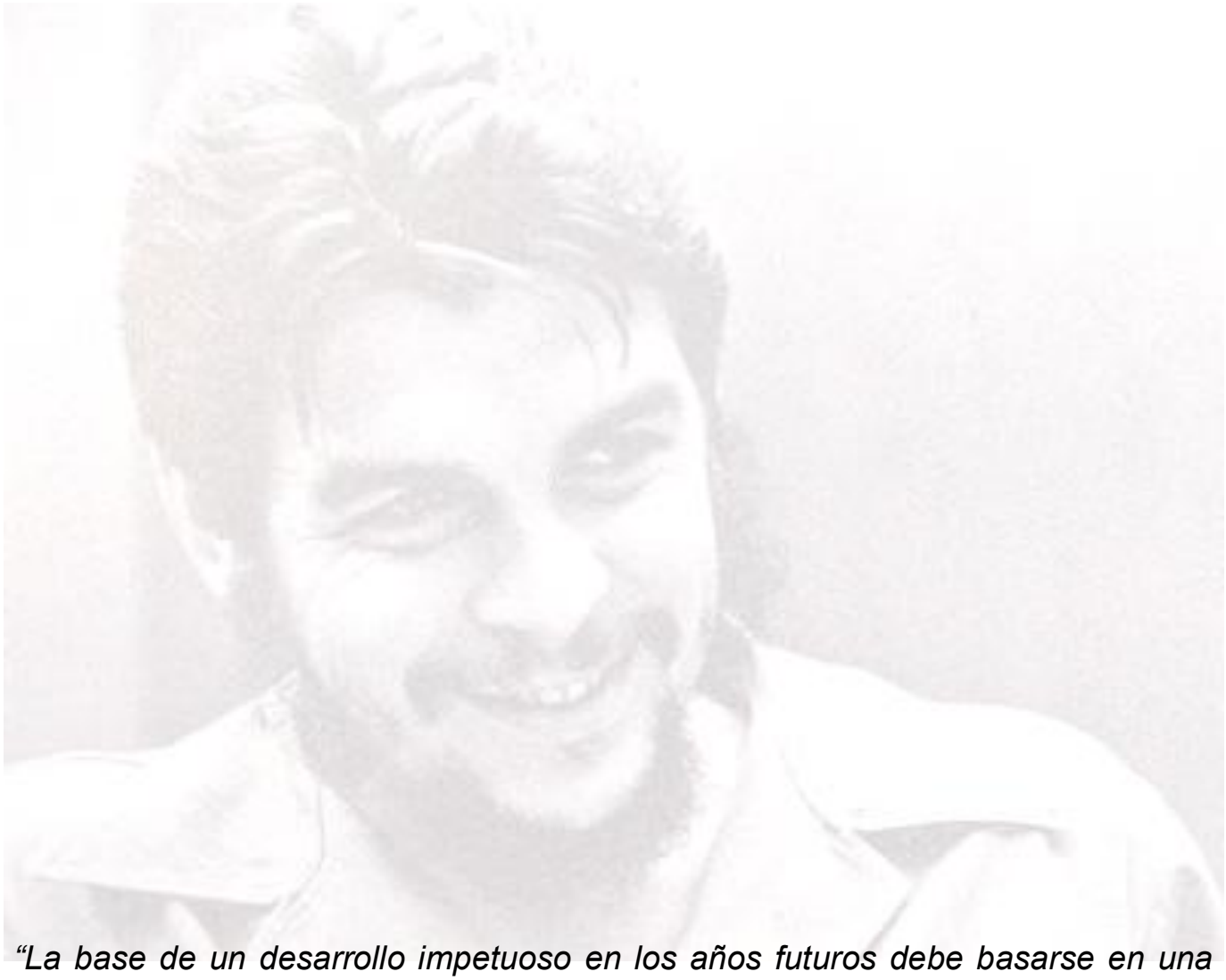
Ing. Yasirys Terry González

Tutora

Msc. Ing. Karina Pérez Teruel

Cotutora

Pensamiento



“La base de un desarrollo impetuoso en los años futuros debe basarse en una Ciencia cada vez más desarrollada”.

Ernesto Che Guevara

Agradecimientos

Agradecer por encima de todo a mi mamita y papito de mi alma, los seres que más quiero en esta tierra, que nunca dejaron de estar ahí y confiaron ciegamente en mí.

A Alian, para crea que no hay proyectos imposibles, solo es cuestión de intentar.

A mi tía Xiomara que siempre me alentó a estudiar y ser mejor, que aunque no está ahora conmigo siempre estuvo en mis sueños cada noche.

A mis compañeros, que juntos disfrutamos y trabajamos estos 5 años, gracias por hacerme cambiar de forma positiva. Aportaron bastante a esta personita que soy. Los extrañaré mucho, a los que están y los que no.

A Annie por aguantar mis locuras como compañera de tesis y juntas aguantar las sugerencias o felicitaciones en este tiempo.

A mi tutora y cotutora que aguantaron todas nuestras perretas, y tuvieron paciencia con nuestros despistes.

A mis primos Denia, Alexis y mi tío Yonny por aguantarme y darme cariño cada fin de semana.

Gracias a mi Pirata por hacerme entender que no soy poca cosa, que soy más de lo que aparento, y ayudarme ante el estrés o la fatiga.

A mi familia por incitarme y hacerme creer que puedo ser mejor cada día.

Pero nunca dejare de agradecer a Fidel, que tuvo la grandiosa idea de unirnos a todos aquí, gracias darme la oportunidad de ser la profesional que soy y gracias por hacer que este sueño se hiciese más que una realidad….

Adriana

Agradecimientos

Mi agradecimiento especial es para toda mi familia, por su preocupación y cariño, por apoyarme y confiar en mí.

Agradezco todo el constante apoyo y amor que me han brindado mis padres Alexis García y Odelis Ramírez, en todo momento en buenas y malas, gracias por existir. A mi hermanito Alexito que lo quiero mucho y que estudie para que pueda alcanzar a ser lo que él desea.

A mis tíos maternos que siempre estaban pendientes de mis estudios, en especial a mi tía Clarisbel gracias por tus consejos y tu ayuda, al igual que a mi prima Yoandra que siempre que necesito me ayuda incondicionalmente.

A mis compañeros de aula, que me han aguantado todo este tiempo y que no debe de ser fácil, que siempre los voy a recordar pues he aprendido mucho de cada uno, y en especial a Adriana que ha sido la mejor compañera de tesis, que me ha soportado las pesadeces y ha tenido paciencia para explicarme cada cosa.

A todos mis profesores que contribuyeron a que me formara como profesional.

A Karina y Yasirys por su exigencia y ayuda porque nosotras estuviésemos cada parte de esta tesis en tiempo para cada corte.

A mi René que siempre me recordaba la tesis que tenía que hacer y prepararme, por regañarme cada vez que quería dormir, por estar a mi lado, apoyarme y quererme.

A Fidel y la Revolución porque sin ellos no habría sido posible que este sueño se hiciese realidad.

Muchas gracias.

Annie.

Dedicatoria

A mi mami Rosa María Castro Sánchez, a mi papi Adrián Almaguer Valdés .Esta victoria es de ellos también, a quien más he de dedicarle, si yo he sido el fruto de sus manos.

A mi hermanito Alian para crea y busque más haya de sueños.

A toda mi familia, que siempre creyó y espero tanto de mi.

A mis amigos que fueron mi otra familia durante estos años.

A todo el que cree y va más allá de sus sueños

Adriana

A mis papitos lindos por ser mis guías, por estar siempre al tanto de mis estudios y apoyándome en todo momento.

Annie.

Resumen

La calidad ha dejado de ser un tópico y es necesario que forme parte de los productos o servicios que se comercializan para los clientes. El cliente es el mejor auditor de la calidad, él exige el nivel que está dispuesto a pagar por ella, pero no más. Por tanto, se debe de cuantificar cuál es el nivel de calidad que se exige para poder planificar la calidad de los productos que se generan a lo largo de la producción del producto o servicio final.

Las métricas de software proporcionan información objetiva que contribuye al mejoramiento de los procesos y productos de software, lo cual favorece al logro de la calidad.

El propósito de este proyecto de investigación es proponer métricas que permitan evaluar la calidad de los Requisitos No Funcionales y la satisfacción de los mismos en la línea base de arquitectura. Para desarrollar la propuesta fue necesario realizar un estudio del estado del arte de las métricas, y propuesto por las autoras se plantea un conjunto de métricas del estándar ISO/IEC 9126 adaptadas a los Requisitos No Funcionales. Con el propósito de obtener una valoración preliminar que permitiera la mejora de la propuesta se realizaron encuestas a expertos en el tema.

Índice

Índice de Contenido

Índice

Índice de Imágenes

Índice Tablas

Introducción

El desarrollo de la producción de software en la Universidad de las Ciencias Informáticas (UCI) ha incrementado su competitividad a través de la difusión, la mejora continua y el conocimiento en tecnologías de información. Hoy se orientan los esfuerzos hacia el entrenamiento, diagnóstico, consultoría y formación según demanda de los proyectos productivos de la Universidad. Se espera elevar la calidad del software que se produce, garantizando y certificando la calidad de los productos, e implantar mecanismos que permitan mejorar la calidad de los procesos de desarrollo de software. En su constante desarrollo se promueve la investigación y la búsqueda de soluciones de los principales problemas en el área de Ingeniería y la Calidad de Software.[1]

La calidad de los productos de software hoy en día ha dado un despunte en Cuba con la creación de la UCI, los productos de desarrollo de software aumentan rápidamente, pero a su vez deben ser mejores, es decir, aumentar la calidad es un elemento indispensable.[2]

Los Requisitos No Funcionales forman una parte importante en el logro de la calidad de los productos de software, principalmente para que clientes y usuarios puedan valorar las características no funcionales del producto. Cumpliendo con toda la funcionalidad requerida entonces las propiedades no funcionales, como cuán usable, seguro, conveniente y agradable, pueden marcar la diferencia entre un producto bien aceptado y uno con poca aceptación.[3]

Los Requisitos No Funcionales luego de ser evaluados son una parte indispensable en la evaluación de la arquitectura, siendo esta tarea una manera más económica de evitar desastres. Cuanto más temprano se encuentre un problema en un proyecto de software, mejor. El costo de arreglar un error durante las fases de tempranas, es mucho menor al costo de arreglar ese mismo error en la fase final. Dado que la arquitectura es un producto temprano de la fase de diseño, esta tiene un profundo efecto en el sistema y en el proyecto. Es mejor cambiar la arquitectura antes que otros artefactos, que están basados en ella, se estabilicen.

El cliente solicita en innumerables ocasiones que su producto posea propiedades que hacen al producto atractivo, usable, rápido o confiable, por ejemplo, pudiera desearse que el sistema responda dentro de un intervalo de tiempo especificado o que obtenga los resultados de los cálculos con un nivel de precisión dado. Debido a esto los Requisitos No Funcionales son fundamentales en el éxito del producto.[4]

Pero regularmente el cliente no domina o conoce las cualidades o propiedades no funcionales necesarias para que su negocio se pueda convertir en software, comenzando desde este momento a introducirse errores que pueden afectar la calidad del producto. Puede que en este momento no causen problemas pero al entregar el producto y enfrentarlo al usuario es donde estos resaltan. Algunos problemas que responden al caso anterior pueden ser: que el usuario no posee el hardware o soporte necesario para que el producto tenga la rapidez requerida, o que las características del software sean superficiales y de poca importancia.

Otro problema frecuente es la captura de requisitos, que no siempre se realiza por los mismos especialistas ni estos se rigen por una guía realizando su registro en documentos de cientos de páginas. [3] Los requisitos no son siempre obvios y vienen dados por distintas fuentes, no son fáciles de expresar textualmente, además hay muchos tipos de requisitos con muy distintos niveles de detalle, su número puede ser inmanejable si no se controla adecuadamente, no todos son igualmente importantes, ni tienen la misma dificultad, cambian constantemente y pueden ser dependientes del tiempo.[5] Los desarrolladores no prestan en el levantamiento de requisitos igual atención a los Requisitos No Funcionales que a los funcionales e incluso no existe en la Universidad propuesta alguna que mida cuán satisfactorios han sido las características no funcionales.

En ningún momento los Requisitos No Funcionales son evaluados con anterioridad a la entrega final del producto. Todos aquellos errores que se han cometido en la etapa de requisitos tienen luego una inmensa repercusión, es decir, son altamente perjudiciales; ante un error, omisión o mala interpretación en los requisitos, los desarrolladores se han visto obligados a rehacer todo el trabajo hecho sobre la base de requisitos incorrectos, provocando un arduo y costoso trabajo de reprogramación para corregirlos.

Se hace necesario entonces evaluar los Requisitos No Funcionales, pues al transcurrir el tiempo las sucesivas evaluaciones suministran una valiosa información permitiendo desarrollar un producto más acertado, mejorar costos y satisfacer en mayor medida las necesidades del cliente, asegurando el cumplimiento en los plazos establecidos.[6]

A raíz de las condiciones descritas hasta el momento, se identificó como **problema científico** de esta investigación:

La calidad de los productos de desarrollo de software se ve afectada por la no satisfacción de los Requisitos No Funcionales en la Línea Base de la Arquitectura.

El problema descrito genera como **objeto de estudio** de esta investigación la Ingeniería de Requisitos (IR).

El **campo de Acción** está conformado por las Métricas para evaluar Requisitos No Funcionales.
Para darle solución al problema científico mencionado se planteó como **objetivo general:** Desarrollar una propuesta de métricas para la evaluación de la calidad y la satisfacción de los Requisitos No Funcionales de los productos de software en la arquitectura candidata y principalmente en la Línea Base de la Arquitectura.
Al mismo tiempo se plantearon los siguientes **objetivos específicos** para un mejor logro del objetivo general descrito anteriormente:

- Elaborar marco teórico de investigación.
- Proponer métricas para la evaluación de la calidad y satisfacción de los Requisitos No Funcionales en la Línea Base de la Arquitectura.
- Validar la propuesta de métricas a través de sistema de expertos.

La investigación científica se guiará por el conjunto de **tareas** que se describen a continuación para darle cumplimiento a los objetivos específicos planteados:

- Elaborar marco teórico de investigación.
- Estudiar e identificar métodos científicos de investigación.
- Realizar un estudio del estado del arte sobre la IR, métricas y las características que hacen necesaria la evaluación de los Requisitos No Funcionales.
- Seleccionar las métricas para la evaluación de RNF.
- Elaborar los criterios de evaluación de acuerdo a las características de la propuesta.
- Seleccionar a los expertos que participarán en la validación de la propuesta.

Se plantea como **Idea a defender** en la presente investigación la siguiente:

El desarrollo e implantación de una propuesta de métricas para la evaluación de la calidad de los Requisitos No Funcionales de software y la satisfacción de los mismos en la Línea Base de la Arquitectura, influirá positivamente en la calidad final de los productos de software.

El presente documento fue estructurado en tres capítulos. A continuación se dará una breve descripción de los contenidos que se abordarán en los mismos:

Capítulo 1 Fundamentación Teórica: Se encuentra todo el contenido que se relaciona con los conceptos fundamentales, necesarios para el dominio de la propuesta como Ingeniería de Requisitos, y el caso en particular Requisitos No Funcionales, métricas y sus clasificaciones, estándares de medición de software que rigen la evaluación de los productos.

Capítulo 2 Descripción de la Propuesta: Se encuentra un estudio de las métricas seleccionadas, dando paso a la selección de los atributos de calidad, contradicciones y relaciones de los atributos que permitirán medir la calidad y la satisfacción de los atributos a través de las métricas más exactas así como un análisis de los resultados en cada uno de los Requisitos No Funcionales evaluados.

Capítulo 3 Validación de la propuesta: Se realiza la evaluación técnica de la propuesta a través del método multicriterios basado en un sistema de expertos que es explicado en dicho capítulo.

Capítulo 1: Fundamentación Teórica

Introducción

En este capítulo se encuentran los conceptos fundamentales sobre Ingeniería de Requisitos, en particular los Requisitos No Funcionales y sus categorías, que ayudan a comprender por qué es necesario hacer una mejor evaluación. Además se encuentra estudios sobre métricas y estándares que son gran relevancia para la realización de la propuesta.

1.1 Ingeniería de Requisitos

Existen diversos conceptos sobre la IR pero los que a continuación se muestran, son los que aportan elementos significativos para la investigación como las metas, los elementos básicos que la conforman y principales artefactos.

“La Ingeniería de Requisitos es el proceso de desarrollar una especificación de software. Las especificaciones pretenden comunicar las necesidades del sistema del cliente a los desarrolladores del sistema”.[7]

Trata de los principios, métodos, técnicas y herramientas que permiten descubrir, documentar y mantener los requisitos para sistemas basados en computadora, de forma sistemática y repetible.[6]

El proceso de recopilar, analizar y verificar las necesidades del cliente o usuario para un sistema es llamado Ingeniería de Requisitos. La meta de la Ingeniería de Requisitos es entregar una especificación de requisitos de software correcta y completa.

1.1.1 Importancia de la Ingeniería de Requisitos

Los principales beneficios que se obtienen de la Ingeniería de Requisitos son:

- Permite gestionar las necesidades del proyecto en forma estructurada: Cada actividad de la Ingeniería de Requisitos consiste de una serie de pasos organizados y bien definidos.

- Mejora la capacidad de predecir cronogramas de proyectos, así como sus resultados: La Ingeniería de Requisitos proporciona un punto de partida para controles subsecuentes y actividades de mantenimiento, tales como estimación de costos, tiempo y recursos necesarios.

- Disminuye los costos y retrasos del proyecto: Muchos estudios han demostrado que reparar errores por un mal desarrollo no descubierto a tiempo, es sumamente caro; especialmente aquellas decisiones tomadas durante la Especificación de Requisitos.

- Mejora la calidad del software: La calidad en el software tiene que ver con cumplir un conjunto de requisitos (funcionalidad, facilidad de uso, confiabilidad, desempeño, etc.).

- Mejora la comunicación entre equipos: La especificación de requisitos representa una forma de consenso entre clientes y desarrolladores. Si este consenso no ocurre, el proyecto no será exitoso.

- Evita rechazos de usuarios finales: La Ingeniería de Requisitos obliga al cliente a considerar sus requisitos cuidadosamente y revisarlos dentro del marco del problema, por lo que se le involucra durante todo el desarrollo del proyecto.[8]

1.1.2 Actividades de la Ingeniería de Requisitos

Existen cuatro actividades básicas (extracción, análisis, especificación y validación) que se tienen que llevar a cabo para completar el proceso. Estas actividades ayudan a reconocer la importancia que tiene, para el desarrollo de un proyecto de software, realizar una especificación y administración adecuada de los requisitos de los clientes o usuarios.[8]

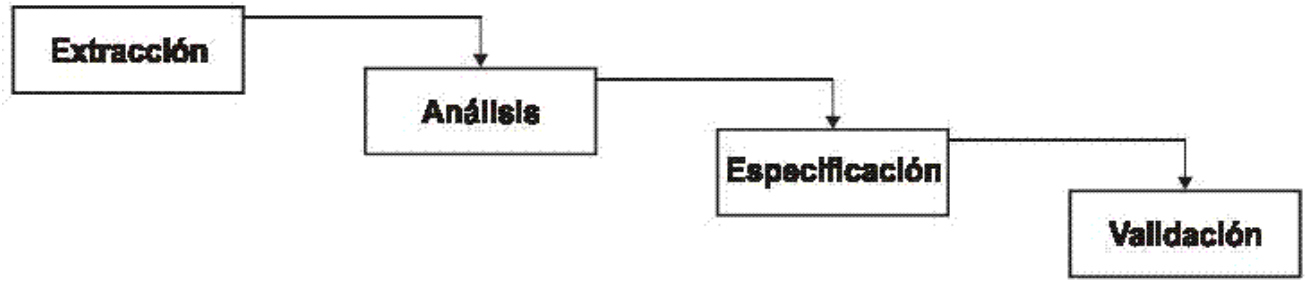


Figura 1 Actividades de la Ingeniería de Requisitos.

Extracción

Esta fase representa el comienzo de cada ciclo. Extracción es el nombre comúnmente dado a las actividades involucradas en el descubrimiento de los requisitos del sistema.

Análisis

Sobre la base de la extracción realizada previamente, comienza esta fase. Usualmente se hace un análisis luego de haber producido un bosquejo inicial del documento de requisitos; aquí se leen los requisitos, se conceptúan, se investigan, se intercambian ideas con el resto del equipo, se resaltan los problemas, se buscan alternativas y soluciones, y luego se van fijando reuniones con el cliente para discutir los requisitos.

Especificación

En esta fase se documentan los requisitos acordados con el cliente, en un nivel apropiado de detalle. En la práctica, esta etapa se va realizando conjuntamente con el análisis, pero se podría decir que la Especificación es el "pasar en limpio" el análisis realizado previamente aplicando técnicas y/o estándares de documentación, como la notación UML.

Validación

La validación es la etapa final de la IR. Su objetivo es verificar todos los requisitos que aparecen en el documento especificado para asegurarse que representan una descripción, por lo menos, aceptable del sistema que se debe implementar. Esto implica verificar que los requisitos sean consistentes y que estén completos.

La validación representa un punto de control interno y externo; interno, porque se debe verificar internamente lo que se está haciendo, y externo, porque se debe validar con el cliente.[9]

1.1.3 Herramientas de la Ingeniería de Requisitos

Existen diversas técnicas y herramientas que se utilizan para llevar a cabo cada una de las actividades del proceso de Ingeniería de Requisitos como las que se muestran en la figura 1.2, una de las razones por las cuales surgen los errores a la hora del levantamiento es la existencia de una gama de herramientas. No existe una especie de guía para el uso de los desarrolladores, estos utilizan incluso en la captura más de una técnica en cada de las actividades que contiene el proceso.

Herramientas	Extracción	Análisis	Especificación	Validación
Entrevistas y cuestionario	X			
Sistemas existentes	X	X		
Grabaciones de video y de audio.	X	X		
Brainstorming (tormenta de ideas)	X	X		
Arqueología de documentos	X	X		
Aprendiz	X			
Observación	X			
Run Use Case WorkShop	X			
Prototipo Bosquejado	X	X	X	
Prototipo Tangible/usable	X		X	X
FODA		X		
Cadena de valor		X		
Modelo de clase conceptual		X	X	
Diagrama de pescado	X	X	X	
Glosario	X	X	X	X
DCO		X	X	
Diagrama de actividad		X	X	
ESRE	X	X	X	X
Casos de uso	X	X	X	X
Casa de calidad o QFD				X
Checklist	X			X

Figura 2 Herramientas que se utilizan en el proceso de Ingeniería de Requisitos.

