UNIVERSIDAD DE GRANADA
Departamento de Ciencias de la Computación e Inteligencia Artificial

Programa Oficial de Doctorado en Tecnologías de la Información y la Comunicación

TESIS DOCTORAL

Modelo computacional para la evaluación y selección de productos de software.

Realizada por: Yamilis Fernández Pérez

Directores: Carlos Cruz Corona
José Luis Verdegay Galdeano

Granada, 2018

Esta memoria titulada “Modelo computacional para la evaluación y selección de productos de software”, que presenta Yamilis Fernández Pérez para optar al grado de Doctora en Informática, ha sido realizada en el Departamento de Ciencias de la Computación e Inteligencia Artificial de la Universidad de Granada bajo la dirección de los Doctores Carlos Cruz Corona y José Luis Verdegay Galdeano.

La doctoranda y los directores de la tesis, garantizamos al firmar esta tesis doctoral, que el trabajo ha sido realizado por la doctoranda bajo la supervisión de los directores de la tesis y hasta donde nuestro conocimiento alcanza, en la realización del trabajo, se han respetado los derechos de otros autores a ser citados, cuando se han utilizado sus resultados o publicaciones.

En Granada, 15 de enero de 2018.

La Doctoranda

Yamilis Fernández Pérez

El Director

Fdo: Carlos Cruz Corona

El Director

Fdo: José Luis Verdegay Galdeano

Agradecimientos

En estas líneas, quisiera agradecer a todas las instituciones y a las personas que me han apoyado en este largo camino, permitiéndome vencer un colosal reto.

Agradezco en primer lugar, a mi familia. A mi hijo Rene, fuente de inspiración de todo lo que hago, por él he llegado hasta aquí, es quien más se ha sacrificado por mis largas ausencias para el desarrollo de esta investigación. A mis padres por sus desvelos en todos estos años, a mi hermano Rey y mi prima Eleusis por su apoyo y ayuda.

A mis directores de tesis, los doctores Carlos Cruz Corona y José Luis Verdegay Galdeano por sus atenciones y paciencia. A Carlos, por su dedicación y apoyo incondicional, por confiar en mí y su inestimable amistad, comprensión y cariño. A Curro, sin dudas la oportunidad de conocerle y el privilegio de trabajar con él han marcado mi formación científica y como ser humano. Gracias por su paciencia y corrección de mis errores.

A mis colegas, en especial a los amigos insuperables de la Dirección de Posgrado de la UCI, por su inestimable ayuda e impulso en esta investigación. En especial al Dr. Novo, quien ya no está entre nosotros; él me animó a concluir este trabajo y apoyó en la realización de la documentación legal del doctorado en Cuba.

A los miembros del Grupo MODO, por su acogida y apoyo durante mis estancias de investigación.

A mis compañeros de CALISOFT, que permitieron la aplicación y demostración de la utilidad de esta tesis; a las otras instituciones que me animaron y están aplicando los resultados de la investigación.

A la Universidad de las Ciencias Informáticas, la Universidad de Granada y la Asociación Universitaria Iberoamericana de Postgrado, por permitirme formar parte del Programa de Doctorado Iberoamericano en *Soft Computing* y Sistemas Inteligentes.

Una “mención especial" para Waldo, mi compañero en la vida, sin cuya ayuda en la etapa final del doctorado no hubiera sido posible concluir; por sus sugerencias, amor, comprensión y ayuda.

A todos los que de una u otra forma contribuyeron a que este empeño llegara a feliz término. MUCHAS GRACIAS.

Índice

INTRODUCCIÓN

Cada día, aumenta la presencia del software en innumerables actividades del ser humano, entre ellas se encuentran todas aquellas relacionadas con el sector industrial, el comercio, la salud, la educación, el transporte, el control de la infraestructura urbana o el medio ambiente. El software, se ha convertido en un producto vital, tanto para empresas, organismos, servicios y tareas cotidianas de los ciudadanos como para la toma de decisiones, el intercambio de información y la gestión del conocimiento.

Su utilización, promueve el aumento de la producción, amplía las posibilidades de negociación, supervisa y controla activamente a los pacientes de manera no invasiva y facilita la comunicación al disminuir las barreras geográficas. Se está convirtiendo, cada vez más, en una tecnología penetrante, omnipresente y habilitadora. En general las tecnologías disruptivas, como la Internet de las cosas (IoT), hacen más interactivos todos los objetos de uso cotidiano. Conceptos como el hogar inteligente, los edificios inteligentes o las ciudades inteligentes, serían imposibles siquiera de imaginar sin el desarrollo de sistemas embebidos confiables, seguros, interoperativos, portables, etc. Se estima que, para 2020, la conexión a Internet de millones de dispositivos, proporcionará a los seres humanos una serie de servicios y aplicaciones inteligentes sin precedentes.

Esta situación induce al incremento de la complejidad del software, a la par que las funcionalidades que debe lograr, surgiendo un ambiente de competencia y especialización entre las entidades productoras y comercializadoras, que provoca en estas un especial interés por la calidad del mismo, convirtiéndose así en uno de los elementos diferenciadores entre las diferentes compañías a nivel mundial por las ventajas competitivas que puede aportar.

La búsqueda de la calidad de los sistemas, ha propiciado por tanto la creación de modelos, estándares, marcos de trabajo y metodologías para evaluarla, asegurarla y controlarla.

El desarrollo de múltiples dominios tecnológicos, como la computación de altas prestaciones, el Big Data, Cloud, la Inteligencia Artificial o la IoT, influye en la forma en que se desarrolla el software y exige la aplicación de métodos y herramientas de Ingeniería de Software novedosas y genéricas, aplicables en diferentes dominios y complementadas con actividades relacionadas con el dominio específico. Estas tecnologías requieren dirigir la transición de los procesos de desarrollo hacia un nuevo paradigma que trate el software, los datos y los recursos de computación y

comunicación, como elementos abstractos, lo que permitirá que los datos fluyan libremente sobre infraestructuras heterogéneas de manera escalable, distribuida y comprensible para las personas [1].

Lo anterior, hace que la tecnología del software constituya uno de los desafíos a abordar como acciones de investigación e innovación del Programa H2020 de la Unión Europea para el período 2018-2020.

Dentro de la Ingeniería de Software, se encuentra la calidad como área de estudio, resultando por tanto pertinente el desarrollo de herramientas, modelos y marcos de trabajo de la calidad, lo que desde un punto de vista general es precisamente el tema a abordar a través de la investigación desarrollada en esta tesis, que de las dos vertientes desde las que se enfoca la calidad del software: la calidad del producto y la del proceso de desarrollo, se orienta hacia la de la calidad, concentrándose en la evaluación y selección de productos de software.

Según la NC ISO/IEC 25000 [2], la calidad del software es el grado en que este satisface las necesidades expresadas o implícitas, cuando se usa bajo condiciones determinadas. Pero el término calidad de producto de software es ambiguo dado su carácter intangible y eso dificulta el proceso de evaluación, en el que intervienen diversos factores.

La evaluación es un proceso costoso y complejo por la cantidad de recursos involucrados, resultando de vital importancia para la toma de decisiones, ya que una evaluación equivoca puede provocar cuantiosos gastos monetarios innecesarios a la entidad donde se implanta o incluso la pérdida de vidas humanas, según el dominio de aplicación. De ahí la relevancia de lograr una adecuada correlación entre los resultados obtenidos al evaluar un producto software con la calidad que este muestra en la realidad. Otro factor esencial es la correcta determinación de los indicadores de problemas importantes, algo que puede implicar un refinamiento del proceso de desarrollo, reducción de costos y mejora de la competitividad de las empresas. Resulta primordial también, lograr que la evaluación de la calidad de un software se traduzca en valores que puedan ser comparados e integrados con criterios de costo y beneficio.

Desde el punto de vista de servicio, es importante la satisfacción de los clientes, entendidos no solo como los usuarios finales, sino también los desarrolladores o quienes desean diseñar nuevas versiones, al poder medir la calidad del producto con precisión e interpretar fácilmente los resultados.

Un inconveniente común en los métodos de decisión existentes, es la necesidad de convertir el conocimiento de los decisores acerca de un problema en medidas cuantitativas (números y funciones). Esto se da justamente en el caso de la evaluación de la calidad del software, donde predomina un criterio cualitativo, más que una evaluación cuantitativa exacta, y por tanto en muchas ocasiones el decisor no es capaz de definir de manera precisa la importancia de los criterios y/o cuan buenas son las alternativas con respecto a cada criterio. En estas situaciones, se usan medidas o cantidades que no son exactas sino aproximadas, siendo una opción natural utilizar modelos que incorporen el manejo de variables cualitativas (descriptivas, lingüísticas, ordinales). Los modelos lingüísticos difusos, que tienen como base teórica la *Soft Computing*, y particularmente los conjuntos difusos, se han mostrado como una técnica eficaz para valorar aspectos de naturaleza cualitativa. Así, la *Soft Computing* ([3], [4], [5], [6] y [7]) aplicada a la toma de decisión multicriterio, puede proporcionar las herramientas necesarias para proponer un modelo de evaluación y selección de productos de software que facilite la toma de decisiones.

Por todo ello, esta tesis aborda un tema de investigación importante, actual y abierto, como es la búsqueda de una selección objetiva de productos, según las características de calidad, y un modelo de evaluación operativo.

Es un tema no suficientemente trabajado, pues a pesar de la existencia de modelos, metodologías y métodos de evaluación de software basados en *Soft Computing* ([8], [9] y [10]), enfocados a considerar el tratamiento de la incertidumbre, persisten deficiencias en los mismos, tales como la incapacidad para ofrecer un marco conceptual capaz de manipular información heterogénea. No existe tampoco una normalización de la escala de valoración; el tratamiento de las relaciones y restricciones entre las características y medidas de calidad, es prácticamente inexistente y, por último, se dificulta el manejo de la información de los usuarios finales al no tener en cuenta que estos pueden proceder de diferentes áreas de conocimiento, poseer diversa formación y dominio de los atributos y necesitar, por tanto, escalas con desigual granularidad.

En ese contexto, el resultado de la revisión bibliográfica realizada, arroja que los modelos y métodos de solución recogidos en la literatura para la evaluación de la calidad de productos de software, no abordan estas situaciones. Concluyendo, existen parámetros importantes que no son contemplados. Entre ellos destacan los siguientes:

(1) la incorporación de criterios esenciales y sus correspondientes restricciones,
(2) la interdependencia entre los criterios de evaluación,

(3) la incertidumbre que caracteriza el proceso, y
(4) la heterogeneidad de la información.

Todo esto provoca pérdida de información, limita el proceso de toma de decisiones y provoca insatisfacción en los clientes de la evaluación.

Como consecuencia de este cúmulo de problemas, las insuficiencias en el proceso de evaluación y selección de productos de software según su calidad, limita la toma de decisiones y la satisfacción del cliente, lo que justifica el desarrollo de esta memoria, dedicada en consecuencia a buscar soluciones que palien estas dificultades.

Por tal motivo, se propone un nuevo modelo basado en técnicas de *Soft Computing*, como soporte a la toma de decisiones durante la evaluación y selección de productos de software según su calidad. El modelo aporta elementos que posibilitan mejorar el proceso de desarrollo y, por supuesto, el producto del software. Además, facilita la flexibilización del marco de evaluación, de modo que se puedan utilizar diversos modelos de calidad con criterios y medidas, representadas en forma cualitativa y cuantitativa e interrelacionadas.

La relevancia de la evaluación de la calidad, tanto como la selección de los productos que serán usados en condiciones específicas, las limitaciones de los métodos tradicionales para llevar a cabo dicho proceso, y la búsqueda de un modelo que obtenga resultados más precisos, facilite la toma de decisiones y logre la satisfacción de los clientes, motivaron la **hipótesis de partida de esta tesis**:

Como se ha puesto de manifiesto, los métodos tradicionales para llevar a cabo el proceso de evaluación y selección de productos de software, no modelan adecuadamente la incertidumbre que caracteriza el proceso, ni la heterogeneidad e interdependencia entre criterios, ni la incorporación de criterios esenciales y las restricciones asociadas a ellos. Todo ello limita el proceso de toma de decisiones y produce resultados imprecisos e insatisfacción de los clientes. Por ello planteamos como hipótesis que el uso de técnicas propias del ámbito de la *Soft Computing*, que facilitan la gestión y el tratamiento de la incertidumbre, modelan rigurosamente la integración de información heterogénea y permiten tener en cuenta la interdependencia entre criterios y las restricciones de criterios esenciales, servirá para paliar todas esas disfunciones, resolviendo los problemas que conllevan y dibujando un nuevo marco de oportunidades prácticas para una correcta evaluación de la calidad.

De manera coherente con esa hipótesis, el **objetivo general** de la tesis es desarrollar un modelo para la evaluación y selección de productos de software, según su calidad,

que facilite la toma de decisiones bajo incertidumbre, trate la información heterogénea, la interdependencia entre los criterios y los criterios esenciales logrando elevar los niveles de satisfacción del usuario.

Para cumplir el objetivo general, se definen los siguientes **objetivos específicos**:

1. Construir el marco teórico referencial relacionado con el análisis de los elementos fundamentales del proceso de evaluación de la calidad y la selección de los productos de software, que facilite la toma de decisiones bajo incertidumbre, con el propósito de deslindar e identificar las limitaciones existentes. Además, analizar en profundidad los métodos más importantes de solución, con énfasis en las técnicas de *Soft Computing* más utilizadas.
2. Considerar la evaluación y selección de productos de software como un problema de toma de decisiones, utilizando las bases teóricas de la lógica difusa y la modelación lingüística, lo que implica el desarrollo de un modelo que formalice el proceso de evaluación y selección bajo incertidumbre, trate la información heterogénea, la interdependencia entre los criterios y los criterios esenciales, haciendo uso de técnicas de *Soft Computing*.
3. Desarrollar una herramienta para soportar la utilización del modelo propuesto, ya sea para la experimentación o el desarrollo de aplicaciones reales, tratando de forma sencilla y flexible los parámetros del problema y métodos de solución.
4. Contrastar la viabilidad, operatividad y eficacia de la propuesta, a partir de su aplicación en casos reales en la industria cubana de software, especialmente en el Centro Nacional de Calidad de Software cubano (CALISOFT) facilitando la toma de decisiones y la satisfacción del cliente.

Para lograr los objetivos propuestos, la memoria se ha ordenado en introducción, tres capítulos principales, conclusiones, líneas de trabajo futuro y bibliografía consultada; además de otra sección complementaria con la lista de las publicaciones obtenidas a lo largo de la realización de la tesis y cuatro apéndices con diferentes aclaraciones sobre conceptos, estándares, medidas definidas para la experimentación, los casos reales desarrollados y el código escrito. A continuación, se describe brevemente cada uno de los capítulos.

El **Capítulo 1**, expone los conocimientos necesarios para la comprensión de la investigación que se ha realizado. Inicialmente se revisan conceptos básicos de la calidad, en general, y de la calidad de software, en particular. Continúa con una exploración exhaustiva de los estándares y modelos de calidad, elementos fundamentales y etapas principales del proceso de evaluación y selección de productos de software, según su calidad. Se analiza este como un problema de toma

de decisiones y se identifican las limitaciones presentes a partir del examen de los métodos de solución. Se abordan también las bases conceptuales de la *Soft Computing*, los conjuntos difusos, la modelación lingüística y los mapas cognitivos difusos. Todos estos elementos constituyen el marco conceptual de la tesis y permiten dar cumplimiento a los objetivos trazados.

Como resultado del análisis bibliográfico realizado en el capítulo, se comprueba que el problema de evaluación y selección de productos de software y los métodos de solución existentes, están limitados a un contexto real. De ahí, la pertinencia de los capítulos 2 y 3.

El **Capítulo 2** presenta las contribuciones más importantes de la tesis. Se propone en primer lugar una definición formal del problema. A continuación, se presenta el modelo de evaluación y selección de productos de software, según su calidad, en contextos heterogéneos, bajo incertidumbre, con interdependencia entre los criterios y teniendo en cuenta los criterios esenciales. Se describe el objetivo, fundamentos teóricos, principios, estructura, descripción general de cada componente y los pasos a realizar, así como un ejemplo demostrativo de la operatividad y viabilidad de este enfoque.

El **Capítulo 3** muestra la herramienta informática SoftQuality que implementa el modelo propuesto. El código de esta se puede descargar en http://modo.ugr.es/SoftQuality/ y también se encuentra en el CD adjunto. A título de ejemplo se muestra una “app” para medir la usabilidad de aplicaciones para móviles, basada en el modelo propuesto. Como demostración de utilidad, eficacia y operatividad, se aplica el modelo a un caso práctico real, asociado a una problemática de CALISOFT.

CAPÍTULO I. PROBLEMAS EN LA EVALUACIÓN DE LA CALIDAD DE PRODUCTOS DE SOFTWARE

Introducción

Al seleccionar un producto o servicio, es esencial una valoración de su calidad, aspecto que se analiza según los intereses del cliente y la finalidad del objeto a adquirir.

La calidad es un concepto ampliamente tratado en la actualidad, pero a la vez su significado puede ser percibido de distintas maneras. Suele ser ambiguo e impreciso, al estar relacionado con las percepciones de cada individuo, al comparar determinado producto o servicio con cualquier otro de su misma especie. Esta propiedad se profundiza al tratarse de productos intangibles, como lo es el software.

En la actualidad, el software se ha convertido en uno de los principales objetivos estratégicos de las organizaciones, debido a que, cada día, sus procesos más importantes dependen del funcionamiento del mismo. Valorar su calidad antes de adquirirlo, implantarlo y lanzarlo al mercado, es una necesidad.

Hasta la fecha, la mayoría de los métodos y procedimientos considerados para evaluar la calidad de los productos de software, se perciben en un entorno con datos precisos y calculados, donde no se valora la relación existente entre los criterios que la definen. Tampoco analizan los criterios esenciales que el software bajo evaluación debe cumplir obligatoriamente. Esto produce resultados irreales, índices de calidad desviados o no representativos del comportamiento del producto en su desempeño.

Partiendo de este diagnóstico, determinada la importancia del problema y con la intención de introducir un nuevo modelo operativo, se dedica el presente capítulo a analizar y detallar los elementos teóricos de la calidad, centrándonos en los productos de software y en los modelos teóricos existentes en el ambiente que nos interesa. También se describen las herramientas metodológicas con que se cuenta para la resolución de las nuevas variantes a considerar, con especial énfasis en las fundamentadas en los conceptos de la teoría de conjuntos borrosos**.**

I.1 Definiciones preliminares

Según el diccionario de la Real Academia Española de la Lengua, la palabra calidad proviene del latín *qualitas* (cualidad), definida como: “Propiedad o conjunto de propiedades inherentes a algo, que permiten juzgar su valor” [11]. Visto así, la calidad puede aplicarse a cualquier contexto en que se desee hacer un juicio o establecer una comparación entre dos o más cosas. Como es posible observar, el concepto tiene un

alto grado de subjetividad porque depende del punto de vista de quien la valore y del patrón con que se juzgue.

Cuando se habla de calidad de un producto, se apunta a lograr una diferenciación de tipo cualitativo y cuantitativo en relación a algún atributo requerido o conjunto de atributos. En cuanto al usuario, implica satisfacer sus expectativas y anhelos, no solo cubrir sus necesidades, incluyendo la relación costo/beneficio, sino que involucra también la obtención de un producto con valor añadido, que supere las expectativas. En cuanto al fabricante, es esencial que cumpla determinadas propiedades técnicas.

De tal modo, puede plantearse que, en la concepción de la calidad, influyen diversos factores o dimensiones. Es un concepto multifactorial, en el cual la dimensión técnica abarca los detalles científicos y tecnológicos relacionados con el producto, y la dimensión humana incluye la satisfacción del cliente, influida por sus necesidades, expectativas y cultura. Por último, está la dimensión económica, que comprende la reducción de los costos, tanto para el productor, como para el cliente.

Un grupo de autores se ha destacado en la conceptualización del término calidad y en la definición e implementación de teorías que han sido la base del desarrollo de la gestión de la misma. Entre ellos están Walter A Shewhart [12], Philip B. Crosby [13], W. Edwards Deming [14], Armand V. Feigenbaum [15], Joseph M. Juran [16] y Kaoru Ishikawa [17]. Estos la definen de diferentes maneras y analizan dos categorías esenciales. La primera, es trabajar en las especificaciones de los productos; o sea, que sus características medibles satisfagan determinados requisitos. En cuanto a la segunda, significa satisfacer al cliente. Esto hace concluir que la calidad es imposible valorarla por una sola característica; es un concepto multidimensional y multifactorial, con alto de grado de subjetividad e incertidumbre.

Es esencial, además, que las características sean medibles, comparables, de manera que pueda realizarse una valoración acertada de la misma. Aunque dichas mediciones no tienen necesariamente que ser en valor numérico, pueden ser apreciaciones del cliente, los especialistas o desarrolladores.

Como se puede apreciar, la calidad tiene un lado objetivo, fácil de medir, y otro subjetivo, ambiguo e impreciso. En la actualidad abarca un poco más; cuando se valora la calidad de un producto, no basta que cumpla los requisitos propios de este y satisfaga las expectativas del cliente, también debe involucrar la reducción de problemas sociales, medioambientales y éticos. [18]

La gestión de la calidad, ha evolucionado a medida que ha evolucionado la industria, los métodos de producción y las relaciones con el cliente. No existe un cambio brusco

de una etapa a otra: se incorporaron de manera gradual conceptos e ideas que marcaron tendencias y fueron después seguidas por otras empresas y países.

Tal evolución aparece resumida en la Figura I.1, donde se observa el desarrollo de la gestión de la calidad desde la inspección y el control, relacionada con la satisfacción de las necesidades técnicas del producto, hasta la calidad total, centrada en la permanente satisfacción de las expectativas del cliente y en la mejora continua.

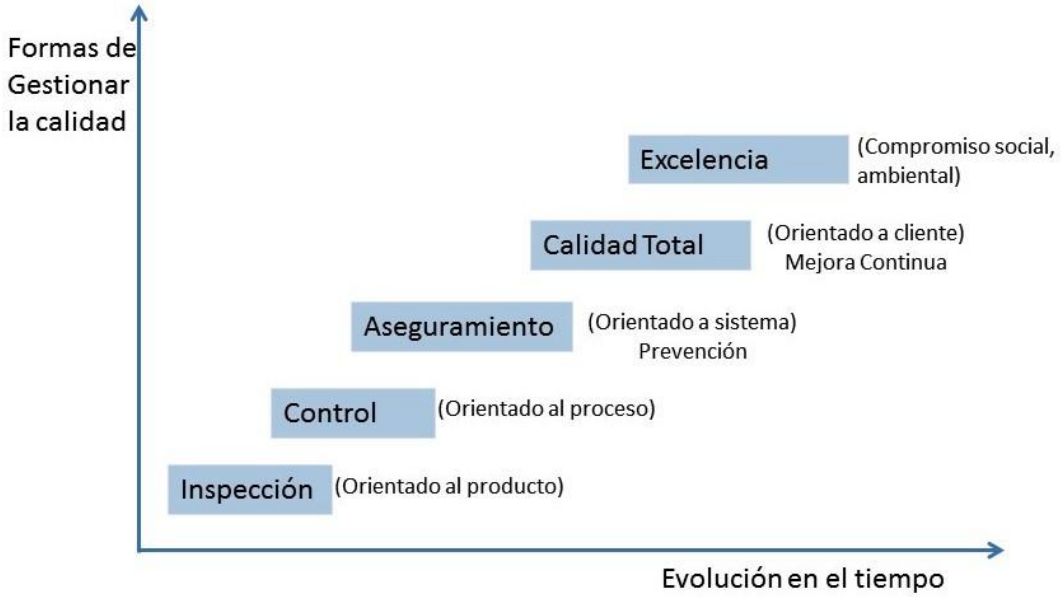

Figura I.1. Evolución de la gestión de la calidad.

En la actualidad, se avanza hacia el logro de la excelencia en todas las actividades de una organización y sus productos, alcanzando un compromiso social y ambiental.

Dentro de la gestión de la calidad total se encuentra el aseguramiento, que engloba el control, y dentro de este la evaluación. La evaluación, es el proceso que desarrolla una organización para supervisar las actividades de control de la calidad, siendo este, precisamente, el objeto de análisis de la presente investigación.

Las definiciones de los diferentes términos y aspectos que conforman la calidad, su proceso de gestión y los modelos, se abordan en estándares. Un estándar es un documento que proporciona los requisitos, especificaciones, directrices o características que se pueden utilizar constantemente para asegurar que los materiales, productos, procesos y servicios sean los adecuados para su propósito [19]. Un estándar sirve como modelo, patrón o referencia.

Hay varias organizaciones que definen estándares, pero la más destacada es la Organización Internacional de Normalización (ISO, siglas en inglés de *International Organization for Standardization*), un organismo consultivo de las Naciones Unidas,

independiente y no gubernamental. La ISO está compuesta por miembros de los organismos nacionales de normalización de 162 países. Sus integrantes desempeñan un papel vital en la forma en que opera.

Las normas ISO, se constituyen en una serie de estándares que se pueden agrupar por familias. Uno de los estándares más reconocidos y populares de la ISO, es la familia de normas ISO 9000, la cual aborda diversos aspectos de la gestión de la calidad.

Dichas normas, proporcionan orientación y herramientas para las empresas y organizaciones que pretenden asegurarse de que sus productos y servicios cumplan consistentemente los requerimientos del cliente y la calidad mejore constantemente. Han evolucionado con el paso del tiempo, surgiendo versiones identificadas con el año de aprobación. (Ver Tabla I.1).

Tabla I.1. Normas de la familia ISO 9000

Normas	***Cubrimiento de la norma***
ISO 9000: 2015	Conceptualiza los términos como producto, proceso, política de la calidad, sistema de gestión, gestión de la calidad, planificación, aseguramiento, control y mejora de la calidad. En el Anexo I, se muestra la definición de estos conceptos según la norma.
ISO 9001: 2015	Establece los requisitos para un sistema de gestión de la calidad. Un sistema de gestión de la calidad es la forma de definir cómo una organización puede satisfacer las necesidades de sus clientes y de los involucrados en el proceso.
ISO/IEC 90003:2014	Presenta una guía para la aplicación de la ISO 9001: 2008, en la industria del software. Proporciona orientación para la adquisición, suministro, desarrollo, operación y mantenimiento de software y servicios de soporte.
ISO 9004: 2009	Se centra en cómo hacer que un sistema de gestión de la calidad sea más eficiente y eficaz, incluyendo la mejora continua.
ISO 19011: 2011	Presenta una guía sobre las auditorías internas y externas de los sistemas de gestión de la calidad

Las empresas, bajo las normas ISO, pueden implementar un sistema de gestión de la calidad, crear su manual, realizar auditorías periódicas y certificarse. La certificación, es el cierre del proceso de implementación del sistema y brinda la documentación para demostrar el logro ante el mercado. La ISO 9001: 2015, es la única norma en la familia que puede ser certificada.

Otra organización que define estándares relacionados con la calidad, es el Instituto de Ingenieros Eléctricos y Electrónicos (IEEE), una asociación internacional dedicada a promover la innovación y excelencia tecnológica en beneficio de la humanidad. Actualmente, cuenta con alrededor de 900 estándares activos. En el Anexo II, se resumen algunos de los estándares de dicha organización con respecto a la calidad. ([20]). De estos, en lo relativo a la evaluación de la calidad de productos de software,

son de consulta obligada las normas IEEE 1061 y los estándares relacionados con el proceso de Verificación y Validación (V&V), como IEEE 1012 y 1028.

I.2 Calidad de software. Conceptos y características

Al particularizar las definiciones anteriores a la industria del software, lo primero que debe quedar claro es el propio concepto de software. Muchas personas asocian software solo con programas de computadoras. El software es más amplio, e incluye la documentación asociada y la configuración de datos necesarios para hacer que estos programas operen de manera correcta [21].

Ahora, si se intenta definir calidad de software, a partir de los enunciados de los autores clásicos, queda como el grado en que el software es capaz de cumplir los requisitos establecidos, los estándares especificados y satisfacer las expectativas del cliente cuando es usado bajo determinadas condiciones.

Un software de calidad es aquel que cumple su función correctamente, de manera eficiente, flexible, confiable, fácil de mantener y usar, portable y seguro. Su calidad la determina un conjunto de cualidades que formaliza su utilidad y existencia, reforzando, además, su carácter multidimensional. No obstante, existen elementos subjetivos que dificultan la medición; por ejemplo, si es atractivo, fácil de usar y aprender y satisface las necesidades del cliente.

Si se analizan los diferentes conceptos expuestos por Pressman y Sommerville, reconocidos investigadores de la ingeniería de software ([22] y [21]) y los estándares internacionales (IEEE 610 [23], ISO 9126, ISO 25000) se aprecia coincidencia con esta definición. Las anteriores insisten en el cumplimiento de tres puntos importantes para lograr la calidad [24]. El primero, es el cumplimiento de los requisitos del software; la falta de concordancia con los requisitos es una falta de calidad; los requisitos son la base de las medidas de la calidad. El segundo, es el cumplimiento de un conjunto de requisitos implícitos o subjetivos (referente a la interfaz, satisfacción); si el software se ajusta a sus requisitos explícitos, pero falla en alcanzar los requisitos implícitos, la calidad del software se debilita. El tercero, se refiere al cumplimiento de los estándares especificados, que definen un conjunto de criterios y buenas prácticas que guían la forma de aplicación de la ingeniería de software en el desarrollo del producto; cuando se irrespeten los mismos, habrá falta de calidad.

Es importante analizar aquí la reflexión de Sommerville [21], quien sostiene que “la calidad del software es un concepto complejo, por ser un bien intangible este producto no es trivial expresar sus requisitos y las expectativas del cliente”. Esto dificulta la especificación de ciertas características, al desconocerse cómo definirlas sin

ambigüedad, debido a que la industria suele tener una apreciación y los clientes otra. Por tanto, aunque un producto se ajuste a su especificación, los usuarios pueden no considerarlo de alta calidad al no responder a sus expectativas.

De la anterior reflexión, se deriva que la evaluación de la calidad de productos de software por la industria, es un proceso complejo por la ambigüedad e imprecisión de algunas de sus características, al depender de la percepción y observación de los diferentes actores que la valoran. Es difícil también expresar los requisitos de manera no ambigua, lo que dificulta especificar la calidad requerida.

En el caso de la industria del software, no solo se mide la calidad del producto; se tiene en cuenta, además, la calidad del proceso de desarrollo del mismo. Es importante valorar aquí el uso de buenas prácticas, criterios y estándares. Ambos niveles aparecen estrechamente relacionados, pues la calidad del producto está en función de la calidad del proceso de desarrollo y viceversa: las metas a establecer para la calidad del producto, determinan las metas para la calidad del proceso de desarrollo.

Sin embargo, se ha incrementado el número de empresas preocupadas por la valoración de la calidad de ambos. Existe poca evidencia de que, cumpliendo determinado modelo de proceso de desarrollo, pudiera mejorarse el producto de software [25].

Con este análisis, se constata que valorar la calidad del producto del software es un proceso ambiguo, impreciso y con alta incertidumbre. Expresar la calidad en términos de cualidades y características, se torna difícil, motivo por el cual surgen diversos modelos y estándares.

I.3 Modelos y estándares de calidad del software

Un modelo de calidad de software es una representación de la calidad a partir de criterios o características. Esta descomposición permite satisfacer las necesidades de los desarrolladores del producto, los encargados de su mantenimiento, clientes que lo compran y usuarios finales. Además, proporciona las bases para la especificación de los requisitos de calidad y su evaluación. Es esencial que el modelo de calidad sea simple, concreto y permita ser utilizado con ciertas garantías de éxito [22]. Los mismos se constituyen en guías para construir mejores productos y asegurar su calidad. Algunos se han establecido como estándares, como la ISO 25010 y la ISO 9126.

En el sector industrial, desde hace varias décadas, se viene desarrollando una gran variedad de modelos y estándares de referencia para la calidad de los productos de software. Se encuentran los estándares que definen un marco de trabajo para la

evaluación de esta (ISO 14598 e ISO 25040); otros, están dirigidos a evaluar características específicas de calidad o iniciativas regionales.

Para la evaluación y mejora de los procesos de desarrollo de software, se pueden citar los estándares ISO 90003, ISO/IEC 12207, ISO/IEC 15504. Se destaca también el *Capability Maturity Model Integration* (CMMI) definido por el Instituto de Ingeniería de Software (SEI) de la *Carnegie Mellon University*. Es un modelo de madurez de mejora de los procesos para el desarrollo de productos y de servicios. Consiste en las mejores prácticas que tratan las actividades de desarrollo y de mantenimiento que cubren el ciclo de vida del producto, desde la concepción a la entrega y el mantenimiento. Una vez que la organización ha decidido adoptar CMMI, aplica la metodología de mejora organizacional IDEAL (*Initiating, Diagnosing, Establishing, Acting and Learning*) como guía para iniciar, planificar e implementar acciones de mejora. El método SCAMPI (*Standard CMMI Appraisal Method for Process Improvement*) se utiliza para la evaluación de las organizaciones que usan CMMI, y un resultado de una evaluación es una calificación [26]. Un gran número de empresas en todo el mundo se han evaluado o certificado según dichas normas o modelos [25].

La Tabla I.2, muestra los estándares agrupados en base a sus propósitos, especificando si es dirigido a la calidad de procesos o a la calidad de productos y clasifica su aplicación en un sector específico o de manera general.

Tabla I.2. Estándares y modelos de calidad de software.

	Calidad de procesos	**Calidad de productos**
General	CMMI ISO 15504 (SPICE) ISO 90003: EFQM	ISO 9126 ISO 14598 ISO 25000 Boehm, McCall, FURPS, Dromey
Servicios	ITIL COBIT	
Seguridad	ISO 17799	ISO 15408 ITSEC Trustmarks
Otros	Iniciativas regionales MOPROSOFT, MP Br, KSPICE, etc	W3C Accesibility Web Accessinility Initiative

Realizar un análisis de todos los estándares y modelos podría resultar extenso y saldría de los objetivos del presente trabajo. Como se expresó en el epígrafe anterior, obtener un modelo estándar de proceso, no garantiza asegurar la calidad del producto. Por eso, la evaluación de esta ha experimentado un notorio auge en los últimos años.

En el presente epígrafe, se analizan los modelos más importantes dirigidos a la evaluación de la calidad de los productos de software, objetivo de esta investigación.

Los modelos de calidad de los productos de software más significativos son: McCall [27], Boehm [28], FURPS [29], Dromey [30], ISO 9126 [31] e ISO 25010. Cada uno de ellos fue introducido con el propósito de definir la calidad de una nueva forma, cubriendo todas las perspectivas con diferentes puntos de vista para la evaluación del producto. Dichos modelos, conocidos como básicos, son genéricos y aplicados para cualquier tipo de software. Sirven como base para los modelos adaptados o modificados, que se personalizan o especifican para una aplicación y sector u área en particular [32].

Dichos modelos de calidad, agrupan los diferentes criterios a tener en cuenta en una evaluación. Para una mejor comprensión de los criterios y facilitar su uso, se clasifican de diferentes formas; están los lineales y los jerárquicos, entre otros.

Los lineales son los más simples y se enfocan en evaluar características de calidad específicas, donde analiza un conjunto no estructurado de atributos. Los más utilizados son los conocidos como jerárquicos integrales. La descomposición jerárquica es una estrategia muy utilizada en diferentes disciplinas científicas.

Estos modelos de calidad, conforman una estructura jerárquica, que puede representarse en forma de árbol *n-ario*. En el nivel más alto de la jerarquía (nivel 0) se encuentra la calidad, que se puede definir a través de una gran variedad de conceptos. Estos se pueden descomponer en sub-conceptos, cada uno de ellos derivados en atributos de calidad que se evalúan a través de un conjunto de medidas. Dichos niveles se nombran según el modelo. La Tabla I.3, resume los niveles presentes en los principales modelos.

Tabla I.3. Niveles de los modelos de calidad.

NIVEL	***McCall*** [27]	***Boehm*** [33]	***FURPS*** [34]	***Dromey*** [30]	***ISO/IEC*** [35]
1	Perspectiva	Características de alto nivel	RF y RNF	Etapas	Características
2	Factor	Características de nivel intermedio	Propiedades del producto	Sub-características	Sub-características
3	Criterio	Características primitivas	(métrica)	(métrica)	(métrica)
4	(métrica)	(métrica)			

La ventaja de los modelos de calidad, es que esta se convierte en algo concreto, que se puede definir, medir y, sobre todo, planificar. Es posible determinar la calidad de manera simple y coherente. Los modelos ayudan a comprender las relaciones existentes entre las diferentes características de un producto de software. Además, brindan criterios e indicadores para la medición de sus fortalezas y debilidades, con el propósito de introducir mejoras. Una desventaja es que aún no ha sido demostrada la

validez absoluta de ninguno de ellos. Las relaciones establecidas entre las diferentes características, atributos y métricas, se derivan de la experiencia, de ahí la existencia de múltiples modelos.

A continuación, se caracterizan los modelos básicos.

Modelo McCall:

El modelo de McCall [27], se originó a solicitud del Departamento de Defensa de los EE.UU., en los años 70. Identifica tres perspectivas: asociados a la operación, revisión y transición. Estas se descomponen en once factores de calidad, asociados a criterios externos. Los factores de calidad son demasiados abstractos para ser medidos directamente por lo que se descomponen en atributos de bajo nivel, denominados criterios de calidad (23 en total). Un nivel más de descomposición es necesario, por lo que se mapea cada criterio de calidad con un conjunto de métricas de calidad.

Debido a su forma de descomponer la calidad del software, es muy utilizado en la actualidad [36]. Destacable es el hecho de que la mayoría de los factores propuestos por McCall, aún continúan definiendo la calidad; las formas de calcular estos criterios son las que han sufrido cambios radicales con el paso de los años. Este modelo, fue propuesto y diseñado para cualquier tipo de sistema. No están explícitamente contemplados en su alcance los atributos de un dominio específico.

La mayor contribución del modelo de McCall, es toda su organización, desde las perspectivas hasta la consideración de las métricas; mientras, la falta de definición de métricas para los factores y criterios, es su desventaja. Los criterios, son aspectos abstractos, que dificultan su medición. Esto provoca que no siempre exista una relación lineal entre los valores de las métricas y las características a estimar.

Es preciso puntualizar, que la mayoría de las organizaciones de software no publican las métricas que han definido para cada uno de los factores y criterios.

Perry, en 1987 [37], definió las interrelaciones entre los factores de calidad expuestos por McCall. Dichas relaciones pueden ser inversa (-), neutral (0) o directa (+). La relación inversa entre factores, representa que cuando hay un alto grado de calidad para un factor, existe un bajo grado de calidad para el otro. La relación directa, significa que cuando hay un alto grado de calidad para un factor, existe un alto grado de calidad para el otro. La neutra, representa que no hay influencias entre los factores.

Estas interdependencias, se muestran en la Figura I.2. Dicho modelo de influencias, se conoce como pirámide de McCall.

Correctitud	C									
Confiabilidad	+	Conf								
Eficiencia	o	o	E							
Integridad	o	o	-	In						
Usabilidad	+	+	-	+	U					
Mantenim	+	+	-	o	+	M				
Testeabilidad	+	+	-	o	+	+	T			
Flexibilidad	+	+	-	o	+	+	+	F		
Portabilidad	o	o	-	o	o	+	+	o	P	
Reusabilidad	o	-	-	-	o	+	+	+	+	R
Interoperab.	o	o	-	-	o	o	o	o	+	o

Figura I.2. Interdependencia entre los 11 factores de calidad expuestos por McCall, según Perry [37].

Modelo de Boehm:

También en los años 70, Boehm define un modelo de calidad para evaluar el software, de manera automática y cuantitativa, centrado en el producto final. Se identifican características de calidad desde el punto de vista del usuario, e incorpora características de rendimiento del hardware. En él, se agregan algunas de las características presentes en el modelo de McCall.

Este modelo representa una estructura jerárquica de características, cada una de las cuales contribuye a la calidad; las descompone en cuatro niveles: alto, intermedio, primitivas y métricas.

Su dificultad radica en no tener bien definidos los niveles [32], haciendo que no queden claros los atributos a controlar. Además, no brinda ningún tipo de sugerencia sobre la medición de las características de calidad [38].

FURPS:

Hewlett Packard, en 1987, desarrolló un conjunto de factores de calidad: funcionalidad, facilidad de uso, fiabilidad o confiabilidad, rendimiento y capacidad de soporte. Por el acrónimo de sus nombres en inglés (*Functionality, Usability, Reliability, Performance y Supportability)* se le conoce como modelo FURPS [32]. Es una adaptación del modelo de McCall, agrupado de manera diferente.

La funcionalidad, se valora mediante la evaluación del conjunto de características y capacidades del software, la generalidad de las funciones entregadas y la seguridad

del sistema global. La facilidad de uso, se estima teniendo en consideración factores humanos como la estética, consistencia y documentación general. La confiabilidad, mide la frecuencia y gravedad de los fallos, la exactitud de las salidas, el tiempo medio entre fallos y la capacidad de recuperación de un fallo, entre otros. El rendimiento, tiene en cuenta la velocidad de procesamiento, el tiempo de respuesta, consumo de recursos, rendimiento efectivo total y la eficacia. La capacidad de soporte, combina la capacidad de ampliar el programa (extensibilidad), la adaptabilidad, capacidad de hacer pruebas, compatibilidad, capacidad de configuración, facilidad de instalación de un sistema y facilidad con que pueden localizarse los problemas.

Se le critica la no existencia de atributos específicos ni la definición de las métricas.

Modelo de Dromey:
Creado por R. Geoff Dromey, en 1996, y derivado también del modelo de McCall [36]. Este modelo se propuso como un marco de trabajo para la aplicación de criterios de calidad durante el desarrollo del producto y el proceso de verificación y validación del software. Se emplea como guía en el desarrollo y centra en hacer corresponder una serie de atributos internos con su implicación en atributos externos, observables y medibles. Es decir, atributos de alto nivel, como la fiabilidad, no pueden aplicarse directamente durante el desarrollo; en lugar de ello, hay que considerar una serie de propiedades cuya resultante sea un comportamiento fiable del software. De esta forma, a diferencia del modelo de McCall, la calidad se construye de abajo hacia arriba, desde las propiedades tangibles de bajo nivel hacia los atributos de calidad de alto nivel [39]. Es una versión reducida de aquel, en el sentido de los factores y criterios que escoge.

Como desventaja, está su complejidad, ya que Dromey para ilustrar su hipótesis, construye modelos de calidad para los requisitos, el diseño e implementación, lo cual dificulta el entendimiento y aplicación de los mismos. Otra desventaja está asociada con atributos de alto nivel, no factibles a evaluar antes que el software sea operacional [40].

ISO 9126:
La norma ISO/IEC 9126, define un marco de representación de los modelos de calidad, siguiendo una estructura jerárquica basada en los modelos anteriores. La calidad se determina mediante la especificación de características generales del software que progresivamente se descomponen en sub-características, compuestas a su vez de atributos, de manera que en la base de la jerarquía estarán atributos del software evaluables mediante algún tipo de métrica.

Se hace una distinción entre calidad interna, externa y en uso (como las perspectivas del modelo de McCall). Los atributos que pueden medirse durante el proceso de desarrollo se consideran internos, mientras los externos son aquellos que se pueden medir durante el proceso de prueba. Por otro lado, la “calidad en uso” es la que percibe el usuario.

El estándar ISO/IEC 9126, está dividido en cuatro partes:

- ISO/IEC 9126-1: Presenta dos modelos de calidad del software: el de la calidad interna y externa, y el de calidad en uso.
- ISO/IEC TR 9126-2: Suministra métricas externas para medir las características de calidad externa y una explicación de cómo aplicar dichas métricas.
- ISO/IEC TR 9126-3: Provee métricas internas para características de calidad interna.
- ISO/IEC TR 9126-4: Define métricas de calidad en uso.

El modelo para la calidad interna y externa categoriza los atributos en seis características: funcionalidad, confiabilidad, usabilidad, eficiencia, mantenibilidad y portabilidad. Estas características a su vez son divididas en sub-características (ver Tabla I.4). Las sub-características pueden medirse a través de métricas internas o externas.

Las métricas externas constituyen una ventaja para usuarios, evaluadores, verificadores, y diseñadores, al permitir evaluar la calidad del producto durante las pruebas u operación.

Los atributos de la calidad durante el uso, se clasifican en cuatro características: eficacia, productividad, seguridad y satisfacción (ver Tabla I.4), similares a las definidas por la ISO 9241 para medir la usabilidad de los productos. En la ISO/IEC TR 9126-4, aparecen ejemplos de las métricas de la calidad durante el uso.

Esta norma, permite especificar y evaluar la calidad del software desde distintas perspectivas, asociadas a la adquisición, requerimientos, desarrollo, uso, evaluación, soporte, mantenimiento, aseguramiento de la calidad, y auditoria. Los criterios de calidad definidos por ella, son aplicables a todo tipo de software. Puede ser usada por diferentes actores: desarrolladores, evaluadores independientes y grupos de aseguramiento de la calidad.

Tabla I.4 Modelos de la ISO 9126.

CALIDAD EXTERNA E INTERNA	
Características	***Sub-características***
Funcionalidad	Adecuación, Exactitud, Seguridad de acceso, Cumplimiento funcional
Fiabilidad	Madurez, Tolerancia a fallos, Capacidad de recuperación, Cumplimiento de fiabilidad
Usabilidad	Capacidad de entender, Capacidad de aprender, Capacidad para operar, Capacidad de atracción, Cumplimiento de usabilidad
Eficiencia	Tiempo de respuesta, Utilización de recursos, Cumplimiento de eficiencia
Mantenibilidad	Capacidad de analizar, Capacidad de cambio, Estabilidad, Capacidad para pruebas
Portabilidad	Adaptabilidad, Instalabilidad, Coexistencia, Capacidad de reemplazo, Cumplimiento de portabilidad

CALIDAD EN USO	
Característica	***Concepto***
Eficacia	Capacidad del producto de software de permitir que los usuarios logren objetivos especificados con precisión e integridad, en un contexto dado.
Productividad	Capacidad del producto de software de permitir que los usuarios dediquen una cantidad de recursos apropiada en relación con la eficacia alcanzada en un contexto de uso dado Entre los recursos, pueden incluirse el tiempo para completar la tarea, los esfuerzos del usuario, los materiales o el costo de utilización en términos financieros
Seguridad	Capacidad del producto de software de alcanzar niveles aceptables de riesgo de daños a las personas, el negocio, el software, la propiedad o el ambiente en un contexto de uso dado.
Satisfacción	Capacidad del producto de software de satisfacer a los usuarios en un contexto de uso dado.

Con este estándar, sin embargo, surgieron determinados problemas prácticos, debido a que algunas de sus características son demasiados abstractas, abiertas a diferentes interpretaciones y a la ambigüedad en las definiciones de las métricas.

Autores como Zulzalil, Pandey y Radulovic, determinan las interdependencias entre las seis características externas e internas, demostrando dependencia inversa (-), directa (+) o neutral (0). ([41], [42], [43] y [44]). Esto provoca que un grupo de criterios puedan influir en la calidad general, de manera contradictoria. (Ver Figura I.3).

La serie original de la ISO / IEC 9126, fue separada en 9126 y 14598, debido a que el modelo de calidad y las métricas son útiles no sólo para la evaluación del producto, sino también para otros propósitos, incluyendo la especificación de requisitos de calidad. La evaluación sólo es posible y significativa cuando los requisitos están claramente especificados.

Funcionalidad (F)	F					
Confiabilidad (R)	+	R				
Usabilidad (U)	+	+	U			
Eficiencia (E)	o	o	-	E		
Mantenibilidad (M)	+	+	+	-	M	
Portabilidad (P)	o	o	o	-	+	P

Figura I.3. Interdependencia en las características de calidad de la ISO 9126 [41].

Una de las desventajas de este modelo es que no muestra claramente cómo se pueden medir las características internas y externas. La especificación de las métricas es insuficiente ([38] y [45]).

ISO 14598:

La Norma ISO 14598, proporciona un marco de trabajo para evaluar la calidad de todo tipo de productos de software, indicando los requisitos a medir y analizar. Describe el proceso de evaluación en general, detallando las actividades y tareas a desarrollar. De cada actividad, explica los propósitos e informaciones de entradas y salidas. Propone el uso de la ISO 9126, como modelo de calidad, y especifica el proceso a seguir por evaluadores, desarrolladores y compradores. Está estructurada en 6 partes:

- ISO/IEC 14598-1: Presenta un resumen de las otras cinco partes. Detalla la relación entre la evaluación del producto de software y el modelo de calidad definido en la 9126.
- ISO/IEC 14598-2: Es una guía para las funciones de soporte, tales como la planificación y gestión de la evaluación del producto del software.
- ISO/IEC 14598-3: Es una guía para realizar evaluaciones de producto cuando este se está desarrollando y evaluando simultáneamente. Se usa también para planear un nuevo producto o mejorar el existente; emplea indicadores que pueden predecir la calidad de los productos finales.
- ISO/IEC 14598-4: Es una guía para la aceptación de determinado producto, también la pueden usar las organizaciones para comprar un software.
- ISO/IEC 14598-5: Es una guía para las organizaciones encargadas de evaluar software.

- ISO/IEC 14598-6: Provee las guías para la documentación del proceso de evaluación.

Se determina que el proceso de evaluación debe cumplir las propiedades siguientes:

- Repetible: cuando al repetir el proceso de evaluación bajo las mismas circunstancias; o sea, la misma configuración de las herramientas utilizadas, el mismo producto y el mismo evaluador, se obtiene igual resultado.
- Reproducible: semejante al anterior, también obtiene el mismo resultado, manteniendo todas las condiciones iguales; la diferencia es que lo hace otro evaluador.
- Imparcial: el resultado no debe estar influenciado por evaluaciones anteriores.
- Objetivo: el evaluador no debe estar influenciado por sentimientos propios o prejuicios sobre el producto o similares.

Mientras se utilizaban las dos series de normas ISO 9126 e ISO 14598, se identificaron problemas causados principalmente por los avances tecnológicos y otras cuestiones adicionales que podían mejorarlas [46]. Además, la existencia de dos series de estándares, complicaba la comprensión y aplicación de estas normas. Por otra parte, las métricas estaban definidas en diferentes normas que era preciso buscar y relacionar. Por todo lo anterior, se decide unir todo en una arquitectura diferente, dando origen a una nueva familia de estándar, la ISO 25000 (proyecto SQuaRE-*System and Software Quality Requirements and Evaluation*).

Familia ISO 25000:
La familia ISO 25000, detalla la arquitectura SQuaRE y cuenta con las siguientes divisiones:

- ISO/IEC 2500n, División de gestión de la calidad,
- ISO/IEC 2501n, División de modelo de calidad,
- ISO/IEC 2502n, División de medición de calidad,
- ISO/IEC 2503n, División de los requisitos de calidad,
- ISO/IEC 2504n, División de evaluación de la calidad.

Las ISO/IEC 25050 e ISO/IEC 25099 son reservadas para ser usadas para extensión del estándar. A continuación, se analiza brevemente cada división.

La **ISO/IEC 25000: 2014** proporciona una guía para el uso de la nueva serie de estándares internacionales SQuaRE. Además, aporta una visión general de los contenidos de SQuaRE, modelos de referencia común y definiciones, permitiendo a los usuarios una guía para comprender correctamente esta serie. También contiene

una explicación del proceso de transición entre las antiguas normas (ISO/IEC 9126 e ISO/IEC 14598) y el proyecto SQuaRE. Su objetivo principal es guiar el desarrollo de los productos de software, mediante la especificación de los requisitos y la evaluación de las características de calidad.

La **ISO/IEC 25010: 2011** define tres modelos de calidad: modelo de calidad en uso, modelo de calidad del producto y modelo de calidad de los datos.

El modelo de calidad en uso está compuesto por cinco características: eficacia, eficiencia, satisfacción, ausencia de riesgo y cobertura de contexto (algunas de las cuales se subdividen en sub-características). Dichas características se relacionan con el resultado de la interacción de un producto en el contexto en que se usa. (Ver Tabla I.5)

Tabla I.5. Características de calidad en uso, según la ISO 25010.

Características	**Resumen**
eficacia	Precisión e integridad con la cual los usuarios logran los objetivos especificados
eficiencia	Recursos gastados en relación con la exactitud e integridad con la que los usuarios logran las metas especificadas.
satisfacción	Grado en el que las necesidades del usuario están satisfechas cuando un producto o sistema se utiliza en un contexto de uso. Se descompone en: utilidad, confianza, placer y comodidad.
ausencia de riesgo	Grado en que un producto o sistema mitiga el riesgo potencial de la situación económica, la vida humana, la salud o el medio ambiente. Se descompone en mitigación del riesgo: económico, de la salud y para la seguridad, y ambientales.
cobertura de contexto	Grado en que un producto o sistema se puede utilizar con eficacia, eficiencia, satisfacción, con ausencia de riesgo en contextos más allá del inicialmente especificado en los requisitos. Se descompone en exhaustividad del contexto y flexibilidad.

El modelo de la calidad del producto está compuesto por ocho características: adecuación funcional, fiabilidad, eficiencia en el rendimiento, facilidad de uso, seguridad, compatibilidad, facilidad de mantenimiento y portabilidad, las cuales se subdividen, a su vez, en sub-características, relacionadas con propiedades estáticas y dinámicas del producto (Ver Tabla I.6).

Se adiciona con la ISO/IEC 25012 un modelo para la calidad de los datos.

El ámbito de aplicación de los modelos de calidad, incluye el apoyo a la especificación y evaluación de software desde diferentes perspectivas: de los asociados a su adquisición, a los requisitos, al desarrollo, uso, evaluación, soporte, mantenimiento, aseguramiento y control de calidad y auditoría.

Tabla I.6. Modelo de calidad del producto.

Características	**Sub-características**
Adecuación funcional	Completitud funcional, corrección funcional, pertinencia funcional
Confiabilidad	Madurez, disponibilidad, tolerancia a fallos, capacidad de recuperación
Eficiencia en el rendimiento	Comportamiento de tiempo, utilización de recursos, capacidad
Facilidad de uso	Inteligibilidad, aprendizaje, operabilidad, protección frente a errores de usuario, estética de interfaz de usuario, accesibilidad
Seguridad	Confidencialidad, integridad, no repudio, responsabilidad, autenticidad
Compatibilidad	Coexistencia, Interoperabilidad
Facilidad de mantenimiento	Modularidad, reusabilidad, analizabilidad, capacidad de ser modificado, capacidad de ser probado
Portabilidad	Adaptabilidad, facilidad de instalación, capacidad de ser reemplazado

La **ISO/IEC 25020: 2007** proporciona una guía para la medición de las características de calidad definidas en la 25010, y un modelo de referencia. Esta división contiene las listas de métricas recomendadas, los requisitos generales y guías para el uso de estas. Además, establece los requisitos para la selección y construcción de medidas de calidad.

La Figura I.4, muestra la relación entre esta norma internacional y los informes técnicos de la División de medición de la calidad. Los elementos de medida de la calidad, evalúan una representación estática del software, su comportamiento y efectos cuando se utiliza. Las medidas presentes en estos informes técnicos no deben interpretarse como un conjunto exhaustivo o requerido. Los desarrolladores, evaluadores, gerentes de calidad, compradores, proveedores, mantenedores y otros usuarios de la evaluación, pueden seleccionar las medidas de estos informes, modificarlas o incluir nuevas medidas.

La **ISO/IEC 25030: 2007** proporciona los requisitos de calidad y una serie de recomendaciones para la especificación de los mismos. Toma una perspectiva de sistema, debido a que el software normalmente se desarrolla y se aplica como parte de un sistema más grande.

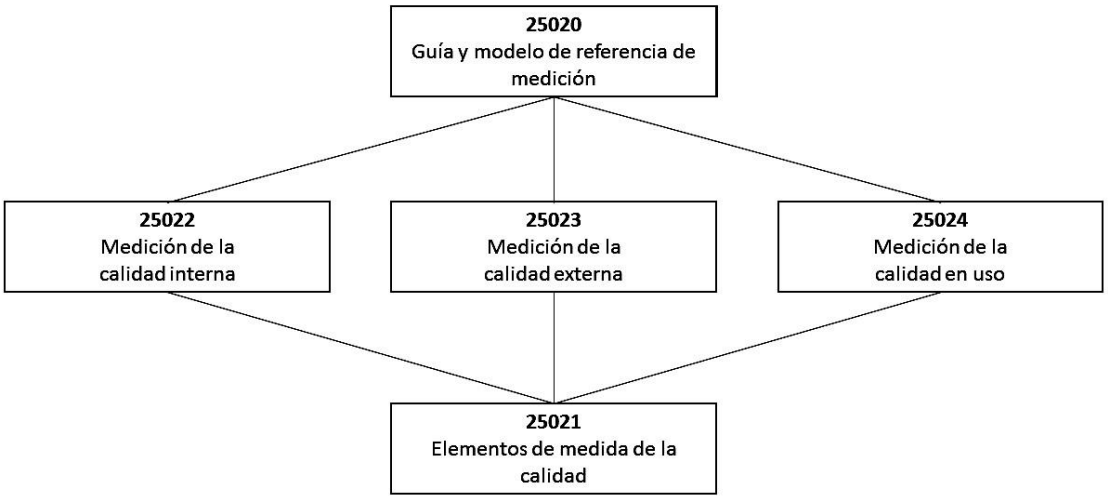

Figura I.4. Norma internacional ISO 2502n [47]

La **ISO/IEC 25040: 2011** proporciona una descripción del proceso de evaluación de la calidad del producto de software, que consta de cinco actividades y 15 tareas, y establece los requisitos para la aplicación de este. Está destinada para los responsables de la evaluación, siendo apropiada para los desarrolladores, compradores y evaluadores independientes. Establece la relación entre el modelo de referencia de la evaluación y los documentos SQuaRE. La ISO/IEC 25040 es una versión revisada que sustituye a la ISO/IEC 14598-1.

Como puede observarse, la familia ISO 25000 es en la actualidad la norma más general y completa para este proceso de evaluación.

Al concluir el análisis de los modelos y estándares de calidad, se considera que:

- Existe gran similitud entre los modelos de calidad, lo que permite modelar una estructura general para representarlos.
- Para juzgar la calidad por razones cognitivas y prácticas, las características de calidad deben ser igual o menor que 9. Es insuficiente tener un solo nivel, motivo por el cual se definen sub-características.
- La mayoría se derivan del modelo de McCall; los criterios son semejantes, lo que varía es la estructura de organización de los mismos.
- Brindan gran variedad de características, sub-características y medidas de calidad, lo que dificulta su selección.
- Las medidas se definen en diferentes dominios de datos y escalas.
- La estructura más usada es la jerárquica; sin embargo, se ha demostrado la interdependencia entre los criterios, lo que conlleva a estructuras más complejas, como los grafos.

- No hay consenso en las métricas a utilizar. Las empresas u organizaciones que valoran la calidad, consideran que la información sobre las métricas y fórmulas de cálculo, se encuentran dentro de su *know how* y no publican los datos referidos a estas. La división ISO 2502n, hace recomendaciones de métricas, pero no se pueden definir como exhaustivas y obligatorias. La mayoría son cuantitativas.
- Constituye un reto lograr medir la calidad con criterios y métricas que cumplan propiedades de completitud y descomposición, sin redundancia.

Se hace necesario ahora analizar cómo dichos modelos y estándares influyen en el proceso de evaluación de la calidad de los productos de software.

I.4 Evaluación de la calidad de los productos de software

Es importante analizar primero el concepto de evaluación. Según la Real Academia [11], evaluación es la acción de estimar, apreciar, calcular o señalar el valor de algo. Al evaluar se emite un juicio o valoración sobre algo, teniendo como base la información del objeto a valorar (evidencias, mediciones) y un conjunto de criterios como patrón. Con el resultado de la evaluación, se toman decisiones y se conduce hacia la mejora continua. Evaluar conlleva una valoración subjetiva (cualificación) y objetiva (cuantificación).

En el caso de la industria de software, suele asociarse de manera errónea la evaluación de la calidad con las pruebas de software. Evaluar es más que probar, es realizar la valoración de un producto a partir de los resultados obtenidos en el proceso de Verificación y Validación (V&V); se analizan los resultados y se continúa con la toma de decisión.

La evaluación de la calidad del producto de software tiene como propósito emitir un juicio de cuán bueno puede ser este para una función determinada. Permite seleccionar entre dos o más alternativas; predecir los valores de las características de calidad y conceptualizar un nuevo producto.

Diferentes autores ([48] y [49]), según el objetivo de la evaluación, proponen diversos tipos de evaluación como:

- La *opción,* cuando se determina en dos subconjuntos: el de los softwares mejores y el de los no mejores.
- La *clasificación,* cuando se obtiene un número de subconjuntos previamente definidos; por ejemplo, softwares buenos, regulares, malos.
- El *ordenamiento,* cuando se obtiene un ranking de los productos.
- La *descripción* cuando se obtiene una descripción formal de cada elección, sin calificación.

- La *conceptualización,* cuando se obtiene una clasificación en productos ideales o cuasi ideales no disponibles en el momento.

Como resultado del proceso de evaluación, se obtiene un valor global de la calidad (cuantificación) y una valoración (cualificación), que permite aislar los problemas que pudiera presentar el producto. Dicho proceso es esencial para la toma de decisiones por las empresas productoras, evaluadoras y las que adquieren el software.

La evaluación, como cualquier proceso, posee un conjunto de actividades interrelacionadas, que transforman elementos de entrada en salidas. En este caso, tiene como entrada los productos, que se conducen por un flujo de actividades, entre las que se incluyen la medición y el análisis, y obtiene como salida el resultado de la valoración de la calidad.

Para evaluar y juzgar la calidad, se necesita determinar los criterios que caracterizan la calidad del producto. En el software, dichos criterios se encuentran estructurados y organizados en los modelos de calidad. Además, están los estándares ISO 14598 y 25040, que ofrecen un *framework* para la evaluación, que detalla el proceso a través de un flujo de actividades y tareas.

El modelo de calidad, representa los criterios a tener en cuenta en el proceso de evaluación, organizados de forma jerárquica. Asociado a estos, se encuentran las medidas que permiten valorarlos. Las medidas provienen de diferentes fuentes: pruebas automáticas, encuestas a expertos, listado de No Conformidades (NC), entre otras.

Azuma, propone una definición formal de la evaluación de la calidad. Se representa el proceso de evaluación de software con la sigla QES y lo describe como la composición de procesos (QES_P) y recursos (QES_R). El QES no es la mera unión de los procesos y recursos, sino la integración de ellos [50]

$$QES = \{QES\_P, QES\_R\} \qquad \text{(I.1)}$$

En la evaluación se involucran recursos (QES_R) como:

$$QES\_R = \{\text{Modelo de calidad, Métricas, Herramientas de medición, Técnicas de evaluación, Herramientas de gestión de datos, Evaluadores, Computadoras}\}$$

Los procesos (QES_P), a su vez, los divide en dos subprocesos: soporte (QES_SP) y evaluación (QES_EP) y finalmente los subprocesos los descompone en tareas.

$$QESP = \{QES\_SP, QES\_EP\} \qquad \text{(I.2)}$$

$$QES\_EP = \{\text{Análisis de requisitos de evaluación, Planificación y ejecución, Medición y calificación, Evaluación total, Gestión de la evaluación}\}$$

$$QES\_SP = \{\text{Desarrollo de tecnologías, Transferencia de tecnología, Evaluación de tecnologías, Gestión de datos (experiencia) y tecnologías, Estandarización , Gestión de apoyo a la evaluación}\}$$

Además, determina las propiedades que el QES debe satisfacer. Estas son:

- Repetitivo, cuando la evaluación repetida en el mismo producto, con iguales condiciones, arroja el mismo resultado.
- Objetivo, cuando se repite la evaluación de un producto por diferentes personas o equipos, en iguales condiciones, debe arrojar el mismo resultado.
- Cuantitativo, cuando la medición y la presentación de los resultados es cuantitativa.
- Indicativo, cuando se presenta alguna discrepancia o problema en la evaluación, se indican las causas y acciones requeridas.
- Económico, cuando el costo requerido para la evaluación debe ser relativamente pequeño; es decir, rentable. Se precisa evaluar en función de la prioridad.
- Inclusivo, cuando la evaluación debe abarcar todas las características de calidad.

Esta primera definición formal, fue la que impulsó la estandarización de los modelos de calidad y el flujo de trabajo logrado por las normas ISO, y aunque después evolucionaron, han mantenido sus ideas iniciales básicas.

Todo lo anterior, se resume en la Figura I.5, que presenta las partes fundamentales de un proceso de evaluación de la calidad de un producto de software, a partir de la formalización anterior y las normas de calidad.

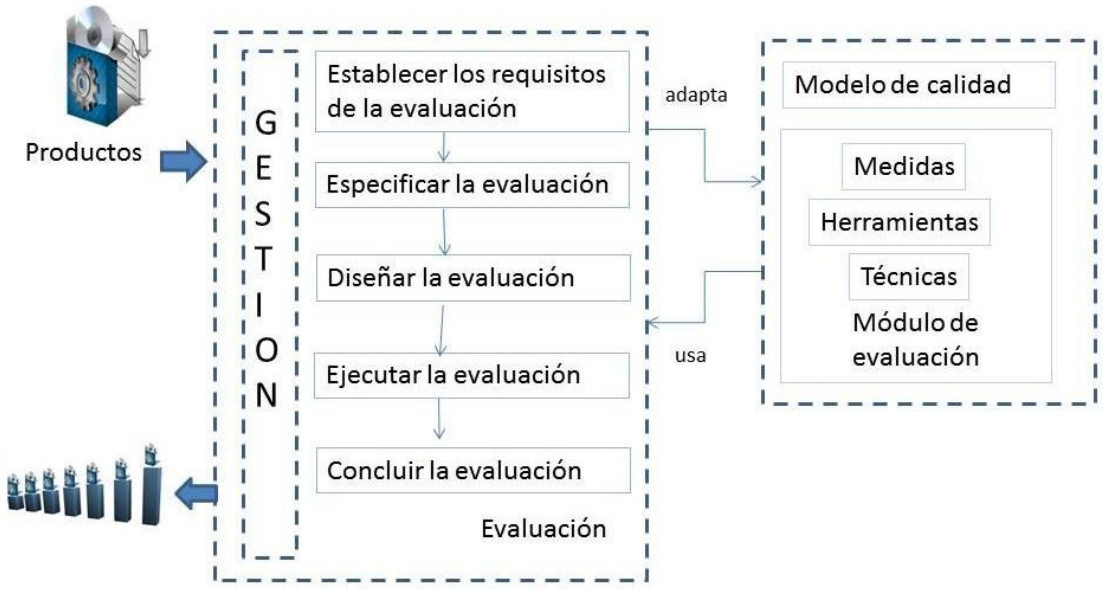

Figura I.5 Proceso de evaluación de la calidad de un producto de software.

El proceso lo ejecutan los evaluadores, involucrando a los clientes, y se realiza sobre la base de la ISO 25040. Tiene como entrada los productos a evaluar y como salida el índice de calidad de cada uno, ordenado de manera descendente. Para la evaluación, se necesitan recursos como el modelo de calidad, y el módulo de evaluación, consistente en las herramientas, técnicas y medidas de los atributos de calidad.

Según el objetivo y rigor de la evaluación, se determina el modelo de calidad y módulo a utilizar. Estos recursos no son estáticos y pueden adaptarse según el tipo de evaluación, producto, presupuesto, etc. Es esencial la valoración de los diferentes criterios, definiendo su importancia o peso, la naturaleza de los datos y el umbral a partir del cual deben tomarse las decisiones.

Requiere un análisis especial el proceso de agregación de la información, que se hace complejo a partir del objetivo a lograr y los datos a operar. Al concluir la evaluación, es importante analizar los resultados con el objetivo de realizar una mejor interpretación de la evaluación para la toma de decisiones.

Es substancial tener en cuenta los subprocesos de soporte y gestión de la evaluación, encargados de gestionar los recursos y controlar los desvíos del cronograma de evaluación. También son importantes los que gestionan la tecnología y calidad.

Este es un proceso complejo y costoso, que implica el uso de gran cantidad de recursos, tanto materiales como humanos. Por esa razón, es preciso lograr una relación entre el resultado de la evaluación y la calidad mostrada por el producto en explotación. De ahí la importancia de la fiabilidad de los resultados, sobre todo cuando conlleva a la certificación de un producto de software.

Su complejidad viene aparejada a la variedad de medidas involucradas y definidas en diferentes dominios de datos y escalas, que valoran factores objetivos y subjetivos. Por tal razón, puede afirmarse que los datos manipulados en el proceso de evaluación son heterogéneos. Las medidas se obtienen a partir de las valoraciones de tiempo, tamaño, cantidad y apreciaciones establecidas por los expertos y probadores, a partir de la observación y percepción. Los expertos suelen tener diferentes criterios y conocimientos acerca del proceso, lo que introduce un alto grado de incertidumbre. La incertidumbre, está caracterizada por la existencia de información vaga, incompleta, conceptos imprecisos y presencia definitoria del juicio humano en la decisión.