Entre las herramientas más usadas se puede encontrar:

Entrevistas y cuestionarios

Las entrevistas y cuestionarios se emplean para reunir información proveniente de personas o grupos, información que se obtiene conversando con el encuestado.

Las preguntas suelen distinguirse en dos categorías: abiertas y cerradas.

Las preguntas abiertas permiten que los encuestados respondan con su propia terminología, mientras que las preguntas cerradas predeterminan todas las posibles respuestas y el interrogado elige entre las opciones presentadas.[9]

Grabaciones de video y de audio

Básicamente existen dos formas de utilizar las grabaciones: como registro y apoyo de las entrevistas, y para analizar algún proceso en particular.

En cuanto a su función de apoyo, es importante porque permite centrar la atención en la entrevista en sí, en vez de distraerse tomando notas de todo lo que se dice.

Cuando se trata de analizar algún proceso en particular, su ayuda es inestimable (sobretodo las filmaciones de video) porque permite ver y analizar en detalle ese proceso la cantidad de veces que sea necesario.[9]

Brainstorming (tormenta de ideas)

Este es un modelo que se usa para generar ideas. La intención en su aplicación es la de generar la máxima cantidad posible de requisitos para el sistema. No hay que detenerse en pensar si la idea es o no del todo utilizable.

1.2 Requisitos

"Una condición o capacidad que debe estar presente en un sistema o componentes de sistema para satisfacer un contrato, estándar, especificación u otro documento formal".[5]

"Un requisito es simplemente una declaración abstracta de alto nivel de un servicio que debe proporcionar el sistema o una restricción de éste".[7]

Los requisitos de software pueden **clasificarse** en 2 categorías: requisitos funcionales y Requisitos No Funcionales.

Los requisitos funcionales son los que definen las funciones que el sistema será capaz de realizar.

Los Requisitos No Funcionales tienen que ver con características que de una u otra forma puedan limitar el sistema.[5]

Analizando las definiciones anteriores, se establece en la investigación que un requisito es una descripción de una condición o capacidad que debe cumplir un sistema. Se caracteriza por ser:

- Especificado por escrito: Como todo contrato o acuerdo entre dos partes.
- Posible de probar o verificar .Si un requerimiento no se puede comprobar, entonces ¿Cómo se sabe si se cumplió con él o no?
- Conciso: Un requerimiento es conciso si es fácil de leer y entender. Su redacción debe ser simple y clara para aquellos que vayan a consultarlo en un futuro.
- Completo: Un requerimiento está completo si no necesita ampliar detalles en su redacción, es decir si se proporciona la información para su comprensión.
- Consistente: Un requerimiento es consistente si no es contradictorio con otro requerimiento.
- No ambiguo: Un requerimiento no es ambiguo cuando tiene una sola interpretación. El lenguaje usado en su definición, no debe causar confusiones al lector.[5]

1.2.1 Requisitos No Funcionales

La Especificación de Requisitos Software (ERS) también contiene Requisitos No Funcionales (o complementarios). Los Requisitos No Funcionales son requisitos que imponen restricciones en el diseño o la implementación como por ejemplo restricciones en el diseño o estándares de calidad.

Los Requisitos No Funcionales son propiedades o cualidades que el producto debe tener. Debe pensarse en estas propiedades como las características que hacen al producto atractivo, usable, rápido o confiable.[3]

Se conocen como un conjunto de características de calidad, que es necesario tener en cuenta al diseñar e implementar el software.

No son parte de la razón fundamental del producto pero si son necesarios para hacer funcionar el producto de la manera deseada. No modifican la funcionalidad del producto y si añaden funcionalidad al producto. Describen la experiencia del usuario cuando trabaja con el producto y fundamentalmente son las características que se representan por casos de usos.[10]

Algunos de los atributos propios de un sistema eficaz no se pueden describir en términos de funcionalidad. En la práctica, los Requisitos No Funcionales son primordiales para el éxito de estos sistemas. Si bien los Requisitos No Funcionales suelen ser difíciles de definir y cuantificar con objetividad, es importante identificarlos, al menos en términos generales, para que puedan estudiarse.

Es muy difícil establecer una separación entre requisitos funcionales y no funcionales, ya que la decisión de si es uno u otro puede venir por el nivel de detalle del documento de requisitos. Además, los Requisitos No Funcionales son difíciles de expresar, y mucho más de ser recogidos en un documento de requisitos utilizando las mismas técnicas que para los requisitos funcionales.[11]

Hay que tener en cuenta, que normalmente, los errores debidos a Requisitos No Funcionales son los más difíciles y caros de resolver. Los RNF deben establecer restricciones en el producto que está siendo desarrollado, en el proceso de desarrollo y en restricciones específicas que el producto pueda tener.

Los Requisitos No Funcionales son difíciles de verificar/testear, y por ello son evaluados subjetivamente.[12]

1.2.2 Categorías de Requisitos No Funcionales

Existen diferentes categorías de los Requisitos No Funcionales entre las que se puede mencionar se encuentran: requisitos de Apariencia, de Usabilidad, de Rendimiento, de Mantenibilidad y Portabilidad, de Seguridad, Culturales y Políticos, y Legales.

Otros especialistas como Brito, Moreira y Araujo señalan que existen los requisitos de Interfaz, de Desempeño, Operacionales y Económicos. Especialistas como Viega y McGraw los requisitos relacionados con la seguridad son los de Funcionalidad, Eficiencia y Simplicidad, como en la figura que se muestra a continuación.[10]

Figura 3 Jerarquía de requisitos.

Orientados al usuario

Existen clasificaciones de requisitos que solapan con otras, o contienen algunas en su interior, como esta clasificación. Tal es el caso los requisitos Orientados al usuario que contiene aquellos que están dirigidos al trabajo específico del cliente con el sistema.

Fiabilidad, Seguridad (security), Seguridad (safety), Usabilidad, Robustez, Disponibilidad,

Rendimiento (Tiempo de respuesta, Capacidad, Throughput).

Requisitos de Seguridad

La seguridad de un sistema no solo tiene en cuenta la seguridad del sistema sino, además, el ambiente en el que se usará el sistema. Por lo que se tiene que contemplar la seguridad física del lugar donde se usa la aplicación, los controles administrativos que se establecen de acceso al sistema y las regulaciones legales que afecta o determina el uso del sistema y que serán tenidas en cuenta si se incumple.

La seguridad puede ser tratada en tres aspectos diferentes:

- Confidencialidad: La información manejada por el sistema está protegida de acceso no autorizado y divulgación.

- Integridad: La información manejada por el sistema será objeto de cuidadosa protección contra la corrupción y estados inconsistentes, de la misma forma será considerada igual a la fuente o autoridad de los datos. Pueden incluir también mecanismos de chequeo de integridad y realización de auditorias.

- Disponibilidad: Significa que los usuarios autorizados se les garantizará el acceso a la información y que los dispositivos o mecanismos utilizados para lograr la seguridad no ocultarán o retrasarán a los usuarios para obtener los datos deseados en un momento dado.[3]

Requisito de Rendimiento y Escalabilidad

Se debe de tener en cuenta el requisito de Rendimiento y Escalabilidad porque convendría que los usuarios examinaran con detenimiento hasta qué punto el proyecto se ajusta a sus expectativas en cuanto a los tiempos de respuesta y es capaz de prestar servicio adecuadamente de acuerdo al tipo y tamaño para el que ha sido concebido.[3]

Requisito de Disponibilidad

También es importante la Disponibilidad del sistema, pues hoy en día las instituciones o centros de trabajos utilizan los sistemas informáticos, conectados a la red. Un cambio fundamental lo constituirá el aumento espectacular de la dependencia de los usuarios con respecto a la red, de modo que si el software dejara de estar disponible, a los usuarios les será imposible continuar su trabajo. Por consiguiente, convendría que los usuarios que se disponen a contratar un sistema identifiquen las necesidades de sus usuarios en cuanto a disponibilidad y las detallen en el momento de la contratación.[3]

Requisito de Fiabilidad

Uno de los más importantes por parte del usuario es de Fiabilidad debido a que debe de tener en cuenta la recuperación frente a fallos de conexión: asegurar que no se pierdan los datos del perfil definido por el usuario. Esto incluye no perderlos en el envío al servidor o en el envío a otras máquinas, como no perderlos si hay un fallo de conexión entre el receptor del usuario y el servidor. La recuperación frente a fallos del sistema: posibilidad de reiniciar el sistema. Verificar la Fiabilidad en la autenticación de los usuarios y la posibilidad de dar marcha atrás en la definición del perfil de cada usuario.[3]

Orientados al desarrollador

Esta es semejante a la clasificación Orientado al usuario, surge por las mismas razones, para unificar aquellas clasificaciones con las que trabaja directamente el desarrollador como: Disponibilidad, Portabilidad, Adaptabilidad, Testabilidad, Comprensibilidad.[6]

Requisito de Facilidad de Uso

Teniendo en consideración los tipos de Requisitos No Funcionales que existen, hay algunos de suma importancia que son aplicables para los proyectos productivos como son los requisitos de Facilidad de Uso. La definición de este requisito es una cuestión de especial importancia, pues un proyecto puede fracasar porque sus usuarios no encuentren sencillo su uso, porque no exista una interfaz sencilla y atractiva, o un manual que describa el funcionamiento y el uso del sistema al usuario final.[6]

Requisito de Soporte

El requisito de Soporte que le brinda al usuario la facilidad de instalación, facilidad de mantenimiento, lo que requiere código y diseño documentado y facilidad de actualización hacia versiones más moderna.[6]

Requisitos de Software

Debe mencionarse el software del que se debe disponer, después de implementado el sistema.

Requisitos de Hardware

Se deben enunciar los elementos de hardware que se necesitan para que el software cumpla sus funcionalidades.

Restricciones en el diseño y la implementación

Especifica o restringe la codificación o construcción de un sistema, son restricciones que han sido ordenadas y deben ser cumplidas estrictamente.

Requisitos de apariencia o interfaz externa

Este tipo de requisito describe la apariencia del producto. Es importante destacar que no se trata del diseño de la interfaz en detalle sino que especifican cómo se pretende que sea la interfaz externa del producto. También pueden ser necesidades de cumplir con normas estándares, o con los estándares de la empresa para la cual se esté desarrollando el software.

Requisitos de Usabilidad

Describen los niveles apropiados de usabilidad, dados los usuarios finales del producto, para ello deben revisarse las especificaciones de los perfiles de usuarios y las clasificaciones de sus niveles de experiencia.

En el estudio anterior se muestra la diversidad en cuanto a las clasificaciones y las diferentes definiciones, mostrando de esta forma la primera interrogante: ¿Hacia cuáles clasificaciones dirigir la propuesta? En vista de la no concordancia entre los especialistas a la hora de clasificar los RNF, la Universidad se ha visto en la necesidad de elaborar sus propios conceptos encaminando sus proyectos productivos hacia estas.

1.3 Relación Requisitos - Arquitectura

La arquitectura de software de un programa o sistema de computación es la estructura o las estructuras del sistema, que contienen componentes de software, las propiedades externamente visibles de dichos componentes y las relaciones entre ellos.[13]

La arquitectura se refiere a una representación abstracta de los módulos/componentes de un sistema y su entorno.

La elección de una determinada arquitectura software debe tener en cuenta los requisitos funcionales pero, sobre todo, los Requisitos No Funcionales. No hay una regla definitiva para establecer, dados los requisitos, el tipo de arquitectura. Tan sólo hay una serie de heurísticas para, dados unos requisitos, elegir la arquitectura.[6]

La relación entre requisitos y arquitectura debe de ser estrecha, debido que para que se diseñe y se implemente el software, los elementos arquitectónicos deben satisfacer la funcionalidad y ejecución de los requisitos del sistema, por ejemplo tiene que cumplir con la fiabilidad, escalabilidad, portabilidad, etc.

Si las decisiones arquitectónicas determinan los atributos de calidad del sistema, entonces es posible evaluar las decisiones arquitectónicas con respecto a su impacto sobre dichos atributos.[6]

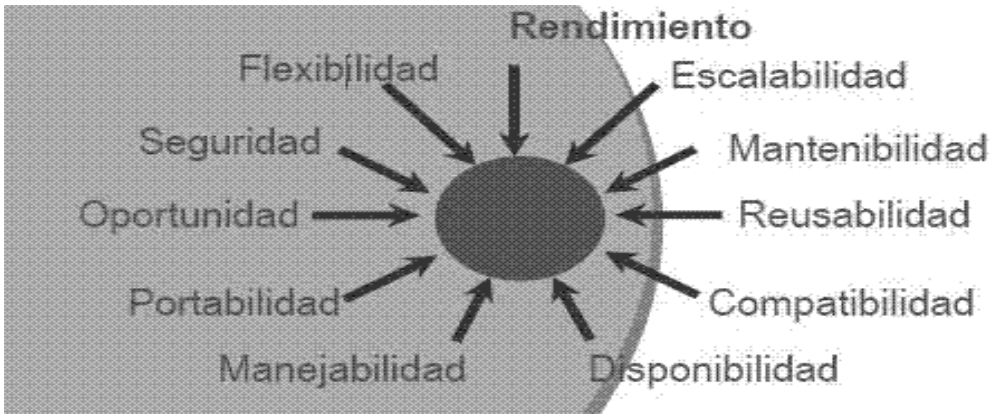


Figura 4 Atributos de calidad de la arquitectura.

1.4 Evaluación

La evaluación es una parte decisiva de esta investigación debido a que no es cuestión de aplicar métricas simplemente, es necesario dominar que razones llevan a medir pero sobre todo a evaluar, cuando es el momento más oportuno para efectuarla de la forma más acertada y así obtener resultados lo más concretos posibles. Por eso es necesario dominar realmente en consiste una evaluación.

Es un proceso que procura determinar, de la manera más sistemática y objetiva posible, la pertinencia, eficacia, eficiencia e impacto de actividades a la luz de los objetivos específicos. Constituye una herramienta administrativa de aprendizaje y un proceso organizativo orientado a la acción para mejorar tanto las actividades en marcha, como la planificación, programación y toma de decisiones futuras.[14]

La evaluación de programas y proyectos es un instrumento de gestión. Es un proceso de duración determinada, que trata de valorar de manera sistemática y objetiva la pertinencia, el rendimiento y el éxito de los programas y proyectos concluidos y en curso. Se realiza con carácter selectivo para dar respuesta a determinadas preguntas e impartir orientaciones a los encargados de tomar decisiones y administradores de programas. Posibilita además, la obtención de información que permite determinar si las teorías e hipótesis básicas que se utilizaron al formular el programa resultaron válidas, qué surtió efecto o no, y por qué.

Además de todo lo anteriormente planteado, la evaluación se expresa como un proceso permanente y continuo de indagación, y valoración de la planificación, la ejecución y la finalización de los programas y proyectos sociales. Su finalidad es generar información, conocimientos y aprendizaje dirigidos a alimentar la toma de decisiones oportunas y pertinentes para garantizar la eficiencia, la eficacia y la calidad de los procesos; todo ello en función del mejoramiento de las condiciones de vida de sus poblaciones beneficiarias. Algo que es necesario aclarar es que no debe ser considerada como una acción de control o fiscalización, es un proceso que permite a los distintos actores involucrados aprender y adquirir experiencia.[14]

Si es el caso de que estas evaluaciones son acertadas y realizadas en el momento oportuno se evitarán entonces costos innecesarios y conduciendo así hacia direcciones más correctas en el desarrollo de las tareas.

Razones como las que se muestran a continuación demuestran la necesidad de medir un producto:

- Para indicar la calidad del producto.

- Para evaluar la productividad de la gente que desarrolla el producto.
- Para evaluar los beneficios en términos de productividad y de calidad, derivados del uso de nuevos métodos y herramientas de la ingeniería de software.
- Para establecer una línea de base para la estimación y para ayudar a justificar el uso de nuevas herramientas o de formación adicional.[6]

La evaluación de los RNF en la Línea Base de la Arquitectura constituye un factor decisivo para la obtención de un producto con alto grado de calidad, en la primera versión del producto que tiene implementado sólo los casos de uso arquitectónicamente significativos. Se verifica la satisfacción de estos siendo determinante en el establecimiento de la línea base, por ello la arquitectura escogida debe cumplir como primer objetivo satisfacer los requisitos funcionales llamados también funciones del sistema y los Requisitos No Funcionales o atributos del sistema. Es necesario aclarar que las clasificaciones de los requisitos son determinados por el propio cliente en el levantamiento de requisitos así como las restricciones de la frontera en aquellos Requisitos No Funcionales vitales para la arquitectura.

Una vez evidenciada la necesidad de evaluar los Requisitos No Funcionales y de que este proceso se haga en la Línea Base de la Arquitectura, el siguiente paso es estudiar las métricas, que sentarán las bases para proponer una solución factible.

1.5 Concepto Métrica

Las métricas son un buen medio para entender, monitorizar, controlar, predecir y probar el desarrollo software y los proyectos de mantenimiento.[15]

La ISO 14598-1:1999 define las Métricas como una forma de medir, la acción de medir (denominada medición), el resultado de la medición (denominada medida).

El concepto de métrica (definido como “una forma de medir + una escala”) y una escala, definidas para realizar mediciones de uno o varios atributos.[16]

En general se entiende por métrica una forma de medir y una escala, definidas para realizar mediciones de uno o varios atributos, donde la escala es un conjunto de valores con propiedades definidas; y atributo es una propiedad mensurable, física o abstracta de una entidad; donde la entidad va a ser un objeto.[17]

Métricas de Software

Las métricas de software se definen como “La aplicación continua de mediciones basadas en técnicas para el proceso de desarrollo del software y sus productos para suministrar información relevante a tiempo, así el administrador junto con el empleo de estás técnicas mejorará el proceso y sus productos”.[18]

También se puede decir que una métrica de software es una medida cuantitativa del grado en el que un sistema, un componente o un proceso poseen un determinado atributo.[19]

Las métricas de software se caracterizan por: unidades de medición, escalas de medición, ser simple y fácil de calcular, empírica e intuitivamente persuasivas, sin ambigüedades y objetiva, consistente en el empleo de unidades y tamaños, independiente del lenguaje de programación y eficaz para aumentar la calidad del software.[20]

Muchos investigadores han intentado desarrollar una sola métrica que proporcione una medida completa de la complejidad del software, debido a que las métricas de software proveen la información necesaria para la toma de decisiones técnicas. Pero les resulta difícil obtener un solo valor de estas métricas de calidad. Es por eso que existen distintos tipos de métricas para poder evaluar, mejorar y clasificar al software, donde serán manejadas dependiendo del entorno de desarrollo del software al cual pretendan orientarse.

1.5.1 Clasificación de Métricas

Las métricas se pueden clasificar atendiendo a los atributos que pueden ser de esfuerzo o tiempo, longitud de un programa, experiencia del equipo de trabajo, consumo de papel y cinta, estado del presupuesto, se clasifican en directas o indirectas, internas y externas, complejas o sencillas, publicas o privadas.[20]

Métricas Directas

Una métrica directa es una métrica de un atributo que no depende de ninguna métrica de otro atributo.[17]

Las métricas directas en el proceso de ingeniería se encuentran el costo, y el esfuerzo aplicado, las líneas de código producidas, velocidad de ejecución, el tamaño de memoria y los defectos observados en un determinado período de tiempo.[21]

Métricas Indirectas

Una métrica indirecta es una métrica de un atributo que se deriva de una o más métricas de otros atributos.[17]

Las métricas indirectas se encuentran la funcionalidad, calidad, complejidad, eficiencia, fiabilidad, facilidad de mantenimiento.[21]

Métricas de Productividad

Las métricas de productividad recogen la eficiencia del proceso de producción de software y relacionan el software que se ha construido con el esfuerzo que ha costado elaborarlo.[19]

Se centran en el rendimiento del proceso de la ingeniería del software. Es decir que tan productivo va a ser el software que se va a diseñar.[21]

Métricas de Calidad

Las métricas de calidad se asocian a determinados atributos medibles, no siempre evidentes, para reconocer la presencia o ausencia de calidad.[19]

Proporcionan una indicación de cómo se ajusta el software a los requisitos implícitos y explícitos del cliente. Es decir cómo se va a medir para que el sistema se adapte a los requisitos que desea el cliente.[22]

Son todas las métricas de software que definen de una u otra forma la calidad del software, tales como exactitud, estructuración o modularidad, mantenimiento, reusabilidad, cohesión del módulo, acoplamiento del módulo, etc.[18]

Métricas Técnicas

Las métricas técnicas se centran en las características de software; como por ejemplo: la complejidad lógica, el grado de modularidad, mide la estructura del sistema, el cómo está hecho.[21]

Se puede decir que las métricas técnicas son aquellas que se centran en las características del software y no en cómo se obtiene. Se puede medir o predecir en un software, es un atributo de cualquier entidad de un producto, proceso o recurso asociado a éste.

Métricas Orientadas a la Persona

Las métricas orientadas a la persona proporcionan medidas e información sobre la forma que la gente desarrolla el software de computadoras y sobre todo el punto de vista humano de la efectividad de las herramientas y métodos.[21]

Métricas Orientadas al Tamaño

Las métricas orientadas al tamaño se utilizan para conocer el tiempo de terminación del software y cuántas personas se necesitan para cumplir con la planificación. Son medidas directas al software y el proceso por el cual se desarrolla, si una organización de software mantiene registros sencillos, se puede crear una tabla de datos orientados al tamaño.[22]

Se relacionan con el tamaño de alguna salida de una actividad. La medida más común son las líneas de código fuente entregadas. También se utiliza el número de instrucciones de código objeto entregado o el número de páginas de la documentación del sistema.[23]

Métricas Orientadas a la Función

Las métricas orientadas a la función son medidas indirectas del software y del proceso por el cual se desarrolla. En lugar de calcular las líneas de códigos (LDC), las métricas orientadas a la función se centran en la funcionalidad o utilidad del programa.[21]

Métricas de Competencia

Las métricas de competencia son todas las métricas que intentan valorar o medir las actividades de productividad de los programadores o practicantes con respecto a su certeza, rapidez y eficiencia.[18]

Métricas de Desempeño

Las métricas de desempeño corresponden a las métricas que miden la conducta de módulos y sistemas de un software, bajo la supervisión del sistema operativo o hardware.[18]

Generalmente tienen que ver con la eficiencia de ejecución, tiempo, almacenamiento, complejidad de algoritmos computacionales, etc.