Si comparamos este proceso con el de toma de decisiones (TD), pueden observarse actividades en común (Ver Figura I.6). En la primera parte de la estructuración del problema, en el proceso de TD, se determinan los criterios y alternativas, semejante a como se realiza en las dos primeras actividades de la evaluación, donde establece el modelo de calidad y productos a evaluar. En ambos, se evalúan las alternativas y se concluye eligiendo una opción o representando un ranking.

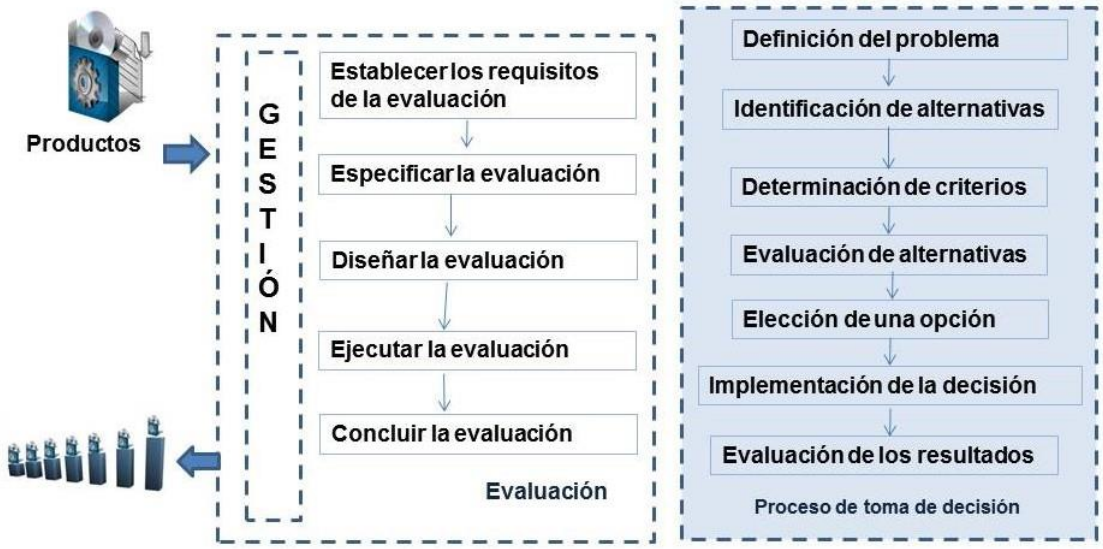

Figura I.6 Comparación del proceso de evaluación de la calidad de un producto de software con el proceso de toma de decisiones.

De lo anterior, puede concluirse que la evaluación de la calidad de los productos de software, pueden modelarse como un problema de toma de decisiones. No obstante, deben tenerse en cuenta las particularidades detalladas a continuación.

Primera: en el caso de los criterios de evaluación, estos se organizan jerárquicamente en el modelo de calidad. En el análisis realizado en la Sección I.3, sobre los modelos y estándares de calidad, se concluyó que además de la relación vertical presente entre los criterios, existe un enlace horizontal de influencia entre ellos. Por ejemplo, una

mayor eficiencia implica más baja usabilidad; un menor comportamiento temporal implica mayor uso de recursos como la memoria; una mejor confiabilidad implica mejor funcionalidad. Esto adiciona complejidad al proceso de evaluación y a la forma de agregación de la información.

Segunda: según Dasso [51] y la norma ISO 25040, debe valorarse un subconjunto de criterios denominados esenciales. Los criterios esenciales, son los que describen aspectos que el software bajo evaluación debe satisfacer y cuya ausencia es preciso penalizar de manera que todo el subárbol se evalúe a cero. Para ello, se definen umbrales relacionados con los requisitos de calidad y los criterios de evaluación para decidir la calidad del producto. Dichos umbrales, pueden establecerse a partir de la comparación de estos aspectos con los productos líderes del mercado (*benchmarking*), límites de control estadísticos, datos históricos, requisitos del cliente, entre otros. Si no se cumple el umbral definido para el criterio, el software no se selecciona o penaliza. Tal requisito vuelve más complejo el método de agregación de la información.

Tercera: los datos obtenidos de la evaluación del producto, están definidos en diferentes dominios y con diversas escalas, lo que implica la agregación de datos heterogéneos.

Cuarta: el proceso de agregación debe ser independiente de los datos y lograr siempre el mismo índice de calidad para un producto, sin importar la cantidad de alternativas a considerar.

A continuación, se analiza la evaluación y selección de productos de software como un proceso de toma de decisiones, determinando si tienen en cuenta o no las particulares anteriores.

I.5 Proceso de toma de decisiones.

La toma de decisiones (TD), es una actividad cotidiana de la vida humana. A diario, las personas se encuentran en situaciones en las que deben decidir entre alternativas u opciones y seleccionar las que cumplan el mayor número de condiciones preestablecidas. Esta acción, es primordial en muchas esferas de la vida y la actividad industrial. La industria del software no es una excepción. Como pudo apreciarse en la Figura I.6, el proceso de evaluación de la calidad de los productos de software, se comporta semejante a un proceso de toma de decisiones.

Un individuo, o grupo de individuos, para tomar decisiones frente a un conflicto o problema, se plantean un conjunto bien definido de alternativas. Además, se establecen un conjunto de criterios para valorarlas, un conjunto de preferencias en

cuanto a los criterios y se selecciona una alternativa o se establece un orden entre estas.

Todo proceso de decisión, transcurre en un contexto que se denomina ambiente o entorno. En este proceso, participan varios elementos: decisores, alternativas, criterios de decisiones y pesos asociados a dichos criterios. Los decisores, son el conjunto de individuos que tienen la responsabilidad de tomar la decisión. Por su parte, los criterios de decisión son los parámetros utilizados para valorar las preferencias del decisor en cuanto a las alternativas; son elementos de referencia en base a los cuales se realiza la decisión. Los pesos o ponderaciones son las medidas de la importancia relativa que los criterios tienen para el decisor. Se definen alternativas como el conjunto finito de soluciones, acciones, opciones posibles a analizar durante el proceso de toma de decisiones que se considere.

El proceso de toma de decisiones se puede clasificar de diversas formas:

- Según la cantidad de criterios: cuando en el problema se tiene en cuenta solo un criterio para el análisis de las alternativas, se trata de toma de decisión de único criterio. Cuando intervienen varios criterios, se denomina toma de decisión multicriterio.
- Según la cantidad de expertos: cuando en el proceso participa un único experto o decisor, se nombra toma de decisión individual. Cuando intervienen varios expertos o decisores, se denomina toma de decisión en grupo.
- Según el entorno de toma de decisiones: se clasifican en decisiones bajo certeza, incertidumbre o riesgo. En las decisiones bajo certeza, los decisores conocen con absoluta seguridad el valor de utilidad de cada alternativa. Las decisiones bajo incertidumbre, son las que ocurren cuando el decisor se enfrenta a situaciones que nunca han ocurrido y tal vez no vuelvan a repetirse en el futuro de esta misma forma; no se conocen las probabilidades de ocurrencia de los criterios y la utilidad de cada una de las alternativas se determina de forma aproximada. Las decisiones bajo riesgos, son aquellas en las que existe un factor probabilístico de ocurrencia ligado a los criterios.

El proceso de toma de decisiones está formado por cuatro fases: estructuración del problema de decisión (definición del problema, identificación de alternativas, determinación de criterios), análisis del problema (evaluación de alternativas y elección de una opción), implementación de la decisión y análisis de los resultados. (Ver Figura I.6).

Durante la definición del problema, se determina la problemática a decidir, el objetivo y la meta de la toma de decisión; se detecta la cantidad de opciones para efectuar la decisión, y se determinan las características más relevantes consideradas por los decisores para luego evaluar en función de ellas la alternativa más conveniente.

Después de estructurar el problema, transcurre la fase de análisis de este. En función de la evaluación de las alternativas, se obtiene una con mejor perspectiva que las demás.

Al tener la alternativa más favorable, se procede a su ejecución o implementación. Después, estando ya en ejecución la alternativa seleccionada, se analizan los resultados generados para evaluar y retroalimentar el proceso de toma de decisiones.

La evaluación de la calidad de los productos de software, se modela en la investigación como un problema de toma de decisión multicriterio, en grupo y en un ambiente de incertidumbre. Se destacan los siguientes elementos que permiten tratarla así:

- Se tienen en cuenta varios criterios, estructurados en un modelo, lo que permite la clasificación de la calidad de los diferentes productos y la selección de uno entre un grupo.
- Se manipula información de diferente naturaleza, tanto objetiva como subjetiva.
- Se maneja información que puede ser considerada vaga e imprecisa, a partir de valoraciones de conceptos y de la experiencia de los decisores.
- Es un proceso en el que intervienen varias personas, por lo que se presenta la incertidumbre en diferentes grados y niveles.
- Se determina la calidad del software, a partir de los datos heterogéneos recopilados en el proceso de V&V, que caracterizan los criterios de evaluación.

Con el objetivo de tratar la incertidumbre y heterogeneidad de los datos, se combinan estos métodos con los conjuntos difusos.

Las soluciones de los problemas de evaluación de la calidad del software, en los últimos años van dirigidas a métodos de decisión multicriterio ([38], [52], [84], [87], [91] y [92]). De forma general, todos estos métodos parten de un conjunto de alternativas que son evaluadas para un conjunto de criterios, y a través de una operatoria dada, establecen un orden entre las alternativas y orientan al decisor sobre cuál o cuáles son las mejores.

En el caso de la evaluación de la calidad de software, con la descripción del problema y la evaluación de cada alternativa, se forma la matriz de evaluación. Dicha matriz, se

compone por las alternativas, criterios y la valoración de cada criterio para cada alternativa. (Figura I.7)

Alternativas $A = \{A_1, A_2, ...,A_m\}$ donde A_i (i=1,...,m)

Criterios de decisión $C = \{c_1, c_2, ...,c_n\}$ donde c_j (j= 1,...,n)

Vector de pesos $W= \{w_1, w_2, ...,w_n\}$

	w_1	w_2		w_n
	C_1	C_2	...	C_n
A_1	x_{11}	x_{12}	...	x_{1n}
A_2	x_{21}	x_{22}	...	x_{2n}
A_m	x_{m1}	x_{m2}	...	x_{mn}

	IQ	Ranking
A_1	0,95	1
A_2	0,60	3
...	...	...
A_m	0,75	2

Figura I.7 Método de toma de decisión multicriterio.

Los diferentes métodos de solución, actúan sobre la matriz para valorar las alternativas, y de esta valoración se devuelve un valor asociado a cada alternativa, por el cual son ordenadas finalmente. Los criterios, representan beneficios (el objetivo es maximizar) o costos (el objetivo es minimizar).

Sin embargo, a la hora de valorar la efectividad de estos métodos en el proceso de evaluación de la calidad de un producto de software, deben tenerse en cuenta los elementos descritos en el Epígrafe I.4, con las siguientes especificaciones: (1), se estima una gran variedad de medidas definidas en diferentes dominios de datos, con diversas escalas, los cuales valoran factores de índole objetiva y subjetiva, conllevando a datos heterogéneos e imprecisos; (2) deben permitir la agregación parcial de los datos cuando sea necesario penalizar la evaluación por los criterios esenciales; (3) la solución debe ser independiente de los datos; (4) el valor obtenido para cada alternativa, debe encontrarse en un rango que sea posible interpretar y clasificar fácilmente; y (5) se ha demostrado en los modelos de calidad la existencia de interdependencia entre los criterios de un mismo nivel de la jerarquía, lo cual lleva a una estructura más compleja, que obliga a valorar la interacción horizontal de los criterios en cada nivel.

A continuación, se analiza cómo cada método aplicado al problema de evaluación de la calidad de los productos de software, cubre dichos elementos.

Los MCDM, para su análisis, pueden agruparse según el concepto que siguen para solucionar el problema de toma de decisiones, en métodos de la escuela americana, escuela europea, y otros.

Los métodos de la escuela americana, se caracterizan por maximizar una función de utilidad para obtener la agregación de la información de los diferentes criterios. Entre ellos, se encuentran la suma ponderada, el Proceso Analítico Jerárquico (*Analytic Hierarchy Process* AHP), el Proceso Analítico en Red (*Analytic Network Process*), entre otros.

La escuela europea, utiliza el método de sobreclasificación o relaciones de superación entre alternativas, basado en comparar entre sí las diferentes alternativas a partir de cada criterio y después agregar dicha información, considerando la fuerza de las evidencias a favor y en contra de una selección respecto a otra. Los métodos más conocidos son los de la familia ELECTRE (*Elimination et Choix Traduisant la Réalité / ELimination Et Choix TRaduisant la realitE*) y PROMETHEE (*Preference Ranking Organisation Methods for Enrichment Evaluations*).

Otros métodos utilizan concepciones diferentes. Por ejemplo, el TOPSIS (*Technique for Order Preference by Similarity to Ideal Solution*), calcula la distancia a las soluciones ideal y no ideal, para ordenar alternativas.

Entre los MCDM más empleados en soluciones para la evaluación de la calidad, se encuentran: la suma ponderada ([53], [54], [55], [56], [57] y [58]), el AHP ([48], [59], [60], [61], [62], [52], [63], [64], [65] y [66]), el Proceso Analítico en Red (Analytic Network Process ANP) ([67] y [68]), el método ELECTRE ([69] y [48]) y TOPSIS ([70], [71] y [72].). Otros trabajos que definen operadores de agregación ([73], [74] y [75]).

Suma Ponderada.

Es la manera más rápida y sencilla para identificar la mejor alternativa en un método de decisión. Los datos de partida son los que expresan el grado de cumplimiento de cada uno de los criterios para cada alternativa. Si los valores de los criterios están expresados en escalas distintas, es necesario normalizar la información para que puedan compararse y no produzcan sesgos. Una vez normalizada la información, se determinan los pesos asociados a cada uno de los criterios considerados. Dicho método, asume que la función de utilidad buscada es un modelo aditivo que se presenta de la forma:

$$v = w_1 x_1 + w_2 x_2 + \cdots + w_n x_n \qquad \text{(I.3)}$$

Donde: w_n *es el peso del criterio* n y x_n *es el valor de la alternativa en el criterio* n.

Con el valor promedio ponderado obtenido por la función de utilidad para cada alternativa, se ordenan las mismas. Con este orden, se resuelve el problema de toma

de decisiones y se determina la mejor alternativa entre las posibles, que será la de suma ponderada mayor/menor.

Los diferentes trabajos que utilizan esta forma de agregación para el caso difuso, utilizan el siguiente procedimiento: se modela cada medida en un número difuso, algunos definen el universo del discurso de los valores de las medidas en el rango [0,1]; para obtener el valor del índice calidad, multiplican dos números difusos (el peso y el valor del criterio para cada alternativa) y después agrega el valor ponderado de cada alternativa, calculando el máximo o sumando los números difusos obtenidos de la multiplicación. Para ordenar las alternativas se defuzifica.

Proceso de Análisis Jerárquico (AHP)

El Proceso de Análisis Jerárquico propuesto por Saaty, en 1980 [76], se basa en la idea de que la complejidad inherente a un problema de toma de decisión con criterios múltiples, se puede resolver mediante la jerarquía de los criterios. Permite la selección de alternativas en función de un grupo de criterios y estructura el problema de decisión complejo como una jerarquía de objetivo, criterios y alternativas; se realiza comparaciones por pares de elementos del mismo nivel de la jerarquía con respecto a cada criterio del nivel superior, y finalmente de manera vertical se sintetizan los juicios sobre los diferentes niveles. Su concepción de trabajo, es la descomposición del problema, la comparación por pares según su importancia relativa, para generar así el vector de prioridad.

Los diferentes trabajos que usan este método asociado con conjuntos difusos en la evaluación de la calidad, realizan la comparación por pares con números difusos. Cuando obtienen la matriz difusa, la defuzifican por un determinado método y termina aplicando el AHP como crisp.

Su aplicación en el proceso de evaluación de la calidad, es ventajosa porque los criterios están organizados de manera jerárquica en un modelo de calidad. Como desventaja tiene que exige la independencia entre criterios, elemento que no ocurre en los modelos de calidad.

Proceso Analítico en Red (Analytic Network Process) ANP.

El Proceso Analítico en Red, desarrollado por Saaty, en 1996, es una generalización del AHP. Permite modelar problemas más complejos, donde se hace necesario contemplar la interdependencia entre los criterios de decisión.

Se establece una estructura de grafo formada por clusters de los criterios y su relación con las alternativas. Se obtiene la supermatriz y esta se normaliza, convirtiéndola en

una matriz estocástica por fila o columna. Le halla el límite hasta llegar a un estado estacionario. El valor obtenido para cada alternativa, se ordena.

Su aplicación en el proceso de evaluación de la calidad, es ventajosa porque valora la interdependencia entre los criterios. Tiene como desventaja que, en su comparación por pares, genera un gran volumen de preguntas, en ocasiones incomprensibles para los expertos. Además, es un método complejo, cuya aplicación se torna engorrosa, difícil y poco intuitiva.

Método TOPSIS

El método TOPSIS, desarrollado por Hwang y Yoon, en 1981 [77], identifica las soluciones dentro de un conjunto finito de alternativas y busca la solución a partir de la definición de un ideal positivo (IP) y otro negativo (IN). El principio básico es encontrar la alternativa que tenga la menor distancia al IP y la mayor distancia al IN. Para ello, define un índice de similitud que se construye combinando la proximidad a la IP y la lejanía a la IN. La ordenación por preferencia de las soluciones, surge de colocar las alternativas en orden decreciente respecto al índice de similitud construido. El valor más alto, representa aquella alternativa más cercana del IP y más lejana del IN.

La Figura I.8, muestra los pasos de TOPSIS.

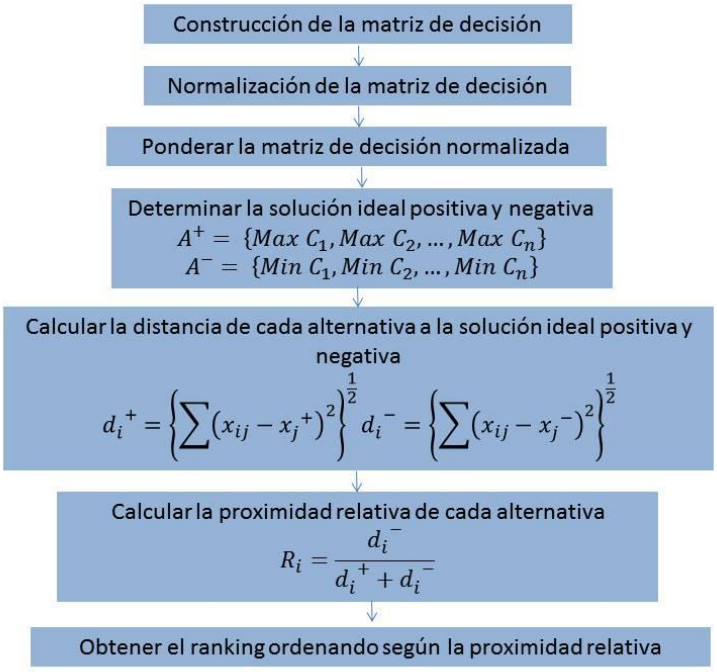

Figura I.8. Pasos del TOPSIS.

La ventaja de este método es que es muy explicativo para los clientes, por la manera de obtener el ranking. Su desventaja es que predetermina la independencia entre criterios.

Familia de métodos ELECTRE

El método ELECTRE, fue propuesto por Benayoun, Roy, en 1968 [78]. Pertenece a una familia de métodos basados en relaciones de superación o sobreclasificación para decidir acerca de la determinación de una solución que, sin ser óptima, pueda considerarse satisfactoria. Además, obtiene una jerarquización de las acciones o alternativas bajo análisis.

Su principio básico consiste en dividir las alternativas en dos subconjuntos, las más y las menos favorables, utilizando una relación de sobreclasificación, mediante la comparación por pares, calculando para cada par de alternativas el índice de concordancia y discordancia, de forma simultánea; determina la matriz de dominancia concordante y discordante, la matriz de dominancia agregada, e interpretan sus valores. De esta forma, se conforma el núcleo del conjunto de alternativas

Han sido desarrolladas las variantes ELECTRE I, II, III, IV, IS; y ELECTRE TRI, que brindan los procedimientos para resolver diferentes tipos de problemas suscitados en el tratamiento de la teoría de la decisión.

Su desventaja radica en que predetermina la independencia entre criterios y es dependiente de los datos.

Resumen comparativo de los métodos de toma de decisión multicriterio

A continuación, se realiza un análisis comparativo de los métodos estudiados, tomando como indicadores los elementos que aportan criterios para el problema abordado en la presente investigación:

1. Punto de partida: información inicial para realizar la evaluación de las alternativas.
2. Tratamiento de la incertidumbre: si da tratamiento a la incertidumbre.
3. Datos heterogéneos: si se ha implementado para usar datos heterogéneos.
4. Interdependencia entre criterios: si tiene en cuenta la interdependencia entre criterios
5. Permite la agregación parcial de los elementos: si tiene en cuenta restricciones de los criterios para permitir la agregación parcial.
6. Independiente de los datos: si obtiene el mismo índice de agregación para una alternativa cuando se evalúe ella sola o cuando se hace dentro de un grupo.

7. Aplicación del método a la evaluación de la calidad: referencias de artículos que aplican estos métodos a los problemas de calidad.

La Tabla I.7, resume el comportamiento de los indicadores en los métodos estudiados.

Tabla I.7. Resumen de los MCDM.

	Suma ponderada	**AHP**	**ANP**	**TOPSIS**	**ELECTRE**
1	Valores de las medidas de calidad	Comparaciones de las alternativas		Valores de las medidas de calidad	Valores de las medidas de calidad
2	Si	Si	Si	Si	No
3	Si, unificando	No	No	Si normalizando	No
4	No	No	Si	No	No
5	No	No	No	No	No
6	No por el proceso de normalización	No	No	No	No
7	[53], [54], [55], [56], [57] y [58]	[48], [59], [60], [61], [62], [52], [63], [64], [65] y [66]	[67] y [68]	[70], [71] y [72]	[69] y [48]

Del análisis de la tabla anterior, se destaca que para la toma de decisión no se permite la agregación parcial; la mayoría no permite tratar la interdependencia entre criterios. Esto dificulta la aplicación de dichos métodos en determinados contextos.

A continuación, se analiza la evolución de las soluciones a los problemas de evaluación y selección de software, haciendo énfasis en las que utilizan técnicas de *Soft Computing.*

I.6 Problemas en la evaluación de la calidad de los productos de software

En 1993, aparecen las primeras soluciones para la evaluación de la calidad de los productos de software, proponiendo métodos, metodologías y modelos. Una de ellas, definía el nivel requerido de cada característica, lo medía en el software y comparaba por medio del porciento [79]. Para el apoyo a la toma de decisiones, dichos valores se representaban en un gráfico circular o de pastel.

Luego, aparecen varios trabajos ([48], [49], [59], [69], [60], [61] y [80]) que presentan una solución al problema a partir de métodos de toma de decisión multicriterio, los cuales ya fueron analizados en el epígrafe anterior. En estos, según el tipo de evaluación y las medidas de cada característica, se proponen diferentes variantes, como el Proceso Analítico Jerárquico (*Analytic Hierarchy Process AHP*) o ELECTRE (*Elimination et Choix Traduisant la Réalité*).

Surgen asimismo trabajos para predecir la calidad del software a través de un árbol de decisión [81] y redes neuronales ([82] y [83]), a partir de datos históricos de proyectos similares o medidas de complejidad. En estos, usan métodos de aprendizaje supervisado para entrenar un clasificador de la calidad o modelo de predicción de fallos de software [84].

A partir del año 2000, se presentan algunas soluciones que consideran medidas imprecisas, a partir de cuestionarios aplicados para evaluar la calidad del software. Por tanto, comienzan a introducirse los conjuntos difusos, siendo uno de los primeros métodos utilizados el de las comparaciones. Se definen etiquetas lingüísticas para valorar cada característica y asocian a un número triangular difuso. Según el software, se evalúa cada característica y compara el valor obtenido contra el valor deseado ([85] y [86]).

Es a partir de 2007, cuando aparecen soluciones que combinan conjuntos difusos y métodos de decisión multicriterio (MCDM). Estas son la Fuzzy AHP ([62], [52] y [63]); Fuzzy Topsis [87] y la integral de Choquet [88].

Para analizar y comparar dichos trabajos, se definieron los siguientes indicadores:

- Objetivo: se utiliza para analizar para qué se va realizar la evaluación de la calidad; o sea, el objetivo de la evaluación.
- Alcance: expresa si la evaluación es enfocada a evaluar características específicas o la calidad en su totalidad.
- Nivel de generalidad: se emplea para revelar si el trabajo está dirigido a evaluar un tipo de producto específico o para cualquier producto de software.
- Modelo de calidad: refleja bajo qué modelo de calidad se evalúa el software.
- Interdependencia entre criterios: valora si en la evaluación del producto se tiene o no en cuenta la interdependencia entre los criterios.
- Naturaleza de los datos: analiza la naturaleza de los datos involucrados en la evaluación; o sea, el dominio de los datos en que está definida cada métrica y su escala.
- Método de agregación: analiza el método de agregación de la información utilizado en cada trabajo.
- Aplicabilidad: valora si en los trabajos analizados la solución se aplica a casos reales o simulados.
- Flexibilidad: analiza si el trabajo permite la modificación de los parámetros de la evaluación.

A continuación, se detalla el análisis de cada uno de los indicadores.

Objetivo, alcance y generalidad.

Los trabajos analizados, persiguen tres objetivos fundamentales. El primero, está enmarcado en obtener un índice global de calidad de los productos de software para facilitar la toma de decisiones. La toma de decisiones, se enfoca en determinar cuál software adquirir, cuáles son las características para la concepción de un nuevo producto, compararlo con la competencia, clasificar la calidad, etc. El segundo, se enfoca en describir o valorar las características del producto, sin llegar a un valor de la calidad, por lo que no se agrega información. La valoración de los criterios es una opción, pero puede dificultar el proceso de toma de decisiones. El tercero, consiste en predecir el valor de una característica de calidad, a partir de los datos históricos almacenados.

Las evaluaciones, según su alcance, pueden ser de dos tipos: las que determinan solamente una característica de calidad y las que obtienen una valoración general de la calidad del software. La diferencia fundamental entre ellas está dada en el modelo de calidad usado durante la evaluación. En el caso de valoración de una característica de calidad, la evaluación se realiza utilizando un fragmento del árbol que conforma el modelo de calidad (una rama), o una estructura lineal al descomponer esa característica en un conjunto de medidas. Si se valora la calidad en general, se utiliza la estructura arbórea completa.

Según su nivel de generalidad, se han desarrollado varios trabajos para evaluar determinados tipos de producto. Entre estos tenemos, por ejemplo, para aplicaciones Web [89]; para aplicaciones de software libre ([90] y [91]); para herramientas CASE ([48] y [92]); para componentes de software ([80], [85], [93] y [94]), entre otros ([95] y [96]). Otros trabajos evalúan cualquier tipo de productos ([97] y [98]).

La diferencia entre evaluar un producto específico o cualquier tipo de producto, radica en el modelo de calidad a utilizar. La evaluación de cualquier tipo producto, se basa en los modelos de calidad genéricos; mientras, en la de un producto específico, se adaptan o redefinen los modelos de calidad básicos, lo cual implica la adición o refinamiento de algunos criterios propios del producto y su agrupamiento de diferentes formas. Asociado a esto, conlleva cambios en las medidas a usar, y por ende en las pruebas a realizar.

Modelo de calidad, naturaleza de los datos e interdependencia entre criterios.

En la actualidad, es con las normas ISO donde se logra mayor consenso entre los expertos en calidad e ingeniería de software. Eso explica la gran adherencia a ellas en la mayoría de los trabajos revisados. En los referidos a la evaluación de la calidad,

proponen el uso de los modelos de McCall [99], ISO 9126 ([79], [69], [80], [85], [62], [89], [91], [100], [101] y [102]) y 25010 ([90], [93], [95] y [103]).

Como se aprecia en el Epígrafe I.3, referido al análisis de los modelos de calidad, la diferencia básica entre estos es la clasificación taxonómica basada en niveles, y la variedad de criterios que definen cada uno. Estos modelos se solapan en gran medida; son muy semejantes, lo que dificulta detallar sus similitudes y diferencias.

Un aspecto importante en los modelos, es su complejidad estructural. En la actualidad, se han extendido taxonomías jerárquicas para formar estructuras más complejas, como los grafos. El tratamiento de dicho problema, ha evolucionado teniendo en cuenta la superposición de características de calidad y las interrelaciones entre ellas. Tal evolución ha sido impulsada por el creciente nivel de comprensión de la calidad, que va más allá de una simple taxonomía.

Los criterios de calidad son, por lo general, interactivos, donde el resultado de la preferencia de un criterio sobre otro está influenciado por los restantes. A pesar de esto, las soluciones propuestas son puramente jerárquicas para evitar la complejidad. No obstante, como se puede observar, valorar la calidad a partir de criterios independientes, causa algún efecto a favor o en contra de la evaluación.

En el contexto real de una empresa de software, el proceso de evaluación suele estar restringido en cuanto a tiempo y recursos. En ocasiones, la documentación está incompleta, lo que dificulta el diseño detallado de los casos de prueba. Constituye un reto seleccionar qué criterios y medidas incluir al inicio de las pruebas. El análisis de la interdependencia entre criterios a partir de experiencias de expertos en las evaluaciones de software, pudiera ser de gran ayuda para recomendar por cuál característica y tipo de pruebas comenzar la evaluación.

El análisis de interdependencias entre criterios, es casi nulo en los trabajos analizados ([48], [59], [60], [61], [62], [52], [63], [53], [54], [55], [56], [57], [58], [64], [65], [66], [70] y [71]); solamente en dos se tiene en cuenta al seleccionar un método de agregación de la información ([67] y [68]). Los criterios esenciales únicamente se evalúan en un trabajo [51] hasta el momento de redacción de esta tesis.

En cuanto a las métricas utilizadas, en la evaluación se manipula una diversidad de estas que en ocasiones no permite dar una estimación más objetiva de los criterios. Las mismas están en función del tiempo, cantidad, tamaño o aspectos subjetivos, por lo que se encuentran expresadas en dominios y escalas diferentes. En el caso de las normas y estándares, en ocasiones se muestran algunos ejemplos de métricas, pero son insuficientes por su cubrimiento; otras se encuentran aún en estudio.

La calidad tiene dos caras, una relacionada con propiedades objetivas, como la velocidad y el tamaño, y otra con la percepción subjetiva del producto. Asignar un valor numérico a dichos criterios subjetivos, puede resultar cuestionable y débil. Es preciso señalar que no todas las métricas toman valores exactos, al estar basadas en factores subjetivos de usuarios finales y equipos de aseguramiento de la calidad. Todo lo anterior lleva al análisis de agregación de datos heterogéneos para obtener un índice que permita clasificar al producto.

De los trabajos analizados, solamente en tres se manipulan datos heterogéneos y unifica la información mediante números difusos ([53], [65] y [66]).

Métodos de agregación de la información

La agregación de la información en el proceso de evaluación de la calidad de un producto de software, se torna complejo debido a los siguientes elementos: (1) los datos a agregar suelen ser heterogéneos e imprecisos; (2) deben permitir la agregación parcial de los datos cuando sea necesario penalizar la evaluación por los criterios esenciales; (3) el método de agregación debe ser independiente de los datos; (4) el índice de calidad obtenido debe encontrarse en un rango que sea posible interpretar y clasificar fácilmente; (5) deberá tener en cuenta una estructura más compleja que la jerárquica; o sea, valorar la interacción horizontal de los criterios en cada nivel.

De los autores consultados, L. Mich decide no agregar la información, y solo muestra los resultados a través de gráficos, con lo cual se dificulta la toma de decisiones [104].

En los trabajos que agregan la información, se usa una gran variedad de métodos que van desde el uso de algoritmos y técnicas estadísticas convencionales, hasta técnicas de *Soft Computing*. En el caso de los primeros, hacen uso fundamentalmente de la estadística descriptiva [105],[106], [107]. La limitante fundamental aquí está dada por la falta de capacidad de adaptación [108] y de fuentes de datos históricos.

Entre los métodos que utilizan *Soft Computing*, se encuentran los sistemas de inferencia difusa, integración difusa y métodos de decisión multicriterio (MCDM). Varios trabajos hacen uso de sistemas de inferencia difusos, como el método de Takagi Sugeno ([109], [110], [111] y [112]). Otros se basan en reglas ([113] y [114]), o redes neuronales y sistemas neurodifusos ([115], [116], [117] y [118]). Los mismos se emplean cuando el objetivo es predecir una característica de calidad.

Cuando se desea calcular un índice global de calidad, se utilizan MCDM. En el caso de las soluciones que combinan estos métodos con números difusos, los que más se emplean son suma ponderada ([53], [54], [55], [56], [57] y [58]), AHP ([48], [59], [60],

[61], [62], [52], [63], [64], [65] y [66]), ELECTRE ([48] y [69]), y otros con operadores de agregación ([73], [74] y [75]). Hay soluciones donde el índice de calidad se calcula a partir de distancias, como es el caso del TOPSIS ([70] y [71]) y otras formas de cálculo de distancias [72]. En todos estos casos, no tienen en cuenta la interdependencia, ni los criterios esenciales.

Entre los MCDM, algunos trabajan con los datos procedentes de la evaluación y otros con preferencias definidas por expertos. El índice de calidad obtenido con los datos procedentes de las medidas, es más preciso y confiable.

Es importante para la evaluación, que el método de agregación utilizado sea independiente de los datos, característica que no cumple AHP, ELECTRE, ni otros métodos, por la forma de normalización de la información que emplean.

El *Analytic Network Process* (ANP), es una generalización de AHP, combinado con números difusos, conocido como Fuzzy ANP. Dicho método, refleja la interdependencia e interactividad entre los diferentes criterios ([67] y [68]). Otra propuesta, valora la interdependencia y usa como operador de agregación la integral de Choquet ([42], [8], [119] y [120]). Ambas, solo analizan la relación, sin tener en cuenta si el impacto es negativo o positivo. Además, la aplicación práctica de ANP resulta engorrosa, difícil y poco intuitiva, debido a la cantidad de preguntas que genera la comparación por pares; la aplicabilidad de la integral, se limita a los problemas con una pequeña cantidad de criterios de entrada.

Otras variantes son soluciones híbridas, cuando combinan dos o varios métodos de los expresados anteriormente [49]. Los algoritmos genéticos, generalmente se utilizan para el cálculo del peso de los criterios ([121] y [122]).

Si se seleccionan los trabajos más destacados se puede observar que no se encuentra una solución que resuelva en su conjunto los problemas encontrados. (Ver Tabla I.8)

Tabla I.8 Principales soluciones de evaluación de la calidad de productos de software.

Soluciones	Trata incertidumbre	Análisis de Interdependencia	Criterios Esenciales	Independencia de los datos	R. fácil de entender
(Morisio & Tsoukias 1997) [48]	No	No	No	No	No
(Stamelos et al. 2000) [49] (BLIN & TSOUKIÀS 2001) [69]	No	No	No	No	No
(L. Etaati, S. Sadi-Nezhad, A 2011) [67]	Sí	Sí, Fuzzy ANP	No	No	No
(Challa et al. 2011) [53]	Sí	No	No	Sí	Sí
(Bhatnagar et al. 2012) [65] (Dubey et al. 2012) [66]	Sí	No	No	No	Sí
(Singh & Dubey 2013) [56]	Sí	No	No	Sí	Sí
(Li et al. 2014) [72]	Sí	No	No	No	No

Aplicabilidad y flexibilidad.

Los trabajos enunciados hasta aquí se han aplicado con datos experimentales, que no muestran la realidad de una industria. Los elementos participantes en la evaluación, son rígidos y apenas se modifican. Por ejemplo, los modelos de calidad no se representan con una estructura flexible y genérica que permita con facilidad evaluar con diferentes modelos.

Dichas soluciones, para garantizar su uso práctico, deben implementarse a través de un sistema automatizado, que abarque todo el proceso y facilite la toma de decisiones. Esto es poco mencionado e implementado en la bibliografía consultada.

A continuación, se analizan un grupo de herramientas metodológicas que ayudan a solucionar las insuficiencias y retos presentes en el proceso de evaluación de la calidad de los productos de software.

I.7 *Soft Computing* en los problemas de evaluación de la calidad de productos de software

Como ya se analizó, el proceso de evaluación de la calidad encierra un alto grado de subjetividad, ya que conlleva estimar, apreciar, señalar el valor de la calidad. La evaluación depende de factores objetivos, subjetivos y otros que no son posibles delimitar. Por ejemplo, el tiempo de ejecución de un algoritmo, es una magnitud de tiempo que se puede medir; pero estimar, apreciar si es un valor óptimo o no, depende de múltiples elementos, como pueden ser para qué se utilizará el algoritmo, si es un procesamiento en tiempo real o no, etc. La necesidad de cuantificar diferentes magnitudes durante los procesos de medición, se vuelve altamente compleja y los

datos no son exactos, demostrando su imprecisión o incertidumbre. Una de las metodologías desarrolladas para lograr una mejor representación de esta información, es la lógica difusa ([123] y [124]).

El uso de la lógica difusa ha sido extraordinario [124] desde 1965, en que Zadeh, en la Universidad de California en Berkeley, introdujo el concepto de conjunto difuso [123] y, por consiguiente, extendió al concepto de variables lingüísticas.

En este epígrafe, describen los elementos principales de la *Soft Computing*, en particular la lógica difusa y el modelado lingüístico difuso, dado que serán utilizados después en la propuesta resultante de esta investigación. Se analizan, también, los mapas cognitivos difusos y su utilización en el modelado de la interdependencia entre criterios. El conformado de los indicadores de calidad, usando variables lingüísticas, se realizará con números difusos triangulares.

Soft Computing y la evaluación del software:
Un conjunto difuso, permite la pertenencia de un elemento a un conjunto de forma gradual y no absoluta, como establece la teoría clásica de conjuntos; es decir, admitiendo pertenencias valoradas en el intervalo [0,1] en lugar de en el conjunto {0,1}. Las aplicaciones basadas en dicha teoría, han evolucionado de tal modo que resulta imposible calcular el volumen de negocio que generan en todo el mundo, debido a su amplio uso en diferentes disciplinas y áreas de la ciencia.

Esta teoría ha evolucionado e impactado en otras teorías y técnicas. En los años 90, las teorías de conjuntos difusos, redes neuronales y el área de la programación evolutiva, comienzan a conocerse como *Soft Computing*. En 1994, Zadeh define por primera vez dicho término como:

"*Soft Computing* no es un cuerpo homogéneo de conceptos y técnicas. Más bien es una mezcla de distintos métodos que de una forma u otra cooperan desde sus fundamentos. En este sentido, el principal objetivo de la *Soft Computing* es aprovechar la tolerancia que conllevan la imprecisión y la incertidumbre, para conseguir manejabilidad, robustez y soluciones de bajo costo. Los principales ingredientes de la *Soft Computing* son la lógica difusa, la neurocomputación y el razonamiento probabilístico, incluyendo este último a los algoritmos genéticos, las redes de creencia, los sistemas caóticos y algunas partes de la teoría de aprendizaje. En esa asociación de lógica difusa, neurocomputación y razonamiento probabilístico, la lógica difusa se ocupa principalmente de la imprecisión y el razonamiento aproximado; la neurocumputación del aprendizaje, y el razonamiento probabilístico de la incertidumbre y la propagación de las creencias." [3].

En esta definición, hace referencia a conceptos aislados e indica el empleo de la lógica difusa. Dicho término ha evolucionado, quedando como la base teórica del área de los Sistemas Inteligentes. Lo anterior se encuentra sustentado en la definición de Verdegay, estableciendo que: *"*Se trata de considerarla como antítesis de lo que podríamos denominar *'Hard Computing',* de manera que podría verse la *Soft Computing* como un conjunto de técnicas y métodos que permitan tratar las situaciones prácticas reales de la misma forma que suelen hacerlo los seres humanos, es decir, en base a inteligencia, sentido común, consideración de analogías, aproximaciones, etc. En este sentido *Soft Computing* es una familia de métodos de resolución de problemas cuyos primeros miembros serían el Razonamiento Aproximado y los Métodos de Aproximación Funcional y de Optimización*"* [4].

Entre los componentes principales de la *Soft Computing*, se encuentran el razonamiento probabilístico, la lógica y conjuntos difusos, redes neuronales y metaheurísticas. No debe entenderse la *Soft Computing* como la suma de todos estos elementos, sino como el resultado de la integración, cooperación, asociación o la hibridación de sus componentes.

La precisión en un modelo, supone que los parámetros de este representan exactamente la percepción del fenómeno o las características del sistema real modelado [125]. Esto no ocurre en la evaluación de los productos de software, donde se manifiesta imprecisión. Dicha imprecisión se entiende como falta de concreción sobre los valores a utilizar, aunque sin afectar su frecuencia ni excluir posibles "errores" en sus mediciones, algo completamente imprevisible, sinónimo de ambigüedad y vaguedad. Existen elementos que determinan el proceso de evaluación de la calidad de los productos de software como un problema de toma de decisiones bajo incertidumbre, entre los que se pueden mencionar los siguientes:

- Se tienen en cuenta varios criterios para la evaluación, organizados en un modelo de calidad, y su valoración integral permite la selección entre alternativas.
- Se manipula información de diferente naturaleza, objetiva y subjetiva. En el caso de los factores subjetivos, se evalúan según las actitudes y niveles de conocimientos de los evaluadores.
- Se manipulan datos vagos e imprecisos, y en su medición existe imprecisión; por ejemplo, la interfaz amigable, la satisfacción del cliente, lo atractivo.
- Es un proceso donde los datos provienen de fuentes diversas: de los clientes, evaluadores, usuarios finales, especialistas en seguridad y expertos en usabilidad. Se involucran deferentes personas con disímiles niveles de conocimiento de los criterios.

- El juicio humano está presente en las actividades del proceso, por lo que se presenta la incertidumbre a partir de las propias características de las personas.
- Esto se manifiesta también en las medidas subjetivas manipuladas.

La *Soft Computing*, es una metodología ampliamente utilizada en situaciones donde los datos a considerar no son exactos, sino imprecisos. El modelado de estos se hace mediante el empleo de la teoría de los conjuntos difusos.

Conjuntos difusos:
La lógica difusa surge para la formalización del razonamiento aproximado; intenta manejar el conocimiento propio del "sentido común" (alto, bajo, poco, mucho, caro, etc.). Es una extensión de los principios de los conjuntos tradicionales, donde los elementos pueden pertenecer o no a estos, y en su lugar, se asigna un determinado grado de pertenencia de los elementos al conjunto.

El conjunto, es una colección de objetos que pueden clasificarse en base a las características que tienen en común. Se define de dos formas: por extensión ({a, e, i, o, u}) o comprensión (las vocales).

Un conjunto booleano A es una aplicación de un conjunto referencial S en el conjunto {0, 1}, $A : S \rightarrow \{0,1\}$, y se define con una función característica:

$$\mu_A(x) = \begin{cases} 1 & si\ x \in A \\ 0 & si\ x \notin A \end{cases} \qquad \text{(I.4)}$$

Los conjuntos difusos, dan un valor cuantitativo a cada elemento, el cual representa el grado de pertenencia al conjunto.

Un conjunto difuso A es una aplicación de un conjunto referencial S en el intervalo [0, 1], i.e., $A: S \rightarrow [0,1]$, y se define por medio de una función de pertenencia: $0 \leq \mu_A(x) \leq 1$.

Así, un 1 significa pertenencia absoluta, 0 pertenencia nula y cualquier valor entre 0 y 1 se considera un grado de pertenencia. Cuanto más cerca está el grado de pertenencia al valor 1, entonces mayor es la pertenencia del objeto al conjunto.

Puede representarse como un conjunto de pares ordenados de un elemento genérico $x,\ x \in A\ y\ su\ grado\ de\ pertenencia\ \mu_A(x)$:

$$A = \{(x, \mu_A(x)),\ \mu_A(x) \in [0,1]\ \} \qquad \text{(I.5)}$$

Ejemplo. En este caso se puede representar la densidad de fracasos de un software es alta cuando es mayor o igual que 0,8 donde su grado de pertenencia es 1, los valores menores que 0,6 no es alta y entre 0,6 y 0,8 la densidad tiene un grado de pertenencia entre [0,1]. (Ver Figura I.9).

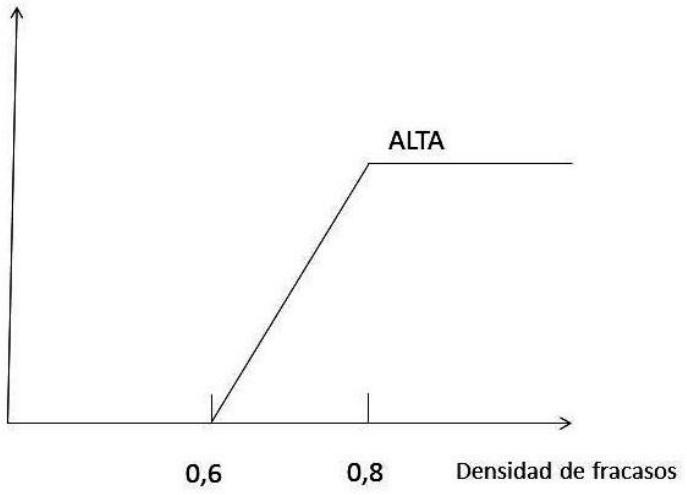

Figura I. 9. Métrica densidad de fracasos.

La definición de la función de pertenencia debe ser simple y depende del concepto a definir, del contexto y aplicación.

Existen funciones de pertenencia típicas y las más usadas en la solución de problemas de calidad de software son:

- Función Triangular: Definido por sus límites inferior *a* y superior *b*, y el valor modal *m*, tal que *a* < *m* < *b* (Ver Figura I.10).

$$A(x) = \begin{cases} 0 & si\ x \le a \\ \frac{(x-a)}{(m-a)} & si\ a < x \le m \\ \frac{(b-x)}{(b-m)} & si\ m < x < b \\ 0 & si\ x \ge b \end{cases} \qquad \text{(I.6)}$$

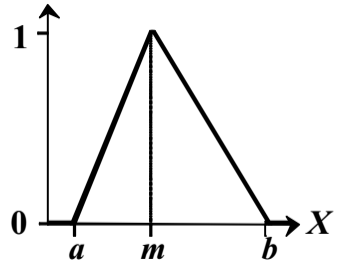

Figura I. 10. Función triangular

- Función Trapezoidal: Definida por sus límites inferior a y superior d, y los límites b y c, correspondientes al inferior y superior respectivamente de la meseta (Ver Figura I.11).

$$A(x) = \begin{cases} 0 & si\ x \leq a\ o\ x \geq d \\ \frac{(x-a)}{(b-a)} & si\ a < x < b \\ 1 & si\ b \leq x \leq c \\ \frac{(d-x)}{(d-c)} & si\ c < x < d \end{cases} \qquad \text{(I.7)}$$

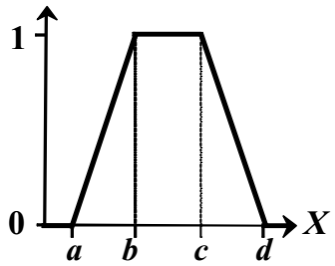

Figura I. 11. Función trapezoidal

- Función Gaussiana. Definida por su valor medio *m* y el valor *k > 0.* Es la típica campana de Gauss (mayor *k*, más estrecha es la campana) (Ver Figura I.12).

$$A(x) = e^{-k(x-m)^2} \qquad \text{(I.8)}$$

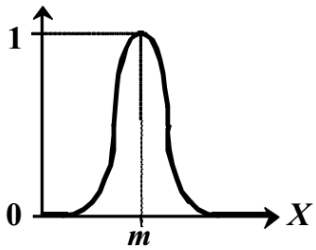

Figura I. 12. Función gaussiana.

Como se ha expresado anteriormente, a cada conjunto difuso se le asocia una función de pertenencia que tiene como imagen el intervalo [0, 1]. Por otra parte, los elementos del conjunto estarán definidos en correspondencia con el problema tratado, y sobre estos elementos se establecen determinadas operaciones aritméticas. A continuación, se exponen las operaciones aritméticas para un conjunto difuso, tal que sus elementos están formados por números difusos triangulares ([126] y [127]).

Sean T_1 y T_2, dos números difusos triangulares, definidos por las tripletas (a_1, b_1, c_1) y (a_2, b_2, c_2), respectivamente, entonces las operaciones aritméticas entre ellos están dadas por:

Adición: $T_1 \oplus T_2 = (a_1 + a_2, b_1 + b_2, c_1 + c_2)$ (I.9)

Sustracción: $T_1 \ominus T_2 = T_1 \oplus (-T_2), donde\ (-T_2) = (-c_2, -b_2, -a_2),$

$por\ lo\ que\ T_1 \ominus T_2 = (a_1 - c_2, b_1 - b_2, c_1 - a_2)$ (I.10)

Multiplicación: $T_1 \otimes T_2 = (a_1 \times a_2, b_1 \times b_2, c_1 \times c_2)$ (I.11)

Division: $T_1 \oslash T_2 = (a_1, b_1, c_1) \otimes (\frac{1}{c_2}, \frac{1}{b_2}, \frac{1}{a_2})$ (I.12)

Multiplicación por un escalar: $k.T_1 = (ka_1, kb_1, kc_1)$ (I.13)

Máximo: $Max(T_1, T_2) = [Max(a_1, a_2), Max(b_1, b_2), Max(c_1, c_2)]$ (I.14)

Mínimo: $Min(T_1, T_2) = [Min(a_1, a_2), Min(b_1, b_2), Min(c_1, c_2)]$ (I.15)

Raíz cuadra: $\sqrt{T_1} = (\sqrt{a_1}, \sqrt{b_1}, \sqrt{c_1})$ (I.16)

Potencia: $T_1^n = (a_1^n, b_1^n, c_1^n)$ (I.17)

Método de defuzificación: $c(T_1) = \frac{(a_1 + 4b_1 + c_1)}{6}$ (I.18)

Variables lingüísticas:

A menudo, en aplicaciones, pretende describirse el estado de un objeto o fenómeno, cuya representación, para que sea útil y apropiada, debe formularse a través de palabras o sentencias, no mediante números. Este es el caso de las variables lingüísticas, cuyo valor establece la descripción.

Una variable lingüística, admite que sus valores sean etiquetas; o sea, términos lingüísticos definidos como conjuntos difusos (sobre cierto dominio subyacente). Una etiqueta, incluye muchos valores posibles. Dichas variables son de utilidad, al constituir una manera de comprimir información [3]; además, ayudan a caracterizar fenómenos que pudieran estar mal definidos o resultar complejos de definir, o ambas cosas [128]. Las mismas son un medio de trasladar conceptos o descripciones lingüísticas a numéricas y tratarlas automáticamente.

Una variable lingüística se define como un conjunto de 5 elementos $(N, U, T(N), G, M)$ [128], donde:

- *N* es el nombre de la variable
- *U* es el dominio subyacente.
- *T(N)* es el conjunto de términos o etiquetas que puede tomar *N.*
- *G* es una gramática para generar las etiquetas de *T(N)*
- *M* es una regla semántica que asocia cada elemento de *T(N)* con un conjunto difuso en *U* de entre todos los posibles: $M: T(N) \otimes F(U)$.

Un ejemplo es la variable evaluación (ver Figura I.13).

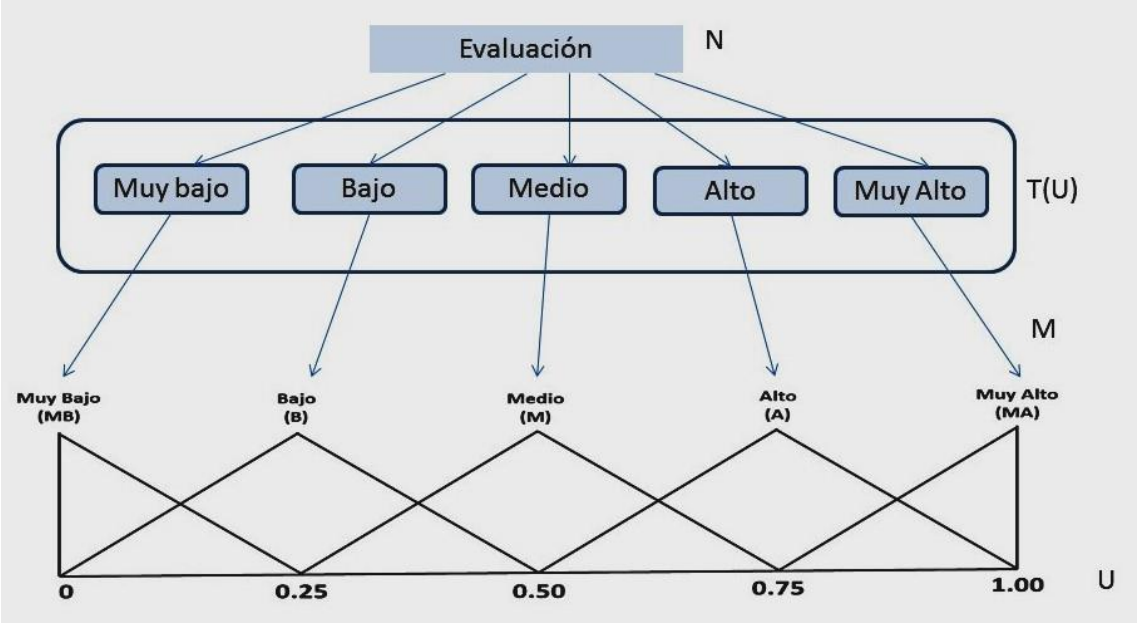

Figura I.13 Definición de la variable lingüística evaluación.

La utilización de etiquetas lingüísticas, facilita la comprensión semántica del estado de un problema dado. No obstante, se requiere de una metodología para operar con ellas.

El modelado lingüístico, tiene como base la teoría de los conjuntos difusos, y ha demostrado su eficacia para la representación de información de naturaleza cualitativa. Un ejemplo de variable lingüística, es la “satisfacción” que se siente al trabajar con el software, normalmente expresada en términos de intervalos que pueden ser asociados a las clasificaciones de alta, media o baja, y tener significados diferentes en diversas regiones.

En cualquier ámbito en el que se desee aplicar un enfoque lingüístico difuso para la resolución de algún problema, deben tomarse dos decisiones:

- Modelo de representación: elección del conjunto de etiquetas lingüísticas junto con su semántica.
- Modelo computacional: definirlo y seleccionar los correspondientes operadores de comparación y agregación.

En la definición de variable lingüística, es importante la selección de las etiquetas y conjuntos difusos asociados a ellas. En estos se concentra el peso significativo en la obtención del ordenamiento de las alternativas y la solución más adecuada.

La cantidad de etiquetas recomendadas en los modelos lingüísticos, es un valor impar entre 5 y 9, sin superar los 11 o 13 términos. Una vez definidas las etiquetas lingüísticas, es necesario definir la semántica del conjunto de ellas, para lo cual existen varios enfoques, aunque el más utilizado es el basado en funciones de pertenencia [129]. Este define la semántica del conjunto de etiquetas lingüísticas, utilizando

números difusos en el intervalo [0,1], donde cada número es descrito por una función de pertenencia [5]. Un método eficiente desde un punto de vista computacional para caracterizar un número difuso, es usar una representación basada en parámetros de su función de pertenencia [130].

Este enfoque, implica establecer las funciones de pertenencia asociadas a cada etiqueta, aunque presenta el problema de determinar los parámetros según los puntos de vista de todas las fuentes de información. En la realidad, es difícil que todas estas propongan exactamente las mismas funciones de pertenencia asociadas a los términos lingüísticos, debido a que cada una de ellas puede interpretar de forma parecida, pero a la vez diferente, el mismo concepto. Por tanto, puede darse el caso de términos lingüísticos con una sintaxis similar pero diferente semántica. En la presente investigación, se trabaja con variables lingüísticas, usando funciones de pertenencia triangular y la sintaxis que los expertos del Centro Nacional de Calidad de Software de Cuba (CALISOFT) han definido para la evaluación de la calidad.

Mapas Cognitivos Difusos:
Los Mapas Cognitivos Difusos (MCD), son una técnica desarrollada por Kosko, en 1986 [131], como extensión de los mapas cognitivos introducidos por Axerold, en 1976. Son una herramienta de modelado cualitativa, basada en los conocimientos y experiencia de los expertos.

Kosko [132], describe un MCD como un grafo difuso dirigido, donde el *feedback* es permitido. Los nodos, representan los conceptos y los arcos pesados, las relaciones causales entre conceptos. En los MCD, existen tres posibles tipos de relaciones entre conceptos: relación positiva, relación negativa o la no existencia de relaciones. El grado de la relación, se describe a través de un número difuso o valor lingüístico. Emplea valores difusos en el intervalo [-1,1] para el peso de los arcos. El enlace causal es dinámico, donde el efecto de un cambio en un concepto o nodo, afecta a los demás nodos.

En el MCD, se representa la influencia de un concepto en otro, y se interpreta como una variable lingüística, tomando valores en el universo U = [- 1,1]; su conjunto de términos T (influencia), podría ser:

T (influencia) = {negativamente muy fuerte, muy negativo, negativamente medio, negativamente débil, cero, positivamente débil, positivamente medio, muy positivo, positivamente muy fuerte} (Ver Tabla I.9)

La regla semántica se define a continuación. Los términos se caracterizan por los conjuntos difusos, cuyas funciones de pertenencia se muestran en la Figura I.14:

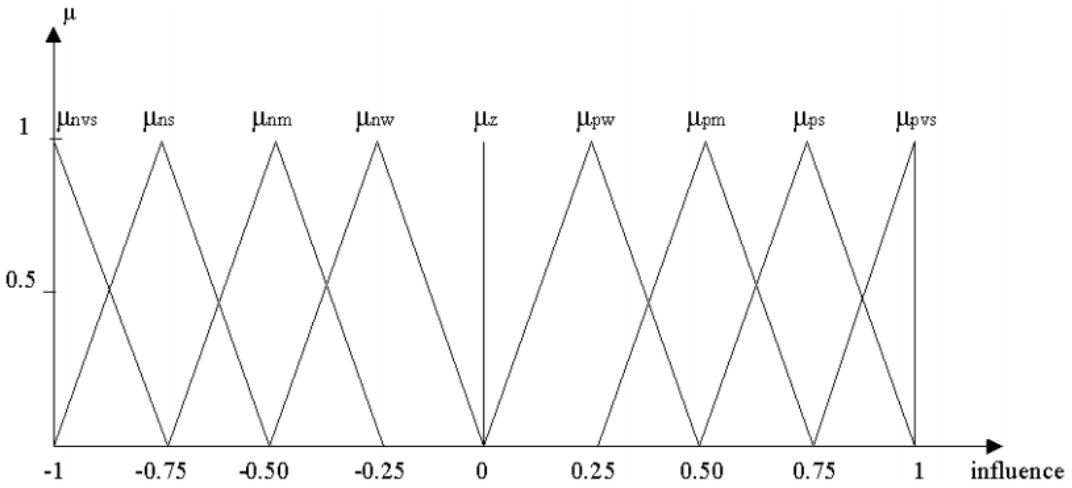

Figura I.14. Representación de las funciones de pertenencia para la variable Influencia [133].

Tabla I.9. Etiquetas lingüísticas de la variable lingüística Influencia

Etiquetas Lingüísticas	***Descripción***	***Función de pertenencia***	$(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c})$
Negativamente muy fuerte	el conjunto difuso de "una influencia a continuación hasta -75%"	μ_{nvs}	$(-1, -1, -0.75)$
Muy negativo	el conjunto difuso de "una influencia cerca de -75%"	μ_{ns}	$(-1, -0.75, -0.50)$
Negativamente medio	el conjunto difuso de "una influencia cerca de -50%"	μ_{nm}	$(-0.75, -0.50, -0.25)$
Negativamente débil	el conjunto difuso de "una influencia cerca de -25%"	μ_{nw}	$(-0.50, -0.25, 0)$
Cero	el conjunto difuso de "una influencia cercana a 0"	μ_z	$(0, 0, 0)$
Positivamente débil	el conjunto difuso de "una influencia cerca del 25%",	μ_{pw}	$(0, 0.25, 0.50)$
Positivamente medio	el conjunto difuso de "una influencia cerca del 50%"	μ_{pm}	$(0.25, 0.50, 0.75)$
Muy positivo	el conjunto difuso de "una influencia cerca del 75%"	μ_{ps}.	$(0.50, 0.75, 1)$
Positivamente muy fuerte	el conjunto difuso de "una influencia por encima de un 75%"	μ_{pvs}	$(0.75, 1, 1)$

Cada etiqueta se representa con la siguiente función de pertenencia:

$$\mu_{etiqueta}(x) = \begin{cases} \frac{(x-a)}{(b-a)} & a \le x \le b \\ \frac{(c-x)}{(c-b)} & b \le x \le c \\ 0 & otro\ caso \end{cases} \qquad \text{(I.19)}$$

Un MCD, formado por *n* conceptos, se representa en una matriz de *nxn* conocida como matriz de adyacencia. Dicha matriz se obtiene a partir de los valores asignados a los arcos.

El mapa está basado en un modelo de ejecución que calcula los niveles de activación en iteraciones sucesivas de los distintos conceptos. Esta simulación, requiere adicionalmente la definición de los valores iniciales para cada concepto en un vector. Los valores de los conceptos, son calculados en cada paso de la simulación, de la siguiente forma:

$$A_i^{(k+1)} = f(A_i^k + \sum_{j=1\ j\neq i}^{n} A_j^k \,.\, w_{ji}) \qquad \text{(I.20)}$$

Donde:

$A_i^{(k+1)}$ es el valor del concepto C_i en el paso $k+1$ de la simulación,

A_j^k es el valor del concepto C_j en el paso k de la simulación

w_{ji} es el peso de la conexión que va del concepto C_j al concepto C_i

y $f(\ \ )$ es la función de activación.

Las principales funciones de activación reportadas en la literatura, son la sigmoide ($f(x) = \frac{1}{1+e^{-\lambda x}}$ donde λ representa la pendiente), y la tangente hiperbólica ($f(x) = \tanh(x)$).

De acuerdo al vector de entrada, el MCD convergerá a uno de los siguientes estados: punto fijo, ciclo límite o comportamiento caótico ([133]).

La principal ventaja de los MCD, es que no existe restricción en el número de expertos ni conceptos. Dichos mapas, juegan un papel esencial en la captura de los modelos mentales de los decisores.

Los MCD, han sido empleados en diversos dominios ([134], [135] y [136]). Entre las áreas de aplicación, se destaca su utilización en la toma de decisiones y la construcción de sistemas de ayuda a la toma de decisiones ([137], [138], [139] y [140]). Su empleo en la toma de decisiones, está orientado fundamentalmente al análisis de alternativas. Ha sido utilizado también para la estructuración de problemas de decisión y eliminar las limitaciones del ANP, en lo relativo a la construcción de las relaciones entre criterios y la elevada demanda cognitiva que significa para los decisores el proceso de comparación por pares [141].

Han sido empleados, además, para la toma de decisión en grupo, debido a las facilidades que brindan para la agregación de modelos causales, provenientes de múltiples expertos [138].

CAPÍTULO II. MODELO COMPUTACIONAL PARA LA EVALUACIÓN Y SELECCIÓN DE PRODUCTOS DE SOFTWARE SEGÚN SU CALIDAD

Introducción

La evaluación de la calidad de los productos de software, es un proceso complejo, debido a la variedad de criterios que se tienen en cuenta. Estos criterios, se organizan en una estructura jerárquica, llamada modelo de calidad. En el Capítulo I, se caracterizaron dichos modelos. Las medidas de calidad ubicadas en el nivel más bajo del modelo, están definidas en dominios de datos diferentes, con escalas distintas y corresponden a factores objetivos y subjetivos.

En el proceso de pruebas de software, los expertos, probadores y usuarios finales, a través de la observación y percepción, establecen elementos y parámetros para a partir de ellos determinar las medidas de calidad. Dichos expertos tienen diferentes criterios y conocimientos acerca del proceso, lo que introduce un alto grado de incertidumbre. Al comparar el proceso de evaluación de la calidad con el proceso de toma de decisiones, tienen una alta semejanza. En ambos, se definen los parámetros, se realiza la evaluación y se selecciona una opción. Por ello, la evaluación de la calidad de los productos de software, en la literatura se trata como un problema de toma de decisión multicriterio ([48], [49], [69], [53], [56], [65], [66] y [67]).

Autores como Tsoukias [48], Challa [53], y Dubey [65], han establecido varias soluciones prácticas basadas en los principales métodos de solución a problemas de decisión multicriterio, como AHP y Fuzzy AHP ([48], [59], [80], [62], [93], [66] y [142]), ANP ([67] y [143]), TOPSIS ([71] y [143]) y ELECTRE ([48], [49] y [69]). Sin embargo, al aplicar dichos modelos a entornos reales, se ha demostrado la existencia de elementos que no han sido suficientemente tratados, como son: 1) la manipulación de información heterogénea, ambigua, imprecisa, y de diferentes fuentes; 2) el análisis de la interdependencia entre los criterios presente en los diferentes modelos de calidad; 3) la incorporación de los criterios esenciales que posibilitan una agregación total y parcial. Tales disfunciones**,** han provocado que en ocasiones no se refleje una correspondencia entre la calidad estimada de un producto y la que realmente demuestra en su uso.

La necesidad surgida de esta problemática, es definir y aplicar un nuevo modelo que incorpore dichos elementos, como vía para lograr una valoración objetiva, de ahí que, en el presente capítulo, se formula una propuesta de modelo computacional para la evaluación y selección de productos de software, según su calidad.

En la primera sección, se justifica la evaluación de la calidad como un problema de toma de decisión multicriterio y se presenta la definición clásica. En la segunda, se

formaliza el nuevo modelo con el uso de mapas cognitivos difusos y un vector de penalización para los criterios esenciales. Además, se muestra la estrategia de solución incorporando la valoración de criterios cualitativos.

II.1 Métodos, metodologías para el problema de evaluación y selección de productos de software según su calidad

Cuando se tiene un conjunto de productos de software a evaluar, y desea comparar la calidad o recomendar el mejor producto, se soluciona como un problema de toma de decisión multicriterio, de manera que a cada uno de ellos se le calcula un índice de calidad, a partir de los resultados de las diferentes métricas a valorar y luego ordena el resultado. Varios son los trabajos que utilizan dicho enfoque ([48], [49], [69], [53], [65], [54], [56] y [122]).

Un elemento interesante es que, si los productos están orientados a resolver el mismo problema, entonces la utilización de un enfoque de toma de decisión multicriterio, permite recomendar el mejor; si los productos evaluados resuelven problemas diferentes, entonces el ordenamiento los diferencia según su nivel de calidad.

Con la metodología IusWare, se define formalmente, por primera vez, la evaluación y selección de productos de software [48]. La Figura II.1, describe los pasos de la misma.

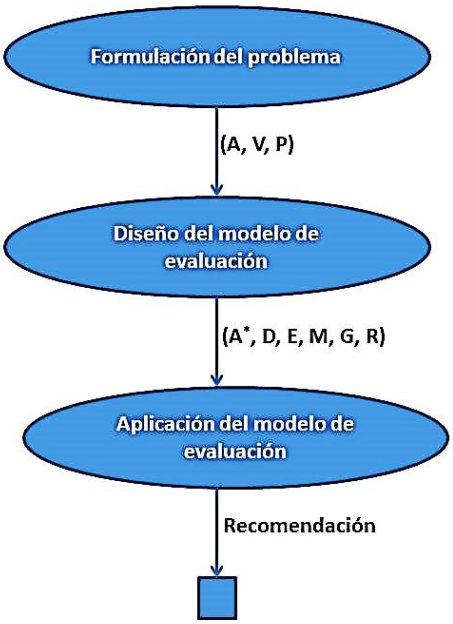

Figura II.1 Metodología IusWare para la evaluación y selección de productos de software, según su calidad [48]

Esta metodología, propone tres actividades: Formulación del problema, Diseño del modelo de evaluación y Aplicación del modelo de evaluación. Las mismas son similares a las fases de un proceso de toma de decisión.

En la primera actividad, Formulación del problema, se enuncia la tripleta (A, V, P), donde:

- **A** representa el conjunto de alternativas (productos de software) posibles a evaluar;
- **V** constituye el conjunto de puntos de vista para la evaluación; o sea, los requisitos a tener en cuenta
- **P** define el propósito de la evaluación.