Las métricas también pueden ser de control o de predicción, ambas métricas influyen en la toma de decisiones administrativas.[23]

Métricas de Control

Las métricas de control por lo general se asocian con los procesos del software. Ejemplo, el esfuerzo y el tiempo promedio requerido para reparar los defectos reportados.[24]

Métricas de Predicción

Las métricas de predicción se asocian con los productos del software. Ejemplo, la complejidad ciclomática de un módulo, la longitud promedio de los indicadores en un programa y el número de atributos y operaciones asociadas con los objetos de un diseño.[24]

Métricas de Proceso

Las métricas del proceso son aquellas que intentan medir determinados atributos y que hacen referencia al entorno de desarrollo del software y tienen en cuenta la manera de construirlo.[19]

Métricas de Proyecto

Permiten evaluar el estado del proyecto en curso, realizar un seguimiento de los riesgos potenciales, detectar las áreas de problemas antes de que se conviertan en "críticas", ajustar el flujo y las tareas de trabajo, evaluar la habilidad del equipo del proyecto para controlar la calidad de los productos de trabajo de la ingeniería del software. Seguir la pista de los riesgos, ajustar tareas y planificaciones y determinar el coste del tiempo.[24]

Métricas de Producto

Las métricas del producto son las que miden diferentes aspectos del software obtenido y a menudo a partir de código fuente expresado en un lenguaje informático determinado, estos lenguajes pueden ser de programación, de diseño, de especificación, etc.[19]

Las métricas del producto se dividen en dos clases:

Métricas Dinámicas

Las métricas dinámicas son recolectadas por las mediciones hechas en un programa en ejecución, relación directa con los atributos de calidad del software y ayudan a valorar la eficiencia y la fiabilidad de un programa.[23]

Métricas Estáticas

Las métricas estáticas son recolectadas por las mediciones hechas en las representaciones del sistema como el diseño, el programa o la documentación y ayudan a valorar la complejidad, la comprensión y la mantenibilidad de un sistema de software. Tienen una relación indirecta con los atributos de calidad.[23]

Tabla 1 Ejemplos de métricas estáticas del producto.

Métricas de producto.	Descripción.
Fan–in / Fan–out	Fan-in: Medida del número de funciones que llaman a otra función X. Fan–out: Número de funciones que son llamadas por una función X. Un valor alto de fan –in indica que X está fuertemente acoplada al resto del diseño y que los cambios en X pueden tener efectos extensivos al resto del sistema. Un valor alto de fan-out indica que la complejidad de la lógica de control necesaria para coordinar los componentes llamados.
Longitud del código.	Medida del tamaño del programa en líneas de código (LDC). Cuanto mayor sea el tamaño de un componente, más complejo y susceptible de incorporar errores serán.
Complejidad ciclomática	Medida de la complejidad del control de un programa, y está relacionada con la comprensión del programa.
Longitud de los identificadores	Medida de la longitud promedio de los diferentes identificadores de un programa. Cuanto mayor sea esta longitud, más probable es que tengas significado, y por tanto el programa será más comprensible.
Profundidad de los if anidados	Medida de la profundidad de anidamiento de las instrucciones de un programa. Instrucciones if profundamente anidadas son difíciles de comprender y son potencialmente susceptibles a errores.
Índice de Fog.	Medida de la longitud promedio de las palabras y declaraciones en los documentos. Cuanto mayor sea el índice de Fog más difícil será comprender el documento.

Métricas de Complejidad

Las métricas de complejidad son todas las métricas de software que definen de una u otra forma la medición de la complejidad; tales como volumen, tamaño, anidaciones, costo (estimación) y flujo.[18]

Métricas Internas

Las métricas internas pueden aplicarse a un producto de software no ejecutable (como una especificación o código fuente). En el desarrollo de un producto de software los productos intermedios deben evaluarse usando las métricas internas que miden las propiedades intrínsecas, incluyendo las que pueden derivarse de un comportamiento simulado. El propósito primario de estas métricas internas es asegurar el logro de la calidad externa y la calidad durante el uso requerido. Estas constituyen una ventaja para los usuarios, evaluadores, verificadores, y diseñadores, pues le permiten evaluar la calidad de producto de software y atender los aspectos de la calidad desde las etapas más tempranas antes de que el producto de software devenga ejecutable.[25]

Métricas Externas

Las métricas externas usan valores de un producto del software derivadas de las mediciones del comportamiento del sistema del que es parte, mediante el ensayo, la operación y observación del software o sistema ejecutable. Antes de adquirir o usar un producto del software, el mismo debe evaluarse usando métricas basadas en objetivos comerciales relacionados con el uso, explotación y gestión del producto en un ambiente organizativo y técnico especificado. Estas son las métricas externas primarias y constituyen una ventaja para los usuarios, evaluadores, verificadores, y diseñadores pues le permiten evaluar la calidad del producto de software durante el ensayo o la operación.[25]

Existen diferentes categorías de métricas, en la investigación se hizo un estudio, en aras de utilizar las que satisfagan las necesidades de la propuesta. Se debe destacar que no existen clasificaciones exactas sobre las métricas, los autores o especialistas las clasifican o nombran en función de las necesidades específicas del producto. En el caso particular de la Universidad la situación no es diferente debido a que no se ha establecido tampoco una clasificación estándar, a pesar de conocer la importancia de la aplicación exacta de estas en la calidad del producto.

1.6 Métricas en la Ingeniería de Software

En la Ingeniería del Software, la idea de medir atributos del producto ayuda a hacerlos más comprensibles y controlables. La medición en el software puede ser hecha usando atributos internos, considerados como cualidades puras del producto, y mediante atributos externos, que son las cualidades del producto relacionándolo con el entorno. Como atributos internos se pueden considerar las líneas de código de un módulo, puntos de función, número de páginas de documentación, número medio de llamadas entre módulos, etc., y como atributos externos se pueden considerar la facilidad de

manejo del usuario, el número de errores encontrados en un período de tiempo, la velocidad de ejecución, etc. A menudo se miden los atributos internos para predecir los atributos externos, ya que algunas cualidades intrínsecas de un producto pueden dar indicios de su comportamiento futuro en un entorno determinado.

La finalidad de las métricas es mejorar las prestaciones del producto y optimizar los procesos de producción en todas sus fases. A continuación se muestran algunas métricas y modelos usados en la Ingeniería de Software a través de las clasificaciones realizada por Frentón:

- Estimación de coste y esfuerzo: que intentan predecir los costos del proyecto en la fase inicio del ciclo de vida a través del modelo de COCOMO o COCOMO II, el modelo SLIM de Putnam, o el modelo de puntos función de Albrecht. *Ver Anexo 1*

- Modelos y medidas de la productividad de la plantilla de desarrollo: como en los modelos COCOMO o los estudios de Jones.

- Modelos y medidas de calidad: miden la calidad del software desde diversos puntos de vista. Se pueden incluir los factores de calidad de McCall. *Ver Anexo 2*

- Modelos de fiabilidad: intentan predecir cuándo fallará un sistema; como el modelo de Jelinski-Moranda, el modelo de Musa o el modelo de Littlewood.

- Métricas de estructura y complejidad. Midiendo estos atributos internos tratan de predecir otros externos como la fiabilidad o la facilidad de mantenimiento. Dos ejemplos serían los modelos de McCabe o de Halstead.

- Valoración de capacidad-madurez del software, para evaluar la calidad de los requisitos de un proyecto. Generalmente basados en el modelo de madurez de la capacidad del SEI, como el SPICE.[26]

1.6.1 Métricas de Reutilización en la Ingeniería del Software

La reutilización del software no puede hacerse de forma aislada y acometerla en un determinado instante de tiempo; es necesario un marco de aplicación de Ingeniería del Software efectiva y una política de empresa que apueste por la reutilización. A continuación se muestran algunas de las métricas reutilizables clasificadas por Frakes:

- Modelos coste-beneficio de reutilización: incluyen el análisis coste/beneficio económico, y de calidad y productividad al reutilizar. Como el modelo de Gaffnet y Durek, la mejora de calidad de Barnes-Bollinger o las métricas de reutilización en negocios de Poulin y colegas.

- Valoración de la madurez: valoran los programas software según el grado en el que aplican la reutilización del software. Como los trabajos de Koltun-Hudson y el modelo de capacidad de Davis.

- Cantidad de reutilización: son métricas para valorar y controlar las mejoras obtenidas por la reutilización mediante porcentajes de reutilización en el tiempo en el ciclo de vida de los objetos. Se pueden mencionar el nivel de reutilización de Frakes y Terry, la fracción de reutilización de Agresti-Evanco y las métricas de orientación a objetos de Bieman-Karunanithi.

- Análisis de los modos de fallo: identifica y clasifica los impedimentos para reutilizar en una organización determinada. Como el método de Frakes y Fox.

- Evaluación de la reusabilidad: indica el grado en el que un artefacto puede ser reutilizable. Se incluyen la clasificación por facetas de Prieto-Díaz y Freeman, la medición para la reusabilidad de componentes Ada de Basili y colegas, la valoración de reusabilidad de REBOOT, y la predicción de reutilización para objetos en el ciclo de vida de Frakes y Fox.

- Métricas de librerías de reutilización: son usadas para controlar y seguir el uso de un repositorio de reutilización. Como las métricas de costos de indexación y efectividad de búsqueda de Frakes y colegas y las métricas de calidad de assets de Frakes y Nejmeh.[26]

Hasta el momento estos modelos o métodos no satisfacen a plenitud las necesidades de la propuesta debido a que estos no evalúan la parte esencial de la investigación: la calidad del producto, a través de los Requisitos No Funcionales. Se hace necesario conocer otras alternativas como los estándares encaminados a la evaluación del producto y seleccionar aquel que se adapte de manera más adecuada para la evaluación de los Requisitos No Funcionales en la fase que se desea.

1.7 Estándares de medición de software

Un estándar puede ser definido como una unidad de medida que sirve como modelo, guía o patrón con base en la cual se efectúa el control.[27]

Los estándares representan el estado de ejecución deseado, de hecho, no son más que los objetivos definidos de la organización.

La Organización de Estándares Internacionales ha introducido conceptos relacionados a la medición del software. [18]

Todo proceso de medición del software tiene como objetivo fundamental satisfacer necesidades de información a partir de las cuales se deben identificar las entidades y los atributos que deben ser medidos.

La medición de software es una disciplina relativamente joven, y no existe consenso general sobre la definición exacta de los conceptos y terminología que maneja.[16]

A continuación se muestra un estudio sobre algunos de los estándares que proponen métricas encaminadas a la evaluación de la calidad del producto de software.

1.7.1 Estándar ISO/IEC 14598

La ISO/IEC 14598 conocida además como Tecnología de la Información y Evaluación del producto software. Esta norma comprende las siguientes seis partes que especifican el proceso a seguir para evaluar software: ISO/IEC 14598-1 Visión general; ISO/IEC 14598-2 Planificación y Gestión; SO/IEC 14598-3 Procedimiento para desarrolladores; ISO/IEC 14598-4 Procedimiento para compradores; ISO/IEC 14598-5, Procedimiento para evaluadores; ISO/IEC 14598-6 Documentación de los módulos de evaluación.[28]

La serie de normas IRAM-ISO/IEC 14598 proporciona un marco de trabajo para evaluar la calidad de todos los tipos de productos de software e indica los requisitos para los métodos de medición y para el proceso de evaluación.

ISO/IEC 14598 se pretende usar por desarrolladores, compradores y evaluadores independientes, particularmente aquellos responsables de la evaluación del producto de software.

Los resultados de la evaluación que se obtienen a partir de la aplicación de ISO/IEC 14598 los pueden utilizar los administradores y desarrolladores/sostenedores para medir el cumplimiento de los requisitos y para realizar mejoras cuando sea necesario. Los resultados de la evaluación también los pueden utilizar los analistas para establecer las relaciones entre las métricas internas y externas. El personal de mejora de los procesos puede utilizar los resultados de la evaluación para definir cómo se pueden mejorar los procesos mediante el estudio y examen de la información de calidad del producto del proyecto.[29]

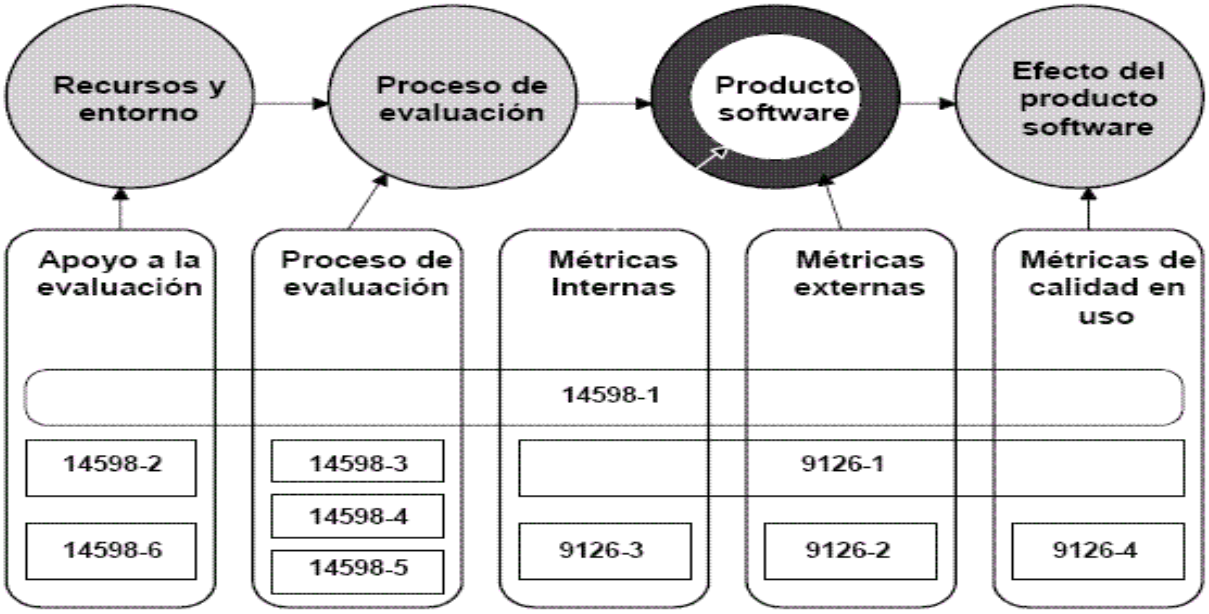


Figura 5 Evaluación del proceso de software: ISO 14598.

1.7.2 ISO / IEC 12119

Encaminada hacia la Tecnología de la Información, los Paquetes de Software, Requisitos de Calidad y Pruebas. Esta norma internacional es aplicable a los paquetes de software. Ejemplos de ello son procesadores de textos, hojas de difusión, programas de base de datos, paquetes gráficos, los programas de carácter científico o técnico funciones, programas y servicios públicos.

Esta estándar establece requisitos de paquetes de software (requisitos de calidad). Instrucciones sobre cómo probar un paquete de software en contra de estos requisitos (instrucciones para la realización de pruebas, en particular para las pruebas de tercero). No se refiere sólo a paquetes de software como ofrecido y entregado. No se refiere a su proceso de producción (incluidas las actividades y los productos intermedios, por ejemplo, pliego de condiciones).[30]

1.7.3 Estándar ISO 9000-3

Las ideas básicas que propone para el estándar ISO 9000-3 son las siguientes:

- El control de calidad debe ser aplicado a todas las fases de la producción de software, incluido el mantenimiento y tareas posteriores a su implantación.
- Debe existir una estricta colaboración entre la organización que adquiere el software y el proveedor del mismo.

- El proveedor del software debe definir su sistema de calidad y asegurarse que toda la organización ponga en práctica este sistema.
- Se compone de las siguientes cláusulas específicas.

La ISO 9000-3 proporciona una guía útil que sirve para detectar y corregir una serie de problemas de los productos software, consiguiendo tras su aplicación una mejora en la calidad de los mismos.[31]

Es importante resaltar que en la ISO 9000-3 trata el concepto de ciclo de vida, pero en ningún momento no desea imponer la utilización de un determinado ciclo como puede ser el ciclo en espiral de Boehm. Pero a parte del ciclo de vida que elijamos, el ISO 9000-3 introduce otras actividades que tienen lugar de forma independiente a las fases del ciclo y que son las actividades referentes a la configuración y distingue entre la verificación y validación.

Además el ISO 9000-3 puede ser utilizado en relaciones contractuales cuando comprador y proveedor establecen que algunos elementos de calidad deben formar parte del sistema de calidad que proporciona el proveedor y que este se compromete a seguir los principios de calidad definidos en el estándar como propone.[31]

1.7.4 Estándar ISO/ IEC 9126

La norma ISO/IEC 9126 permite que puedan ser utilizadas para especificar tanto los requisitos funcionales como no funcionales de los clientes y usuarios. Permite especificar la calidad del producto y evaluarlo desde perspectivas diferentes: Adquisición, Requisitos, Desarrollo, Uso, Mantenimiento y Aseguramiento de Calidad. Puede ser usada junto a:

- ISO/IEC 15504 (Evaluación del proceso software).
- ISO/IEC 12207 (Ciclo de Vida)
- ISO 9001 (Aseguramiento de Calidad)

Este estándar permite especificar y evaluar la calidad del producto de software desde las perspectivas de aquellos asociados con la adquisición, regulación, desarrollo, uso, evaluación, soporte, mantenimiento, aseguramiento de la calidad y auditoria del software.

- Parte1. Modelo de Calidad ISO/IEC 9126-1
- Parte2. Métricas Externas ISO/IEC 9126-2
- Parte3. Métricas Internas ISO/IEC 9126-3

- Parte4. Métricas de Calidad en el uso ISO/IEC 9126-4

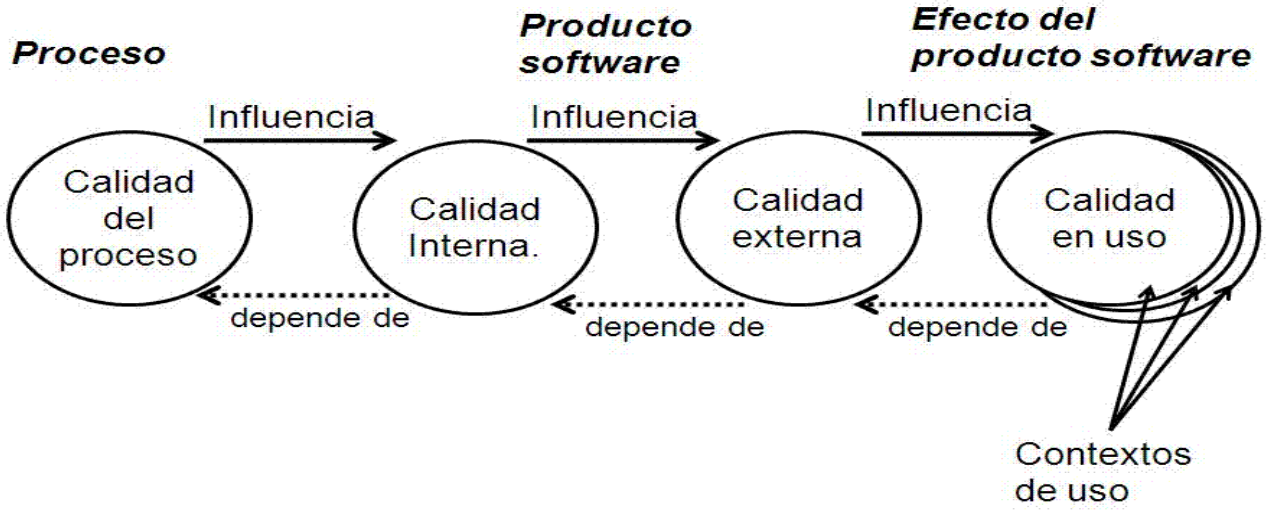


Figura 6 Procesos del estándar ISO/IEC 9126.

Este estándar está pensado para los desarrolladores, adquirentes, personal de aseguramiento de calidad y evaluadores independientes, responsables de especificar y evaluar la calidad del producto software. Por tanto, puede servir para validar la completitud de una definición de requisitos, identificar requisitos de calidad de software, objetivos de diseño y prueba, criterios de aseguramiento de la calidad, etc.

La calidad de cualquier proceso del ciclo de vida del software influye en la calidad del producto software que, a su vez, contribuye a mejorar la calidad en el uso del producto. La calidad del software puede evaluarse midiendo los atributos internos (medidas estáticas o productos intermedios) o atributos externos (comportamiento del código cuando se ejecuta).[25]

1.7.5 Estándar de Medición: IEEE 1061-1998

Trata de definir la calidad del software para un sistema mediante una lista de atributos de calidad del software requeridos por el propio sistema.

El propósito de las métricas del software es hacer evaluaciones a través del ciclo de vida del software para comprobar si los requisitos de calidad del software se están cumpliendo, aunque sin que ello elimine la necesidad de un juicio humano en las evaluaciones de software.

Tiene como objetivos:

- Facilitar a una organización:

- Lograr sus objetivos de calidad.
- Establecer requisitos de calidad para un sistema en su inicio.
- Establecer criterios de aceptación y estándares.
- Evaluar el nivel de calidad logrado frente a los requisitos establecidos.
- Detectar anomalías o problemas en el sistema.
- Predecir el nivel de calidad que se logrará en el futuro.
- Evaluar la facilidad de cambio en el sistema durante la evolución del producto.
- Normalizar, escalar, calibrar o validar una métrica.[32]

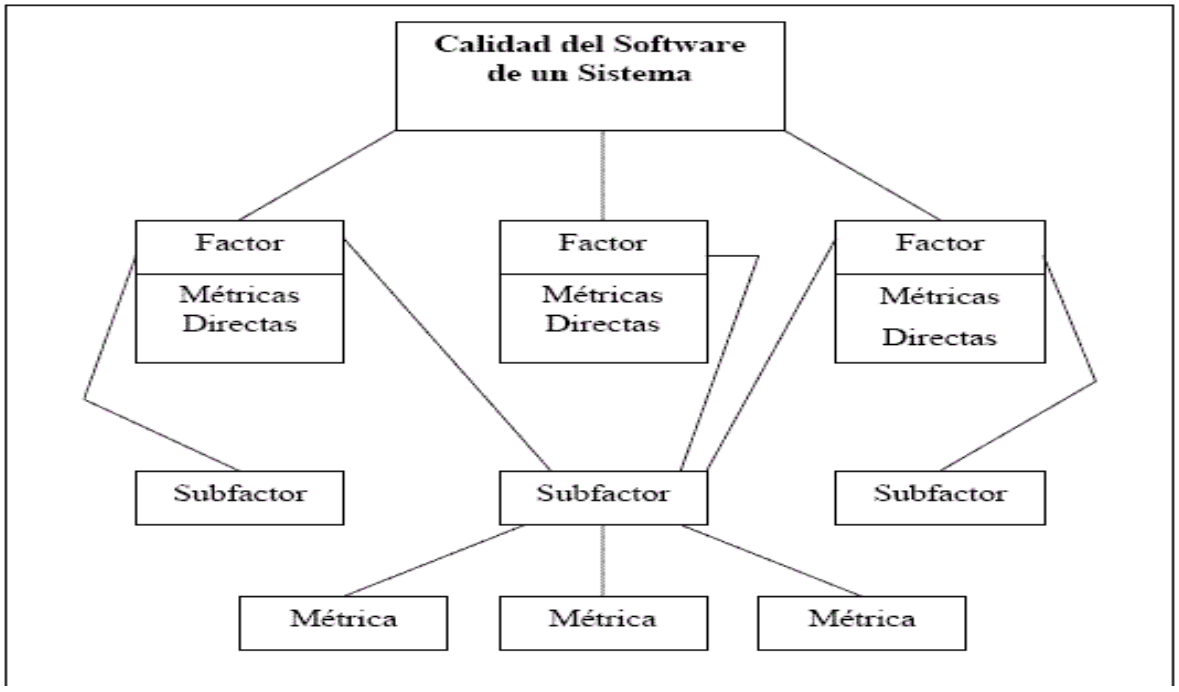


Figura 7 Marco de trabajo para métricas de calidad de software.