Dichos elementos, se determinan mediante entrevistas con los clientes. En la actividad siguiente, Diseño del modelo de evaluación, se define la 6-tupla {A^*, D, E, M, G, R}, donde:

- **A^*** subconjunto de A; representa el conjunto de alternativas a evaluar, dado que se seleccionan las alternativas a valorar de entre todas las posibles.
- **D** constituye el conjunto de criterios de evaluación.
- **M** representa el conjunto de medidas asociadas a los criterios
- **E** es el conjunto de escalas asociadas a cada medida; los datos se representan en una escala numérica u ordinal.
- **G** son las preferencias del usuario para el conjunto de criterios.
- **R** significa el método de agregación de información; describe un algoritmo capaz de transformar el conjunto de datos de entrada en una recomendación para el cliente.

En esta actividad, las tareas se organizan como muestra la Figura II.2. Dicha operación, tiene como entrada los elementos definidos en la actividad anterior (A, V, P). A partir de V, se definen los criterios de evaluación y con ellos las métricas y escalas. A continuación, se determinan las preferencias de los criterios y paralelamente, se selecciona el conjunto de alternativas a evaluar y el método de agregación a utilizar.

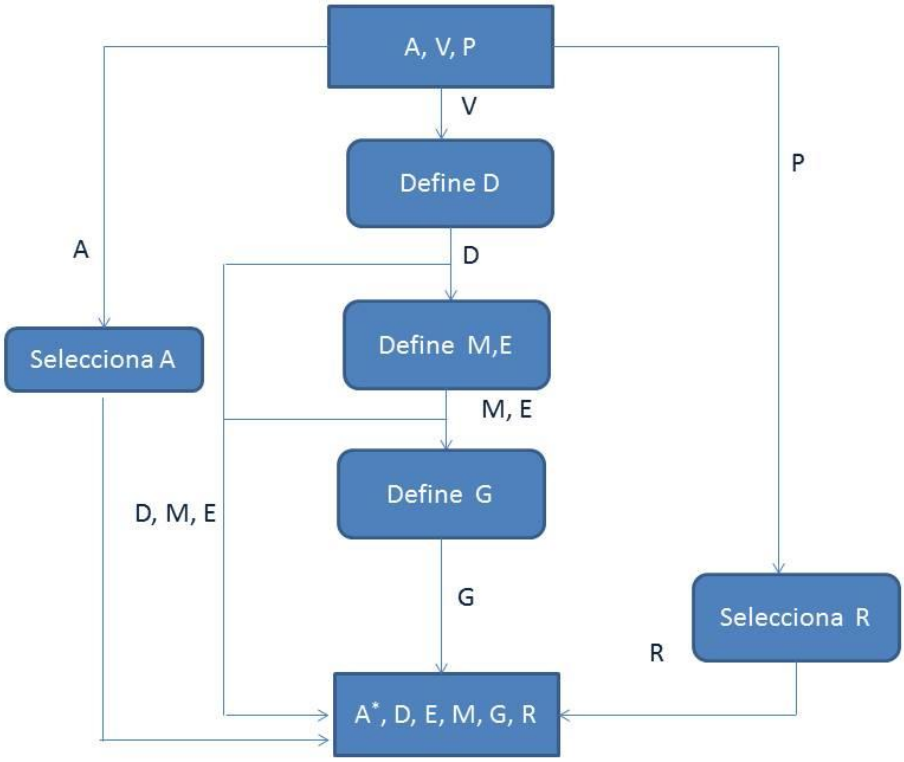

Figura II.2. Descomposición en tareas de la actividad Diseño del modelo de evaluación [48].

Otra solución, redefine la tupla anterior como 7-tupla **{A,T,Q,M,E,G,R}** [49], donde:

- **A** representa el conjunto de alternativas a evaluar.
- **T** constituye el tipo de evaluación, determinado por el tipo de resultado deseado. En el caso de una clasificación, se responde con una valoración de la calidad; si es la selección de un producto entre varios, se responde con un ranking de productos; en otro caso, simplemente con una valoración descriptiva o gráfica de las características del producto.
- **Q** significa el modelo de calidad a utiliza; o sea, es el árbol de los criterios de evaluación. La raíz, representa el índice global de calidad, las ramas, las características y subcaracterísticas y las hojas, las diferentes medidas de calidad.
- **M** es el conjunto de medidas asociadas a los criterios.
- **E** es el conjunto de escalas asociadas a cada medida.
- **G** representa las preferencias del usuario para el conjunto de criterios
- **R** es el método de agregación de la información. Permite obtener la recomendación, dependiendo del tipo de resultado deseado.

En esta propuesta, la selección del método de agregación (R), no solo depende del propósito de la evaluación, sino también del conjunto de posibles opciones; de las escalas de medición; preferencia de los atributos; dependencia entre los atributos y la incertidumbre presente.

Por último, se desarrolla la actividad Aplicación del modelo de evaluación, donde las tareas se desarrollan como muestra la Figura II.3.

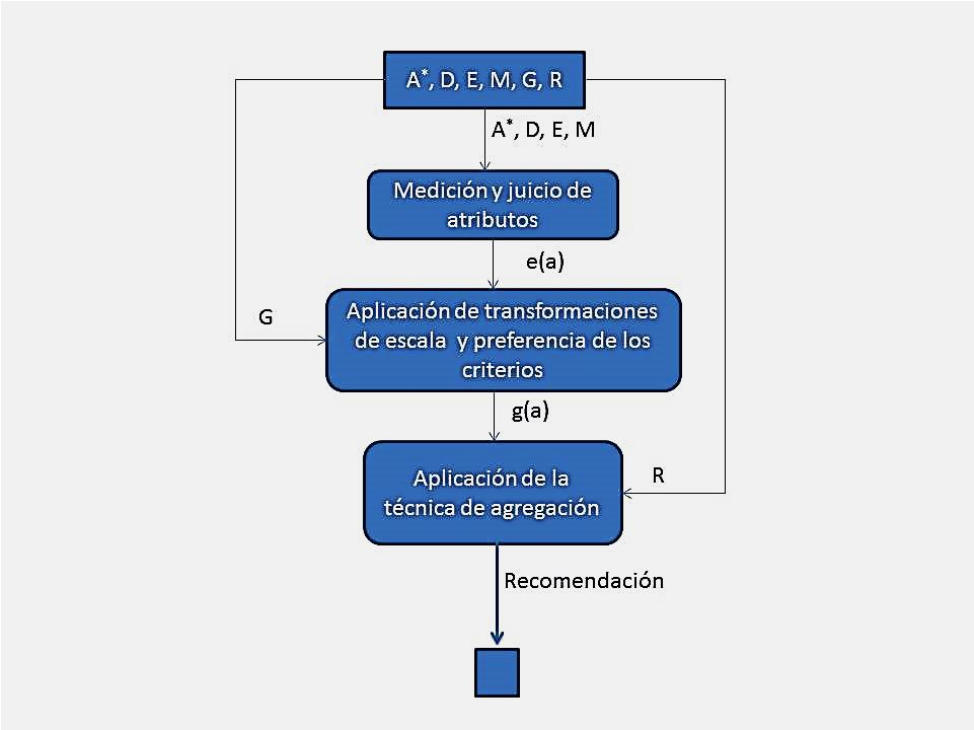

Figura II.3. Descomposición en tareas de la actividad Aplicación del modelo de evaluación [48].

La primera tarea, se corresponde con la evaluación de las alternativas. En ella, se obtiene el valor de cada medida de calidad (e(a)); se aplican las transformaciones necesarias de las escalas de medidas para llevar todos los datos a un dominio (g(a)) y se adicionan las preferencias de los criterios. Por último, se aplica la técnica de agregación seleccionada. En esta tarea, se obtiene una valoración de la calidad de cada producto de software, se ordena según el resultado, y en base al propósito de la evaluación, se elabora la recomendación. Con ello es posible recomendar el mejor producto o ranking de productos.

Otros trabajos revisados que abordan el problema, no modifican dicha definición. La asumen como referencia y solo describen determinados elementos, según un problema más específico. Por ejemplo, si evalúan un tipo de software, describen el modelo de calidad y el conjunto de medidas a utilizar. ([80], [56], [65], [113], [144] y [145])

En el caso de transformar los distintos tipos de datos a un dominio, lo realizan de dos formas. La primera, convierte todos los datos a un dominio numérico, en ocasiones definido en el intervalo [0,1] ([48], [49] y [69]). Otra solución, define todos los datos como conjuntos difusos ([53], [54], [56] y [65]).

En el momento de agregación de la información, si los datos son números difusos, soluciona mediante dos enfoques. El primero, desfuzifica, agrega y fuzificar. El segundo, agrega los números difusos y los compara para obtener la recomendación. A partir del enfoque asumido, se detallan y se modifican diferentes métodos de agregación ([53], [54], [56], [66], [67] y [113]).

Otros trabajos, tienen en cuenta la interdependencia entre criterios pero solo la expresan al seleccionar ANP como método de decisión multicriterio ([67] y [143]).

La solución, no define las restricciones de los criterios esenciales y la interdependencia entre los diferentes criterios. En la próxima sección, se propone un modelo, donde se especifica cada elemento y se adicionan los que faltan.

II.2 Modelo computacional para la evaluación y selección de productos de software según su calidad

Como ya se expresó anteriormente, la evaluación de la calidad de los productos de software, consiste en juzgar cuán bueno puede ser un software o conjunto de ellos para determinados objetivos.

En los modelos asociados a la evaluación analizados, hay parámetros importantes que no se contemplan; tal es el caso de: (1) la incorporación de criterios esenciales y las restricciones asociadas a ellos, (2) la interdependencia entre los criterios de evaluación y (3) la incertidumbre que caracteriza el proceso.

Los criterios esenciales, son aquellos que describen los aspectos que el software bajo evaluación debe cumplir. A estos se le asocian restricciones determinadas a partir de un valor umbral. Si el producto no cumple dichas restricciones, debe penalizarse el valor de la característica o subcaracterística, asignándole el valor 0.

En los modelos de calidad, se ha determinado la interdependencia entre criterios ubicados en un mismo nivel. La calidad se descompone en características, las cuales, entre sí, pueden tener relaciones directas, inversas o no tener relaciones. Lo mismo sucede con las subcaracterísticas o medidas.

En el proceso de evaluación, es importante tener en cuenta que el aumento de un criterio puede influir en el decrecimiento o aumento de los otros. Es significativo que el análisis de dicha interdependencia a nivel de características, permite ordenar las pruebas de software a ejecutar, lo que implica un ahorro de tiempo y costo.

En la evaluación, es manipulada una gran variedad de medidas, basadas en factores objetivos y subjetivos. Esto provoca que la valoración de algunas de ellas sea vaga e imprecisa. Además, la incorporación de expertos con diferentes criterios y conocimientos acerca del proceso, aumenta la incertidumbre del mismo. Es usual que los decisores modelen el problema usando conocimientos de expertos, pero casi siempre la información la expresan lingüísticamente, tanto para la toma de decisiones, como para concretar medidas de calidad.

La revisión documental practicada durante la presente investigación, arrojó que los modelos, métodos, metodologías para la evaluación de la calidad de los productos de software, no abordan las situaciones expresadas con anterioridad ([48], [49], [53], [56], [65], [66], [67], [69] y [72]); por ello, en este epígrafe se propone un nuevo modelo y diferentes mecanismos operativos de solución.

La propuesta resultante de la presente investigación, parte del planteamiento del problema, prosigue con la formulación del objetivo del modelo, y luego presenta sus bases y principios, que aportan los elementos esenciales sobre los cuales se sustenta. Finalmente, se realiza una descripción detallada del mismo.

Planteamiento del problema:
Se tiene un grupo de productos de software $\mathbf{A} = \{\mathbf{A_1}, \dots \mathbf{A_m}\},\ \mathbf{m} > \mathbf{0},\ \mathbf{m} \in \mathbb{N}$**,** a evaluar a partir de un conjunto de requisitos de calidad $\boldsymbol{REQ} = \{\boldsymbol{R_1}, \dots \boldsymbol{R_r}\}$.

La evaluación de la calidad de los productos de software ($ECPS$), se define como la tupla $\{\boldsymbol{A}, \boldsymbol{MC}\}$, donde:

- $\boldsymbol{A}$ es el conjunto no nulo de productos a evaluar.
- $\boldsymbol{MC}$ representa el modelo de calidad a utilizar para evaluar cada $A_i \in A, 1 \leq i \leq m$.
- **r** significa la cantidad de requisitos a tener en cuenta durante la evaluación

A continuación, se detalla cada elemento de $ECPS$.

$\boldsymbol{A}$ (Conjunto de alternativas)

El conjunto $\boldsymbol{A}$ contiene los productos de software a evaluar, ya sea para recomendar uno o para ordenarlos de acuerdo a su calidad.

$$\boldsymbol{A} = \{a_1, a_2, \dots, a_m\},\ m > 0, \boldsymbol{m} \in \mathbb{N} \qquad \text{(II.1)}$$

Donde **m** es la cantidad de productos a evaluar

MC (Modelo de Calidad)

La evaluación, se realiza sobre un modelo de calidad que estructura jerárquicamente los diferentes criterios, como ya se expresó en el Capítulo I. A partir de los requisitos, se define el modelo a utilizar. Un modelo de calidad (MC) es un grafo (Ver Figura II.4).

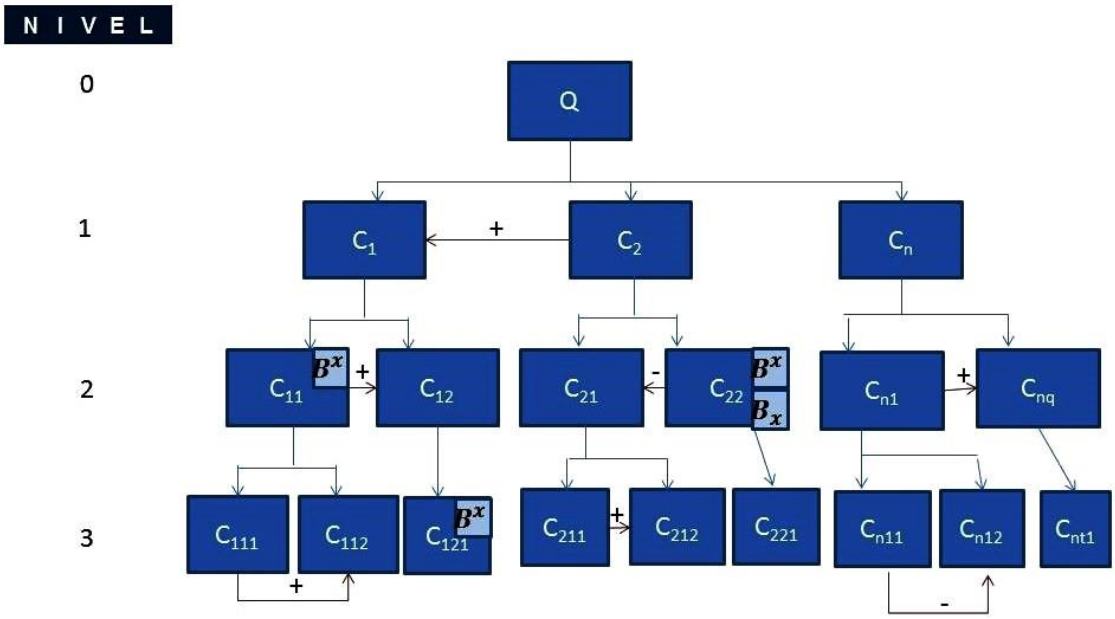

Figura II.4 Modelo de Calidad

$$MC = (V,\ E_v,\ E_h, CE) \qquad \text{(II.2)}$$

Donde:

- $\boldsymbol{V}$ es el conjunto de criterios de evaluación.
- $\boldsymbol{E_v}$ es el conjunto de enlaces verticales.
- $\boldsymbol{E_h}$ es el conjunto de enlaces horizontales (influencia)
- $\boldsymbol{CE}$ es el conjunto de criterios esenciales

El modelo está formado por cuatro niveles: en el nivel 0, se representa la calidad (ver Ecuación II.3); en el nivel 1, se representan las características; en el nivel 2, se encuentran las sub características; y en el nivel 3, están las medidas obtenidas a partir del proceso de pruebas de software y de los juicios de los expertos.

$$Nivel_0 = \{Q\} \quad \bigcap_{0 \le j \le l} Nivel_j = \emptyset, \qquad \bigcup_{0 \le j \le l} Nivel_j = V \qquad \text{(II.3)}$$

Un criterio solamente se encuentra en un nivel y la unión de todos se corresponde con la totalidad de los criterios a valorar.

Los enlaces verticales ($\boldsymbol{E_v}$) representan la relación entre vértices de niveles consecutivos, representando la relación entre padre e hijos (Ver Ecuación II.4).

$$E_v \subset VxV$$

$$E_v = \{(y,x) \ / \ x,y \in V,\ x \in Nivel_j,\ y \in Nivel_{j-1}\} \qquad \text{(II.4)}$$

Los criterios de todos los niveles tienen un padre que se encuentra en el nivel consecutivo anterior, excepto el nivel 0, que representa el índice de calidad (ver Ecuación II.5)

$$\forall_{0<j\leq l}\, Nivel_j = \left\{x/(y,x) \in E_v,\ x \in V,\ y \in Nivel_{j-1}\right\} \qquad \text{(II.5)}$$

Cada criterio, tiene asociado un peso (W). En cada nivel hay un conjunto de vectores de peso. La suma de los pesos de los criterios hermanos (hijos de un mismo padre) es igual a 1 (Ver Ecuación II.6).

$$W: V \to \mathbb{R},\ es\ el\ peso\ de\ los\ criterios$$

$$\forall_{x \in V-Nivel\ l} \sum_{\forall_{y,\ (x,y)\in E_v}} w(y) = 1. \qquad \text{(II.6)}$$

Como se mostró en el Capítulo I, en los modelos de calidad más utilizados existe dependencia entre los criterios definidos en cada nivel. La relación entre los criterios pudiera ser directa (+), inversa (-) o neutra (o). La relación neutra significa que no hay influencia; la directa, se presenta cuando al aumentar un criterio, aumenta también el criterio sobre el cual él influye (por ejemplo, al aumentar la confiabilidad, aumenta la funcionalidad); la inversa, por su parte, es cuando al aumentar un criterio, el criterio dependiente disminuye (por ejemplo, al aumentar la eficiencia, la usabilidad disminuye).

Estas dependencias, pueden representarse a través de los enlaces horizontales (E_h). Cada enlace se presenta por una tripleta (x, y, I_{xy}), que significa la influencia del criterio *x* sobre el criterio *y* con un peso (I_{xy}). El valor (I_{xy}) es una etiqueta lingüística. Las etiquetas lingüísticas se definen como: negativamente alto (na), negativamente bajo (nb), sin influencia (sin), positivamente bajo (pb) y positivamente alto (pa), definidas todas en el universo de [-1,1] (Ver Ecuación II.7).

$$E_h \subset VxVx\ \{na, nb, sin,\ pb, pa\}$$

$$E_h = \left\{\left(x, y,\ I_{xy}\right) / \ x,y \in V,\ \exists z\ [(z,x) \in E_v,\ \ (z,y) \in E_v],\ I_{xy} \in \{na, nb, sin,\ pb, pa\}\right\}$$

$$\forall_{x,y}[(x, y, I_{xy}) \in E_h \Rightarrow (y, x, I_{xy}) \notin E_h] \qquad \text{(II.7)}$$

En el modelo, existe un subconjunto de criterios clasificados como esenciales (CE), determinados a partir de los requisitos de calidad y el propósito de la evaluación. A

cada criterio esencial, se asocia una restricción, a la cual se le asocia, a su vez, un intervalo ($\boldsymbol{B_x}$, $\boldsymbol{B^x}$). Los valores del intervalo, están definidos en el mismo dominio de los datos donde se define el criterio y el valor que alcanza este en la evaluación, debe encontrarse en el intervalo. A partir del resultado de esta comparación, se decide si se penaliza o no la alternativa objeto de valoración. Los valores $\boldsymbol{B_x}$, $\boldsymbol{B^x}$ se determinan a partir de estudios históricos, de mercado o de control estadístico.

$$CE \subset Vx\mathbb{R}x\mathbb{R},$$

$$CE = \{(x, B_x,\ B^x)\ \ B^x\ , B_x\ \in \mathbb{R},\ , B_x < B^x\} \qquad \text{(II.8)}$$

Al realizar la evaluación de los productos de software, a través de pruebas y valoraciones de expertos, se obtiene un conjunto de elementos de medidas de calidad $E = \{x_i\}$; donde x_i es un dato que representa una unidad de tamaño, cantidad, tiempo o apreciación cualitativa. Dicho dato, constituye la base para obtener las medidas de calidad, y estas a su vez son funciones del dato $(M_j = f(x_i))$.

Cada medida se define en un dominio de datos con una escala. Si la medida es de naturaleza cualitativa, la escala a utilizar es ordinal; si fuera cuantitativa, se usan escalas de intervalo o razón. En la Tabla II.1, se muestran ejemplos de medidas de calidad, el nombre de estas, la función que permite obtenerlas, el dominio de datos en que se definen y el valor óptimo de las mismas. El valor óptimo, se corresponde con el valor máximo, si la medida es de beneficio, o con el valor mínimo si es de costo.

Tabla II.1. Ejemplos de medidas de calidad.

Medida de Calidad	***Función que determina la medida de calidad f(x)***	***Dominio***	***Valor óptimo***
Inclusividad funcional	$X = \frac{A}{B}$ *A:* Número de funciones que se ejecutan de manera similar y sin cambios. *B:* Número de funciones probadas que deben tener comportamiento similar	$[0,1]$	1
Tiempo medio entre fallos (MTBF)	$X = \frac{T}{A}$ *T*: Tiempo de operación *A* Número total de errores detectados	$[0,\infty)$	$V_{máx}(\infty)$
Precisión	$X = \frac{A}{T}$ A = Número de resultados encontrados por los usuarios con un nivel de precisión diferente al requerido T = tiempo de operación		0

En estas valoraciones intervienen varios expertos, que evalúan aspectos específicos del software, a partir de la observación y percepción, por lo que se caracteriza este entorno como cambiante y complejo, con incertidumbre.

Objetivo del modelo:

El objetivo principal del modelo es obtener la valoración de la calidad de uno o varios productos de software, a partir de la información obtenida de las pruebas y opiniones de los expertos involucrados en el proceso de evaluación, sobre la base de un modelo de calidad (incluye la interdependencia entre criterios y las restricciones de los criterios esenciales). Es importante que este modelo sea operativo y los resultados que brindan sean de fácil comprensión por la industria.

El modelo, brinda un marco de trabajo que facilita el análisis de la información y predice el valor de la calidad de los productos de software. Al aplicarlo, permite la selección del producto que mejor responde a los objetivos previamente establecidos, u ordenarlos de acuerdo a su calidad.

Fundamentos teóricos y principios del modelo:

Partiendo de las consideraciones anteriores, y del análisis teórico realizado, se establecen como fundamentos teóricos del modelo:

- La modelación de la información se realiza con la normalización de las medidas, el análisis de los conflictos costo/beneficio de los criterios y la unificación de las medidas en valores lingüísticos. A cada término lingüístico, se asocia un número triangular difuso.
- La interdependencia entre criterios, se trata mediante Mapas Cognitivos Difusos.
- Los criterios esenciales, se aplican sobre la base de restricciones y definición de un vector de penalización.

El modelo, se sustenta sobre los siguientes principios:

1. **Principio de la interpretación/comprensión basada en estándares.** Para garantizar la interpretación correcta y comprensión por la industria, se basa en los estándares de calidad de software definidos.

2. **Principio de la agregación basada en vectores de ponderación.** Para garantizar su fiabilidad, se necesita considerar las particularidades de los expertos, sin desechar ningún experto o criterio.

3. **Principio de adecuación a los cambios y racionalidad**. Permite cambios en los parámetros, identificados y definidos formalmente. Además, es capaz de obtener una respuesta para los usuarios en el tiempo adecuado, con un mínimo gasto computacional.

4. **Principio de la adecuación del resultado al cliente.** El resultado estará expresado con datos numéricos y/o etiquetas lingüísticas, según el deseo del cliente.

Descripción general del modelo:

El nuevo modelo, se basa en las fases definidas para la solución de un problema de toma de decisiones, al cual se adiciona la modelación de la información, la determinación de los criterios esenciales, el análisis/modelación de la interdependencia entre criterios y el manejo de datos heterogéneos y con incertidumbre. Tiene como referencia la metodología analizada en el epígrafe anterior y los estándares de calidad.

La representación gráfica del nuevo modelo, se presenta en la Figura II.5.

Entradas y salidas del modelo.

El modelo tiene como entradas:

- El conjunto de los diferentes productos de software a valorar (las alternativas).

$$\mathrm{A} = \{\mathrm{A}_1, \ldots \mathrm{A}_\mathrm{m}\},\ \mathrm{m} > 0,\ \ \mathrm{m} \in \mathbb{N} \qquad \text{(II.9)}$$

- El conjunto de requisitos a tener en cuenta para la evaluación.

$$REQ = \{R_1, \ldots R_\mathrm{r}\}, \mathrm{r} > 1, \mathrm{r} > 1, \mathrm{r} \in \mathbb{N} \qquad \text{(II.10)}$$

Como resultado, se obtiene la recomendación de un producto o el ordenamiento de un grupo de estos, de acuerdo a su calidad. Dicha recomendación se realiza según el Índice de Calidad (**IQ**) calculado y el tipo de evaluación deseada. El IQ, definido en la presente investigación, indica la medida en que la calidad del producto es valorada, y se hace corresponder con la variable lingüística Nivel de Calidad (**NQ**). NQ representa la valoración de la calidad del producto y describe el IQ.

NQ se describe sin especificar su representación, debido a que el modelo propuesto se abstrae de la definición de los términos lingüísticos y el dominio de datos al que pertenece la variable. Estos son definidos de acuerdo a los intereses del cliente y deben ser ajustados en dependencia del contexto de aplicación del modelo.

Para la recomendación u ordenamiento, se utiliza el IQ de cada A_i.

$$\{(A_i, NQ_i)\}\ 1 \leq \mathrm{i} \leq \mathrm{m}\ \mathrm{A}_\mathrm{i}^1 \succcurlyeq \mathrm{A}_\mathrm{i}^2 \succcurlyeq \cdots \succcurlyeq \mathrm{A}_\mathrm{i}^\mathrm{m} \qquad \text{(II.11)}$$

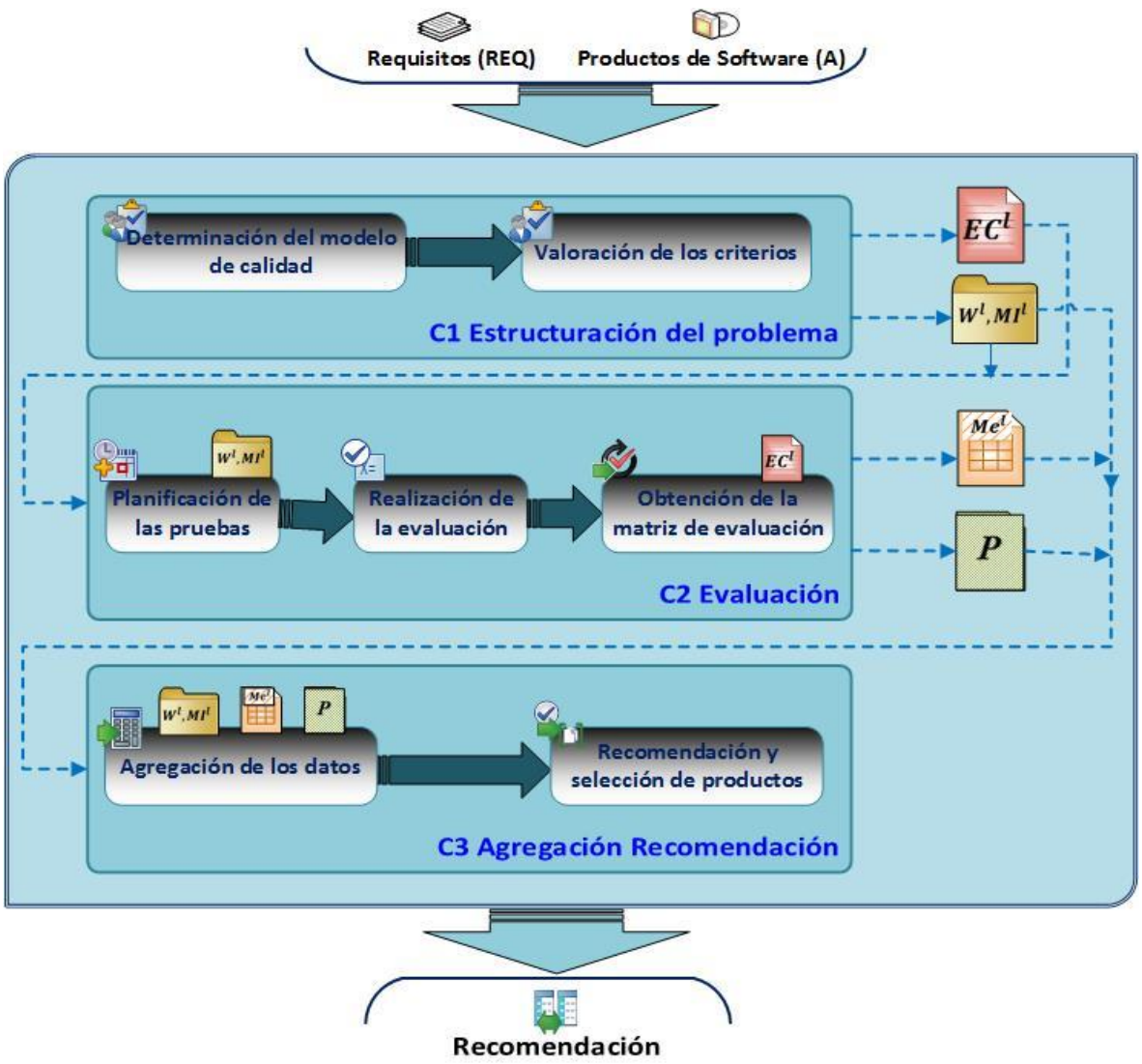

Figura II.5. Nuevo modelo para la evaluación y selección de productos de software según su calidad.

Descripción de los componentes del modelo

La articulación entre los diferentes componentes del modelo, está dada sobre la base de las propias relaciones de información establecidas entre las entradas y salidas que en cada uno de ellos se utilizan y se procesan. Se dirige a cumplir el objetivo declarado de obtener la valoración de la calidad de uno o varios productos de software, a partir de un modelo de calidad, que incluya la interdependencia entre criterios y las restricciones de los criterios esenciales.

Para facilitar la comprensión del modelo, se desarrollará un ejemplo de evaluación de la calidad de tres productos de software, paralelamente a la explicación de este, con lo cual se ilustra la funcionalidad de cada componente.

C1. "Estructuración del problema"

El primer componente del modelo se denomina "Estructuración del problema" y se identifica como **C1.** C1 tiene como entradas el conjunto de alternativas a evaluar y requisitos de calidad. Su función es identificar el modelo de calidad (MC) sobre el cual se realizará la evaluación. Se determinan los elementos del MC: los criterios y medidas a evaluar; el conjunto de criterios esenciales y restricciones asociadas a éste; el peso de los criterios e interdependencia entre los mismos.

Los enlaces horizontales (E_h) reflejan la interdependencia entre criterios. Si se analiza la definición de estos (Ecuación II.7) y los mapas cognitivos difusos (MCD), tratados en la Sección I.7, del Capítulo I, es factible modelar y representar la influencia a través de esta herramienta.

Para cada nivel, se analizan los enlaces horizontales con un MCD. Para su modelación, los vértices representan los criterios de un mismo nivel y las aristas la influencia de un criterio sobre otro. Para colocar el valor de las aristas, se propone la variable lingüística Influencia (I), donde I describe la influencia entre criterios de un mismo nivel y se valora con una etiqueta de la variable. La variable Influencia tiene cinco etiquetas lingüísticas: negativamente alto (NA), negativamente bajo (NB), sin Influencia (SInf), positivamente bajo (PB) y positivamente alto (PA).

En la Figura II.6, se definen las etiquetas, teniendo como universo del discurso el intervalo $[-1,1]$.

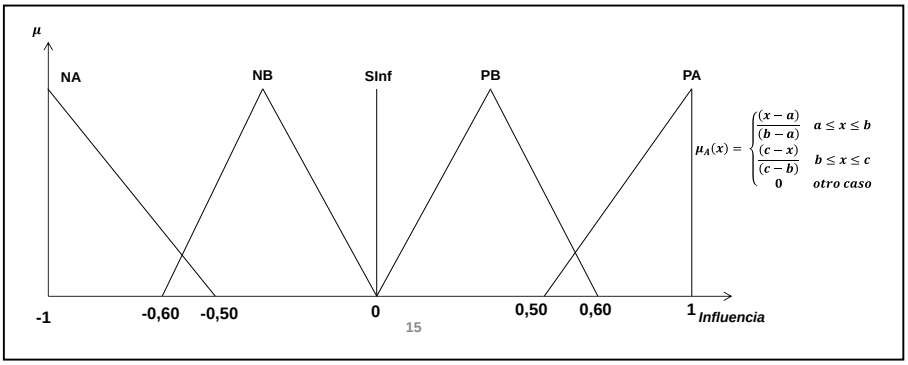

Figura II.6 Variable lingüística Influencia.

El MCD se representa con una matriz, denominada matriz de interdependencia entre criterios (MI). Se obtiene una por cada nivel.

$$\widehat{MI}^{l} = \begin{bmatrix} \widehat{y_{1,1}} & \cdots & \widehat{y_{1,J}} & \cdots & \widehat{y_{1,n}} \\ \vdots & & \vdots & & \vdots \\ \widehat{y_{\iota,1}} & \cdots & \widehat{y_{\iota,J}} & \cdots & \widehat{y_{\iota,n}} \\ \vdots & & \vdots & & \vdots \\ \widehat{y_{n,1}} & \cdots & \widehat{y_{n,J}} & \cdots & \widehat{y_{n,n}} \end{bmatrix} \qquad \text{(II.12)}$$

A continuación, se explica la organización y tareas fundamentales del componente **C1**.

C1 se divide en dos subcomponentes:

- **C1.1 Determinación del modelo de calidad**: determina y adapta el modelo de calidad a usar a partir de los requisitos, sin el análisis de los enlaces horizontales; establecen los criterios a valorar (características, subcaracterísticas y medidas) con sus enlaces verticales y niveles; formalizan para cada medida, el dominio de los datos, la escala en que está definida; y prefijan los criterios esenciales y restricciones asociadas a ellos.
- **C1.2 Valoración de los criterios**: determina el peso de los criterios en dependencia de los requisitos de calidad; modela y analiza la interdependencia entre los criterios, a través de los MCD.

El primer subcomponente **C1.1 Determinación del modelo de calidad**, se estructura como muestra la Figura II.7.

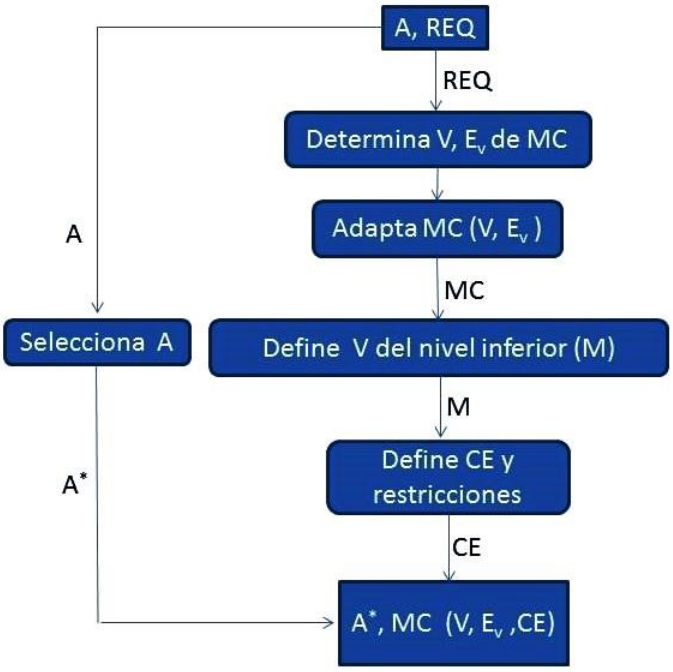

Figura II.7. Subcomponente **C1.1 Determinación del modelo de calidad.**

A partir de los requisitos de calidad (REQ), se determinan las características, sub-características y medidas a evaluar, lo que conlleva a la definición de un modelo de

calidad con una estructura jerárquica, donde aún no precisan los enlaces horizontales. A continuación, se analiza si es preciso agregar una característica no definida en el modelo, lo que implica su adaptación. Cada medida seleccionada, se define en un dominio de datos, con una escala determinada. Después, se analiza cada criterio y se establece si es esencial o no. A cada criterio esencial, se le asocian dos valores (B_x, B^x), que determinan la restricción $(B_x \leq x \leq B^x)$, con el objetivo de penalizar la evaluación. Dicho proceso, se ejecuta en los diferentes niveles del grafo.

Para facilitar la comprensión del modelo, se demuestra cada paso mediante el siguiente ejemplo:

> *Ejemplo II.1* Se desea obtener la evaluación de la calidad de tres productos de software (S1, S2, S3). Para ello, se precisa tener en cuenta los siguientes requisitos:
>
> - REQ 1. Todas las funcionalidades incluidas deben ejecutarse correctamente, con alta precisión.
> - REQ 2. Alcanzar un tiempo de respuesta medio y hacer un uso adecuado de la memoria.
> - REQ 3. Debe ser de fácil operación.
> - REQ 4. Interfaz amigable, que satisfaga al usuario.
> - REQ 5. Poder ejecutarse en diferentes sistemas operativos.
> - REQ 6. De fácil instalación.
>
> A partir de los requisitos de software, se determinan las características y subcaracterísticas a evaluar.
>
> - REQ 1.
> - Características: Adecuación funcional, Fiabilidad
> - Subcaracterísticas: Completitud funcional, corrección funcional y madurez
> - REQ 2.
> - Característica: Eficiencia
> - Subcaracterísticas: Comportamiento temporal y uso de recursos
> - REQ 3, 4.
> - Característica: Usabilidad
> - Subcaracterísticas: Capacidad para operar, Capacidad para reconocer su adecuación, Capacidad de aprendizaje, Estética.

- REQ 5,6.
    - Característica: Portabilidad
    - Subcaracterísticas: Capacidad para ser instalado, Adaptabilidad.

  Se decide usar el modelo de calidad definido en la norma 25010 [146] con adaptaciones, ya que no se evalúan todas las características.

En el Anexo III, se detallan las medidas a utilizar en la evaluación. De cada medida, se determina el dominio de datos en que se define (Ver Tabla II.2), y si es de costo (minimizar) o beneficio (maximizar). En este caso los dominios numéricos son intervalos y el resto son ordinales. Es esencial, para una mayor confianza en la evaluación de la calidad, procesar las medidas que respondan tanto a factores cualitativos como cuantitativos. Como por ejemplo en el caso de la usabilidad las medidas de satisfacción y de estética.

Se determinan como criterios esenciales del nivel más bajo el conjunto de medidas mostrado en la Tabla II.3, con el correspondiente valor umbral. Además, define que no existen criterios esenciales del nivel de las características y subcaracterísticas.

El otro subcomponente, **C1.2 Valoración de los criterios**, se organiza como muestra en la Figura II.8.

La primera tarea a realizar es valorar la importancia de cada criterio, a partir de la opinión de los expertos. Hasta el momento, no se han analizado los enlaces horizontales. El peso de los criterios, puede establecerse a partir de la utilización de diferentes métodos, ya sea por determinación directa de los expertos o mediante la comparación por pares, obteniendo el autovector ([80], [62] y [66]). Debe tenerse en cuenta, que la suma de los pesos de los hijos de igual padre (hermanos), ha de ser igual a 1.

Para determinar la interdependencia entre criterios de un mismo nivel, se conforma el MCD a partir de la información brindada por los expertos. Se analiza primero la relación existente entre los criterios hermanos y se conforma un MCD para cada grupo de hermanos.

La propuesta del MCD inicial, se obtiene a partir de la revisión bibliográfica respecto a la interdependencia de criterios presente en el modelo de calidad seleccionado. Los vértices son los criterios hermanos y las aristas representan la influencia de un criterio sobre otro.

Tabla II.2 Definición del dominio de datos de las medidas.

Criterios	Adecuación funcional		Eficiencia	
	Completitud funcional	Corrección funcional	Comportamiento temporal	Uso de recursos
	Cobertura de aplicación funcional (CAF)	Corrección Computacional (CC)	Tareas por tiempo (TpT)	Uso de Memoria (MEM)
Dominio	[0,1]	[0,1]	[0, 40]	[0, 10]
Costo/ Beneficio	B	B	B	C

Criterios	Fiabilidad			Portabilidad		
	Madurez			Capacidad para ser instalado	Adaptabilidad	
	Tiempo medio entre fallos (MTBF)	Densidad de fracasos frente caso de pruebas (DF)	Cobertura de las pruebas (CP)	Facilidad de instalación (Fins)	Adaptabilidad de Hardware (Ahw)	Adaptabilidad de Software (Asw)
Dominio	[0, 100]	[0,1]	[0,1]	(MB, B,M, A, MA)	[0,1]	[0,1]
Costo/ Beneficio	B	C	B	B	B	B

Usabilidad			
Capacidad para reconocer su adecuación	Capacidad de aprendizaje	Capacidad para operar	Estética
Descripción Exhaustiva (DE)	Integridad de la documentación (IntDoc)	Satisfacción (Sat)	Apariencia (Apa)
[0,1]	[0,1]	(MB, B,M, A, MA)	(NA, PA, A, MA, AA)
B	B	B	B

Tabla II.3 Medidas que se consideran criterios esenciales con su valor umbral.

	Cobertura de aplicación funcional (CAF)	Corrección Computacional (CC)	Tareas por tiempo (TpT)	Uso de Memoria (MEM)	Tiempo medio entre fallos (MTBF)	Densidad de fracasos frente caso de pruebas (DF)	Cobertura de las pruebas (CP)	Satisfacción (Sat)
Umbral	[0,25;1]	[0,25;1]	[3;40]	[0;1,25]	[5, 100]	[0;0,5]	[0,5;1]	(M, A, MA)

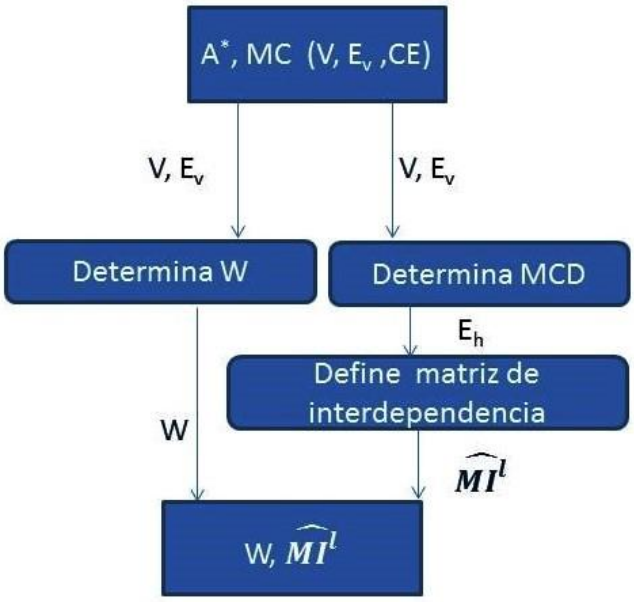

Figura II.8. Subcomponente **C1.2 Valoración de los criterios.**

El MCD resultante, es revisado por cada experto y a cada arista se asocia un valor de la variable I (Influencia). El MCD obtenido por cada experto, debe ser agregado luego, utilizando una técnica que permita el consenso; para lo cual es recomendable utilizar un algoritmo de logro de consenso [135], que puede ser simple, considerando que se cuenta con *k* expertos y *n* criterios.

Algoritmo 1 Pseudocódigo del algoritmo para la unificación de k MCD con logro de consenso

Entradas:

K grafos de n vértices,

p $umbral\ de\ decisión,\ p\ \in\ [0,1]$

Salida:

G *Grafo*

Inicio

1. Para cada par de vértices:
2. Examina el valor que cada *k* experto ha asignado
3. Si existen valores con diferentes signos y el número de valores con igual signo es menor que pk , entonces:
4. Volver a preguntar por ese valor, e ir al paso 3
5. Si no
6. Con los valores de igual signo, calcular el máximo valor y asignar el valor de ese arco a G.
7. Si no han sido examinados todos los nxn arcos, ir al paso 2.
8. Retorno G

Fin

Se obtiene el mapa cognitivo difuso consensuado ($MCDc$) para cada grupo de hermanos. De cada $MCDc$, se halla la matriz de adyacencia (MAc), se combinan las

diferentes MAc de cada nivel y se determina la matriz de interdependencia entre criterios.

La unificación de las diferentes MAc es sencilla, porque no existen criterios comunes entre los diferentes mapas. La combinación se realiza de acuerdo a la Ecuación II.13.

$$\begin{bmatrix} MAc_1 & sinf & sinf \\ sinf & MAc_2 & sinf \\ sinf & sinf & MAc_k \end{bmatrix} \qquad \text{(II.13)}$$

$\widehat{\boldsymbol{MI}}^{\boldsymbol{l}}$ representa la matriz de interdependencia entre los criterios del nivel l.

$$\widehat{\boldsymbol{MI}}^{\boldsymbol{l}} = \begin{bmatrix} \widehat{\boldsymbol{y}_{1,1}} & \cdots & \widehat{\boldsymbol{y}_{1,J}} & \cdots & \widehat{\boldsymbol{y}_{1,n}} \\ \vdots & & \vdots & & \vdots \\ \widehat{\boldsymbol{y}_{\iota,1}} & \cdots & \widehat{\boldsymbol{y}_{\iota,J}} & \cdots & \widehat{\boldsymbol{y}_{\iota,n}} \\ \vdots & & \vdots & & \vdots \\ \widehat{\boldsymbol{y}_{n,1}} & \cdots & \widehat{\boldsymbol{y}_{n,J}} & \cdots & \widehat{\boldsymbol{y}_{n,n}} \end{bmatrix} \qquad \text{(II.14)}$$

El primer componente, obtiene como salida $MC(V, E_v, E_h, CE)$.

Desarrollando este subcomponente para el Ejemplo II.1, lo primero sería obtener el peso de los criterios, a partir de su valoración por los expertos. Se realiza la comparación y se deriva el conjunto de vectores de peso de cada nivel ($W = \{W^1, W^2, W^3\}$) (Ver Tabla II.4).

Posteriormente, se definen la relación existente entre criterios en cada nivel, con el uso de los MCD (Figura II.9), y las matrices de interdependencia de cada nivel ($\widehat{\boldsymbol{MI}}^{\boldsymbol{l}}$).

Para el nivel 1, se deriva del MCD la matriz MI^1, quedando:

$$\begin{array}{c} \\ AF \\ E \\ F \\ P \\ U \end{array} \begin{array}{c} \begin{array}{ccccc} AF & E & F & P & U \end{array} \\ \begin{bmatrix} Sinf & Sinf & Sinf & Sinf & Sinf \\ Sinf & Sinf & Sinf & Sinf & NB \\ PA & Sinf & Sinf & Sinf & Sinf \\ Sinf & NA & Sinf & Sinf & Sinf \\ PA & Sinf & PB & Sinf & Sinf \end{bmatrix} \end{array} \qquad \text{(II.15)}$$

Tabla II.4 Vectores peso para cada nivel.

Para $l = 1$

Características	**Peso**
Adecuación funcional (AF)	0.324
Eficiencia (E)	0.075
Fiabilidad (F)	0.324
Portabilidad (P)	0.072
Usabilidad (U)	0.206

Para $l = 2$

Características	Subcaracterísticas	Peso
Adecuación funcional (AF)	Completitud funcional (CpF)	0.75
	Corrección funcional (CcF)	0.25
Eficiencia (E)	Comportamiento temporal (Ct)	0.667
	Uso de recursos (UR)	0.333
Fiabilidad (F)	Madurez (M)	1
Portabilidad (P)	Capacidad para ser instalado (CapI)	0.25
	Adaptabilidad (A)	0.75
Usabilidad (U)	Capacidad para reconocer su adecuación (Comp)	0.152
	Capacidad de aprendizaje (Apr)	0.211
	Capacidad para operar (Op)	0.400
	Estética	0.237

Para $l = 3$

Características	Subcaracterísticas	Métrica	Peso
Adecuación funcional (AF)	Completitud funcional (CpF)	Cobertura de aplicación funcional (CAF)	1
	Corrección funcional (CcF)	Corrección Computacional (CC)	1
Eficiencia (E)	Comportamiento temporal (Ct)	Tareas por tiempo (TpT)	1
	Uso de recursos (UR)	Uso de Memoria (MEM)	1
Fiabilidad (F)	Madurez (M)	Tiempo medio entre fallos (MTBF)	0.4
		Densidad de fracasos frente caso de pruebas (DF)	0.4
		Cobertura de las pruebas (CP)	0.2
Portabilidad (P)	Capacidad para ser instalado (CapI)	Facilidad de instalación (Fins)	1
	Adaptabilidad (A)	Adaptabilidad de Hardware (Ahw)	0.5
		Adaptabilidad de Software (Asw)	0.5
Usabilidad (U)	Capacidad para reconocer su adecuación (Comp)	Descripción Exhaustiva (DE)	1
	Capacidad de aprendizaje (Apr)	Integridad de la documentación (IntDoc)	1
	Capacidad para operar (Op)	Satisfacción (Sat)	1
	Estética	Apariencia (Apa)	1

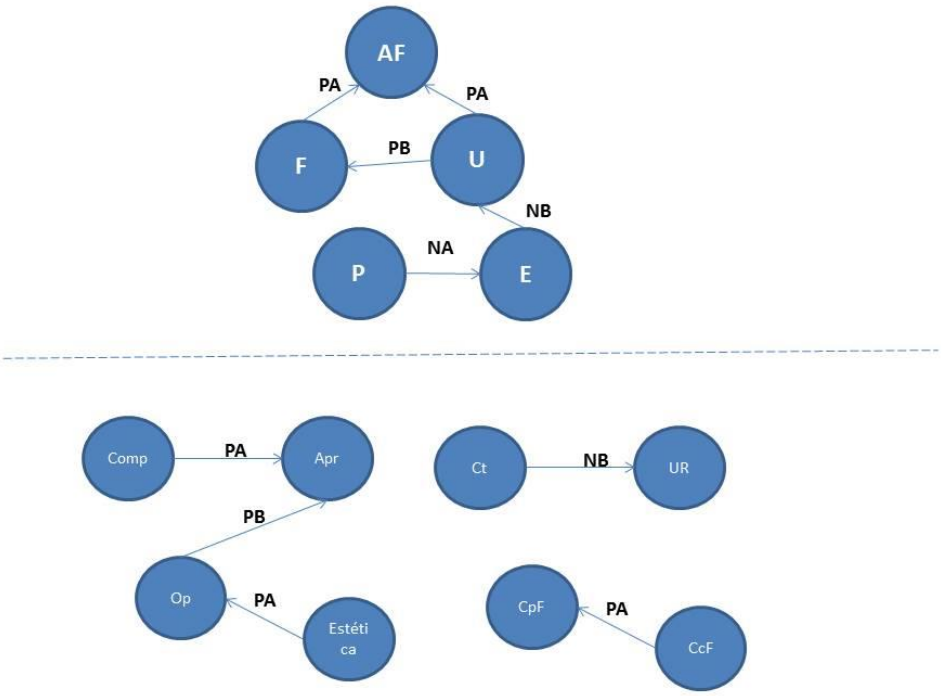

Figura II.9. Mapas Cognitivos Difusos del nivel 1 y 2.

C2. "Evaluación"

El segundo componente del modelo se denomina **"Evaluación" y s**e identifica con **C2**. Su función principal es realizar la evaluación de los diferentes productos de software y obtener una matriz de evaluación normalizada.

C2 se descompone en tres subcomponentes:

- **C2.1 Planificación de las pruebas de software**: Se determinan los tipos de pruebas a realizar, según los criterios y medidas seleccionadas. A partir del análisis de la interdependencia entre los criterios del nivel 1 y el vector peso asociado a dicho nivel, se recomienda un orden para la realización de las pruebas y se precisa el plan a ejecutar.
- **C2.2 Realización de la evaluación**: Se ejecutan los diferentes tipos de pruebas de software, evaluando cada alternativa y teniendo en cuenta el juicio de los expertos. Este proceso puede ser automático, manual o ambos. Con los datos recopilados, calcula el valor de cada medida.
- **C2.3 Obtención de la matriz de evaluación**: Se analizan las restricciones para los criterios esenciales y se define el vector de penalización.

Paralelamente, se normaliza la información de las diferentes medidas y se unifican al transformarlas a un mismo dominio de datos. Se conforma la matriz de evaluación con los valores de las medidas normalizadas y unificadas para cada alternativa.

El subcomponente **C2.1 Planificación de las pruebas de software** se estructura como muestra la Figura II.10. A partir del modelo de calidad, se establecen los tipos de pruebas a realizar y herramientas a utilizar. Posteriormente, se definen los especialistas necesarios.

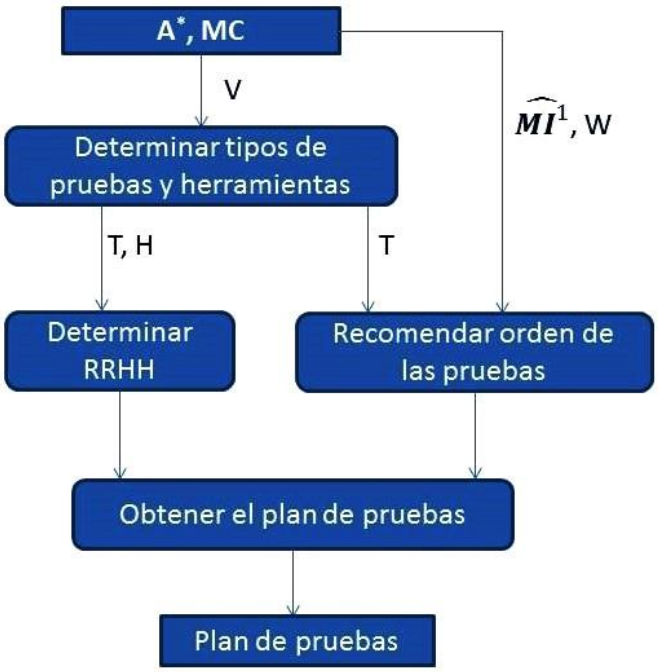

Figura II.10 Subcomponente **C2.1 Planificación de las pruebas de software.**

La interdependencia de los criterios de nivel 1 y el peso de los mismos, sirve de base para recomendar sobre cuáles características comienza la evaluación. Para ello, se aplica el algoritmo 2.

Lo primero es transformar el MCD, teniendo en cuenta solo el peso de los arcos sin signo; esto no es más que hallar el valor absoluto de cada elemento de la matriz de interdependencia ($\overline{MI^1}$), para lo cual define el valor absoluto de cada etiqueta, según Tabla II.5.

Algoritmo 2 Pseudocódigo del algoritmo para recomendar el orden de las pruebas de software.

Entrada:

Matriz de interdependencia de nivel 1 (MI^1),

Vector peso de nivel 1 (W^1),

Salida:

Orden de las pruebas a realizar a los productos de software

INICIO

1. *Hallar el valor absoluto de cada elemento de la MI^1*
2. *Calcular Matriz de efecto total*
3. *Calcular la centralidad de cada C_i*
4. *Calcular el ranking por la centralidad*
5. *Calcular el ranking por el peso.*
6. *Hallar la mediana de los rankings.*
7. *Ordenar por la mediana.*
8. *Retornar el orden*

FIN

Tabla II.5. Las etiquetas de la Variable Influencia y el valor absoluto de cada etiqueta.

Etiqueta (l_g)	$Abs(l_g)$
Negativamente Alta (NA)	Positivamente Alta (PA)
Negativamente Baja (NB)	Positivamente Baja (PB)
Sin influencia (Sinf)	Sin influencia (Sinf)
Positivamente Baja (PB)	Positivamente Baja (PB)
Positivamente Alta (PA)	Positivamente Alta (PA)

Se utiliza la Ecuación II.16 para el cálculo del valor absoluto.

$$Abs(\tilde{A}) = (\max((|b| - \delta), 0), |b|, min(|b| + \delta, 1)), \qquad \text{(II.16)}$$

$$\text{siendo } \delta = \begin{cases} \frac{|a-c|}{2} & si\ a \neq b \neq c \\ |a-c| & si\ a = b\ or\ b = c \end{cases}$$

Posteriormente, se halla la Matriz de efecto total, según la Ecuación II.17.

$$T = \sum_{i=1}^{n-1} \left(\widehat{MI}^1\right)^i \qquad \text{(II.17)}$$

Donde *MI* es la matriz de interdependencia de nivel 1 y *n* el número de criterios de ese nivel.

Se determina la centralidad de cada característica con la Ecuación II.18:

$$CEN\ (C_i) = Inf(C_i) + Dep(C_i) \qquad \text{(II.18)}$$

$$Inf(C_i) = \sum_{k=1}^{n} \bar{t}_{ik}$$

$$Dep(C_i) = \sum_{k=1}^{n} \bar{t}_{ki}$$

Se defuzifica, según la Ecuación II.19:

$$c(\tilde{A}) = \frac{a+4b+c}{6}. \qquad \text{(II.19)}$$

Se calcula el ranking, según el valor defuzificado de $CEN\,(C_i)$. A continuación, se calcula el ranking según el peso, se halla la mediana de los rankings y se ordena por la mediana. Es este el orden recomendado.

A partir de los datos anteriores, se confecciona el plan de pruebas.

Para explicar este algoritmo, se utiliza el Ejemplo II.1.

En este ejemplo, se evalúan las características: adecuación funcional (AF), eficiencia (E), fiabilidad (F), portabilidad (P) y usabilidad (U). Por tanto, el valor de *n* es 5. A la matriz de interdependencia de nivel 1, se le haya el valor absoluto de cada elemento, se eleva la matriz desde *1* hasta *n-1* (4 en este caso) y se suma cada matriz resultante:

$$\begin{array}{c} \\ AF \\ E \\ F \\ P \\ U \end{array}\begin{bmatrix} (0,0,0) & (0,0,0) & (0,0,0) & (0,0,0) & (0,0,0) \\ (0,0,0) & (0,0,0) & (0,0,0) & (0,0,0) & (0,0.3,0.6) \\ (0.5,1,1) & (0,0,0) & (0,0,0) & (0,0,0) & (0,0,0) \\ (0,0,0) & (0.5,1,1) & (0,0,0) & (0,0,0) & (0,0,0) \\ (0,0.3,0.6) & (0,0,0) & (0,0.3,0.6) & (0,0,0) & (0,0,0) \end{bmatrix}$$

En la Tabla II.6, se muestra el valor de la Influencia, Dependencia y Centralidad de cada característica.

Tabla II.6. Valores de Influencia, Dependencia y Centralidad de cada característica.

	Influencia	***Dependencia***	***Centralidad***	***Defuzifico***
AF	(0,0,0)	(1,3.08,4.52)	(1,3.08,4.52)	2.97333333
E	(0,0.78,1.92)	(0.5,1,1)	(0.5,1.78,2.92)	1.75666667
F	(0.5,1,1)	(0,0.48, 1.32)	(0.5,1.48,2.32)	1.45666667
P	(0.5,1.78,2.92)	(0,0,0)	(0.5,1.78,2.92)	1.75666667
U	(0.5,1.6,2.2)	(0,0.6,1.2)	(0.5,2.2,3.4)	2.11666667

Por la centralidad, quedan ordenadas las características de la siguiente forma: AF, U, E, P, F. Por el peso, el orden queda: AF, F, U, E, P. Se halla la mediana de la posición de cada característica, se ordena por la mediana y se obtiene una propuesta de orden para realizar las pruebas. En este caso quedaría AF, U, F, E, P (Ver Tabla II.7).

Tabla II.7. Rankings de cada característica por su centralidad, peso y la mediana de los rankings anteriores.

	Ranking por Centralidad	***Ranking por Peso***	***Mediana de los rankings***
AF	1	1	1
E	3	3	3
F	4	1	2.5
P	3	4	3.5
U	2	2	2

A continuación, se desarrolla el subcomponente **C2.2. Realización de la evaluación**. Siguiendo el plan de pruebas, se ejecutan las mismas y se obtienen los elementos de medidas de calidad. A partir de estos elementos, se calculan las medidas de calidad.

Para el Ejemplo II.1, los valores de las medidas de calidad, se muestran en la Tabla II.8.

Tabla II.8. Medidas de calidad para cada uno de los productos de software (S1, S2, S3)

Medidas de calidad	**S1**	**S2**	**S3**
Cobertura de aplicación funcional (CAF)	1	0.4	0.2
Corrección Computacional (CC)	0.2	0.3	0.1
Tareas por tiempo (TpT)	20	10	2
Uso de Memoria (MEM)	0.1	0.6	0.1
Tiempo medio entre fallos (MTBF)	90	20	15
Densidad de fracasos frente caso de pruebas (DF)	0.6	0.2	0.4
Cobertura de las pruebas (CP)	0.4	0.7	0.8
Facilidad de instalación (Fins)	B	M	MB
Adaptabilidad de Hardware (Ahw)	0.8	0.8	0.3
Adaptabilidad de Software (Asw)	0.9	0.8	0.4
Descripción Exhaustiva (DE)	0.9	0.8	0.5
Integridad de la documentación (IntDoc)	0.7	0.7	0.6
Satisfacción (Sat)	A	M	B
Apariencia (Apa)	MA	AA	A

Las medidas están definidas con diferentes escalas, por lo que se necesita normalizar los datos y obtener una matriz de evaluación. Lo anterior es el principal objetivo del subcomponente **C2.3**, además de la obtención del vector de penalización. Dicho vector

se utiliza para penalizar la calidad del software si no se cumple con las restricciones definidas en los criterios esenciales.

En el subcomponente **C2.3 Obtención de la matriz de evaluación**, se obtienen la matriz de evaluación y, paralelamente, el vector de penalización a nivel de l-1 (subcaracterística).

Para obtener la matriz, se normaliza la información resultante de las pruebas y se unifica en un dominio de datos; en este caso, los números difusos triangulares. Dicha operación es válida, sólo en problemas donde la información con que se cuenta es heterogénea; o sea, las medidas de calidad están definidas en dominios de datos distintos y en diferentes escalas. Se precisa transformar la información a un único dominio. (Ver Figura II.11)

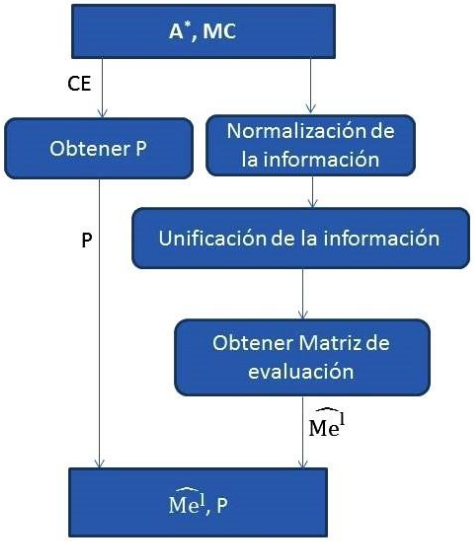

Figura II.11 Subcomponente **C2.3 Obtención de la matriz de evaluación.**

Para penalizar la solución del nivel superior, si no cumple las restricciones definidas para los criterios esenciales, se determina el vector de penalización del nivel l-1 (subcaracterística). El objetivo de este vector es controlar que las medidas de calidad consideradas esenciales cumplan las restricciones definidas; en caso contrario, a las sub-características asociadas, se les asigna el valor 0.

Se calcula el vector de penalización, según el método propuesto por Cables [147].

Se define el vector de penalización P^{l-1} para las sub-características, según la Ecuación II.20.

$$P^{l-1} = \{\min_{1 \le k \le r} {z_{ik}}^{l}\}, \qquad \text{(II.20)}$$

Donde:

- *r* es la cantidad de hermanos (hijos de un mismo padre).
- ${z_{ik}}^{l} es\ un\ elemento\ de\ la\ matriz\ Z.$
- Z es una matriz de valores de 0 y 1, que se calcula con la Ecuación II.21.

$$Z = \begin{bmatrix} z_{11} & \cdots & z_{1ln} \\ \cdots & \cdots & \cdots \\ z_{m1} & \cdots & z_{mln} \end{bmatrix}$$

$$\boldsymbol{z_{ij}} = \begin{cases} \mathbf{1} & \boldsymbol{si\ C_j\ \notin CE} \\ \boldsymbol{g(x, uf,\ us)} & \boldsymbol{si\ C_j\ \in CE} \end{cases} \qquad \text{(II.21)}$$

$$\boldsymbol{g(x, uf, us)} = \begin{cases} 1 & si\ uf \le x \le us \\ 0 & en\ otro\ caso \end{cases}$$

Regresando al Ejemplo II.1, el valor de Z se muestra en la Tabla II.9

Tabla II.9 Valores de Z y del vector de penalización P.

Medida	**Z**			**Subcaracterísticas**	**P**		
Cobertura de aplicación funcional (CAF)	1	1	0	Completitud funcional (CpF)	1	1	0
Corrección Computacional (CC)	0	1	0	Corrección funcional (CcF)	0	1	0
Tareas por tiempo (TpT)	1	1	0	Comportamiento temporal (Ct)	1	1	0
Uso de Memoria (MEM)	1	1	1	Uso de recursos (UR)	1	1	1
Tiempo medio entre fallos (MTBF)	1	1	1	Madurez (M)	0	1	1
Densidad de fracasos frente caso de pruebas (DF)	0	1	1				
Cobertura de las pruebas (CP)	0	1	1				
Facilidad de instalación (Fins)	1	1	1	Capacidad para ser instalado (CapI)	1	1	1
Adaptabilidad de Hardware (Ahw)	1	1	1	Adaptabilidad (A)	1	1	1
Adaptabilidad de Software (Asw)	1	1	1				
Descripción Exhaustiva (DE)	1	1	1	Capacidad para reconocer su adecuación (Comp)	1	1	1
Integridad de la documentación (IntDoc)	1	1	1	Capacidad de aprendizaje (Apr)	1	1	1
Satisfacción (Sat)	1	1	0	Capacidad para operar (Op)	1	1	0
Apariencia (Apa)	1	1	1	Estética	1	1	1

Para normalizar la información, se analiza para cada medida en qué dominio de datos y en qué escala está definida.

Para el dominio de datos numéricos:

Si x_i es un dato numérico definido en el intervalo de $[V_{min}, V_{max}]$, se transforma al intervalo [0,1], según la Ecuación II.22:

$$x_{ni} = \frac{x_i - V_{min}}{V_{max} - V_{min}} \qquad \text{(II.22)}$$

Si x_i pertenece al conjunto de medida de *Costo*, se transforma en *Beneficio*. Un dato x_{ij}, $0 \leq x_{ij} \leq 1$, se transforma de un criterio de costo a uno de beneficio, al aplicar la ecuación $1 - x_{ij}$.

$$x_{ni} = 1 - x_{ni} \qquad \text{(II.23)}$$

Para el dominio de datos ordinales:

Se evalúan con un conjunto de etiquetas lingüísticas, dispuestas en orden de maximizar, y asocian a cada una de ellas un número triangular difuso. Si la medida es de costo, se realiza de mayor a menor. Se definen, en un universo del discurso de [0,1], a partir de las Ecuaciones II.24:

$$VL = \{e_0, e_1, e_2, \ \ldots e_k \ldots, \ e_g\}, \ \ d = \left(\frac{1}{g}\right) \qquad \text{(II.24)}$$

$$\psi(l_k, \ \mathrm{d}) = (a_k, b_k, c_k)$$

Donde:

$$a_k = \begin{cases} 0 & k = 0 \\ b_k - d & k \neq 0 \end{cases}$$

$$b_k = \frac{k}{g}$$

$$c_k = \begin{cases} b_k + d & k \neq g \\ 1 & k = g \end{cases}$$

Para el dominio de datos booleanos:

Se mantiene con los valores 0 y 1 si fuera una medida de beneficio. Caso de tratarse de una medida de costo, el 0 se transforma en 1 y el 1 en 0. Es simplemente aplicar un not (negado).

Los estándares (ISO 25000, ISO 9126) recomiendan trabajar en el intervalo $[0,1]$ maximizando; o sea, que todas las medidas sean beneficios.

A continuación, se unifican las medidas en números difusos triangulares.

Para los valores numéricos, se transforma a la variable lingüística Evaluación (Ver Figura II.12), asociándola con la etiqueta de mayor grado de pertenencia.