1.7.6 ISO 15939

Establece actividades y tareas necesarias para identificar, definir, seleccionar, aplicar y mejorar de manera exitosa la medición de software dentro de un proyecto general o de la estructura de medición de una empresa. Proporciona las definiciones de los términos de uso común relativos a la medición dentro de la industria del software.[32]

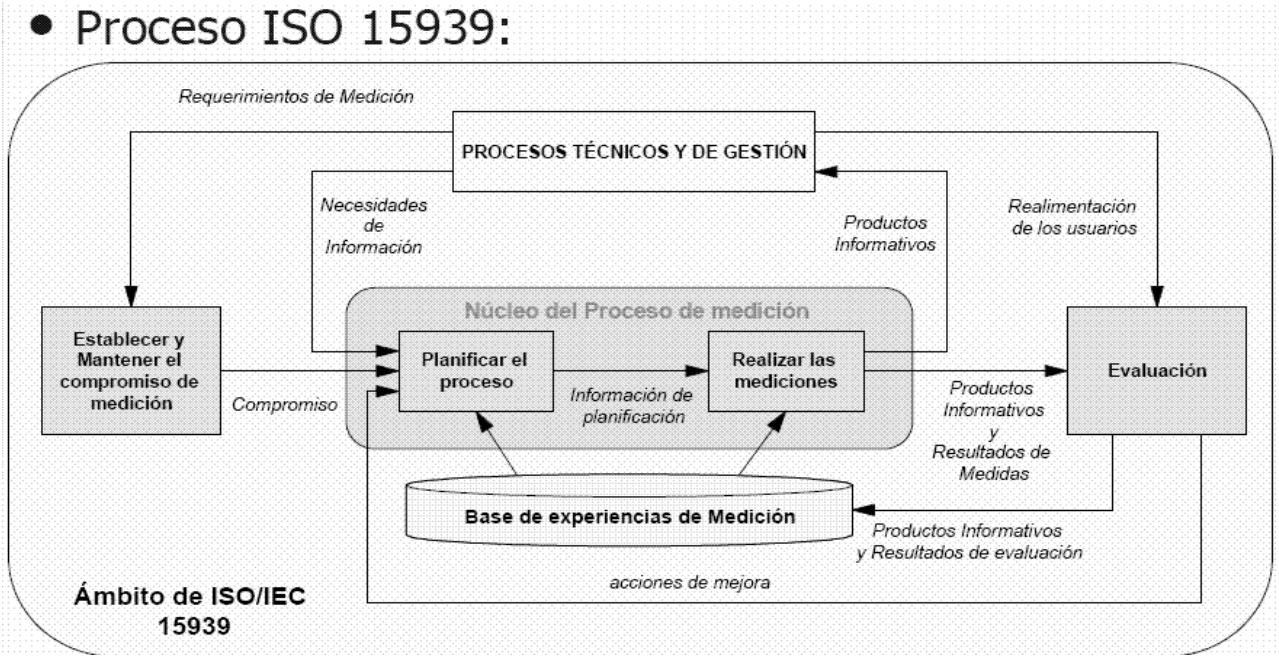


Figura 8 Ámbito de la ISO/IEC 15939.

Anteriormente se mostró un estudio de algunos de los estándares que evalúan la calidad del producto, a través del análisis de sus características se conoció cuál es él que satisface las necesidades de la propuesta que se plantea.

Se estableció entonces que cuando se va a evaluar la calidad del software, se puede considerar la norma ISO 9126 porque esta norma define las características de calidad como un conjunto de atributos del producto de software a través de los cuales la calidad es descripta y evaluada, dentro de sus características y subcaracterísticas coincide en muchas ocasiones con las mismos atributos necesarios para evaluar una arquitectura.

Por estas razones se ha escogido el estándar 9126 y además decir que es favorable para la propuesta debido a que es una normativa cubana, la mayoría de los proyectos productivos de la Universidad encaminan la evaluación de la calidad de sus productos hacia esta norma.

1.8 Conclusiones

A lo largo de este capítulo se han ofrecido los elementos teóricos que sirven de sustento científico a la investigación. El análisis de cada uno de estos aspectos estudiados permite arribar de manera parcial a las siguientes conclusiones:

Capítulo 1

El estudio sobre algunas de las diversas técnicas y herramientas que se utilizan para llevar a cabo cada una de las actividades del proceso de Ingeniería de Requisitos son una evidencia de la necesidad de la propuesta, pues en las diferentes actividades que tiene IR se introducen los primeros errores en los Requisitos No Funcionales. Debido a que no existe una guía para el uso de los desarrolladores, estos aplican las técnicas según la forma en que cada uno las entiende y según las condiciones en que se encuentra, utilizan incluso más de una técnica en cada de las actividades que contiene el proceso.

El uso de las métricas se extiende rápidamente entre la comunidad de desarrolladores de software debido a que las mismas producen indicadores a partir de los cuales se pueden tomar decisiones importantes. Existen diferentes tipos de métricas para poder evaluar, mejorar y clasificar al software desde sus inicios hasta el producto final, sus clasificaciones son dadas por muchos autores, no coincidiendo estos entre sí.

Estudiados los estándares se decidió tomar las métricas internas y externas que son las clasificaciones que ofrece el estándar seleccionado en este caso el 9126, además es necesario aclarar que al igual que las métricas, los Requisitos No Funcionales también poseen diferentes clasificaciones determinadas siempre por las necesidades del cliente.

Es evidente la necesidad de que exista una forma de evaluar a las características no funcionales primero porque es de vital importancia la evaluación temprana del producto, segundo porque propuestas anteriores no han profundizado lo suficiente es un tema tratado con demasiada superficialidad de los Requisitos No Funcionales.

Capítulo 2: Descripción de la Propuesta de Solución

Introducción

En este capítulo se muestra de forma detallada la propuesta de solución, las clasificaciones de los Requisitos No Funcionales hacia los que está encaminada la investigación, así como la repercusión de un requisito en los restantes. Se propone además la estructura para mostrar de forma estándar todas las métricas utilizadas, así como los momentos en que se realiza la evaluación, las métricas adaptadas a cada requisito y un análisis de los resultados obtenidos posteriormente.

2.1 Propuesta de solución

La propuesta ofrece inicialmente la selección de los Requisitos No Funcionales sobre los cuales va encaminada, luego momentos en que se realizarán las evaluaciones así como las métricas adaptadas a los requisitos, consideradas como las que mejor se ajustan a sus necesidades actuales así como un análisis de los resultados obtenidos luego de su aplicación para convertir de cuantitativos a cualitativos.

Requisitos No Funcionales seleccionados

Siendo tan complejos como pueden llegar a ser los proyectos de software no se puede atender al 100% cada una de estas características, de hecho el intento de maximizar algunas de ellas crea conflictos con el intento de lograr lo mismo con otras. Es la búsqueda del punto óptimo lo que hará la diferencia entre productos de Software de calidad. Para ubicar este punto es necesario definir los objetivos que se tienen para el producto final. El enfocarse en maximizar la calidad en ciertas características afecta a otras tanto positiva como negativamente. El hecho de enfocarse a ciertas características compromete la calidad en otros factores pero algunas de ellas se benefician también, por ejemplo enfocarse en hacer el software adaptable ayuda a hacerlo robusto y viceversa.

Teniendo en cuenta las clasificaciones hacia las cuales la dirección de producción de la Universidad orienta cada uno de los proyectos productivos, la propuesta estará dirigida hacia los siguientes Requisitos No Funcionales:

- Requisitos de Software
- Requisitos de Hardware
- Restricciones en el diseño y la implementación

- Requisitos de apariencia o interfaz externa
- Requisitos de Seguridad
- Requisitos de Usabilidad
- Requisito de Soporte

La propuesta va encaminada a algunas de las métricas que se encuentran plasmadas en la ISO/IEC 9126, en él se utilizan los modelo de la calidad interna y externa, además se utilizan otras que están dirigidas a la evaluación particular con la Ingeniería de Requisitos.

2.1.1 Declaración de las métricas

Para conseguir una alta probabilidad de entendimiento total de las métricas que se proponen, formar parte de la propuesta la elaboración de una común estructura para plantear todas estas métricas. Las métricas relacionadas con el desarrollo y evolución de los requisitos no se encontraban con la misma estructuración que ofrecen las métricas que muestra la ISO/IEC 9126 se hizo entonces necesario adaptarlas.

A continuación se muestran la estructura común que se planteó para todas las métricas:

a) Nombre de la métrica: Denominación de la métrica.

b) La métrica se propone medir: Se expresa como la interrogante a ser respondida por la aplicación de la métrica.

c) Método de aplicación: Provee una secuencia de pasos para la aplicación de la métrica.

d) Medición (fórmula): Provee la fórmula de medición y el significado de los datos empleados. En algunos casos más de una fórmula es brindada como métrica.

e) Interpretación del valor obtenido: Provee valores preferidos y su delimitación.

f) Escala: Tipo de escala usada por la métrica. Los tipos son: escala valorativa, escala relativa (ejemplo la longitud, el peso, el tiempo, el tamaño) y escala absoluta (ejemplo número de líneas de código).

2.1.2 Evaluación Inicial

La primera evaluación se realiza posteriormente de ser levantados los Requisitos No Funcionales donde mayormente se utilizan las métricas que ayudan el trabajo con Requisitos, estas ofrecen al

evaluador resultados que muestran el estado de la Ingeniería de Requisitos, verifican cuán satisfactorio a sido hasta ahora su procesamiento. Para la captura de requisitos se plantea una inmensa necesidad de métricas, pues una correcta gestión de requisitos contribuye en gran medida al éxito de los proyectos software, aportando el entendimiento y la comprensión de los problemas que se necesita solucionar y cómo resolverlos. En ellos se define, con claridad, sin ambigüedades, en forma consistente y compacta lo que se desea producir. La calidad y la cantidad de información suministrada durante la entrevista con el cliente es la clave del éxito.

2.1.2.1 Métricas para propuestas para requisitos

Entendimiento de los requisitos

En una especificación de requisitos es de vital importancia que los desarrolladores y miembros del equipo sean capaces de entender el significado de los requisitos, o sea, que no exista ambigüedad o lo que es lo mismo que cada requisito tenga una sola interpretación; por lo que el lenguaje usado en su definición no debe causar confusiones al lector.

Con la aplicación de esta métrica los proyectos de la Universidad podrán guardar registros del porcentaje de requisitos ambiguos con respecto al total en una determinada revisión técnica formal, y compararla con la anterior, de manera que se indique el avance o retroceso en el entendimiento de los requisitos y por lo tanto se tenga una idea de cómo ha sido el proceso de especificación y el entendimiento con el cliente.

a) Nombre de la métrica: Entendimiento de los requisitos.

b) La métrica se propone medir: ¿Pueden desarrolladores y miembros del equipo entender el significado de los requisitos?

c) Método de aplicación: Conducir las pruebas a los desarrolladores. Realizar entrevistas a miembros del equipo con cuestionarios preparados. Contar el número requisitos ambiguos con respecto al total en una determinada revisión técnica formal.

d) Medición (fórmula): ER= ((RT- RA)/ RT) *100

ER: Entendimiento de los requisitos.

RA: Requisitos ambiguos.

RT: Requisitos totales.

e) Interpretación del valor obtenido: 90 < ER <= 100

Mientras el valor de ER se acerque a 100, será mejor, pues significará que no ha habido ambigüedad en los requisitos.

f) Escala: Absoluta.

Estabilidad de los requisitos

Los requerimientos son la manera de comprender mejor el desarrollo de las necesidades de los usuarios y como los problemas a resolver, los objetivos de la organización, el entorno de trabajo entre otros factores pueden cambiar. Es esencial planear posibles cambios a los requerimientos cuando el sistema sea desarrollado y utilizado. Muchos son los cambios que pueden sufrir los requisitos a lo largo de todo el ciclo de vida del software incluyendo la eliminación, inserción y modificación.

Es de vital importancia establecer con qué frecuencia se analizará está métrica en cada proyecto. Teniendo en cuenta que en cada iteración del sistema pueden surgir nuevos requisitos, si se aplica esta métrica con una frecuencia que se delimite en el proyecto. Los requisitos insertados se referirán a los que surgieron después de haber quedado aprobados los requisitos iniciales para esa iteración si se está comparando con una iteración anterior entonces no se tendrán en cuenta los insertados.

Es muy importante lograr estabilidad, pues los continuos cambios en la especificación de los requisitos traen consigo un atraso en el cronograma de trabajo lo que puede ocasionar el incumplimiento del plazo de entrega pactado con el cliente. También puede afectar los costos y variar la cantidad de trabajo que ha de desarrollarse respecto a lo planificado. Se propone la métrica Estabilidad de los Requisitos para tener una medida de cuan estable son los requisitos de un sistema.

Con la aplicación de esta métrica los proyectos de la Universidad podrán tener una idea del nivel de madurez alcanzado durante el proceso de entrevista con el cliente y el levantamiento de los requisitos, pues mientras mejor haya sido este proceso menos cambios serán necesarios realizar en el transcurso del proyecto.

a) Nombre de la métrica: Estabilidad de los requisitos.

b) La métrica se propone medir: ¿Cuán estables son los requisitos?

c) Método de aplicación: Conducir las pruebas al sistema. Realizar entrevistas al personal capacitado con cuestionarios preparados al efecto. Contar el número de veces que han sido cambiados.

d) Medición (fórmula): ETR= ((RT- RC)/ RT) *100

ETR: Estabilidad de los requisitos.

RT: Requisitos totales.

RC: Requisitos cambiados (Sumatoria de los requisitos Insertados, Modificados y Eliminados).

e) Interpretación del valor obtenido: 90 < ETR <= 100

Mientras el valor de ETR se acerque a 100, será mejor, pues mostrará que no se están aplicando más cambios de requisitos y que la definición de estos es estable.

f) Escala: Absoluta.

Tiempo invertido en el proceso de análisis de requisitos

Para el análisis de requisitos es necesario disponer de cierto tiempo, la métrica que se propone a continuación tiene como objetivo analizar el tiempo invertido para cada requisito específico.

Es importante tener una idea de cuanto tiempo puede demorar una correcta especificación de un requisito pues así puede estimarse el tiempo de análisis de requisitos para futuros proyectos que tengan una complejidad similar. Esta estimación tal vez no sea muy precisa debido a que no se puede conocer exactamente el tiempo que se demorará la especificación de requisitos, pues nunca se sabe cual será el último incluido (esto depende en gran medida de los continuos cambios solicitados por el cliente) sin embargo puede establecerse la comparación entre proyectos que tengan objetivos similares a partir de la información inicial que se tenga de los mismos. Esta medida contribuye a la optimización del recurso tiempo.

a) Nombre de la métrica: Tiempo invertido en el proceso de análisis de requisitos.

b) La métrica se propone medir: ¿Cuánto tiempo se invierte para cada requisito específico?

c) Método de aplicación: Comenzar una tarea específica. Medir el tiempo que puede demorar una correcta especificación de un requisito. Guardar los registros de cada intento.

d) Medición (fórmula): TAR =TiT / RT

TAR: Tiempo de Análisis de un Requisito.

TiT: Tiempo total destinado al análisis de los requisitos.

RT: Requisitos totales.

e) Interpretación del valor obtenido: 0 <= TAR

Mientras más pequeño sea el valor, mejor porque es menor el tiempo que se dedica a cada requisito.

f) Escala: Relativa.

2.1.2.2 Métricas de Requisitos propuestas por Davis

Davis propone una lista de características que pueden emplearse para valorar la calidad del Modelo de Análisis y la correspondiente Especificación de Requisitos: especificidad (ausencia de ambigüedad), compleción, corrección, comprensión, capacidad de verificación, consistencia interna y externa, capacidad de logro, concisión, trazabilidad, capacidad de modificación, exactitud y capacidad de reutilización. Aunque muchas de las características anteriores parecen ser de naturaleza cualitativa, Davis sugiere que todas pueden representarse usando una o más métricas. Describe Pressman tres de estas métricas:

- Especificidad de los requisitos.
- Compleción de los requisitos funcionales.
- Grado de validación de los requisitos. [4]

En esta propuesta la compleción de requisitos no se utilizará debido a que está dirigido al trabajo con Requisitos No Funcionales

Especificidad de los requisitos

Basada en la consistencia de la interpretación de los revisores para cada requisito.

a) Nombre de la métrica: Grado de Validación de los Requisitos.

b) La métrica se propone medir: Medir el grado de validez que tienen los requisitos de un sistema.

c) Método de aplicación: Contar los requisitos que tuvieron igual interpretación y compararlos contra el total de requisitos.

d) Medición (fórmula): $Q_1 = n_{ui} / n_r$

n_{ui}: número de requisitos para los que todos los revisores tuvieron interpretaciones idénticas.

n_r: cantidad de requisitos en una especificación, es decir, la suma de los requisitos funcionales y los no funcionales.

e) Interpretación del valor obtenido: A medida que la razón se acerque a uno, mayor será la consistencia de la interpretación de los revisores para cada requerimiento. Esto es: menor será la ambigüedad en la especificación.

f) Escala: Absoluta.

Grado de Validación de los Requisitos

Mide el grado de validez que tienen los requisitos de un sistema.

a) Nombre de la métrica: Grado de Validación de los Requisitos.

b) La métrica se propone medir: Medir el grado de validez que tienen los requisitos de un sistema.

c) Método de aplicación: Contar el número de requisitos que no se validaron y reportar informe.

d) Medición (fórmula): $Q_3 = n_c / (n_c + n_{nv})$

n_c: número de requisitos que se han validado como correctos.

n_{nv}: número de requisitos que no se han validado todavía.

e) Interpretación del valor obtenido: Este valor debe acercarse a uno tanto como pueda.

f) Escala: Absoluta.[33]

Tabla 1 Métricas en la Evaluación Inicial.

Métricas	**Fórmula**	**Análisis de los Resultados**
Entendimiento de los requisitos.	ER= ((RT- RA)/ RT) *100	90 < ER <= 100. Mientras el valor de ER se acerque a 100, será mejor, pues significará que no ha habido ambigüedad en los requisitos.
Estabilidad de los requisitos.	ETR= ((RT- RC)/ RT) *100	90 < ETR <= 100. Mientras el valor de ETR se acerque a 100, será mejor, pues mostrará que no se están aplicando más cambios de requisitos y que la definición de

		estos es estable.
Tiempo invertido en el proceso de análisis de requisitos.	TAR =TiT / RT	0 <= Tiempo de Análisis de un Requisito.
Especificidad de los requisitos.	$Q_1 = n_{ui} / n_r$	A medida que la razón se acerque a uno, mayor será la consistencia de la interpretación de los revisores para cada requerimiento. Esto es: menor será la ambigüedad en la especificación.
Grado de Validación de los Requisitos.	$Q_3 = n_c / (n_c + n_{nv})$	Este valor debe acercarse a uno tanto como pueda.

2.1.3 Segunda evaluación

Segunda etapa de evaluación debe realizarse una vez definida la Línea Base de la Arquitectura. Se aconseja aplicar métricas que se encuentran en el modelo de la calidad interna y externa, y métricas desarrolladas para el trabajo específico con los requisitos, en el caso de esta propuesta han sido adaptadas a los Requisitos No Funcionales.

Se aplican también métricas que se encuentran en el modelo de la calidad interna y externa plasmado en la ISO 9126, el cual clasifica los atributos de calidad del software en seis características:

- Funcionalidad
- Confiabilidad
- Usabilidad
- Eficiencia
- Mantenibilidad
- Portabilidad

Estas son divididas en sub-características. Las sub-características pueden medirse a través de métricas internas o externas.[34]

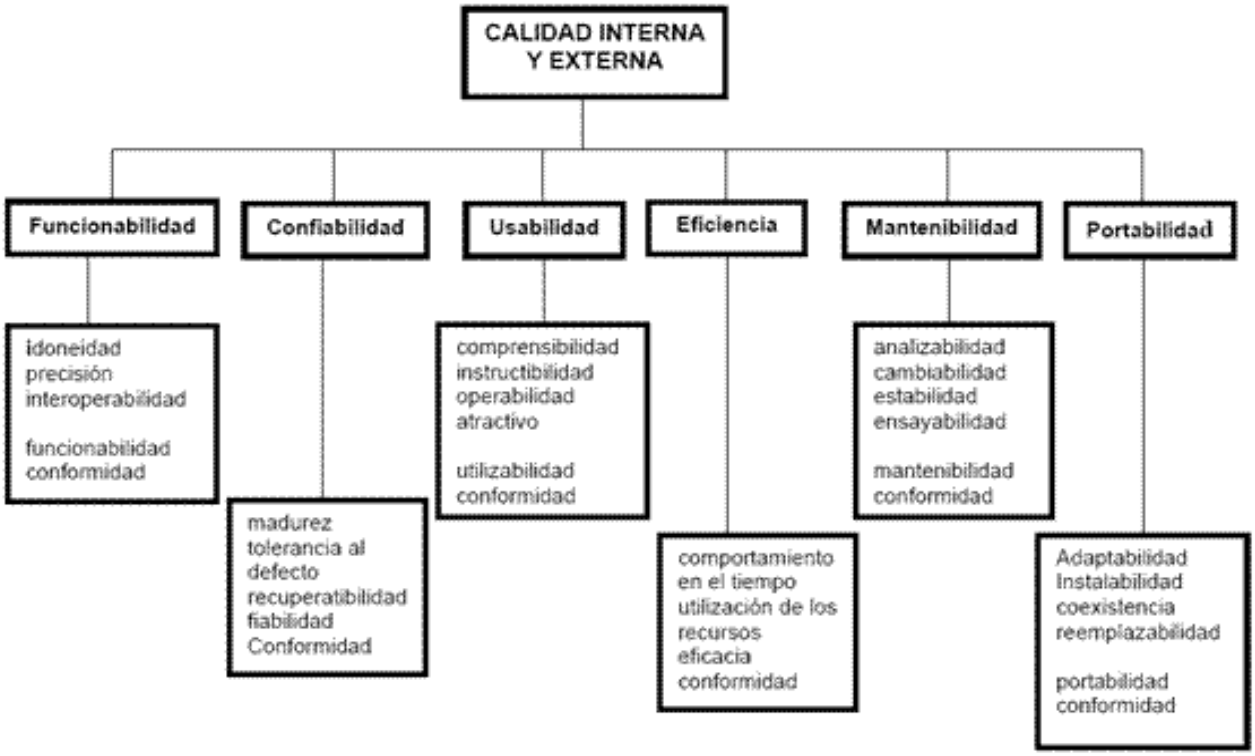


Figura 9 Características y subcaracterísticas del modelo de calidad externa.