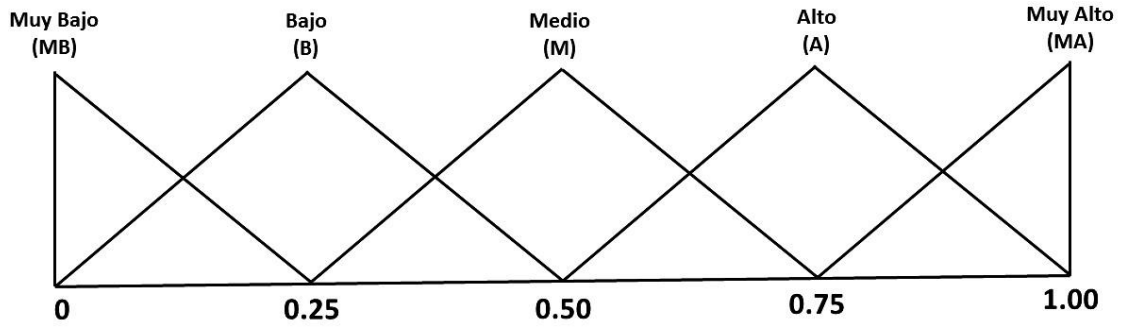

Figura II.12 Variable lingüística Evaluación.

Al valor booleano, se le asocia a 0 el número (0,0,0) y a 1 el número (1,1,1).
El resultado, se integra en una matriz de evaluación (Ver Ecuación II.25).

$$\widehat{\mathrm{Me}^{\mathrm{l}}} = \begin{bmatrix} \widehat{\mathrm{x}_{1,1}} & \cdots & \widehat{\mathrm{x}_{1,\mathrm{J}}} & \cdots & \widehat{\mathrm{x}_{1,\mathrm{ln}}} \\ \vdots & & \vdots & & \vdots \\ \widehat{\mathrm{x}_{\mathrm{i},1}} & \cdots & \widehat{\mathrm{x}_{\mathrm{i},\mathrm{J}}} & \cdots & \widehat{\mathrm{x}_{\mathrm{i},\mathrm{ln}}} \\ \vdots & & \vdots & & \vdots \\ \widehat{\mathrm{x}_{\mathrm{m},1}} & \cdots & \widehat{\mathrm{x}_{\mathrm{m},\mathrm{J}}} & \cdots & \widehat{\mathrm{x}_{\mathrm{m},\mathrm{ln}}} \end{bmatrix} \quad 1 \leq i \leq \mathrm{m},\ 1 \leq j \leq ln \qquad \text{(II.25)}$$

Donde:

$\widehat{x_{i,j}}$ representa el valor normalizado de la alternativa a_i en la medida c_j

ln representa la cantidad de medidas resultantes.

m representa el número de alternativas a evaluar.

Como salida de este componente, se obtiene la matriz de evaluación normalizada ($\widehat{\mathrm{Me}^{\mathrm{l}}}$) y el vector de penalización (P^{l-1}). En el componente que sigue, se agrega la información.

> En el Ejemplo II.1, que estamos desarrollando, todas las medidas son numéricas, pero no están definidas en el intervalo de [0,1]. Se normalizan según las Ecuaciones II.22 y II.23 y se obtiene la matriz de evaluación para el nivel 3. (Me^3) (Ver Tablas II.10 y II.11).

Tabla II.10. Normalización de los datos para obtener la matriz de evaluación.

Medidas de calidad	**S1**	**S2**	**S3**
Cobertura de aplicación funcional (CAF)	1	0.4	0.2
Corrección Computacional (CC)	0.2	0.3	0.1
Tareas por tiempo (TpT)	0.5	0.25	0.05
Uso de Memoria (MEM)	0.99	0.94	0.99
Tiempo medio entre fallos (MTBF)	0.9	0.2	0.15
Densidad de fracasos frente caso de pruebas (DF)	0.4	0.8	0.6
Cobertura de las pruebas (CP)	0.4	0.7	0.8
Facilidad de instalación (Fins)	B	M	MB
Adaptabilidad de Hardware (Ahw)	0.8	0.8	0.3
Adaptabilidad de Software (Asw)	0.9	0.8	0.4
Descripción Exhaustiva (DE)	0.9	0.8	0.5
Integridad de la documentación (IntDoc)	0.7	0.7	0.6
Satisfacción (Sat)	A	M	B
Apariencia (Apa)	MA	AA	A

Tabla II.11. Unificación de los datos para obtener la matriz de evaluación.

	CAF	CC	TpT	MEM	MTBF	DF	CP	Fins	Ahw	Asw	DE	IntDoc	Sat	Apa
S1	MA	B	M	MA	MA	M	M	B	A	MA	MA	A	A	MA
S2	M	B	B	MA	B	A	A	M	A	A	A	A	M	AA
S3	B	MB	MB	MA	B	M	A	MB	B	M	M	M	B	A

C3. "Agregación y recomendación"

El tercer componente se denomina "**Agregación y recomendación**", y se identifica como **C3**. En **C3,** se realiza la agregación de la información para obtener el índice de calidad (IQ) de cada alternativa. Dicho índice, permite ordenar las alternativas de mejor a más baja calidad. También, puede asociarse una valoración cualitativa de cada alternativa, a través de la variable lingüística nivel de calidad (NQ). Este componente tiene dos subcomponentes:

- **C3.1 Agregación:** Al valor de cada criterio, se agrega la influencia del resto, mediante la operatoria de los MCD; a continuación, se pondera el valor obtenido con los pesos de cada uno. Seguidamente, se suman las medidas que conforman las sub-características, se obtiene el valor de cada sub-característica para cada alternativa, y luego se penalizan las sub características con el vector de penalización. El proceso, se repite para obtener los valores de las diversas características. Por último, se itera de nuevo para conseguir los diferentes IQ de las distintas alternativas.

- **C3.2 Recomendación:** Según el tipo de evaluación deseada, se selecciona la o las alternativas de mejores soluciones. Para esto, deben ordenarse los distintos IQ.

La función del subcomponente **C3.1. Agregación**, es agregar la información para obtener el IQ de cada alternativa (Ver Figura II.13)

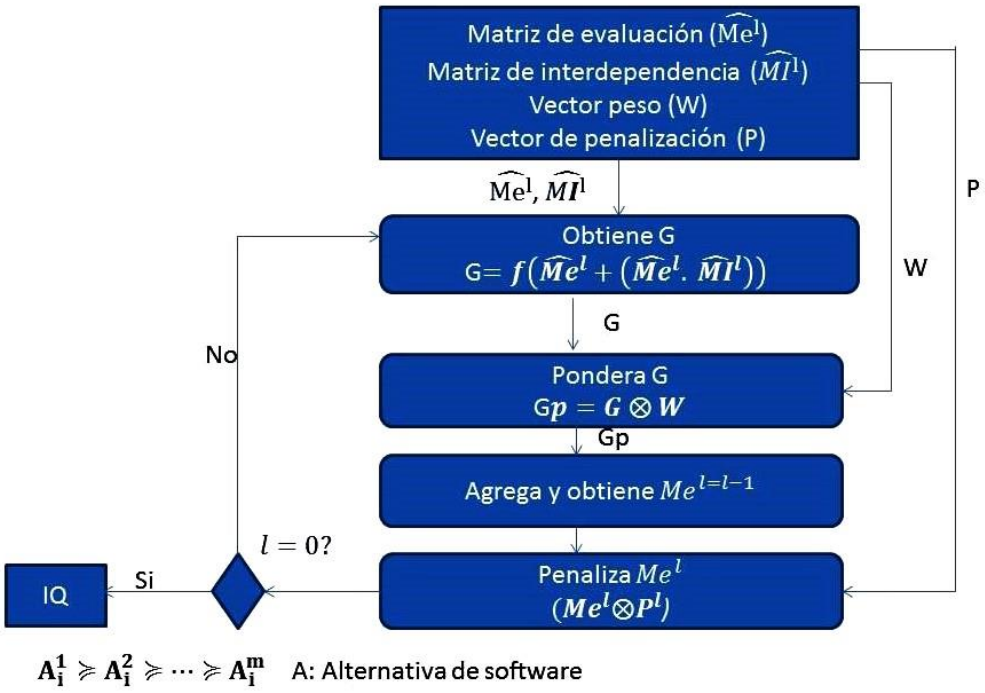

Figura II.13 Subcomponente **C3.1. Agregación.**

Lo primero que se realiza, es obtener el valor de los diferentes criterios, teniendo en cuenta la interdependencia. Para ello, se agrega la influencia entre los criterios, expresada en la matriz de interdependencia, con el valor de las medidas de calidad.

Se obtiene la matriz G, contentiva del valor de los criterios con la influencia del resto de los mismos. Es importante que los resultados se encuentren en el intervalo [0,1], lo cual se logra evaluando cada elemento de G con una función umbral (Ver Ecuación II.26).

Dicha función, puede ser la siguiente: $f(x) = \frac{1}{1+e^{-\lambda x}}, \text{donde } \lambda \succ 0$ determina la pendiente de la función continua, que en este caso es 1.

$$G = f\left(\widehat{Me}^l + \left(\widehat{Me}^l \times \widehat{MI}^l\right)\right) \qquad \text{(II.26)}$$

Seguidamente, se pondera el valor de cada criterio con el vector peso definido en el componente 1, y se logra la matriz ponderada (*Gp*) (Ver Ecuación II.27).

$$Gp = G \otimes W\,, \quad gp_{ij} = g_{ij}\,.\,w_j \qquad \text{(II.27)}$$

El próximo paso, es la agregación de las medidas para obtener los valores de las sub-características. Esto se realiza a través de la suma ponderada.

Por último, se penalizan las subcaracterísticas, utilizando la Ecuación II.28:

$$Me^{l-1} \times P^{l-1} = (Me_{i1}.P_{i1},\ Me_{i2}.P_{i2}, \ldots\ldots, Me_{in}.P_{in}). \qquad \text{(II.28)}$$

Los valores de las subcaracterísticas ya penalizadas para el Ejemplo II.1, se muestran en la Tabla II.12.

Tabla II.12. Valores de las subcaracterísticas.

Completitud funcional (CpF)	0.6792	0,7311	0,7311
	0.5622	0,6225	0,6792
	0.0000	0,0000	0,0000
Corrección funcional (CcF))	0.0000	0.0000	0.0000
	0.5000	0.5622	0.6225
	0.0000	0.0000	0.0000
Comportamiento temporal (Ct)	0.5622	0.6225	0.6792
	0.5000	0.5622	0.6225
	0.0000	0.0000	0.0000
Uso de recursos (UR)	0.6792	0.7311	0.7311
	0.6792	0.7311	0.7311
	0.6792	0.7311	0.7311
Madurez (M)	0.0000	0.0000	0.0000
	0.5735	0.6324	0.6876
	0.5494	0.6097	0.6669
Capacidad para ser instalado (CapI)	0.5000	0.5622	0.6225
	0.5622	0.6225	0.6792
	0.5000	0.5000	0.5622
Adaptabilidad (A)	0.6508	0.7051	0.7311
	0.6225	0.6792	0.7311
	0.5311	0.5923	0.6508
Capacidad para reconocer su adecuación (Comp)	0.6792	0.7311	0.7311
	0.6225	0.6792	0.7311
	0.5622	0.6225	0.6792
Capacidad de aprendizaje (Apr)	0.6225	0.6792	0.7311
	0.6225	0.6792	0.7311
	0.5622	0.6225	0.6792
Capacidad para operar (Op)	0.6225	0.6792	0.7311
	0.5622	0.6225	0.6792
	0.0000	0.0000	0.0000
Estética	0.6225	0.6792	0.7311
	0.6792	0.7311	0.7311
	0.5622	0.6792	0.7311

Las tareas de este subcomponente de agregación, son ejecutadas iterativamente hasta llegar al nivel 0. Se ejecutan los diferentes pasos para obtener los valores de las diferentes características y lo mismo ocurre para los índices de calidad de cada alternativa.

El valor de cada característica de calidad para el Ejemplo II.1, se muestra en la Tabla II.13. Por ejemplo, el valor de la adecuación funcional para el software 1, se expresa en el numero triangular difuso (0.623, 0.631, 0.631).

Tabla II.13. Valores de las características.

	AF			**E**			**F**		
S1	0.623	0.631	0.631	0.620	0.645	0.667	0.500	0.500	0.500
S2	0.675	0.734	0.752	0.613	0.637	0.659	0.640	0.653	0.665
S3	0.500	0.500	0.500	0.554	0.558	0.558	0.634	0.648	0.661

	P			**U**		
S1	0.649	0.661	0.669	0.695	0.754	0.771
S2	0.647	0.660	0.672	0.692	0.753	0.767
S3	0.628	0.639	0.652	0.623	0.685	0.699

Se realiza otra iteración de los pasos anteriores para obtener el IQ en este ejemplo, como se muestra en la Tabla II.14.

Tabla II.14. Valor de IQ.

	IQ	**IQ defuzificado**	**Ranking**
S1	(0.516,0.586,0.621)	0.580513299	3
S2	(0.522,0.594,0.630)	0.588252946	1
S3	(0.521,0.591,0.623)	0.584475493	2

Cuando se concluye la agregación, se obtiene el valor del índice de calidad para cada alternativa y se asocia al IQ un valor cualitativo, a través de la variable lingüística Nivel de calidad.

El subcomponente **C3.2. Recomendación**, tiene como su función principal formular una recomendación, según el tipo de evaluación deseada. Si el objetivo de la evaluación es recomendar un producto o conjunto de productos, entonces lo primero que se hace es ordenar los IQ, de mayor a menor, y se recomienda los definidos como de calidad alta. Si el objetivo es un ordenamiento de productos, entonces se les recomienda un ranking según su calidad.

El objetivo puede ser, en ocasiones, tan simple como obtener la valoración de la calidad de un producto determinado. Entonces, se muestra el mismo con su índice global de calidad y una valoración cualitativa de este.

En cada uno de los casos, debe adicionarse una valoración de las diferentes características de calidad.

La recomendación para el Ejemplo II.1, quedaría en el orden S2, S3 y S1.

Modificaciones del modelo, incorporando la interdependencia como peso:
En el epígrafe anterior, se refleja la influencia entre criterios, representada en el modelo de calidad con los E_h. Se trata estos enlaces horizontales con los MCD, los cuales se representan a través de la matriz de interdependencia (MI). Esta influencia actúa de manera directa sobre los valores de la matriz de evaluación, expresada en el modelo, durante el proceso de agregación de la información, cuando se obtiene la matriz G.

Existe otra manera de tratar la interdependencia y es a través de la modificación del peso de los criterios. En el presente epígrafe, se detallan los cambios a realizar al modelo anterior para tratarla de esta forma.

En la literatura, se destacan dos trabajos que de manera general abordan las interrelaciones entre criterios a través de los pesos ([141] y [148]). Estos, combinan AHP, MCD y ANP para determinar los pesos.

Los pasos propuestos por Yu y Tzeng [148], son:

Paso 1. Calcular el vector peso local (z), mediante la comparación por pares, usando el enfoque de autovector semejante a AHP.

Paso 2. Construir el MCD y su matriz de adyacencia (*E*), para expresar la influencia entre los criterios.

Paso 3. Determinar la matriz de estado estacionario con la Ecuación II.29:

$$C^{t+1} = f(C^t E), \qquad \text{(II.29)}$$

Donde:

- C^t es la matriz de estado a cierta iteración.
- $C^0 = I$, Es la matriz inicial, que coincide con la matriz idéntica.
- $f(x)$, es la función umbral recomendada, en este caso la tangente hiperbolica o la función lineal.
- C^{t+1} es la matriz de estado a cierta iteración t+1

Se itera varias veces hasta que los valores converjan a un vector (C^*).

Paso 4. Obtener el vector peso global. Se normaliza C^* y aplica la Ecuación II.30.

$$w = z + C_n^* z \qquad \text{(II.30)}$$

Kiszová y Mazurek [141], proponen el método híbrido MCD-autovector (HEFCM, por sus siglas en inglés). El mismo consta de los siguientes pasos:

Paso 1. Calcular el peso por AHP *(w)*.

Paso 2. Construir el MCD, y determinar la matriz de adyacencia (E).

Paso 3. Normalizar la matriz de adyacencia (E*) de manera que la suma de sus columnas sea 1, lo que hace a E* matriz estocástica por columnas.

Paso 4. Buscar la matriz límite $M = \lim_{n\to\infty} (E^*)^n$ y obtener el vector *m*.

Paso 5 Obtener el vector peso global, combinando el valor de *m* y *w* por una función $v = f(w, m)$, donde la función puede ser el promedio.

Ambos trabajos, solo tienen en cuenta que la relación entre criterios sea positiva. Sobre la base de los mismos, se propone un nuevo método para tener en cuenta la interdependencia entre criterios, a través de la modificación de los pesos para la evaluación de la calidad de software.

El método propuesto, consta de los siguientes pasos:

Paso 1. Calcular el conjunto de los vectores pesos ($R = \{R^1, R^2, R^3\}$), determinando la preferencia de importancia de los criterios. Esto se realiza por comparación por pares o de forma directa.

Paso 2. Para determinar la influencia de un criterio sobre los otros, construir los MCD para los diferentes niveles. A partir de estos, obtiene la matriz de interdependencia (*MI*).

Paso 3. Determinar la matriz de efecto total, a partir de la Ecuación II.31.

$$T = \sum_{i=1}^{n-1} MI^i \qquad \text{(II.31)}$$

Paso 4. Determinar el vector de influencia $S = \{s_i\}$, mediante la Ecuación II.32.

$$s_i = f(\sum_{k=1}^{n} t_{ik}) \qquad \text{(II.32)}$$

Donde:

- *n*: cantidad de criterios.
- $f(x)$: función de agregación

- t_{ij} : Influencia total del criterio c_i sobre c_j

Paso 5. Normalizar los valores, de manera que $\sum_{i=1}^{n} s_i = 1$

Paso 6. Obtener el vector peso global, aplicando $v = f(w, s_n)$.

El método, no implica grandes modificaciones al modelo. Estos pasos se adicionan al subcomponente **C1.2 Valoración de los criterios** (ver Figura II.14). Se simplifica el subcomponente **C3.1 Agregación**, pues no es necesario obtener la matriz G, solo se trabaja directamente con la matriz de evaluación normalizada, ya que el peso tiene incluido la interdependencia.

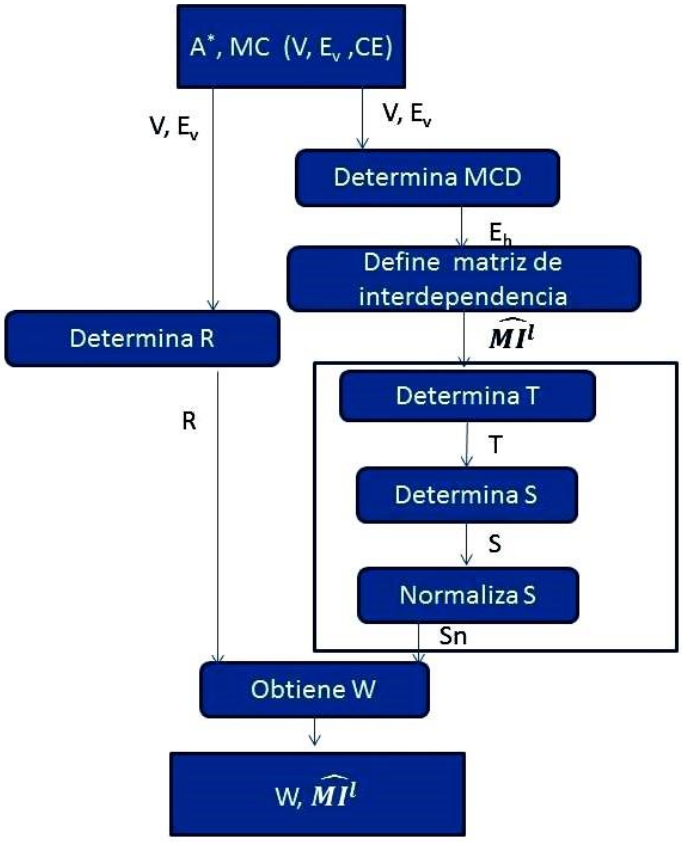

Figura II.14 Modificaciones en el Subcomponente C1.2 Valoración de los criterios.

En el caso del Ejemplo II.1, se obtiene la matriz de interdependencia y el vector S normalizado, resultante del cálculo. Los vectores resultantes, se muestran en la Tabla II.15.

Tabla II.15. Vector de peso modificado.

Para $l = 1$

Características	**R**	**S**	**W**
Adecuación funcional (AF)	0.324	0.1785	0.251
Eficiencia (E)	0.075	0.1217	0.098
Fiabilidad (F)	0.324	0.2545	0.289
Portabilidad (P)	0.072	0.1567	0.114
Usabilidad (U)	0.206	0.2886	0.247

Para l = 2

Características	Subcaracterísticas	R	S	W
Adecuación funcional (AF)	Completitud funcional (CpF)	0.75	0.412	0.581
	Corrección funcional (CcF)	0.25	0.588	0.419
Eficiencia (E)	Comportamiento temporal (Ct)	0.667	0.460	0.564
	Uso de recursos (UR)	0.333	0.540	0.436
Fiabilidad (F)	Madurez (M)	1	1	1
Portabilidad (P)	Capacidad para ser instalado (CapI)	0.25	0.500	0.375
	Adaptabilidad (A)	0.75	0.500	0.625
Usabilidad (U)	Capacidad para reconocer su adecuación (Comp)	0.152	0.279	0.216
	Capacidad de aprendizaje (Apr)	0.211	0.196	0.203
	Capacidad para operar (Op)	0.400	0.225	0.312
	Estética	0.237	0.300	0.269

Para l = 3

Características	Subcaracterísticas	Métrica	R, S, W
Adecuación funcional (AF)	Completitud funcional (CpF)	Cobertura de aplicación funcional (CAF)	1
	Corrección funcional (CcF)	Corrección Computacional (CC)	1
Eficiencia (E)	Comportamiento temporal (Ct)	Tareas por tiempo (TpT)	1
	Uso de recursos (UR)	Uso de Memoria (MEM)	1
Fiabilidad (F)	Madurez (M)	Tiempo medio entre fallos (MTBF)	0.4
		Densidad de fracasos frente caso de pruebas (DF)	0.4
		Cobertura de las pruebas (CP)	0.2
Portabilidad (P)	Capacidad para ser instalado (CapI)	Facilidad de instalación (Fins)	1
	Adaptabilidad (A)	Adaptabilidad de Hardware (Ahw)	0.5
		Adaptabilidad de Software (Asw)	0.5
Usabilidad (U)	Capacidad para reconocer su adecuación (Comp)	Descripción Exhaustiva (DE)	1
	Capacidad de aprendizaje (Apr)	Integridad de la documentación (IntDoc)	1
	Capacidad para operar (Op)	Satisfacción (Sat)	1
	Estética	Apariencia (Apa)	1

Al tener los vectores de peso (w) y la matriz de evaluación normalizada, el procedimiento sería ponderar, agregar y penalizar. El resultado del índice de calidad, se muestra en la Tabla II.16.

Tabla II.16 Valor de IQ y ranking de productos

	IQ	Ranking
S1	0,298614586	3
S2	0,56816364	1
S3	0,476906445	2

Si se comparan los resultados obtenidos por esta vía y la anterior, los mismos son semejantes, aunque el valor de IQ se expresa en escalas diferentes.

CAPÍTULO III. HERRAMIENTAS OPERATIVAS PARA RESOLVER LOS PROBLEMAS DE EVALUACIÓN Y SELECCIÓN DE PRODUCTOS DE SOFTWARE SEGÚN SU CALIDAD

Introducción

Es importante para los especialistas que desarrollan procesos de evaluación de la calidad y selección de productos de software, tener una aplicación que soporte el modelo a utilizar. Las soluciones planteadas y obtenidas hasta ahora, de poco sirven si no van acompañadas de un marco operativo que permita aplicarlas en situaciones reales concretas. Para el proceso de toma de decisiones, es esencial disponer de varios modelos de calidad que permitan evaluar y seleccionar el producto. También se necesitará la implementación de varios métodos de agregación de la información, para poder comparar y evaluar las soluciones obtenidas por estos métodos.

Con esta base, el presente capítulo se enfoca en describir la plataforma SoftQuality, una herramienta informática de software libre que permite resolver el problema de evaluación y selección de productos de software según su calidad, a partir del modelo creado. La sección III.1, muestra el funcionamiento de SoftQuality, y describe la arquitectura, las herramientas y lenguajes utilizados en su desarrollo. Además, explica las interfaces del sistema. El código de esta herramienta se obtiene a través del Anexo IV y está disponible en http://modo.ugr.es/SoftQQuality/, También a título de ejemplo se muestra el funcionamiento de ValoraApp, un prototipo de app que permite evaluar la usabilidad de aplicaciones para móviles, desarrollada sobre el modelo de propuesto.

La sección III.2, como demostración de utilidad, eficiencia y operatividad, estudia un caso práctico real del Centro Nacional de Calidad de Software (CALISOFT), entidad evaluadora de la calidad del proceso y producto de la industria cubana del software. Por último, se valida el enfoque de solución, según la satisfacción del usuario y el apoyo a la toma de decisiones.

III.1 Plataforma para evaluar y seleccionar productos de software según su calidad

El modelo propuesto está soportado por SoftQuality, la plataforma de ayuda a la toma de decisiones en el proceso de evaluación y selección de productos de software, según su calidad. Además, se implementan las variantes difusas de otros métodos de agregación, que facilita al usuario la comparación de estos, permitiendo, por tanto, elegir la solución más apropiada en base a la información disponible.

El diseño flexible de la herramienta, hace posible la incorporación de soluciones con diferentes parámetros; o sea, varios modelos de calidad y diversas métricas. De igual manera, proporciona un conjunto de componentes reutilizables. La misma fue

desarrollada en Phyton, con la incorporación de robustos paquetes de cálculo numérico, tales como Numpy [149],[150] y Scipy [151]. Sobre estos, se han desarrollado soluciones generales, como Scikit-Learn [152], para el aprendizaje automático; Scikit-Criteria [153], una librería que desarrolla varios métodos de decisión multicriterios; y scikit-fuzzy, con herramientas útiles que incluyen la lógica difusa.

Herramientas, metodologías y lenguajes:

Metodología de desarrollo.

Para el desarrollo de SoftQuality, se utilizó la metodología ágil Programación extrema (XP), dadas sus bondades, enfocadas más en la colaboración con el cliente, en el proceso de desarrollo incremental del producto basado en iteraciones de corto plazo, muy apropiada para pequeños equipos de desarrollo.

Las metodologías ágiles, han mostrado una alta efectividad en proyectos con requerimientos muy cambiantes y en aquellos casos en los que se exige reducir notablemente los tiempos de desarrollo, sin descuidar la alta calidad del producto. Además, XP genera poca cantidad de artefactos y enfoca principalmente al código.

Herramienta de modelado.

Se utiliza el Visual Paradigm 8.0, para representar el modelado de los procesos del negocio. Es una herramienta CASE, fácil de instalar y actualizar; es multiplataforma y presenta capacidades de ingeniería directa e inversa. Brinda facilidades para el desarrollo de programas informáticos, desde la planificación, pasando por el análisis y el diseño, hasta la generación del código fuente de los programas y la documentación. Además, potencia la reutilización del software y soporta la notación BPMN (Modelado de Procesos de Negocio, por sus siglas en inglés).

Lenguajes de programación.

Python 3.4, es un lenguaje interpretado, multiplataforma, que usa tipado dinámico; se emplea como lenguaje de programación del lado del servidor. Su sintaxis, favorece el código legible. Como lenguaje multiparadigma, soporta orientación a objetos, programación imperativa y, en menor medida, programación funcional. Posee una licencia de código abierto, denominada Python Software Foundation License, llegando a ser uno de los más difundidos en el ámbito científico, y siendo la base para los desarrollos de robustos paquetes de cálculo numérico, como *numpy, scikit-fuzzy, scikit-criteria, scikit-learn, mathplolib* [153].

HTML 5 y JavaScript, son utilizados como lenguajes de programación en el lado del cliente; HTML 5 (*HyperText Markup Language*), para estructurar y presentar el

contenido para la web. Debido a su alto grado de portabilidad, se visualizan las páginas con cualquier sistema operativo. JavaScript, por su parte, es un lenguaje de programación interpretado, que no necesita ser compilado. Consigue mejoras en la interfaz de usuario y páginas web dinámicas, además, está totalmente integrado con HTML y CSS.

Entorno integrado de desarrollo.

Para el desarrollo del sistema, se seleccionó como IDE Visual Studio Code 1.17, de código abierto, desarrollado por Microsoft. La licencia es MIT License y su desarrollo se encuentra activo. Está escrito en TypeScript, JavaScript, CSS y posee resaltado para Phyton.

Visual Studio Code, puede extenderse mediante complementos disponibles en un repositorio central. Se integra con Git, para el control de versiones.

Sistema Gestor de Base de Datos.

PostgreSQL, es un sistema que brinda una alta seguridad de la información manejada y posibilita la optimización de consultas; posee una arquitectura cliente-servidor. Este gestor, implementa transacciones, integridad referencial, vistas y multitud de funcionalidades; es, además, una aplicación de código abierto.

Framework: Django 1.9

Django es un *framework* web gratuito, de código abierto, escrito en Python, que sigue el patrón arquitectónico *Model-Template-View* (MVT). Su objetivo principal es facilitar la creación de sitios web complejos basados en bases de datos. Enfatiza la reutilización y la "capacidad de conexión" de los componentes. Además, posibilita el desarrollo rápido y el principio de no repetir. Asimismo, proporciona una interfaz administrativa opcional de creación, lectura, actualización y eliminación, generada dinámicamente mediante la introspección. Se configura a través de modelos de administración.

Django 1.9, requiere Python 2.7, 3.4 o 3.5. Dicha versión, continúa extendiendo el soporte para funcionalidades específicas de PostgreSQL, integrando ahora funciones específicas de agregación. Se usan los paquetes d*jango-grappelli* y *django-reversion*, que son extensiones del *framework* para el trabajo con las interfaces gráficas y el control de la administración.

Control de versiones

Git es un software de control de versiones que permite la gestión distribuida, diseñado por Linus Torvalds, pensando en la eficiencia y confiabilidad del mantenimiento de

versiones de aplicaciones cuando las mismas tienen un gran número de archivos de código fuente. Entre sus ventajas se encuentra la gestión eficiente de proyectos grandes, dada la rapidez de gestión de diferencias entre archivos, entre otras mejoras de optimización de la velocidad de ejecución. Es destacable, asimismo, el fuerte apoyo al desarrollo no lineal, por ende, su rapidez en la gestión de ramas y mezclado de diferentes versiones. Los almacenes de información pueden publicarse por HTTP, FTP, rsync, o mediante un protocolo nativo, ya sea a través de una conexión TCP/IP simple o de cifrado SSH [154].

Arquitectura del software:
El patrón de arquitectura implementado, ha sido el N-Capas, por las ventajas que esta ofrece (Ver Figura III.1).

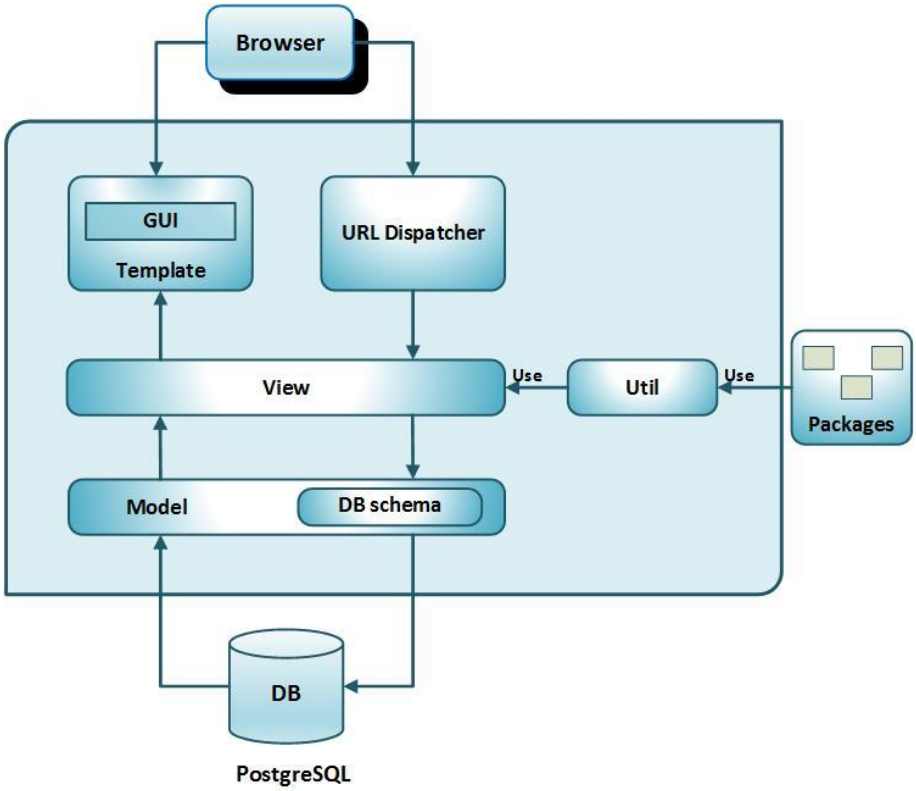

Figura III.1. Arquitectura de SoftQuality.

El concepto básico de esta arquitectura es diseñar en base a niveles de abstracción, separar la aplicación en componentes situados en diferentes niveles funcionales. Llevar a cabo el desarrollo en varios niveles, permite que, en caso de error o necesidad de cambio, sólo se precise modificar el nivel afectado, sin perturbar el correcto funcionamiento del resto del sistema. Además, facilita la interoperabilidad, el mantenimiento, la escalabilidad y reutilización del código.

La arquitectura lógica, está basada en el modelo MTV de Django, que difiere de MVC (*Model-View-Controller*).

El Modelo (*Model*), es la capa de acceso a la base de datos, donde se declaran las estructuras que contendrán los datos para transportarlos desde el *View* hacia el *Template.* Esta capa contiene toda la información necesaria para acceder a estos y validarlos, sobre su comportamiento y las relaciones entre ellos. Permite independizar la lógica de negocio de la lógica de acceso a datos, obteniendo mayor organización en la herramienta.

Template (Plantilla), se refiere a la capa de presentación. Toma las decisiones de cómo mostrar los datos sobre una página web u otro tipo de documento. Solo contiene el paquete de la interfaz gráfica del usuario (*GUI*) y las clases encargadas de garantizar la interacción del usuario con la herramienta (capturando movimiento, eventos y acciones). Se comunica únicamente con la capa *View*, a la cual envía los datos a través de los formularios.