Característica: Eficiencia

Capacidad del producto de software para proporcionar una ejecución o desempeño apropiado, en relación con la cantidad de recursos utilizados, bajo condiciones establecidas. Entre los recursos se pueden incluir otros productos del software, la configuración del software y el hardware del sistema y los materiales, (por ejemplo el papel de la impresión o disquetes).

Subcaracterística: Rendimiento

Capacidad del producto de software para proporcionar apropiados tiempos de respuesta y procesamiento, así como tasas de producción de resultados, al realizar su función bajo condiciones establecidas.

Métricas:

- Tiempo de respuesta.

- Tiempo de espera.

Tiempo de respuesta

a) Nombre de la métrica: Tiempo de respuesta.

b) La métrica se propone medir: ¿Cuánto tiempo toma completar una tarea en particular? ¿Cuánto toma antes de que el sistema responda a determinada operación?

c) Método de aplicación: Comenzar una tarea específica. Medir el tiempo que esta toma en completar una muestra hasta que concluya la operación emprendida. Guardar los registros de cada intento.

d) Medición (fórmula): T = Tiempo empleado en obtener el resultado.

e) Interpretación del valor obtenido: 0 < T

A mayor prontitud, (menor tiempo) resultará mejor.

f) Escala: Relativa.

Tiempo de espera

a) Nombre de la métrica: Tiempo de espera.

b) La métrica se propone medir: ¿Qué proporción del tiempo los usuarios dedican a esperar por una respuesta del sistema?

c) Método de aplicación: Ejecutar los casos de pruebas en situaciones características y tareas concurrentes. Medir el tiempo que toma completar las operaciones seleccionadas. Guardar un registro de cada intento y calcule el tiempo medio para cada caso o situación.

d) Medición (fórmula): X = Ta/ Tb

Ta - Tiempo total dedicado a esperar.

Tb - Tiempo consumido por la tarea.

e) Interpretación del valor obtenido: 0 <= X

Cuanto menor resultará mejor.

f) Escala: Absoluta.

TA = tiempo

TB = tiempo

Subcaracterística: Utilización de recursos

Capacidad del producto de software para utilizar la cantidad y el tipo apropiado de recursos cuando el software realiza su función bajo las condiciones establecidas.

Métricas:

- Tiempo de espera del usuario ante la utilización de los equipos de Entrada/Salida (E/S).
- Relación entre los errores de memoria y el tiempo.

Tiempo de espera del usuario ante la utilización de los equipos de Entrada /Salida (E/S)

a) Nombre de la métrica: Tiempo de espera del usuario ante la utilización de los equipos de Entrada/Salida.

b) La métrica se propone medir: ¿Cuál es el impacto de la utilización de los equipos de E/S en el tiempo de espera de los usuarios?

c) Método de aplicación: Ejecutar un amplio número de tareas concurrentes y medir el tiempo de espera del usuario ante la utilización de los equipos de E/S.

d) Medición (fórmula): T = Tiempo empleado en esperar por la culminación de una operación de un equipo de E/S.

e) Interpretación del valor obtenido: 0 < T

Más corto resultará mejor.

f) Escala: Relativa.

Relación entre los errores de memoria y el tiempo

a) Nombre de la métrica: Relación entre los errores de memoria y el tiempo.

b) La métrica se propone medir: ¿Cuántos errores de memoria fueron identificados durante el período de tiempo especificado, y con una utilización de recursos especificada?

c) Método de aplicación: Calibrar las condiciones de prueba. Emular una condición en que el sistema haya alcanzado un máximo de carga. Ejecutar la aplicación y registrar el número de errores debido a fallos de memoria.

d) Medición (fórmula): X = A/ T

A = número de mensajes de alerta o fallos del sistema.

T = tiempo de operación del usuario.

e) Interpretación del valor obtenido: 0 < X

Más pequeña, mejor.

f) Escala: Relativa.

A = contable

T = tiempo

Característica: Funcionalidad

Es la capacidad del software para proporcionar funciones que satisfacen las necesidades declaradas e implícitas cuándo el software se usa bajo las condiciones especificadas. Esta característica está relacionada con lo que hace el software para satisfacer las necesidades, al tiempo que las otras características principalmente están relacionadas con el cuando y cómo lo hace.

Subcaracterística: Precisión

Capacidad del software para proporcionar efectos o resultados correctos o convenidos con el grado de exactitud necesario.

Métricas:

- Exactitud esperada.

Exactitud esperada

a) Nombre de la métrica: Exactitud esperada.

b) La métrica se propone medir: ¿Existen diferencias entre los resultados actuales y los razonablemente esperados?

c) Método de aplicación: Deben ejecutarse los casos de pruebas de entrada versus salida y comparar los resultados actuales y los razonablemente esperados. Contar el número de casos encontrados con diferencias inaceptables en relación con los resultados razonablemente esperados.

d) Medición (fórmula): X = A/ T

A - Número de casos encontrados con diferencias entre los resultados razonable-mente esperados y aquellos resultantes más allá de lo permisible.

T - Tiempo de operación.

e) Interpretación del valor obtenido: 0 <= X

A mayor cercanía al 0 resultará mejor.

f) Escala: Valorativa.

A = contable

T = tiempo

Subcaracterística Interoperabilidad

Capacidad del producto de software para interactuar recíprocamente con uno o más sistemas especificados. La interoperabilidad se usa en lugar de la compatibilidad para evitar la posible ambigüedad con la reemplazabilidad.

Métricas:

- Intercambiabilidad de datos, en base su formato.
- Intercambiabilidad de datos, en base éxito del intento.

Intercambiabilidad de datos, en base su formato

a) Nombre de la métrica: Intercambiabilidad de datos, en base su formato.

b) La métrica se propone medir: ¿Cuán correctamente ha sido implementado el intercambio de funciones de interfaces para una transferencia de datos específica?

c) Método de aplicación: Ejecutar las pruebas a cada registro de salida de las funciones de interfaces de acuerdo con la especificación de los campos de datos. Contar el número de formatos de datos que deben ser intercambiados con otro software o sistemas durante las pruebas en comparación con el número total.

d) Medición (fórmula): X = A/ B

A- Número de formatos de datos intercambiados exitosamente con otro software o sistemas durante las pruebas del intercambio de datos.

B - Número total de formatos de datos a intercambiar.

e) Interpretación del valor obtenido: 0 <= X <= 1

Nota: A mayor cercanía al 1 resultará mayor intercambiabilidad.

f) Escala: Absoluta.

A = contable.

B = contable.

Intercambiabilidad de datos, en base éxito del intento

a) Nombre de la métrica: Intercambiabilidad de datos, en base éxito del intento.

b) La métrica se propone medir: ¿Cuán frecuentemente falló el intento de intercambio de datos entre el software objeto de la prueba y otro software? ¿Cuán frecuentemente es satisfactoria la transferencia de datos entre el software objeto de la prueba y otro?

c) Método de aplicación: Ejecutar las pruebas. Contar el número de casos en que las funciones de interfaces fueron usadas y fallaron.

d) Medición (fórmula):1) X = 1 - A/ B

A - Número de casos en que se falló al proceder a un intercambio de datos con otro software o sistemas.

B - Número de casos en que se intentó proceder a un intercambio de datos.

Y = A / T

T - Período de tiempo de operación.

e) Interpretación del valor obtenido: 0 <= X <= 1

Nota: A mayor cercanía al 1 resultará mejor.

0 <= Y

A mayor cercanía al 0 resultará mejor.

f) Escala: Absoluta, valorativa

Característica: Confiabilidad

La capacidad del producto de software para mantener un nivel de ejecución especificado cuando se usa bajo las condiciones especificadas El software no sufre desgaste ni envejecimiento. Las limitaciones en la fiabilidad son debidas a los fallos en los requisitos, el diseño y la implementación. Los fallos totales debidos a estos fallos dependen de la manera en que el producto del software se utilice y las opciones del programa seleccionado y no del tiempo de uso transcurrido.

Subcaracterística: Tolerancia ante fallos

Capacidad del producto de software de mantener un nivel de ejecución o desempeño especificado en caso de fallos del software o de infracción de su interfaz especificada.

Métricas:

- Protección de desastre.
- Protección de operaciones incorrectas.

Protección de operaciones incorrectas

a) Nombre de la métrica: Evitación de operaciones incorrectas.

b) La métrica se propone medir: ¿Cuántas funciones están implementadas con capacidad de evitación de operaciones incorrectas?

c) Método de aplicación: Contar el número de casos de prueba de operaciones incorrectas que fueron evitadas para que no causaran fallos totales críticos o serios, y compararlos con el número de casos de pruebas ejecutados a los patrones de operaciones incorrectas considerados período de pruebas bajo condiciones similares.

d) Medición (fórmula):X = A/ B

A1 - Número de ocurrencia de fallos totales críticos o serios evitada.

A2 - Número de casos de pruebas ejecutados a los patrones de operaciones incorrectas (casi causantes de fallos) durante las pruebas.

e) Interpretación del valor obtenido: 0 <= X <= 1

A mayor cercanía al 1 resultará mejor, cuantas más operaciones incorrectas del usuario sean evitadas.

f) Escala: Absoluta.

Protección de desastre

a) Nombre de la métrica: Protección de desastre.

b) La métrica se propone medir: ¿Cuán frecuentemente el producto de software causa un desastre en el ambiente de la producción total?

c) Método de aplicación: Contar el número de desastres detectados con respecto al número de fallos totales.

d) Medición (fórmula): X = 1- A/ B

A - Número de desastres

B - Número de fallos totales

e) Interpretación del valor obtenido: 0 <= X <= 1

A mayor cercanía al 1 resultará mejor.

f) Escala: Absoluta.

Subcaracterística: Madurez

Capacidad del producto de software de evitar un fallo total como resultado de haberse producido un fallo del software.

Métricas:

- Latencia estimada de la intensidad de fallos.
- Intensidad de fallos totales contra casos de prueba.
- Grado de solución ante fallos totales.
- Intensidad de fallos.
- Erradicación de fallos.
- Tiempo medio entre fallos totales.

Latencia estimada de la intensidad de fallos

a) Nombre de la métrica: Latencia estimada de la intensidad de fallos.

b) La métrica se propone medir: ¿Cuántos problemas aún existen que pueden emerger como futuros fallos?

c) Método de aplicación: Contar el número de fallos detectados durante el período definido de pruebas y pronosticar el número potencial de futuros fallos utilizando un modelo de estimación de la fiabilidad.

d) Medición (fórmula): X = (ABS (A1- A2))/ B

ABS () - Valor absoluto.

A1 - Número total de fallos latentes predecibles en el producto de software.

A2 - Número total de fallos detectados realmente.

B - Tamaño del producto.

e) Interpretación del valor obtenido: 0 <= X

En dependencia del estadio de las pruebas. En etapas más avanzadas, mientras más pequeño, mejor.

f) Escala: Absoluta.

Intensidad de fallos totales contra casos de prueba

a) Nombre de la métrica: Intensidad de fallos totales contra casos de prueba.

b) La métrica se propone medir: ¿Cuántos fallos totales fueron detectados durante un período de pruebas definido?

c) Método de aplicación: Contar el número de fallos totales detectados y el número de casos de pruebas.

d) Medición (fórmula):X = A1/ A2

A1 - Número total de fallos totales detectados.

A2 - Número de casos de pruebas ejecutados.

e) Interpretación del valor obtenido: 0 <= X

En dependencia del estadío de las pruebas. En etapas más avanzadas, mientras más pequeño, mejor.

f) Escala: Absoluta.

Grado de solución ante fallos totales

a) Nombre de la métrica: Grado de solución ante fallos totales.

b) La métrica se propone medir: ¿Cuántas condiciones de fallo total están resueltas?

c) Método de aplicación: Contar el número de fallos totales que no se repitieron en determinado período de pruebas bajo condiciones similares. Mantener un reporte de solución de problemas describiendo la situación de todos los fallos totales.

d) Medición (fórmula): X = A1/ A2

A1 - Número de fallos totales solucionados.

A2 - Número total de problemas reales detectados.

e) Interpretación del valor obtenido: 0 <= X <= 1

A mayor cercanía al 1 resultará mejor, cuantos más fallos totales estén resueltos.

f) Escala: Absoluta.

Intensidad de fallos

a) Nombre de la métrica: Intensidad de fallos.

b) La métrica se propone medir: ¿Cuántos fallos fueron detectados durante un período de pruebas definido?

c) Método de aplicación: Contar el número de fallos detectados y computar su intensidad.

d) Medición (fórmula): X = A/ B

A - Número total de fallos detectados.

B - Tamaño del producto.

e) Interpretación del valor obtenido: 0 <= X

Depende del estadío de las pruebas. En etapas más avanzadas, mientras más pequeño, mejor.

f) Escala: Absoluta.

Erradicación de fallos

a) Nombre de la métrica: Erradicación de fallos.

b) La métrica se propone medir: ¿Cuántos fallos han sido corregidos?

c) Método de aplicación: Contar el número de fallos resueltos durante el período de pruebas y compararlos con el número total de fallos detectados y el número total de fallos pronosticados.

d) Medición (fórmula):

X = A1/ A2

A1 - Número de fallos solucionados.

A2 - Número total de fallos reales detectados.

Y = A1/ A3

A3 - Número total de fallos latentes pronosticados.

e) Interpretación del valor obtenido: 0 <= X <= 1

A mayor cercanía al 1 resultará mejor (cuantos menos fallos queden).

0 <= Y

A mayor cercanía al 0 resultará mejor (cuantos menos fallos queden).

f) Escala: Absoluta, Absoluta.

Tiempo medio entre fallos totales

a) Nombre de la métrica: Tiempo medio entre fallos totales.

b) La métrica se propone medir: ¿Cuán frecuente-mente el software fracasa en su operación?

c) Método de aplicación: Contar el número de fallos totales ocurridos durante el período de operación definido y computar el intervalo promedio entre fallos totales.

d) Medición (fórmula):

X = T1/ A

Y = T2/ A

T1 – Tiempo de operación.

T2 – suma de los intervalos de tiempo entre los fallos totales consecutivos producidos.

A - Número total de fallos realmente detectados (fallos totales ocurridos durante el tiempo de operación observado).

e) Interpretación del valor obtenido: 0 < X, Y

A mayor resultará mejor, cuanto mayor sea el período de tiempo entre fallos.

0 <= Y

A mayor cercanía al 0 resultará mejor (cuantos menos fallos queden).

f) Escala: Valorada, Valorada.

Subcaracterística: Recuperabilidad

Capacidad del producto de software de restablecer un nivel de ejecución especificado y recuperar los datos directamente afectados en caso de fallo total.

Métricas:

- Grado de disponibilidad.

- Tiempo medio de inactividad.
- Tiempo medio de recuperación

Grado de disponibilidad

a) Nombre de la métrica: Grado de disponibilidad.

b) La métrica se propone medir: ¿Cuán disponible está el sistema para su uso durante?

c) Método de aplicación: Probar el sistema en la producción como ambiente durante un período de tiempo especificado ejecutando todas las operaciones del usuario.

d) Medición (fórmula): X = (To/ To+Tr)

Y = A1/ A2

To – Tiempo de operación.

Tr – Tiempo de reparación.

A1 – Total de casos de uso del software disponibles exitosamente, cuando se han intentado utilizar.

A2 - Total de casos de uso del software que se han intentado utilizar durante el tiempo de observación.

e) Interpretación del valor obtenido: 0 <= X <= 1

A mayor cercanía al 1 resultará mejor.

f) Escala: 1), 2) Absolutas.

Tiempo medio de inactividad

a) Nombre de la métrica: Tiempo medio de inactividad.

b) La métrica se propone medir: ¿Cuál es el tiempo promedio en que el sistema se mantiene no disponible cuando ocurre un fallo total y antes de la arrancada gradual?

c) Método de aplicación: Medir el tiempo de inactividad cada vez que el sistema se encuentre no disponible durante un período de prueba especificado y computar el tiempo medio.

d) Medición (fórmula): X = T/ N

T - tiempo total de inactividad.

N - Número de desastres observados.

El peor caso o la distribución del tiempo de inactividad deben ser medidos.

e) Interpretación del valor obtenido: 0 < X

Cuanto menor sea mejor, el sistema estará inactivo por menos tiempo.

f) Escala: Valoración.

Tiempo medio de recuperación

a) Nombre de la métrica: Tiempo medio de recuperación.

b) La métrica se propone medir: ¿Cuál es el tiempo promedio que toma el sistema para completar la recuperación desde el inicio de la recuperación parcial?

c) Método de aplicación: Medir el tiempo total de recuperación cada vez que haya pasado a la inactividad (caído el sistema) y calcular el tiempo promedio invertido en la recuperación.

d) Medición (fórmula): X =SUM (Tn)/ N

T - tiempo de recuperación de la inactividad en cada n oportunidad,

N - Número de oportunidades en que el sistema entró en recuperación.

e) Interpretación del valor obtenido: 0 < X

Cuanto menor sea mejor.

f) Escala: Valoración.

Característica: Usabilidad

Capacidad del producto de software de ser comprendido, aprendido, utilizado y de ser atractivo para el usuario, cuando se utilice bajo las condiciones especificadas. La usabilidad debe abordar todos los ambientes del usuario que el software puede afectar, lo cual puede incluir la preparación para el uso y la evaluación de resultados.

Subcaracterística: Comprensibilidad

Capacidad del producto de software para permitirle al usuario entender si el software es idóneo, y cómo puede usarse para las tareas y condiciones de uso particulares.

Métricas:

- Integridad de la descripción de producto.
- Accesibilidad a demos.

- Comprensibilidad de entradas y salidas.

Integridad de la descripción de producto

a) Nombre de la métrica: Integridad de la descripción de producto.

b) La métrica se propone medir: ¿Qué proporción de funciones (o tipos de funciones) son comprendidas después de leer la descripción del producto?

c) Método de aplicación: Conducir las pruebas de usuario. Realizar entrevistas a usuarios con cuestionarios preparados al efecto. Observar el comportamiento del usuario. Contar el número de funciones que fueron adecuadamente comprendidas en comparación con el número total de funciones en el producto.

d) Medición (fórmula): X = A/ B

A - número de funciones comprendidas.

B - número total de funciones.

e) Interpretación del valor obtenido: 0 <= X <= 1

A mayor cercanía al 1 resultará mejor

f) Escala: 1), 2) Absolutas.

Accesibilidad a demos

a) Nombre de la métrica: Accesibilidad a demos.

b) La métrica se propone medir: ¿A qué proporción de demos/tutoriales pueden acceder los usuarios?

c) Método de aplicación: Conducir las pruebas de usuario. Observar el comportamiento del usuario.

d) Medición (fórmula):X = A/ B

A - número de demos/tutoriales a los que pueden acceder los usuarios exitosamente.

B - número total de demos/tutoriales a los que se puede acceder.

e) Interpretación del valor obtenido: 0 < X <= 1

Mientras cercano al 1, mejor.

f) Escala: Absoluta.

Comprensibilidad de entradas y salidas

a) Nombre de la métrica: Comprensibilidad de entradas y salidas.

b) La métrica se propone medir: ¿Pueden los usuarios comprender lo que se requiere como entrada y lo que suministra el sistema de software como salida?

c) Método de aplicación: Conducir las pruebas de usuario. Realizar entrevistas a usuarios con cuestionarios preparados al efecto. Observar el comportamiento del usuario. Contar el número de elementos de entrada.

d) Medición (fórmula):X = A/ B

A - número de elementos de entrada y que suministra el sistema de software como salida comprendidas correctamente.

B - número total de elementos de entrada y que suministra el sistema de software como salida proporcionados por el interfaz.

e) Interpretación del valor: 0 <= X <= 1

Mientras cercano al 1, mejor.

f) Escala: Absoluta.

Subcaracterística: Atracción

Capacidad del producto del software de ser atractivo o amigable para el usuario. Esto se refiere a los atributos del software que se aplican para hacer el software más atractivo al usuario, tales como el uso del color y la naturaleza del diseño gráfico.

Métricas:

- Interacción por atracción.
- Adaptabilidad de la apariencia de la interfaz.

Interacción por atracción

a) Nombre de la métrica: Interacción por atracción.

b) La métrica se propone medir: ¿Cuán atractiva es la interfaz para el usuario?

c) Método de aplicación: Ejecutar las pruebas, por medio de las métricas planteadas en usabilidad.

d) Medición (fórmula): Resultados de las respuestas de la usabilidad con posterioridad al uso de la interfaz.

e) Interpretación del valor obtenido: Depende del método de cómputo del resultado de las preguntas del cuestionario empleado en la prueba.

f) Escala: Relativa.

Adaptabilidad de la apariencia de la interfaz

a) Nombre de la métrica: Adaptabilidad de la apariencia de la interfaz.

b) La métrica se propone medir: ¿Qué proporción de los elementos de la interfaz puede ser, por su apariencia, adaptado por el usuario para la satisfacción del mismo?

c) Método de aplicación: Conducir las pruebas de usabilidad. Observar el comportamiento del usuario.

d) Medición (fórmula):X = A/ B

A - número de elementos de la interfaz del sistema cuya apariencia puede ser adaptada por el usuario.

B - número de elementos de la interfaz del sistema cuya apariencia querría adaptar el usuario.

e) Interpretación del valor obtenido: 0 <= X <= 1

Mientras cercano al 1, mejor.

f) Escala: Absoluta.

Subcaracterística: Conformidad con la usabilidad

Capacidad del producto de software para adherirse a las normas, convenciones, guías de estilo o regulaciones relativas a la usabilidad.

Métrica:

- Conformidad con la usabilidad.

Conformidad con la usabilidad

a) Nombre de la métrica: Conformidad con la usabilidad.

b) La métrica se propone medir: ¿Cuánto se adhiere el producto de software a las regulaciones aplicables, las normas, convenciones, estilos y guías relacionadas con la usabilidad?

c) Método de aplicación: Especificar los elementos que requieren estar en conformidad, diseñar los casos de prueba de esos elementos, conducir las pruebas funcionales para esos elementos que requieren estar en conformidad.

d) Medición (fórmula): X = 1 - A/ B

A - número de elementos especificados que requiriendo estar en conformidad no han sido implementados durante las pruebas.

B - Número total de elementos que requieren estar en conformidad especificados.

e) Interpretación del valor obtenido: 0 <= X <= 1

A mayor cercanía al 1 mejor.

f) Escala: Absoluta.

Característica: Mantenibilidad

Capacidad del producto de software de ser modificado. Las modificaciones pueden incluir las correcciones, mejoras o adaptaciones del software a cambios en el ambiente, así como en los requisitos y las especificaciones funcionales.

Subcaracterística: Diagnosticabilidad

Capacidad del producto del software de ser objeto de un diagnóstico para detectar deficiencias o causas de los fallos totales en el software, o para identificar las partes que van a ser modificadas.

Métrica:

- Grado de implementación de las funciones de diagnóstico.

Grado de implementación de las funciones de diagnóstico

a) Nombre de la métrica: Grado de implementación de las funciones de diagnóstico.

b) La métrica se propone medir: ¿Cuán capaces son las funciones de diagnóstico implementadas para efectuar análisis de causas? ¿Puede el usuario identificar una operación que causa un fallo? (y evitarla, recurriendo a una operación alternativa) ¿Puede el serviciador fácilmente encontrar la causa del fallo?

c) Método de aplicación: Observar el comportamiento del operador que trata de resolver los fallos utilizando funciones de diagnóstico.

d) Medición (fórmula):X = A/ B

A - número de fallos que pueden ser diagnosticadas (utilizando las funciones de diagnóstico) para entender la correlación causa- efecto.

B - número total de fallos registrados.

e) Interpretación del valor obtenido: 0 <= X <= 1

Mientras más cercano al 1, mejor.

f) Escala: Absoluta.

Características: Portabilidad

Capacidad de producto de software de ser transferido de un ambiente a otro. El ambiente puede incluir el ambiente del software, del hardware u organizacional.

Subcaracterísticas: Instalabilidad

Capacidad del producto de software de ser instalado en un ambiente especificado. Si el software será instalado por un usuario final, la instalabilidad puede influir en la idoneidad y la operabilidad.

Métricas:

- Facilidad de instalación.
- Facilidad de reintento de ejecución del instalador.

Facilidad de instalación

a) Nombre de la métrica: Facilidad de instalación.

b) La métrica se propone medir: ¿Puede el usuario o serviciador instalar fácilmente el producto de software en su ambiente de operación?

c) Método de aplicación: Observar el comportamiento del usuario o serviciador cuando tratan de instalar el producto de software en su ambiente de operación.

d) Medición (fórmula): X = A/B

A = número de casos en los cuales el usuario tiene éxito en las operaciones para una instalación adecuada a su conveniencia.

B = número total de casos en los cuales el usuario intenta adecuar la instalación a su conveniencia.

e) Interpretación del valor obtenido: 0 <= X <= 1

Más cercano al 1 resultará mejor.

f) Escala: Absoluta.

Facilidad de reintento de ejecución del instalador

a) Nombre de la métrica: Facilidad de reintento de ejecución del instalador.

b) La métrica se propone medir: ¿Puede el usuario o serviciador fácilmente reintentar ejecutar el programa instalador del software?

c) Método de aplicación: Observar el comportamiento del usuario o serviciador cuando tratan de reinstalar el producto de software.

d) Medición (fórmula): X = A/ B

A = número de casos en los cuales el usuario no tiene éxito al reintentar ejecutar el instalador.

B = número total de casos en los cuales el usuario reintenta ejecutar la instalación durante la operación de instalación.

e) Interpretación del valor obtenido: 0 <= X <= 1

Más cercano al 1 resultará mejor.

f) Escala: Absoluta.

Subcaracterística: Conformidad con la portabilidad

Capacidad del producto de software de adherirse a las normas o convenciones relativas a la portabilidad.

Métricas:

- Conformidad de la portabilidad.

Conformidad de la portabilidad

a) Nombre de la métrica: Conformidad de la portabilidad.

b) La métrica se propone medir: ¿Cuán conforme es la portabilidad del producto de software con las regulaciones, las normas y convenciones que les son aplicables?

c) Método de aplicación: Contar el número de elementos que se encontraron en conformidad y compararlo con el número de elementos que requieren estar en conformidad según la especificación.

d) Medición (fórmula): X = 1 -A/ B

A - Número de elementos especificados que requiriendo estar en conformidad no han sido implementados durante las pruebas.

B - Número total de elementos especificados que requieren estar en conformidad.

e) Interpretación del valor obtenido: 0 <= X <= 1

A mayor cercanía al 1 mejor.

f) Escala: Absoluta.[34]

2.1.4 Resumen de Métricas por Requisitos

Requisito de Software			
Características	Subcaracterística	Métricas	Análisis de los Resultados
Eficiencia	Rendimiento	**Tiempo de respuesta** T = Tiempo empleado en obtener el resultado.	¿Cuánto tiempo toma completar una tarea en particular? ¿Cuánto toma antes de que el sistema responda a determinada operación?
		Tiempo de espera X = Tiempo total dedicado a esperar. /Tiempo consumido por la tarea.	¿Qué proporción del tiempo los usuarios dedican a esperar por una respuesta del sistema?

Requisito de Hardware

Características	Subcaracterística	Métricas	Mide
Eficiencia	Rendimiento	**Tiempo de respuesta** T = Tiempo empleado en obtener el resultado.	¿Cuánto tiempo toma completar una tarea en particular? ¿Cuánto toma antes de que el sistema responda a determinada operación?
		Tiempo de espera X = Tiempo total dedicado a esperar. /Tiempo consumido por la tarea.	¿Qué proporción del tiempo los usuarios dedican a esperar por una respuesta del sistema?
	Utilización de recursos	**Tiempo de espera del usuario ante la utilización de los equipos de E/S.** T = Tiempo empleado en esperar por la culminación de una operación de un equipo de E/S.	¿Qué proporción del tiempo los usuarios dedican a esperar por una respuesta del sistema?
		Relación entre los errores de memoria y el tiempo. X = Número de mensajes de alerta o fallos del sistema /tiempo de operación del usuario.	¿Cuántos errores de memoria fueron identificados durante el período de tiempo especificado, y con una utilización de recursos especificada?

Requisitos de apariencia o interfaz externa			
Características	Subcaracterística	Métricas	Mide
Usabilidad	Comprensibilidad	**Comprensibilidad de entradas y salidas** X = A / B A- número de elementos de entrada y que suministra el sistema de software como salida comprendidas correctamente. B- número total de elementos de entrada y que suministra el sistema de software como salida proporcionados por el interfaz.	¿Pueden los usuarios comprender lo que se requiere como entrada y lo que suministra el sistema de software como salida?
	Atracción	**Interacción por atracción** Resultados de las respuestas de la usabilidad con posterioridad al uso de la interfaz.	¿Cuán atractiva es la interfaz para el usuario?

		Adaptabilidad de la apariencia de la interfaz. ¿Qué proporción de los elementos de la interfaz puede ser, por su apariencia, adaptado por el usuario para la satisfacción del mismo?	¿Qué proporción de los elementos de la interfaz puede ser, por su apariencia, adaptado por el usuario para la satisfacción del mismo?

Restricciones en el diseño y la implementación			
Características	Subcaracterística	Métricas	Mide
Funcionalidad	Precisión	**Exactitud esperada** X = Número de casos encontrados con diferencias entre los resultados razonablemente esperados y aquellos resultantes más allá de lo permisible. /Tiempo de operación.	¿Existen diferencias entre los resultados actuales y los razonablemente esperados?
Confiabilidad	Tolerancia	**Protección de operaciones incorrectas.** X = Número de ocurrencia de fallos totales críticos o serios	¿Cuántas funciones están implementadas con capacidad de evitación de operaciones incorrectas?

		evitada/ Número de casos de pruebas ejecutados a los patrones de operaciones incorrectas (casi causantes de fallos) durante las pruebas.	
Usabilidad.	Comprensibilidad	**Comprensibilidad de entradas y salidas** X = Número de elementos de entrada y que suministra el sistema de software como salida comprendidas correctamente. / Número total de elementos de entrada y que suministra el sistema de software como salida proporcionados por el interfaz.	¿Pueden los usuarios comprender lo que se requiere como entrada y lo que suministra el sistema de software como salida?
Mantenibilidad	Diagnosticabilidad.	**Grado de implementación de las funciones de diagnóstico.** X = Número de fallos que pueden ser diagnosticadas (utilizando las funciones	¿Cuán capaces son las funciones de diagnóstico implementadas para efectuar análisis de causas? ¿Puede el usuario identificar una operación que causa un fallo? (y evitarla, recurriendo a una operación alternativa)

		de diagnóstico) para entender la correlación causa - efecto. / Número total de fallos registrados.	¿Puede el serviciador fácilmente encontrar la causa del fallo?

Requisitos de Seguridad			
Características	Subcaracterística	Métricas	Mide
Usabilidad	Interoperabilidad	**Intercambiabilidad de datos, en base su formato.** X = Número de formatos de datos intercambiados exitosamente con otro software o sistemas durante las pruebas del intercambio de datos. / Número total de formatos de datos a intercambiar.	¿Cuán correctamente ha sido implementado el intercambio de funciones de interfaces para una transferencia de datos específica?
		Intercambiabilidad de datos, en base éxito del intento. X = 1 - Número de casos en que se falló al proceder a un intercambio de datos con otro software o sistemas.	¿Cuán frecuentemente falló el intento de intercambio de datos entre el software objeto de la prueba y otro software? ¿Cuán frecuentemente es satisfactoria la transferencia de datos entre el software objeto de la prueba y otro?

		/ Número de casos en que se intentó proceder a un intercambio de datos. Y = Número de casos en que se falló al proceder a un intercambio de datos con otro software o sistemas / Período de tiempo de operación	
Confiabilidad	Madurez	**Latencia estimada de la intensidad de fallos.** X = (ABS (Número total de fallos latentes predecibles en el producto de software - Número total de fallos detectados realmente)) / Tamaño del producto. ABS () - Valor absoluto	¿Cuántos problemas aún existen que pueden emerger como futuros fallos?
		Intensidad de fallos totales contra casos de prueba. X = Número total de fallos totales detectados / Número de casos de pruebas ejecutados.	¿Cuántos fallos totales fueron detectados durante un período de pruebas definido?
		Grado de solución ante fallos totales.	¿Cuántas condiciones de fallo total están resueltas?

		X = Número de fallos totales solucionados/ Número total de problemas reales detectados.	
		Intensidad de fallos. X = Número total de fallos detectados/ Tamaño del producto.	¿Cuántos fallos fueron detectados durante un período de pruebas definido?
		Erradicación de fallos. X =Número de fallos solucionados/ Número total de fallos reales detectados	¿Cuántos fallos han sido corregidos?
Confiabilidad	Madurez	**Tiempo medio entre fallos totales.** X = T1 / A Y = T2 /A T1– T2– suma de los intervalos de tiempo entre los fallos totales consecutivos producidos. A- Número total de fallos realmente detectados (fallos totales ocurridos durante el tiempo de operación observado).	¿Cuán frecuente-mente el software fracasa en su operación?
	Tolerancia ante	**Protección de**	¿Cuán frecuentemente el

	fallos.	**desastre.** X = 1 - A /B A - Número de desastres. B – Número de fallos totales.	producto de software causa un desastre en el ambiente de la producción total?
		Protección de operaciones incorrectas. X = A/ B A1- Número de ocurrencia de fallos totales críticos o serios evitada. A2- Número de casos de pruebas ejecutados a los patrones de operaciones incorrectas (casi causantes de fallos) durante las pruebas.	¿Cuántas funciones están implementadas con capacidad de evitación de operaciones incorrectas?
	Recuperabilidad	**Grado de disponibilidad.** X = (To/ To+Tr) Y = A1/ A2 To–Tiempo de operación Tr–Tiempo de reparación A1–Total de casos de uso del software	¿Cuán disponible está el sistema para su uso durante?

		disponibles exitosamente, cuando se han intentado utilizar. A2 - Total de casos de uso del software que se han intentado utilizar durante el tiempo de observación.	
		Tiempo medio de inactividad. X = T/ N T - tiempo total de inactividad N- Número de desastres observados. El peor caso o la distribución del tiempo de inactividad deben ser medidos.	¿Cuál es el tiempo promedio en que el sistema se mantiene no disponible cuando ocurre un fallo total y antes de la arrancada gradual?
		Tiempo medio de recuperación. X =SUM (Tn) / N T-tiempo de recuperación de la inactividad en cada n oportunidad N-Número de oportunidades en que el sistema entró en	¿Cuál es el tiempo promedio que toma el sistema para completar la recuperación desde el inicio de la recuperación parcial?

		recuperación	

Requisitos de Usabilidad			
Características	Subcaracterística	Métricas	Mide
Usabilidad	Comprensibilidad	**Integridad de la descripción de producto** X = A/ B A - número de funciones comprendidas. B-número total de funciones.	¿Qué proporción de funciones (o tipos de funciones) son comprendidas después de leer la descripción del producto?
		Accesibilidad a demos. X = A/ B A-número de demos/tutoriales a los que pueden acceder los usuarios exitosamente. B-número total de demos/tutoriales a los que se puede acceder.	¿A qué proporción de demos /tutoriales pueden acceder los usuarios?
		Comprensibilidad de entradas y salidas. X = A/ B A- número de elementos de entrada y que	¿Pueden los usuarios comprender lo que se requiere como entrada y lo que suministra el sistema de software como salida?

		suministra el sistema de software como salida comprendidas correctamente. B-número total de elementos de entrada y que suministra el sistema de software como salida proporcionados por el interfaz.	
	Conformidad con la usabilidad	**Conformidad con la usabilidad** X = 1 - A/ B A- número de elementos especificados que requiriendo estar en conformidad no ha sido implementada durante las pruebas. B- Número total de elementos que requieren estar en conformidad especificados.	¿Cuánto se adhiere el producto de software a las regulaciones aplicables, las normas, convenciones, estilos y guías relacionadas con la usabilidad?

Requisitos de Soporte			
Características	Subcaracterística	Métricas	Mide
Portabilidad	Instalabilidad	**Facilidad de instalación.** X = A/ B A = número de casos en los cuales el usuario tiene éxito en las operaciones para una instalación adecuada a su conveniencia. B = número total de casos en los cuales el usuario intenta adecuar la instalación a su conveniencia.	¿Puede el usuario o serviciador instalar fácilmente el producto de software en su ambiente de operación?
		Facilidad de reintento de ejecución del instalador. X = A/ B A = número de casos en los cuales el usuario no tiene éxito al reintentar ejecutar el instalador. B = número total de casos en los cuales el usuario reintenta	¿Puede el usuario o serviciador fácilmente reintentar ejecutar el programa instalador del software?

		ejecutar la instalación durante la operación de instalación.	
	Conformidad de la portabilidad	**Conformidad de la portabilidad.** X = 1 - A/ B A- número de elementos especificados que requiriendo estar en conformidad no han sido implementados durante las pruebas. B- Número total de elementos especificados que requieren estar en conformidad.	¿Cuán conforme es la portabilidad del producto de software con las regulaciones, las normas y convenciones que les son aplicables?

2.2 Análisis de los Resultados

Terminada la aplicación de las métricas a cada clasificación de Requisito No Funcional surge entonces la importante necesidad de llevar estos datos cuantitativos a cualitativos. En este momento de la propuesta, el evaluador para establecer una conclusión lo más realista posible sobre el requisito en cada una de las métricas que se le aplicó debe tener en cuenta la restricciones de la frontera especificada por el usuario en el levantamiento de requisitos.

2.2.1 Análisis de los Resultados: Requisito de Software

Característica Eficiencia. Subcaracterística Rendimiento

En estas métricas debe tener en cuenta la restricción establecida por el usuario, su punto medio depende del valor establecido.

Métrica	Valor	Rango	Análisis de los Resultados
Tiempo de respuesta		0<X>1	• **Caso óptimo**: Si su valor de respuesta y su valor de espera es menor que el punto medio su comportamiento es adecuado. • **Peor caso**: Si su valor de respuesta y su valor de espera es mayor que el punto medio este valor no es el más satisfactorio. • **Caso medio**: Si su valor de respuesta es menor que el punto medio, pero su valor de espera es mayor que el punto medio entonces se debe trabajar en la mejora de este último tiempo su comportamiento no es el más adecuado. • **Caso medio**: Si su valor de espera es menor que el punto medio, pero su valor de respuesta es mayor que el punto medio entonces se debe trabajar en la mejora de este último tiempo su comportamiento no es el más adecuado.
Tiempo de espera		0<X>1	

2.2.2 Análisis de los Resultados: Requisito de Hardware

Eficiencia. Subcaracterística Rendimiento

En estas métricas debe tener en cuenta la restricción establecida por el usuario, su punto medio depende del valor establecido, para un excelente funcionamiento de un producto este no debe exceder este tiempo, pues el usuario pierde el interés sobre lo que se le ofrece.

Métrica	Valor	Rango	Análisis de los Resultados
Tiempo de respuesta		0<X>1	• **Caso óptimo**: Si su valor de respuesta y su valor de espera es menor que el punto medio su comportamiento es adecuado. • **Peor caso**: Si su valor de respuesta y su valor de espera es mayor que el punto medio existen gran problema en este requisito es decir debe ser mejorado.
Tiempo de espera		0<X>1	• **Caso medio**: Si su valor de respuesta es menor que el punto medio, pero su valor de espera es mayor que el punto medio entonces se debe trabajar en la mejora de este último tiempo su comportamiento no es el más adecuado. • **Caso medio**: Si su valor de espera es menor que el punto medio, pero su valor de respuesta es mayor que el punto medio entonces se debe trabajar en la mejora de este último tiempo su comportamiento no es el más adecuado.

Característica Eficiencia: Subcaracterística Utilización de recursos

Métrica	Valor	Rango	Análisis de los Resultados
Tiempo de espera del usuario ante la utilización de los equipos de E/S.		En este caso las restricciones de la frontera las establece el cliente en el levantamiento de requisitos.	• **Caso óptimo**: Mientras más pequeño sean los valores entre los errores de memoria y tiempo de espera del usuario ante la utilización de los equipos de E/S más efectiva es este requisito. • **Peor caso**: En caso que sean grandes los valores de ambas métricas, el requisito de hardware presenta problemas en esta subcaracterística, podía dañar mucho la efectividad de este requisito. • **Caso medio**: Si solo presenta altos valores en el
Relación entre los errores de		En este caso las restricciones de	

memoria y el tiempo.		la frontera las establece el cliente en el levantamiento de requisitos.	tiempo de espera debe revisar de forma cuidadosa este requisito pues existe algún problema, debería consultar a su cliente. • **Caso medio**: Si solo presenta altos valores entre los errores de memoria y el tiempo debe ejecutar las mismas acciones que en el caso anterior.

2.2.3 Análisis de los Resultados: Restricciones en el diseño y la implementación

Característica Funcionalidad: Subcaracterística Precisión

Métrica	Valor	Rango	Análisis de los Resultados
Exactitud esperada		En este caso las restricciones de la frontera las establece el cliente en el levantamiento de requisitos.	• **Caso óptimo**: El valor obtenido resultará mejor siempre que sea mayor o cercano al 0, en este caso el requisito es óptimo en esta subcaracterística. • **Peor caso**: En caso contrario al antes expuesto el desarrollador deberá revisar y mejorar esta subcaracterística que no es de mucho peso sobre el requisito pero también forma parte de él.

Característica Confiabilidad: Subcaracterística Tolerancia

Métrica	Valor	Rango	Análisis de los Resultados
Protección de operaciones incorrectas.		0 <= X <= 1	• **Caso óptimo**: En caso de que el valor este más cercano a 1 resultará mejor, cuantas más operaciones incorrectas del usuario sean evitadas más efectivo será el software, siempre tomando como valor intermedio el punto medio del valor establecido. • **Peor caso**: Pero en caso contrario el desarrollador debe introducir en el código más funciones que eviten funciones incorrectas, pues las existentes no son

			suficientes.

Característica Usabilidad. Subcaracterística Comprensibilidad

Métrica	Valor	Rango	Análisis de los Resultados
Comprensibilidad de entradas y salidas		0 <= X <= 1	• **Caso óptimo**: Mientras más cercano al 1, mejor para el sistema, siempre tomando como valor intermedio el punto medio. • **Peor caso**: En caso contrario el desarrollador debe trabajar más en comprensibilidad de entradas y salidas lo más aceptable para el comportamiento del usuario.

Característica Mantenibilidad. Subcaracterística Diagnosticabilidad

Métrica	Valor	Rango	Análisis de los Resultados
Grado de implementación de las funciones de diagnóstico.		0 <= X <= 1	• **Caso óptimo**: Mientras más cercano al 1, mejor para este requisito más exacto se encuentra, siempre tomando como valor intermedio el punto medio del valor establecido. • **Peor caso**: En caso contrario el desarrollador debe verificar esta subcaracterística y revisar si son capaces las funciones de diagnóstico implementadas y si son las más correctas.

2.2.4 Análisis de los Resultados: Requisito de apariencia o interfaz externa

Característica Usabilidad. Subcaracterística Comprensión

Métricas	Valor	Rango	Análisis de los Resultados
Intercambiabilidad de datos, en		0 <= X <= 1	• **Caso óptimo**: A mayor cercanía al 1 en ambas

base su formato.			métricas resultará mayor intercambiabilidad, es decir y más correcto el intercambio de datos, siempre tomando como valor intermedio el punto medio establecido por el usuario. • **Peor caso**: En caso contrario el intercambio de datos no seria el más eficaz existiendo perdida de información los requisitos de seguridad no serian los más eficaces.
Intercambiabilidad de datos, en base éxito del intento.			

2.2.5 Análisis de los Resultados: Requisito de Seguridad

Característica Funcionalidad: Subcaracterística Interoperabilidad

Métricas	Valor	Rango	Análisis de los Resultados
Intercambiabilidad de datos, en base su formato.		0<X>1	• **Caso óptimo**: Si el valor obtenido en la intercambiabilidad de datos, en base a su formato es cercano al 1, resultará un éxito pues ha sido implementado correctamente las funciones de interfaz para la transferencia; si el valor obtenido en la intercambiabilidad de datos, en base éxito del intento es cercano al 1 es que hubo pocos fallos al intentar el intercambio, en caso de ser cercano al 0 resultará satisfactoria la transferencia de datos. • **Peor caso**: En caso de que la intercambiabilidad de datos, en base su formato sea cercano al 0 es casi no hubo transferencia, debe acudir a los desarrolladores o evaluadores debido a que debe de haber ocurrido un error al ser implementado; si el valor obtenido en la intercambiabilidad de datos, en base éxito del intento es cercano al 0, es que ocurrió un fallo en el intento de la transferencia debe acudir a los serviciadores o evaluadores, otro caso es que
Intercambiabilidad de datos, en base éxito del intento.		0<X>1	

			el valor puede ser 1 o mayor se recomienda revisar el informe de pruebas debido a que hubo transferencia satisfactoria. • **Caso medio**: Si la intercambiabilidad de datos, en base su formato es cercano al 1 resultará satisfactoria la intercambiabilidad mientras que si la intercambiabilidad de datos, en base éxito del intento es cercano al 0 se debe de volver a ejecutar las pruebas pues falló el intento de transferencia, pero puede ocurrir que el valor obtenido cercano al 0 signifique que fue satisfactoria la transferencia. • **Caso medio**: En caso de que la intercambiabilidad de datos, en base éxito del intento sea cercano al 1 es que no falló ningún intento de transferencia y haber obtenido otro valor cercano al 0 de que fue satisfactoria de transferencia, mientras que si la intercambiabilidad de datos, en base su formato es cercano al 0 se debe trabajar más en esta, debido a que la implementación de las funciones de interfaces para la transferencia de datos específicos debe de tener algún error.

Característica Confiabilidad: Subcaracterística Madurez

El mayor peso sobre esta Subcaracterística lo tienen las métricas Grado de solución ante fallos totales y Erradicación de fallos, porque a través de ellas se conoce los fallos que han quedado y cuantos fallos totales han sido realmente resueltos. Juntas ofrecen la situación final con respecto a esta subcaracterística, en que medida es efectivo ante fallos. Por tanto es necesario guiarse siempre por los valores más óptimos de las métricas c, e.

Métricas	Valor	Rango	Análisis de los Resultados
a) Latencia		0 <= X	• **Caso óptimo**: En caso de que las métricas a, b, d

estimada de la intensidad de fallos.			tengan valores cercanos a 0 y las métricas c, e, f tengan valores cercanos a 1 esta subcaracterística tiene un excelente comportamiento en este requisito. • **Peor caso**: En caso de que las métricas a, b, d tengan valores lejanos a 0 y las métricas c, e, f tengan valores lejanos a 1 esta subcaracterística tiene comportamiento que debe ser analizado otra vez por el desarrollador, siempre teniendo en cuenta que el peso esta en las métricas c y e. • **Caso medio**: En caso de que las métricas c, e, f tengan valores cercanos a 1 no existirán muchos fallos y estarán solucionados y las métricas a, b, d también entonces esta subcaracterística no está funcionando completamente bien deben analizarse de nuevo los aspectos relacionados con las métricas a, b, d .lo mismo cuando ocurra en caso inverso.
b) Intensidad de fallos totales contra casos de prueba.		0 <= X	
c) Grado de solución ante fallos totales.		0 <= X <= 1	
d) Intensidad de fallos.		0 <= X	
e) Erradicación de fallos.		0 <= X <= 1; 0 <= Y	
f) Tiempo medio entre fallos totales.		0 < X, Y ; 0 <= Y	

Característica Confiabilidad: Subcaracterística Tolerancia ante fallos

Métricas	Valor	Rango	Análisis de los Resultados
Protección de desastre.		0 <= X <= 1	• **Caso óptimo**: El valor obtenido de la evitación de desastre sea cercano al 1 y la evitación de operaciones incorrectas sea también cercano al 1. • **Peor caso**: En caso de que el valor de la protección de desastre sea cercano al 0 y la protección de operaciones incorrectas sea también cercano al 0.
Protección de operaciones incorrectas.			

			• **Caso medio**: Si en la evitación de desastre se obtiene un valor cercano a 1 es bueno debido a que no habrá ningún problema con el producto, mientras que si la evitación de operaciones incorrectas tiene un valor cercano al 0. • **Caso medio**: Si en la evitación de operaciones incorrectas se obtiene un valor cercano a 1 es señal que las funciones con capacidad de evitación de operaciones incorrectas han sido corregidas con tiempo, mientras que si la evitación de desastre es cercano al 0.

Característica Confiabilidad: Subcaracterística Recuperabilidad

Métricas	**Valor**	**Rango**	**Análisis de los Resultados**
Grado de disponibilidad.		0<X>1	• **Caso medio**: En caso de que el grado de disponibilidad sea cercano al 1 será mejor el uso del sistema, cuanto menor sea el valor obtenido del tiempo medio de inactividad el sistema estará inactivo por menos tiempo, al igual que mientras menor sea el tiempo medio de recuperación es satisfactoria debido que tomará menos tiempo el sistema para completar recuperación. • **Caso medio**: En caso de que el grado de disponibilidad sea cercano a 0 más tiempo se pasa ejecutando las operaciones del usuario, si el valor del tiempo medio de inactividad es mayor que 1 mayor será el tiempo que estará no disponible al ocurrir un fallo, también si el tiempo medio de recuperación sea mayor que 1 mayor tiempo pasará en inactividad una ves caído el sistema.
Tiempo medio de inactividad.			
Tiempo medio de recuperación.			

			• **Peor caso**: Si el valor del grado de disponibilidad es 1 es bueno debido a que pasará menos tiempo ejecutando todas las operaciones del usuario, mientras que si el valor del tiempo medio de inactividad es un número mucho mayor que 1 y el valor tiempo medio de recuperación es igual mucho mayor que 1 mayor será el tiempo de inactividad cuando el sistema se encuentre no disponible y más tiempo le tomara al sistema recuperarse, por lo que debe tener en cuenta el informe de prueba, de operación. • **Caso óptimo**: Si el valor del tiempo medio de inactividad es cercano a 0 y el valor del tiempo medio de recuperación es igual cercano a 0 es que el sistema está en perfecto funcionamiento en cuanto al tiempo de respuesta, mientras que si el valor del grado de disponibilidad es cercano a 0 se tiene que trabajar más con este debido a que el sistema no estará disponible en un tiempo especificado.

2.2.6 Análisis de los Resultados: Requisito de Usabilidad

Característica Usabilidad: Subcaracterística Comprensibilidad

Métricas	Valor	Rango	Análisis de los Resultados
Integridad de la descripción de producto.		0<X>1	• **Caso óptimo**: En caso de que la integridad de la descripción de producto obtener un valor cercano al 1 resultaría bueno debido a que las funciones serán comprendidas después de leer la descripción del producto, si el valor de la accesibilidad a demos es cercano a 1 sería bueno debido a que los usuarios pueden acceder más a los demos/tutoriales, y si la
Accesibilidad a demos.		0<X>1	
Comprensibilida		0<X>1	

d de entradas y salidas.			comprensibilidad de entradas y salidas tiene un valor cercano al1 resultaría mejor la comprensibilidad del usuario. • **Peor caso**: En caso de la integridad de la descripción de producto obtenga un valor cercano al 0 sería una mala proporción de funciones comprendida por los usuarios por lo que debería ver el manual de usuario, en caso del valor cercano al 0 de la accesibilidad a demos es perjudicial debido a que casi ningún usuario puede acceder a los demos pues debería contactar al evaluador, y si el valor de comprensibilidad de entradas y salidas es cercano al 0 casi ningún usuario puede comprender lo que el sistema suministra como salida o requiere como entrada. • **Caso medio**: Si el valor obtenido en la integridad de la descripción de producto es cercano al 1 es que la descripción del producto ha sido claro y preciso, mientras que si la accesibilidad a demos y la comprensibilidad de entradas y salidas tienen un valor cercano al 0 es recomendable que se trabaje con cuidado en cuanto a las pruebas de usuarios debido al no poder acceder a los demos y al usuario no entender el formato en el cual los datos deben ser introducidos por lo que se debe contactar con el probador. • **Caso medio**: Si los valores obtenidos en el accesibilidad a demos y en la comprensibilidad de entradas y salidas son cercanos a 1 da un resultado bueno debido a que cumple la proporción de los usuarios que acceden a los tutoriales y los que

			comprenden la E/S del sistema; mientras que si el valor de la integridad de la descripción de producto es cercano al 0 es que se ha cometido error o no se ha sido claro en el manual de usuario, por lo que se debe contactar con los evaluadores, debido es que un artefacto importante.

Característica Usabilidad: Subcaracterística Conformidad con la usabilidad

Métricas	Valor	Rango	Análisis de los Resultados
Conformidad con la usabilidad.		0<X>1	• **Caso óptimo**: En caso del valor ser cercano a 1 en la conformidad con la usabilidad es que el producto tiene en cuenta las regulaciones aplicables, las normas, convenciones, estilos y guías relacionadas con la usabilidad por lo que no debe tener problemas con las misma. • **Peor caso**: En caso de que la conformidad con la usabilidad obtenga un valor cercano al 0 debe revisar la descripción del producto (manual del usuario o especificación) y las convenciones, normas y regulaciones relacionadas con el producto.

2.2.7 Análisis de los Resultados: Requisito de Soporte

Característica Portabilidad: Subcaracterística Instalabilidad

Métricas	Valor	Rango	Análisis de los Resultados
Facilidad de instalación.		0<X>1	• **Caso óptimo**: Si la facilidad de instalación obtiene un valor cercano al 1 al igual que la facilidad de reintento de ejecución del instalador un valor cercano al 1 es que el usuario no va tener problemas al instalar el producto de software en su
Facilidad de reintento de		0<X>1	

ejecución del instalador.			entorno de trabajo, al igual que fácilmente podrá reintentar ejecutar el programa instalador del software. • **Peor caso**: Si el valor obtenido en la facilidad de instalación y el valor de la facilidad de reintento de ejecución del instalador es en ambos cercano al 0, puede ocasionar problemas al instalar el producto de software en su ambiente de operación y al reinstalar el producto de software por lo que debe consultar el informe de problemas durante las pruebas. • **Caso medio**: En caso de la facilidad de instalación obtenga un valor cercano al 1 es bueno porque el usuario podrá instalar fácilmente el producto de software en su ambiente de trabajo, pero si la facilidad de reintento de ejecución del instalador se obtiene un valor cercano al 0 debe de trabajar más en este requisito pues tendrá problemas al reintentar ejecutar el programa por lo que debe consultar a los desarrolladores. • **Caso medio**: En el caso de que la facilidad de reintento de ejecución del instalador obtenga un valor cercano al 1 es un buen resultado debido a que hubo un buen trabajo en la reinstalación del producto, por lo que no tendrá problemas, mientras que si en la facilidad de instalación se obtiene un valor cercano al 0 se debe poner empeño en el arreglo de la instalación del producto software debido a que el mismo se debe ajustar a cualquier ambiente de trabajo, por lo que debe asistir a los desarrolladores del mismo.

Característica Portabilidad: Subcaracterística Conformidad de la portabilidad

Métricas	Valor	Rango	Análisis de los Resultados
Conformidad de la portabilidad.		0<X>1	• **Caso óptimo**: En el caso de que la conformidad de la portabilidad tenga un valor cercano al 1 es un buen resultado debido a que la portabilidad del producto de software con las regulaciones, las normas y convenciones que les son aplicables. • **Peor caso**: En caso de que la conformidad de la portabilidad obtenga un valor cercano al 0 es que se tiene que trabajar más en cuanto a los elementos que requieren estar en conformidad según la especificación, por lo que debe de consultar la descripción del producto y trabajar cuidadosamente con las regulaciones, las normas y convenciones que les son aplicables.

2.3 Resultados por Requisitos

Terminada la aplicación de las métricas en cada uno de los Requisitos No Funcionales según los resultados obtenidos se establecen 3 casos: Caso Óptimo, Caso Medio, Peor Caso en dependencia del por ciento que cada uno de los casos representen del total de métricas aplicadas a ese requisito, el mayor por ciento permitirá clasificar la subcaracterística en general:

Mayor Por ciento de métricas		Valor de la Subcaracterística
Caso Óptimo	----------	Alta
Caso Medio	----------	Media
Peor Caso	----------	Baja

En caso de que existan casos de igual porcentaje la clasificación de la subcaracterísticas dependerá de la importancia que tengan estas métricas para este requisito, escogiendo preferentemente el caso con peores resultados para así obtener la mayor cantidad de errores, logrando ser lo más exigente y una evaluación lo más exacta y realista posible.

La evaluación por subcaracterísticas permite ahorrar tiempo a los evaluadores, permite evaluar cada requisito por características y hacer énfasis en la toma de decisiones en aquellas áreas que fueron evaluadas como bajas.

Tabla 2 Tabla Ejemplo.

Nombre del requisito evaluado.				
Características	Subcaracterística	Valor	Métricas	Análisis de los Resultados
Característica que contiene algunas métricas aplicada a este requisito.	Subcaracterística que contiene algunas métricas aplicada a este requisito.	Puede tomar tres valores que sería: Alto, Medio, Bajo.	Nombre de las métricas aplicadas a este requisito.	Muestra los valores ofrecidos por las métricas, con una breve explicación de la situación exacta de la evaluación en este RNF.
Observaciones Alguna acotación que desee realizar el evaluador que no se contenga anteriormente.				

Tabla 3 Requisito de Software.

Requisito de Software				
Características	Subcaracterística	Valor	Métricas	Análisis de los Resultados
Eficiencia	Rendimiento		a) Tiempo de respuesta	
			b) Tiempo de espera	

Observaciones

Tabla 4 Requisito de Hardware.

Requisito de Hardware				
Características	Subcaracterística	Valor	Métricas	Análisis de los Resultados
Eficiencia	Rendimiento		a) Tiempo de respuesta	
			b) Tiempo de espera	
	Utilización de recursos		a) Tiempo de espera del usuario ante la utilización de los equipos de E/S.	
			b) Relación entre los errores de memoria y el tiempo.	
Observaciones				

Tabla 5 Requisito de apariencia o interfaz externa.

Requisito de apariencia o interfaz externa

Características	Subcaracterística	Valor	Métricas	Análisis de los Resultados
Usabilidad	Comprensibilidad		Comprensibilidad de entradas y salidas.	
	Atracción		Interacción por atracción.	
			Adaptabilidad de la apariencia de la interfaz.	
Observaciones				

Tabla 6 Requisito Diseño – Implementación.

Restricciones en el diseño y la implementación				
Características	Subcaracterística	Valor	Métricas	Análisis de los Resultados
Funcionalidad	Precisión		Exactitud esperada	
Confiabilidad	Tolerancia		Protección de operaciones incorrectas.	
Usabilidad.	Comprensibilidad		Comprensibilidad de entradas y	

			salidas	
Mantenibilidad	Diagnosticabilidad.		Grado de implementación de las funciones de diagnóstico.	
Observaciones				

Tabla 7 Requisito de Seguridad.

Requisito de Seguridad				
Características	Subcaracterística	Valor	Métricas	Análisis de los Resultados
Funcionalidad	Interoperabilidad		a) Intercambiabilidad de datos, en base su formato.	
			b) Intercambiabilidad de datos, en base éxito del intento.	
Confiabilidad	Madurez		a) Latencia estimada de la intensidad de fallos.	
			b) Intensidad de fallos totales contra casos de prueba.	

			c) Grado de solución ante fallos totales.	
			d) Intensidad de fallos.	
			e) Erradicación de fallos.	
			f) Tiempo medio entre fallos totales.	
	Tolerancia ante fallos.		Protección de desastre.	
			Protección de operaciones incorrectas.	
	Recuperabilidad		Grado de disponibilidad	
			Tiempo medio de inactividad.	
			Tiempo medio de recuperación.	
Observaciones				

Tabla 8 Requisito Usabilidad.

Requisito de Usabilidad

Características	Subcaracterística	Valor	Métricas	Análisis de los Resultados
Usabilidad	Comprensibilidad		Integridad de la descripción de producto	
			Accesibilidad a demos.	
			Comprensibilidad de entradas y salidas.	
	Conformidad con la usabilidad		Conformidad con la usabilidad	
Observaciones				

Tabla 9 Requisito Soporte.

Requisito de Soporte				
Características	Subcaracterística	Valor	Métricas	Análisis de los Resultados
Portabilidad	Instalabilidad		Facilidad de instalación.	
			Facilidad de reintento de ejecución del instalador.	
	Conformidad de la portabilidad		Conformidad de la portabilidad.	

Observaciones

2.4 Conclusiones

En este capítulo se hizo una selección de las métricas de la Ingeniería de Requisitos para un primer momento de evaluación de la propuesta, donde se evalúan los Requisitos No Funcionales luego del levantamiento de estos. Se seleccionó las métricas internas y externas de la norma ISO/IEC 9126 que satisfacen el desarrollo de una mejor calidad y satisfacción de los Requisitos No funcionales definidos, en la Línea Base de la Arquitectura. Se brindan Análisis de los Resultados para el usuario en caso de existir algún problema, y un epígrafe importante es los resultados por requisitos.

.

Capítulo 3 Validación de la Propuesta

Introducción

Como parte del ciclo de desarrollo del software, es importante verificar si los objetivos previstos se han cumplido en la práctica o si los ajustes que se introdujeron en el desarrollo efectivamente mejoran la calidad de los Requisitos No Funcionales.

Es conveniente recurrir a especialistas de la materia que sean preferiblemente ajenos al centro de elaboración de la propuesta en aras de ganar objetividad. Cada uno de los expertos consultados, velará por la eficacia y eficiencia de la propuesta, realizando una valoración de los aspectos.

Analizar los resultados de una propuesta propicia la realización de valoraciones acerca de la misma sobre la base de sus ventajas y limitaciones. Los resultados se consideran en virtud de su impacto y suele tomar como referencia opiniones tanto de personas conocedoras del fenómeno como de otras que han interactuado con este.

También determinará si las métricas seleccionadas responden a la necesidad real de los Requisitos No Funcionales, y se pronunciarán sobre la actualidad, pertinencia, exactitud y completitud del contenido, dentro del entorno de trabajo en el que se presente.

Según la bibliografía consultada se considera y está de acuerdo en que la revisión por expertos no implica, necesariamente, que la propuesta va a funcionar apropiadamente y producir los resultados esperados al ser usado por los distintos destinatarios. Tan solo se tiene una alta probabilidad de que así sea, pero habrá que probar la propuesta por usuarios reales.

La correcta elección de los expertos propicia obtener resultados con calidad y una opinión grupal con un alto grado de consenso. En este capítulo se describirá la forma para aplicar este método y los elementos necesarios para el mismo, posteriormente se presentarán los resultados obtenidos de la evaluación.

3.1 Guía para la evaluación técnica de la propuesta

Para validar la propuesta se utilizó el método de experto que permite tomar decisiones para aceptar o no la propuesta de acuerdo a los criterios definidos.

A continuación se muestra la guía utilizada para la evaluación técnica de la propuesta de solución, para la misma se efectuaron un conjunto de pasos.

Se elaboran los criterios de evaluación de acuerdo a las características de la propuesta y se organizan por grupos en las siguientes categorías:

Grupo No 1: Criterios de mérito científico

C1. Valor científico de la propuesta.

C2. Calidad de la investigación.

C3. Contribución científica.

Grupo No 2: Criterios de implantación

C4. Necesidad de utilización de la propuesta.

C5. Posibilidades de aplicación.

C6. Obtención de productos finales con calidad.

Grupo No 3: Criterios de flexibilidad

C7. Adaptabilidad a los proyectos productivos que se encuentren en las fases tempranas.

C8. Capacidad de admitir cambios en el uso de métricas necesarias para la correcta evaluación.

Grupo No 4: Criterios de impacto

C9. Repercusión en los proyectos productivos en la Ingeniería de Requisitos.

C10. Organización en el proceso de obtención y especificación de los Requisitos No Funcionales.

3.1.1 Peso relativo de cada grupo

Se determina el peso relativo de cada grupo asignándole el por ciento que representa cada uno de los criterios del total, teniendo en cuenta el número de criterios y los intereses a evaluar.

Grupo No.1…… 30

Grupo No.2…… 30

Grupo No.3…… 20

Grupo No.4…….20

3.1.2 Selección de los expertos

Primeramente se puede señalar que, se entiende por experto tanto al individuo en sí como a un grupo de personas u organizaciones capaces de ofrecer valoraciones conclusivas de un problema en cuestión (el procedimiento propuesto) y hacer, además, las recomendaciones que considere válidas para su enriquecimiento.

En el caso de esta investigación fueron recomendados inicialmente expertos que trabajan directamente con el tema de requisitos, ingenieros que constantemente se encuentran relacionados con la Ingeniería de Requisitos, y la calidad.

Para la selección de los expertos se tuvieron en cuenta las siguientes características:

- Competencia.
- Creatividad.
- Disposición a participar en la encuesta.
- Capacidad de análisis y de pensamiento.
- Espíritu colectivista.
- Espíritu autocrítico.

Dentro de las características de selección la competencia, que define el nivel de calificación del experto en una determinada esfera del conocimiento, resulta imprescindible; para su determinación se empleó la metodología descrita en Landeta.

Según Landeta el procedimiento para la selección de los expertos se enmarca en cuatro etapas fundamentales:

- Determinación de la cantidad de expertos.
- Confección del listado de posibles expertos y determinación de su consentimiento para su participación en el peritaje.
- Estudio de las características de los posibles expertos y encuesta para determinar su coeficiente de competencia.
- Determinación final de los expertos que serán utilizados para valorar la propuesta.[35]

Un paso muy importante que no debe ser obviado es la confirmación del listado, se invitó personalmente a cada experto elegido para participar en la evaluación. Allí se les explicó en qué consistía el trabajo en general, la propuesta a evaluar así como el plazo de entrega. Una vez recibida la respuesta positiva, se estableció el listado final de los expertos, informando a cada especialista su inclusión en el proceso a evaluar y las instrucciones necesarias para contestar las preguntas. De esta forma culmina el proceso de selección, logrando la participación de los siete expertos escogidos.

3.1.3 Entrega de Encuestas a los expertos

Para conformar el listado se invitó formalmente a cada experto elegido a participar en la validación de la propuesta. En ella se le explicó el trabajo en general y la propuesta a evaluar. Una vez recibida la respuesta positiva, se estableció el listado final de los expertos y se le envió un artículo donde se explicaba detalladamente la propuesta y las instrucciones necesarias para contestar las preguntas. De esta forma culmina el proceso de selección, logrando la participación de los siete expertos escogidos.

Tabla No 1 Resultado del trabajo de expertos.

G	C/E	E1	E2	E3	E4	E5	E6	E7	EP
30	C1	8	7	7	8	7	10	7	7.71
	C2	15	12	13	11	13	11	15	12.86
	C3	7	11	10	11	10	9	8	9.43
30	C4	10	14	13	10	9	10	13	11.29
	C5	10	6	8	10	11	10	10	9.28
	C6	10	10	9	10	10	10	7	9.43
20	C7	10	10	15	9	9	10	9	10.29
	C8	10	10	5	11	11	10	11	9.71
20	C9	10	11	10	10	10	11	11	10.43
	C10	10	9	10	10	10	9	9	9.57

T		100	100	100	100	100	100	100	100

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). Se sigue el procedimiento siguiente:

- 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:

∑E: Sumatoria del peso dado por cada experto

Ep: Puntuación promedio del peso dado por cada experto

M∑E: media de los ∑E

ΔC: 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

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

	∑E	**∑E/C**	**∑E - ∑ ∑E / C**	**$(\sum E - \sum \sum E / C)^2$**
C1	54	5.4	16	256
C2	90	9	20	400
C3	66	6.6	-4	16
C4	79	7.9	9	81
C5	65	6.5	-5	25
C6	66	6.6	-4	16
C7	72	7.2	2	4
C8	68	6.8	-2	4

C9	73	7.3	3	9
C10	67	6.7	3	9
$\sum \sum E / C$		70		
$S = \sum (\sum E - \sum \sum E / C)^2$				820

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

$W = S / E^2 (C^3 – C)/ 12$

$W = 820 / (7)^2 (10^3 – 10) /12$

W = 820 / 49 (1000 - 10) /12

W = 820/ 49 *(990 /12)

W = 820 / 48510 /12

W = 820 / 4042.5

W = 0.20

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

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

$X^2 = 7(10 -1) * 0.20$

$X^2 = 7 *9 *0.20$

$X^2 = 12.6$

- Los valores obtenidos se muestran en la Tabla No.2.

Tabla No.2 Tabla para el cálculo de concordancia de Kendall.

Expertos/Criterios	**E_1**	**E_2**	**E_3**	**E_4**	**E_5**	**E_6**	**E_7**	**$\sum E$**	**Ep**	**ΔC**	**ΔC^2**
C_1	8	7	7	8	7	10	7	54	7.71	15	225

C_2	15	12	13	11	13	11	15	90	12.86	21	441
C_3	7	11	10	11	10	9	8	66	9.43	3	9
C_4	10	14	13	10	9	10	13	79	11.29	10	100
C_5	10	6	8	10	11	10	10	65	9.28	4	16
C_6	10	10	9	10	10	10	7	66	9.43	3	9
C_7	10	10	15	9	9	10	9	72	10.29	3	9
C_8	10	10	5	11	11	10	11	68	9.71	1	1
C_9	10	11	10	10	10	11	11	73	10.43	4	16
C_{10}	10	9	10	10	10	9	9	67	9.57	2	4
DC	100	100	100	100	100	100	100	700	100	76	830
M ∑E	69										
W	0.20										
X^2	12.6										

- El Chi cuadrado calculado se compara con el obtenido del las tablas estadísticas.
- Si se cumple:

$$X^2_{real} < X^2_{(\alpha,\ c-1)}$$

12.6 < 21.6660

Existe concordancia en el trabajo de expertos.

- Después de comprobar la consistencia del trabajo de expertos se puede definir el peso relativo de cada criterio (P).

- Conociendo el peso de cada criterio y la calificación dada por los evaluadores en una escala de 1-5 se puede construir la Tabla No.3, para obtener el valor de P × c, donde ©, es el criterio promedio concebido por los expertos.

Para recoger la calificación dada por los expertos a cada uno de los criterios se definió un modelo el cual se expone en el anexo 4 de la investigación.

Tabla No.3 Tabla de calificación de cada criterio

Criterios	**Calificación ©**					**P**	**P × c**
	1	**2**	**3**	**4**	**5**		
C_1				x		0.0771	0.3084
C_2					x	0.1286	0.643
C_3				x		0.0943	0.3772
C_4					x	0.1129	0.5645
C_5					x	0.0928	0.464
C_6				x		0.0943	0.3772
C_7				x		0.1029	0.4116
C_8				x		0.0971	0.3884
C_9					x	0.1043	0.5215
C_{10}					x	0.0957	0.4785
Total							4.5281

Se calcula el Índice de aceptación del proyecto (IA).

$IA = \Sigma (P \times c) / 5$

$IA = 4.5281/5$

IA= 0.90562

Por último se determina la probabilidad de éxito de la propuesta, para esto se ubica el Índice de Aceptación (IA) calculado anteriormente en rangos que están ya predefinidos, en dependencia de donde se ubique, será la probabilidad de éxito que tenga la propuesta.

El Índice de Aceptación calculado es 0.90562.

Rangos predefinidos de Índice de Aceptación.

IA > 0,7 Existe alta probabilidad de éxito

0,7 > IA > 0,5 Existe probabilidad media de éxito

0,5 > IA > 0,3 Probabilidad de éxito baja

0,3 > IA Fracaso seguro.

Por lo que existe alta probabilidad de éxito.

3. 2 Conclusiones

En este capítulo se ha mostrado todo el proceso de validación de la propuesta, proceso que ayudó a mejorar la investigación, pues el trabajo con los expertos no sólo aportó valores numéricos sobre la investigación en los modelos, sino que también brindó Análisis de los Resultados que hicieron madurar más la propuesta y luego de analizar los resultados se obtuvo una alta probabilidad de éxito de la propuesta.

Conclusiones

El creciente desarrollo de la Industria de Software ha traído consigo la necesidad de producir software de Calidad, y para lograrlo se tienen en cuenta numerosos factores entre los que se encuentran las métricas de software, una herramienta indiscutible para ayudar a mantener el control de los procesos y productos durante el desarrollo del software.

El desarrollo de esta investigación tuvo gran repercusión en la preparación profesional de las autoras por los conocimientos que aportó a la misma, lo que ha permitido llegar a las siguientes conclusiones, de manera que se evidencia el cumplimiento a los objetivos propuestos:

Las métricas de software contribuyen al control, seguimiento y mejora de la calidad del proceso de desarrollo de software, y este caso particular el de los Requisitos No Funcionales.

Luego de un estudio del estado del arte sobre la IR, métricas y estándares para la evaluación de los Requisitos No Funcionales se trazó una propuesta dirigida por ISO/IEC 9126.

Se realizó una propuesta de métricas encaminada hacia las clasificaciones establecidas dentro de la Universidad, así como momentos necesarios para comenzar a aplicarlas en los proyectos específicamente al área de los Requisitos No Funcionales.

Se validó la propuesta mediante el método de expertos, logrando una alta probabilidad de éxito.

Recomendaciones

Comenzar a aplicar las métricas propuestas en todos los proyectos de la Universidad que no presenten o ejecuten métricas en los Requisitos No Funcionales.

Nombrar miembros del equipo de desarrollo que se encarguen de aplicar las métricas y evaluar los resultados.

Destinar un especialista en Arquitectura de Software para el desarrollo de la toma de decisiones luego de expuestos los resultados de la evaluación.

El trabajo de diploma puede ser utilizado como material de estudio por cualquier persona interesada en conocer acerca del papel de las métricas en el perfeccionamiento de la gestión de Requisitos No Funcionales.

Referencias bibliografías

Referencias Bibliográficas

1. (11-12-07) *Calidadsoft(Sitio de la UCI)*. **Volume**,
2. (27/11/2007) *Sitio de Calidad(UCI)*. **Volume**,
3. RUMBAUGH, I.J.G.B.J., *El Proceso Unificado de Desarrollo de Software*. 2000: Addison Wesley. Capítulos 7, 8 páginas 125-163, 187-202.
4. PRESSMAN, R., *Ingeniería del Software. Un enfoque práctico.*, ed. t. edicion. Vol. 2. 2002, McGraw-Hill/Interamericana de España. Páginas 184-186, pagina 601.
5. Chaves, M.A. (2006) *La ingeniería de requerimientos y su importancia en el desarrollo de proyectos de software.* **Volume**, 13
6. Giraldo, O.P. (2007) *Ingeniería de Requisitos*. **Volume**, 13
7. Sommerville, I., *Ingeniería del Software*. Sétima edición ed. 2005, México DF.
8. Herrera, L.J. *Ingeniería De Requerimientos, Ingeniería De Software*. **Volume**, 22
9. Dávila, N.D. (2001) *INGENIERÍA DE REQUERIMIENTOS*. **Volume**, 51
10. Hurtado, G.P.G. (2006) *"Solo requisitos 2006"*. **Volume**, 37
11. Department of Energy EEUU, D.O.M. (2006) *"Safety Software Quality Assurance Draft"*. **Volume**,
12. Thayer, M.D.a.R. (1990) *"Standards, Guidelines and Examples on System and Software Requirements Engineering"*. **Volume**,
13. Grupo10 *Gestión de Software, Evaluación de Arquitecturas de Software* **Volume**, 42
14. (2004) *SEGUIMIENTO Y RENDICIÓN DE CUENTAS DE LA EJECUCIÓN DE LOS PLANES DE ACCIÓN Y APLICACIÓN*. **Volume**, 17
15. Briand, L.C., Daly, J.W. y Wüst, J.K (1999) *A Unified framework for Coupling Measurement in Object-Oriented Systems*. IEEE transactions on software engineering **Volume**, 91-121
16. Muñoz, D.C.C. (2006) *MÉTRICAS DEL SOFTWARE:Conceptos básicos, definición y formalización.* **Volume**, 13
17. Olsina, D.L. (2003) *Métricas e Indicadores: Dos Conceptos Claves para Medición y Evaluación.* **Volume**, 52

Referencias bibliografías

18. González, D. (2001) *CAPÍTULO 2*. **Volume**, 9

19. García, M.B. *Estimación de costes de un proyecto informático.* **Volume**, 60

20. Robles, M.C.E.A. *Métricas para la Gestión de Proyectos de Software.* **Volume**, 65

21. Giraldo, O.P. *Métricas, Estimación y Planificación en Proyectos de Software.* **Volume**, 10

22. *Tema 1 -Métricas del Proyecto de Software.* **Volume**, 22

23. *Métricas Técnicas del Software(ppt)*. **Volume**, 101 diapositivas

24. *Metricas(ppt)*. **Volume**, 34

25. CUBANA, N. *NC ISO/IEC 9126-1 TECNOLOGÍA DE LA INFORMACIÓN. CARACTERÍSTICAS DE CALIDAD Y MÉTRICAS DEL SOFTWARE. Parte 1: Las características y sub-características de calidad* **Volume**,

26. Tello, A.L. (2002) *MÉTRICA DE IDONEIDAD DE ONTOLOGÍAS*. **Volume**, 236

27. Española, R.A. (1999) *Diccionario de la Lengua Española"*. **Volume**,

28. Sastre, B.d. *NORMAS DE CALIDAD EN SOFTWARE, UNA LLAVE QUE ABRE MERCADOS*. **Volume**,

29. Angeleri, M.P.M. *NORMAS DE CALIDAD DEL PRODUCTO SOFTWARE* **Volume**,

30. *ISO / IEC 12119* **Volume**,

31. Paredes, J.L.T.N.S.P. (29/01/03) *ESTANDAR ISO 9000-3*. **Volume**,

32. *UCLM-TSI. Curso Doctorado PSGC. Parte 2c El Proceso de Medición Software*. **Volume**, 65

33. Interamericana, U.A. (2002) *"Trabajo de campo 1. Segundo Parcial."* **Volume**,

34. CUBANA, N. (ISO/IEC 9126-1: 2005) *ANEXO B -TABLAS CONTENTIVAS DE LAS MÉTRICAS*. **Volume**, 38

35. J. LANDETA RODRIGUEZ, J.M.D.A., V. RUIZ HERRAN,O. VILLARREAL LARRINAGA (2002) *ALIMENTACI ON DE MODELOS CUANTITATIVOS CON INFORMACION SUBJETIVA: APLICACIÓN DELPHI EN LA ELABORACION DE UN MODELO DE IMPUTACION DEL GASTO TURISTICO INDIVIDUAL EN CATALUNYA*. **Volume**, , p. 175-196

Bibliografías

Bibliografía

1. Pressman, R. (2004). *Ingeniería del Software: Un enfoque Práctico.* McGraw Hill.

2. (jcgranja@ugr.es), D. J. (13 de febrero de 2007). Metricas técnicas del Software. España, Universidad de Granada: Grupo GILSIIS.

3. Alvarez, D. J. (13 de febrero de 2007). Ingeniería de requisitos y tecnología web. España, Universidad de Granada.

4. Ávila, D. L. (s.f.). Procedimiento para el desarrollo del proceso de Ingeniería de requisitos en un proyecto software (PROCIR). *PROCIR* .

5. Corporation., R. U. (Copyright 1987-2001). *"Rational Unified Process".* Version 2001A.04.00,.

6. Pressman, R. (2004). *Ingeniería del Software: Un enfoque Práctico.* McGraw Hill.

7. Pressman, R. S. (2002). *"Ingeniería de Software. Un enfoque práctico." Quinta edición.* Madrid: McGraw-Hill.

Anexos

Anexo 1 Modelo Cocomo

- Modelo empírico
- Se obtuvo recolectando datos de varios proyectos de software grandes, y después analizando esos datos para descubrir fórmulas que se ajustarán mejor a las observaciones.
- Esté bien documentado, es de dominio público y lo apoyan herramientas comerciales.
- Se ha utilizado y evaluado ampliamente.
- Ha evolucionado del COCOMO 81(1981) al COCOMO 2 (1995)

Cocomo Básico:

Complejidad del proyecto	Fórmula	Descripción
Simple	$PM = 2.4 (KDSI)^{1.05} \times M$	Aplicaciones bien comprendidas desarrolladas por equipos pequeños
Moderada	$PM = 3.0 (KDSI)^{1.12} \times M$	Proyectos más complejos donde los miembros del equipo tienen experiencia limitada en sistemas relacionados
Incrustada	$PM = 3.6 (KDSI)^{1.20} \times M$	Proyectos complejos donde el software es parte de un complejo fuertemente acoplado de hardware, software, reglas y procedimientos operacionales.

Cocomo II

- Considera diferentes enfoques para el desarrollo del software.
- Los niveles del modelo no reflejan simplemente estimaciones detallas con complejidad creciente.
- Los niveles se asocian a las actividades del proceso por lo que las estimaciones iniciales se llevan cabo al inicio del proceso y las estimaciones detalladas se llevan a cabo después del que se definió la arquitectura del sistema.

Niveles del Cocomo II

- Nivel de construcción de prototipo inicial:
  - Las estimaciones de tamaño se basan en puntos de objeto y se utiliza una fórmula sencilla tamaño/productividad para estimar el esfuerzo requerido.
- Nivel de diseño inicial:
  - Corresponde a la terminación de requerimientos del sistema con algún diseño inicial. Las estimaciones se basan en puntos de función que se convierten a número de líneas de código fuente.
- Nivel postarquitectónico:
  - Una vez que se diseña la arquitectura del sistema se hace una estimación razonablemente precisa del tamaño del software. En este nivel se utiliza un conjunto más grande de multiplicadores que reflejan la capacidad del personal, el producto y las características del proyecto.

Anexo 2 Modelo factores/criterios/métricas (McCall)

Modelo factores/criterios/métricas (McCall) (i)

Descompone el concepto de calidad en tres usos o capacidades importantes para un producto de software:

- Operación
- Revisión
- Transición

Cada capacidad se descompone en una serie de factores que determinan la calidad en cada una de ellas:

Operación

- Facilidad de Uso

- Integridad
- Eficiencia
- Corrección o exactitud
- Fiabilidad

Revisión

- Facilidad de prueba
- Facilidad de Mantenimiento
- Flexibilidad

Transición

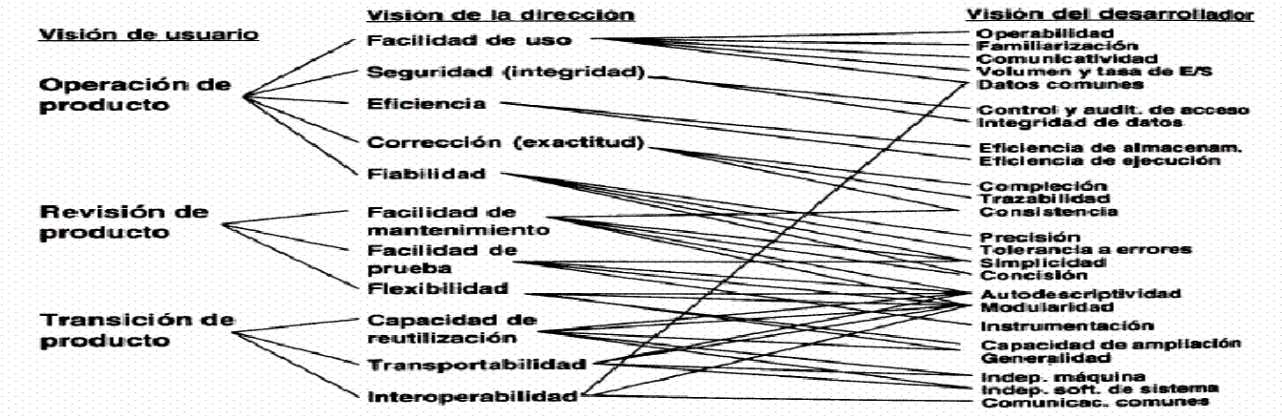
- Reusabilidad
- Portabilidad
- Interoperabilidad

Modelo factores/criterios/métricas (McCall) (ii)

Cada factor determinante de la calidad se descompone, a su vez, en una serie de criterios o propiedades que determinan su calidad.

Los criterios pueden ser evaluados mediante un conjunto de métricas. Para cada criterio deben fijarse unos valores máximo y mínimo aceptables para cada criterio.

Modelo factores/criterios/métricas (McCall) (iii)



Anexo 3 Guía para informar el peso de los criterios

Modelo No.1

Guía para informar el peso de los criterios.

Fecha de recepción _____

Fecha de entrega _____

Nombre y Apellidos de evaluador _____

Le otorgará un peso a cada criterio e acuerdo a su opinión y el peso total de cada grupo debe sumar:

Grupo No.1…… 30

Grupo No.2…… 30

Grupo no.3…… 20

Grupo No.4…….20

Para que el peso total asignado sea 100.

Grupo 1: Criterios de mérito científico

1. Valor científico de la propuesta.

Peso……..

2. Calidad de la investigación.

Peso……..

3. Contribución científica.

Peso……..

Grupo 2: Criterios de implantación

4. Necesidad de utilización de la propuesta.

Peso……..

5. Posibilidades de aplicación.

Peso……..

6. Obtención de productos finales con calidad.

Peso……..

Grupo 3: Criterios de flexibilidad

7. Adaptabilidad a los proyectos productivos que estén destinados a la producción de Software Educativo y Multimedia.

Peso……..

8. Capacidad de la estrategia para la admisión de cambios que impliquen mejoras.

Peso……..

Grupo 4: Criterios de impacto

9. Repercusión en los proyectos productivos de Software Educativo y Multimedia.

Peso……..

10. Organización en el proceso de selección y capacitación de estudiantes en los proyectos de Software Educativo y Multimedia.

Peso……..

Anexo 4 Guía para la evaluación

Modelo No.2

Guía para la evaluación

Fecha de recepción _____

Fecha de entrega _____

Nombre y Apellidos de evaluador _____

Criterios de medida que se evalúan en una escala de 1-5.

Grupo 1: Criterios de mérito científico

1. Valor científico de la propuesta.

Peso……..

2. Calidad de la investigación.

Peso……..

3. Contribución científica.

Peso……..

Grupo 2: Criterios de implantación

4. Necesidad de utilización de la propuesta.

Peso……..

5. Posibilidades de aplicación.

Peso……..

6. Obtención de productos finales con calidad.

Peso……..

Grupo 3: Criterios de flexibilidad

7. Adaptabilidad a los proyectos productivos que estén destinados a la producción de Software Educativo y Multimedia.

Peso……..

8. Capacidad de la estrategia para la admisión de cambios que impliquen mejoras.

Peso……..

Grupo 4: Criterios de impacto

9. Repercusión en los proyectos productivos de Software Educativo y Multimedia.

Peso……..

10. Organización en el proceso de selección y capacitación de estudiantes en los proyectos de Software Educativo y Multimedia.

Peso……..

Categoría final de la propuesta

___ 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.

Valoración final a través de una encuesta.

1. ¿Cree usted que la propuesta de métricas está a la altura de las posibilidades y necesidades de la Universidad?

Sí____ No_____ ¿Por qué?

_____

_____

_____

2. Con la propuesta establecida para los Requisitos No Funcionales, ¿cree usted que logrará efectividad en el proceso de desarrollo de los Requisitos No Funcionales?

Sí____ No_____ ¿Por qué?

_____

_____

_____

3. ¿Cree usted que estas métricas son los necesarios en el desarrollo y efectividad de los Requisitos No Funcionales en la Universidad?

Sí____ No_____ ¿Por qué?

_____

_____

_____

4. En la escala de 1 al 5 otorgue una evaluación en relación a la Propuesta según los criterios siguientes :

____Satisfacción de las necesidades de la Universidad.

____Adaptabilidad a todos los proyectos productivos de la Universidad.

____Repercusión en los proyectos productivos de de la Universidad.

____ Posibilidad de aplicación.

5. ¿Cuáles serían los argumentos que usted expondría a favor de la aplicación del procedimiento propuesto y cuáles estarían en contra?

6. Realice alguna observación o aporte sobre la propuesta que está evaluando.

Glosario de Términos

CMM: Modelo de capacidad y madurez

CMMI: Modelo de capacidad y madurez integrado

FMC: Modelo" Factores/Criterios/Métricas".

GQM: Método Metas/ Preguntas/ Métricas.

IEEE: The Institute of Electrical and Electronics Engineers, el Instituto de Ingeniería Eléctrica y Electrónica, una asociación técnico-profesional mundial dedicada a la estandarización, entre otras cosas.

Indicador: soporte de información (habitualmente expresión numérica) que representa una magnitud, que a través del análisis del mismo se permita la toma de decisiones sobre los parámetros de actuación (variables de control) asociados.

ISO: International Organization for Standardization.

ISO 15504: Estándar para la evaluación y mejora de los procesos

ISO 15939: Estándar sobre la medición del software.

ISO 9000-2000: Familia de Normas y directrices internacionales para el establecimiento de sistemas de gestión de la calidad.

ISO 90003: Guía de interpretación de la norma ISO 9001:2000 para la producción de software y servicio de soporte asociados al software.

RNF: Requisitos No Funcionales.

UCI: Universidad de las Ciencias Informáticas.