View (Vista), es la capa de la lógica de negocios. Contiene la lógica que accede al modelo y la delega a la plantilla apropiada: Es como un puente entre los modelos y las plantillas. En esta encontramos todas aquellas funciones que hacen algún tipo de tratamiento de los datos, se aplica las reglas de negocio, etc. Incorpora la lógica para la manipulación y generación de las interfaces visuales. A través de *Util*, se relaciona con los paquetes de cálculo numérico, tales como *Numpy, Scipy, Scikit-Criteria* y *scikit-fuzzy*.

La capa de control en esta arquitectura es manejada por el *framework* Django, siguiendo la *URL Dispatcher* y llamando a la función apropiada de Python para la URL obtenida [155]. Como capa base, se representa a la Base de Datos.

Diseño de interfaz:

Las interfaces, se diseñaron de manera intuitiva, de manera tal que el usuario en todo momento conozca las acciones que puede y debe realizar. Se concibió un proceso secuencial donde el usuario transite desde la carga de los datos del problema, pasando por cada componente del modelo, hasta los resultados (Ver Figuras III.2 a III.4). Para la administración del sitio, se utiliza el *admin-Django*, que facilita la implementación y cumple los principios de diseño de interfaces.

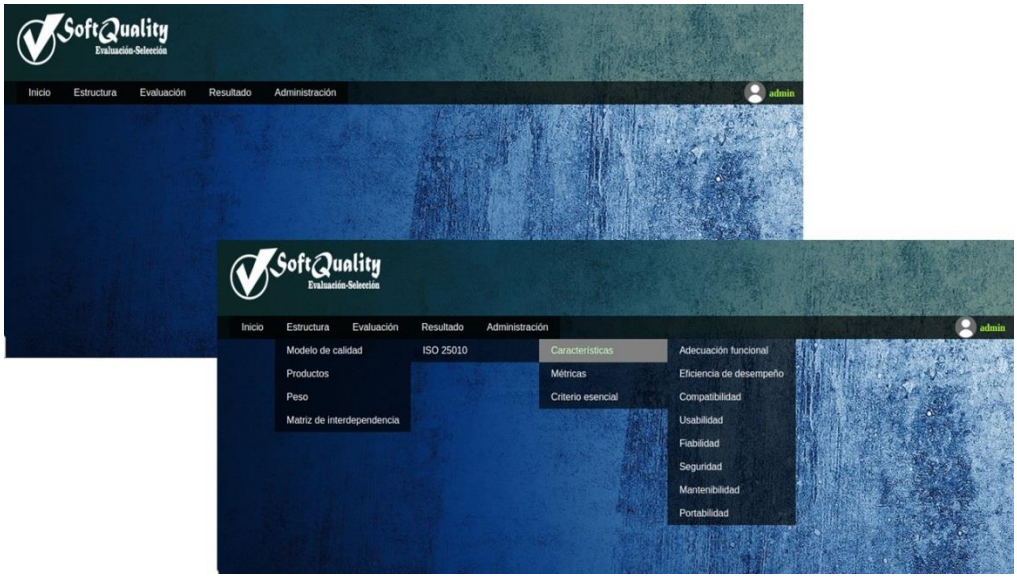

Figura III.2. Página principal y menú desplegado del sistema.

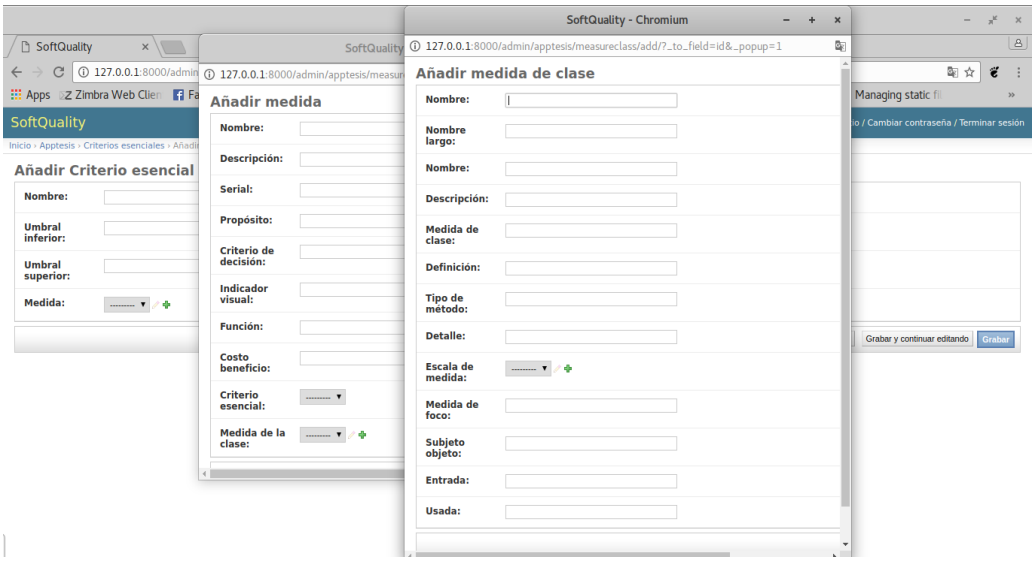

Figura III.3 Formulario de entrada de datos

Evaluación de la calidad

Código	Ranking
P1	7
P2	4
P3	8
P4	3
P5	5
P6	12
P7	9
P8	1
P9	11
P10	6
P11	10
P12	2

Figura III.4 Resultado de la evaluación.

Además, a título de ejemplo como se indicó en la introducción de este capítulo, a continuación, se muestra una app que permite evaluar la usabilidad de aplicaciones para móviles. El modelo propuesto es la base de la implementación de esta aplicación.

La app se desarrolla en Android y mediante servicios web, implementa una API que permite acceder, a través del protocolo de HTTP, a su base de datos. Implementa varios scripts en lenguaje PHP para acceder e interactuar con la base de datos y gestionar las consultas realizadas por los clientes.

Las interfaces, se han diseñado de manera agradable, entendible y fácil de operar, de manera que el usuario, en todo momento, conozca las acciones que puede y debe realizar. Se transita por la aplicación, desde el registro de los usuarios, hasta el ranking de las *apps*, de manera intuitiva y con una navegación cómoda para el usuario, concibiendo un proceso secuencial. Las Figuras III.5 a III.10, muestran las interfaces de usuario de cada paso a realizar.

La Figura III.5, presenta el registro de los usuarios y dos botones, uno para completar el registro y otro para entrar las credenciales, a través de la pantalla de acceso, también mostrada en la figura.

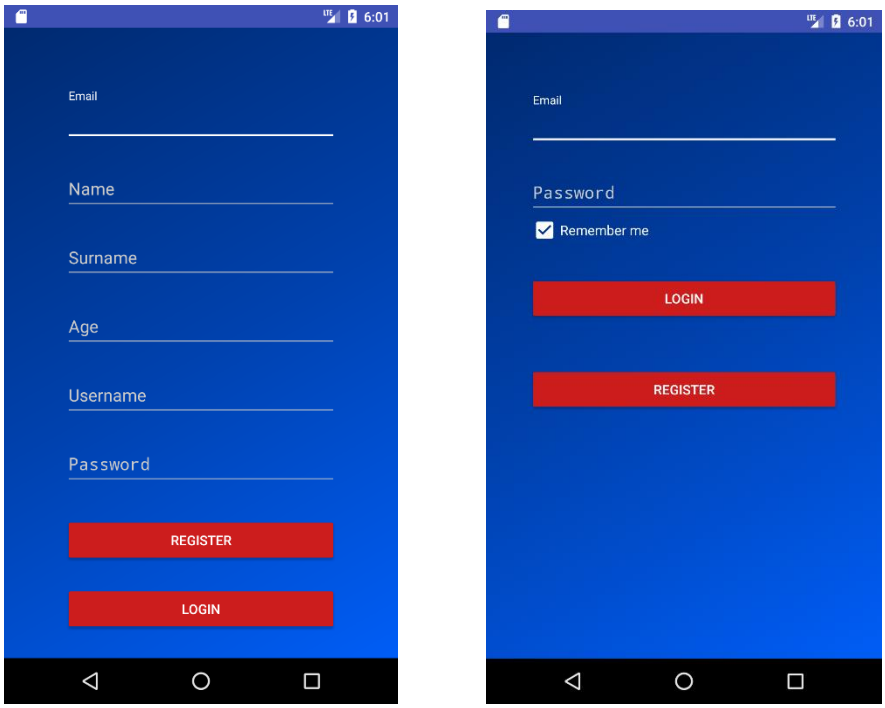

Figura III.5 Formularios mostrados al usuario para su registro y entrada de las credenciales para acceder a la app.

Una vez que se accede a la aplicación, el usuario se encuentra con la pantalla principal que contiene el ranking de las aplicaciones más valoradas por los usuarios de la comunidad junto a su valoración media (Ver Figura III.6).

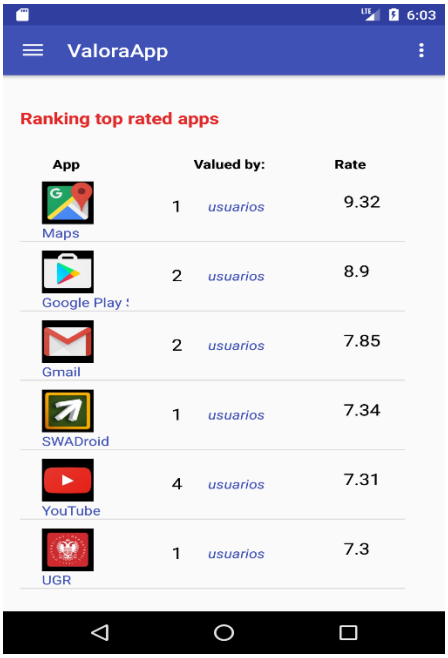

Figura III.6 Pantalla principal de la app.

La Figura III.7, muestra el menú lateral, compuesto por las siguientes opciones:

- *Home:* muestra al usuario la pantalla principal.
- *Profile*: presenta la pantalla del perfil.
- *Evaluate app*: navega a la pantalla de valoración de las aplicaciones instaladas.
- *My ratings*: muestra la lista de las valoraciones realizadas por el usuario
- *Community ratings*: expone la lista de las valoraciones realizadas por la comunidad.
- *Preferences*: presenta el formulario de las preferencias de los pesos, matriz de influencias de las sub-características y medidas.
- *Logout:* cierra la sesión y redirige el usuario a la pantalla de acceso.

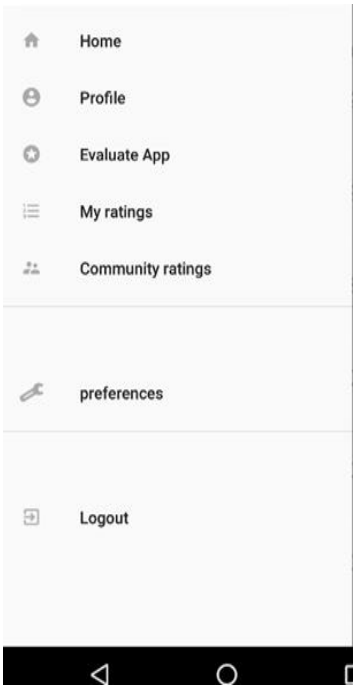

Figura III.7 Menú lateral de la app.

La Figura III.8, muestra la entrada de datos, en este caso las sub-características de la usabilidad junto con una medida de calidad y un deslizador para la entrada del valor. Seguidamente, se obtiene el índice de usabilidad de las apps seleccionadas para su valoración (Ver Figura III.9).

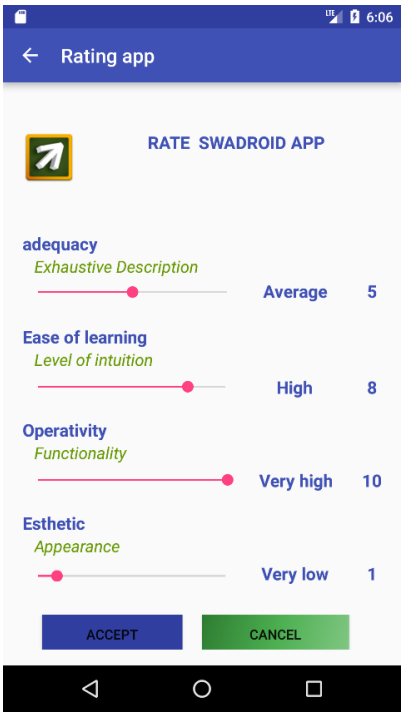

Figura III.8 Formulario para añadir los valores de las medidas de calidad de una app.

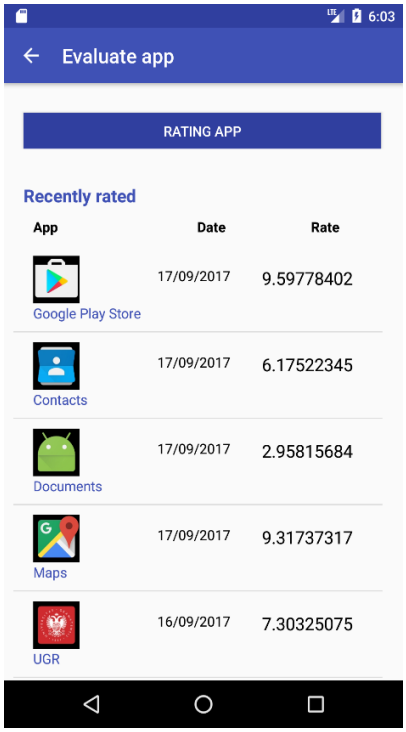

Figura III.9 Resultado de la evaluación.

Mediante la opción *Preferences*, se introduce el peso de las sub-características y medidas, así como la matriz de influencia de cada nivel (Ver Figura III.10)

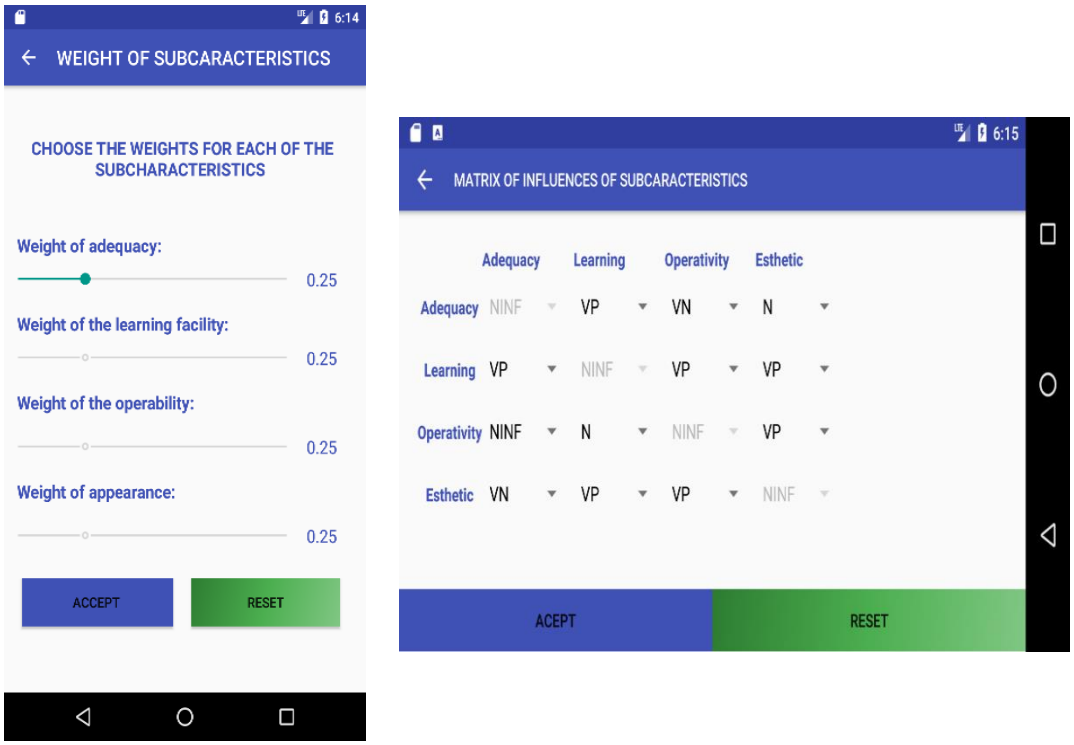

Figura III.10 Formulario para la entrada de las preferencias (peso y matriz de influencia).

III.2 Caso real de un problema de evaluación y selección de productos de software, según la calidad.

Una de las estrategias fundamentales del gobierno cubano, con implicaciones en el modelo económico nacional, es la de informatización de la sociedad. El país apuesta por el desarrollo de aplicaciones y software que permitan la independencia tecnológica. Inscrito entre los objetivos de desarrollo del programa, diversos organismos, rectorados por el Ministerio de Comunicaciones (MINCOM), promueven soluciones que se insertan en el quehacer diario de instituciones del país. Esto implica desarrollar productos de software de alta calidad y garantizar que también la tengan los adquiridos del exterior. En ese entorno, se precisa contar con un modelo y una directiva especializada en la medición de los patrones de calidad. Así surge el Centro Nacional de Calidad de Software (CALISOFT), adscrito al MINCOM.

En esta sección, se detalla el marco conceptual en el cual desarrolla el presente caso de estudio. Se presenta una descripción general de CALISOFT y de su forma de evaluar productos de software, sin descuidar la satisfacción de los clientes, lo cual constituyó la motivación principal para el desarrollo del ejemplo, con el objetivo de constatar la operatividad, precisión, confiabilidad del modelo y de la herramienta de software.

Centro Nacional de Calidad de Software (CALISOFT)

Constituye una entidad enfocada a contribuir al desarrollo de esta industria, proponiendo al Ministerio normativas técnicas, políticas, procedimientos y estándares para las instituciones desarrolladoras de aplicaciones informáticas [156].

CALISOFT, tiene como principales objetivos, además, de establecer los estándares técnicos y procedimientos que normalicen la industria informática cubana, dirigiéndola hacia niveles de calidad superiores, ofrecer servicios de evaluación de la calidad de los procesos de desarrollo de software, productos informáticos y procesos de prestación de servicios de tecnología de la información.

Es responsable de evaluar la conformidad con las normas y emitir evaluación a procesos y productos informáticos nacionales o importados; brindar servicios de consultorías asociados a la ingeniería y calidad de las aplicaciones; controlar y auditar el uso de normativas técnicas, procedimientos, documentos estandarizados y buenas prácticas para el desarrollo de aplicaciones y sistemas, y brindar servicios de formación en temas de calidad [157].

La Empresa Nacional de Software (Desoft), la Empresa de Informática y Automatización para la Construcción (Aicros), los Joven Club de Computación y

Electrónica y Transoft (grupo perteneciente a la Empresa de Ingeniería del Transporte), constituyen ejemplos de organizaciones beneficiadas con sus servicios.

CALISOFT, cuenta con dos estructuras productivas: una de evaluación de productos (Laboratorio Industrial de Pruebas de Software, LIPS) y otro de evaluación a procesos (Departamento de Consultoría y Evaluación de Procesos), protagonistas ambos de los servicios que ofrece. Entre estos, destacan:

- En cuanto a productos:
    - Evaluaciones a productos: ejecución de pruebas a aplicaciones informáticas, dirigidas a evaluar las características de calidad del software, según la norma NC ISO IEC 25010: 2011.
    - Pruebas de aceptación y/o piloto: evaluación a soluciones informáticas en el entorno real de aplicación, donde el cliente y/o usuario final se desempeña como probador del producto.
    - Pruebas de aceptación en adquisiciones de productos.
- En cuanto a procesos:
    - Evaluación a procesos: brinda una visión objetiva de la adherencia de la organización cliente a las disposiciones normalizadas por esta en el desarrollo de software o prestación de servicios de tecnologías de la información.
    - Consultorías: dirigidas a la mejora de procesos basados en modelos y estándares de calidad, enfocados al desarrollo de software o la prestación de servicios de tecnologías de la información.
    - Diagnósticos: dirigidos a solventar las necesidades de información relacionadas con los procesos productivos y entornos tecnológicos de las organizaciones desarrolladoras de software o prestadoras de servicios de tecnologías de la información.

Actualmente CALISOFT se encuentra inmerso, junto a reconocidas entidades desarrolladoras de software de Cuba, en la definición del Modelo de Calidad para el Desarrollo de Aplicaciones Informáticas (MCDAI), que permitirá incorporar buenas prácticas internacionalmente reconocidas en la industria nacional del software. La entidad desarrolla, además, un proyecto de mejora de sus procesos, basado en los requisitos de la NC ISO 9001:2012.

Las misiones principales del LIPS, consisten en evaluar técnicamente productos informáticos nacionales e importados, según lo establecido en las normas cubanas e

internacionales; proponer técnicas, políticas, procedimientos y estándares de calidad y realizar pruebas de liberación con la calidad requerida. Para asegurar que el resultado de las pruebas realizadas resulte correcto y confiable, se enfocan en acreditar el laboratorio en base a la norma NC ISO/IEC 17 025 del 2008.

En la actualidad, el proceso de evaluación que ejecutan es semejante a los descritos con anterioridad, basado en los estándares internacionales. A pesar de la existencia de aspectos subjetivos valorados por expertos, estos no se incluyen en la agregación de los datos para la obtención del índice de calidad, calculado a través del promedio de los criterios numéricos.

El proceso de evaluación y selección de productos de software, según su calidad, suele resultar costoso. Montar un escenario de pruebas, implica una elevada utilización de recursos, tanto materiales como humanos. No se configura un entorno de pruebas, que en ocasiones conlleva la participación de usuarios finales, si no existe seguridad en la estabilidad del software a evaluar.

En CALISOFT, antes de comenzar la evaluación de un producto o conjunto de productos con todos los requisitos, someten estos a pruebas exploratorias, constituyendo un desafío determinar el conjunto mínimo de criterios para realizarlas. El análisis de la interdependencia entre criterios de calidad y el algoritmo para recomendar el orden de las pruebas (Ver Algoritmo 2), ayuda a excluir criterios y seleccionar otros, lo que apoya y facilita el proceso.

Si no se realiza una correcta evaluación de la calidad, habrá gastado recursos y tiempo de manera innecesaria, además de conllevar a las diferentes instituciones a la toma de decisiones incorrectas, que pudieran implicar grandes pérdidas económicas para el país.

A continuación, se describe el modelo propuesto, aplicado a una muestra de doce productos de software.

Descripción de los datos del problema:
Para este caso real, se decidió aplicar el modelo en un entorno controlado en CALISOFT, con el objetivo de evaluar el índice de calidad de doce aplicaciones web, clasificadas en sistema de información para servicios de salud, de alta y media complejidad; todas están desplegándose en Cuba y ocho también en el extranjero. Para preservar información confidencial de la empresa, se sustituyen los nombres reales de los productos por los identificadores P1 hasta P12.

El proceso, inició con la solicitud de la evaluación de los productos, detallando los requisitos de calidad a valorar al laboratorio de pruebas por parte de los clientes (los

propietarios de los productos, en este caso un organismo del Estado responsable de estos servicios, las empresas productoras y adquisidoras). El laboratorio, aceptó la solicitud y requirió los productos y documentación asociada a cada uno. El líder de la evaluación, convocó a los involucrados a la reunión de inicio, donde explicó el proceso de evaluación, precisaron los requisitos de calidad y otros detalles.

En la reunión, fijaron los siguientes requisitos:

REQ 1. Todas las funcionalidades incluidas deben ejecutarse correctamente, con alta precisión. Garantizar que en el producto o sistema estén desarrolladas todas las funcionalidades. El fallo, si ocurre, debe estar controlado.

REQ 2. Atender las peticiones en un tiempo menor a los 5 segundos y las peticiones a la base de datos en un tiempo inferior a los 3 segundos, garantizar un consumo de CPU y RAM inferior al 80%.

REQ 3. Debe ser de fácil operación, los formularios bien estructurados, tipos y tamaños de fuente legibles, distinguir los campos obligatorios, utilización de opciones de cancelar, cerrar, etc.

REQ 4. Interfaz amigable que satisfaga al usuario, mantener una tipografía coherente, utilizar los estilos (negritas, cursivas, etc.) con moderación, garantizar que las imágenes, gráficos, tablas, etc. utilizadas tengan buena resolución y que no existan errores ortográficos.

REQ 5. Fácil de aprender a utilizarlo y de comprensión de sus funciones, emplear nombres estandarizados, ofrecer una navegación sencilla.

REQ 6. Poder ejecutarse en diferentes sistemas operativos.

REQ 7. De fácil instalación.

Solución:
Se comenzó a realizar la evaluación de productos de software, con la ejecución de las tareas del componente C1 Estructuración del problema.

A partir de los requisitos de calidad, se determinó el modelo de calidad a utilizar con sus características, subcaracterísticas y medidas. Esta empresa utiliza el modelo de calidad definido en la norma NC ISO 25010, en este caso con adaptaciones, ya que no evalúan todas las características (Ver Tabla III.1).

Tabla III.1 Correspondencia entre requisitos y modelo de calidad.

Requisitos de calidad	Características	Sub características
REQ 1	Adecuación funcional	Completitud funcional, corrección funcional y pertinencia funcional
	Fiabilidad	Tolerancia a fallos, madurez y disponibilidad
REQ 2	Eficiencia	Comportamiento temporal y uso de recursos
REQ 3, 4, 5.	Usabilidad	Capacidad para operar, capacidad para reconocer su adecuación, capacidad de aprendizaje, estética
REQ 6, 7.	Portabilidad	Capacidad para ser instalado, adaptabilidad

De cada medida, determinó el dominio de datos (Ver Tabla III.2), y si era de costo (minimizar) o beneficio (maximizar). Además, determinaron los criterios esenciales y sus umbrales (Ver Tabla III.3).

A continuación, ejecutó el subcomponente de valoración de los criterios. Lo primero fue determinar los vectores pesos para las características, subcaracterísticas y medidas, según el dominio de solución donde se desarrollaron los productos y el uso final de los mismos. Aunque tienen diferentes niveles de importancia, todas las características de calidad debían evaluarse, pues cada una juega su papel, atendiendo a que permiten evaluar diferentes aristas de la calidad del producto de software. Para estimar los vectores pesos realizaron encuentros con cada grupo involucrado en la evaluación (expertos, evaluadores y clientes). Al no lograrse consenso, se decide la realización de un grupo focal, con el objetivo de alcanzarlo. Se utilizaron dos especialistas neutrales, uno para la moderación y otro encargado de la relatoría. Realizaron dos sesiones de trabajo, siendo necesario auxiliarse de procedimientos para la estimación de pesos preferenciales. El resultado, se muestra en la Tabla III.4.

Tabla III.2. Definición del dominio de datos de las medidas.

Características	**Sub características**	**Medidas de calidad**	**Dominio**	**Costo/ Beneficio**
Adecuación funcional	Completitud funcional	Cobertura de aplicación funcional (CAF)	[0,1]	B
		Idoneidad funcional *(IF)*	[0,m]	C
	Pertinencia funcional	Estabilidad de la especificación funcional *(EEF)*	[0,1]	B
	Corrección funcional	Corrección Computacional (CC)	[0,1]	B
Eficiencia	Comportamiento temporal	Tareas por tiempo (TpT)	[0, 40]	B
	Uso de recursos	Uso de Memoria (MEM)	[0, 10]	C
Fiabilidad	Tolerancia a fallos	Evitar el fracaso (EF)	[0,m]	B
	Madurez	Densidad de fracasos frente caso de pruebas (DF)	[0,1]	C
		Cobertura de las pruebas (CP)	[0,1]	B
	Disponibilidad	Tiempo de inactividad (Tina)	[0,m]	C
Portabilidad	Capacidad para ser instalado	Facilidad de instalación (Fins)	(MB, B,M, A, MA)	B
	Adaptabilidad	Adaptabilidad de Hardware (Ahw)	[0,1]	B
		Adaptabilidad de Software (Asw)	[0,1]	B
Usabilidad	Capacidad para reconocer su adecuación	Descripción Exhaustiva (DE)	[0,1]	B
	Capacidad de aprendizaje	Integridad de la documentación (IntDoc)	[0,1]	B
	Capacidad para operar	Satisfacción (Sat)	(MB, B,M, A, MA)	B
	Estética	Apariencia (Apa)	(NA, PA, AM, A, MA)	B

Tabla III.3. Criterios esenciales y sus umbrales.

Cobertura de aplicación funcional (CAF)	Estabilidad de la especificación funcional *(EEF)*	Corrección Computacional (CC)	Tareas por tiempo (TpT)	Uso de Memoria (MEM)	Densidad de fracasos frente caso de pruebas (DF)	Cobertura de las pruebas (CP)	Satisfacción (Sat)	Apariencia (Apa)
[0,25;1]	[0,25;1]	[0,25;1]	[3;40]	[0;1,25]	[0;0,5]	[0,5;1]	(M, A, MA)	(AM, A, MA)

Tabla III.4 Vectores peso para los criterios.

Características	Peso (W)	Sub características	Peso (W)	Medidas de calidad	Peso (W)
Adecuación funcional	0.443182997	Completitud funcional	0.607962213	Cobertura de aplicación funcional (CAF)	0.75
				Idoneidad funcional *(IF)*	0.25
		Pertinencia funcional	0.11993927	Estabilidad de la especificación funcional *(EEF)*	1
		Corrección funcional	0.27209852	Corrección Computacional (CC)	1
Eficiencia	0.097085233	Comportamiento temporal	0.75	Tareas por tiempo (TpT)	1
		Uso de recursos	0.25	Uso de Memoria (MEM)	1
Fiabilidad	0.239853101	Tolerancia a fallos	0.23948761	Evitar el fracaso (EF)	1
		Madurez	0.623224728	Densidad de fracasos frente caso de pruebas (DF)	0.75
				Cobertura de las pruebas (CP)	0.25
		Disponibilidad	0.13728766	Tiempo de inactividad (Tina)	1
Portabilidad	0.061346944	Capacidad para ser instalado	0.25	Facilidad de instalación (Fins)	1
		Adaptabilidad	0.75	Adaptabilidad de Hardware (Ahw)	0.5
				Adaptabilidad de Software (Asw)	0.5
Usabilidad	0.158531725	Capacidad para reconocer su adecuación	0.27714047	Descripción Exhaustiva (DE)	1
		Capacidad de aprendizaje	0.0959699	Integridad de la documentación (IntDoc)	1
		Capacidad para operar	0.4658194	Satisfacción (Sat)	1
		Estética	0.161070234	Apariencia (Apa)	1

Posteriormente, se definió la relación existente entre criterios, con el uso de los MCD (Ver Figuras III.11 y III.12). Para establecer la interdependencia entre criterios de un mismo nivel, se conformó el MCD consensuado para el grupo de hermanos. Esto se realizó con la ayuda de la herramienta NEXO para la toma de decisiones en grupo, mediante la aplicación del modelo Ágora (modelo para el proceso de logro de consenso) a los MCD individuales de los usuarios [158]. Se agregaron los diferentes

$MCDc$ de un mismo nivel y se derivó la matriz de adyacencia (MAc), del nivel la cual se denomina matriz de interdependencia entre criterios de un nivel l (MI^l).

Aquí concluye el componente C1, obteniendo como salida los parámetros de la evaluación: el modelo de calidad, que involucra criterios, criterios esenciales (CE^l), medidas, pesos de los criterios (W^l), e interdependencia entre criterios (MI^l).

Seguidamente, se ejecutó el componente C2 Evaluación. Se estableció la planificación de las pruebas; aplicando el algoritmo para recomendar el orden a seguir. Como resultado de este subcomponente, se realizó un subconjunto de pruebas de funcionalidad, usabilidad y estrés, para determinar si los productos eran estables y qué escenarios montar. Además, se aconsejó iniciar con las pruebas de funcionalidad, usabilidad y fiabilidad (específicamente estrés, integridad y estructura), continuando con las de rendimiento y por último las de portabilidad.

Siguiendo el plan de pruebas, se ejecutaron las mismas y se obtuvieron los elementos de medidas de calidad (Ver Anexo III). A partir de estos últimos, se calcularon las medidas de calidad, cuyo resultado se muestra en la Tabla III.5.

Las medidas están definidas en dominios diferentes y con diferentes escalas, por lo que resultaba necesario normalizar la información derivada de las pruebas y unificar en un dominio de datos; en este caso, los números difusos triangulares, y obtuvo la matriz de evaluación para el nivel 3 (Me^3) (Ver Tablas III.6 y III.7).

Paralelamente, se determinó el vector de penalización (*P*), con el objetivo de penalizar los productos que no cumplían con las restricciones de los criterios esenciales

Concluyó el componente C2, obteniendo la matriz de evaluación y el vector de penalización.

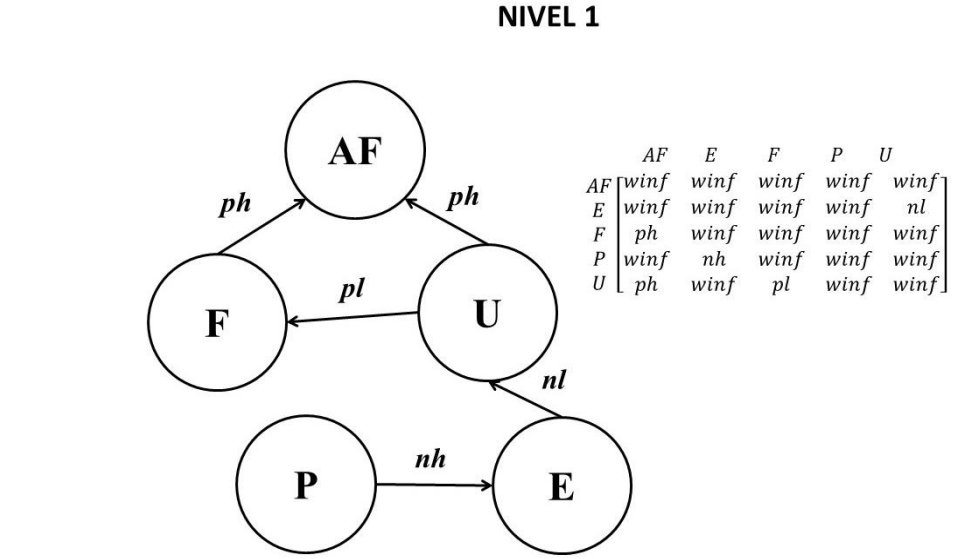

Figura III.11. Mapa Cognitivo Difuso del nivel 1 (Características)

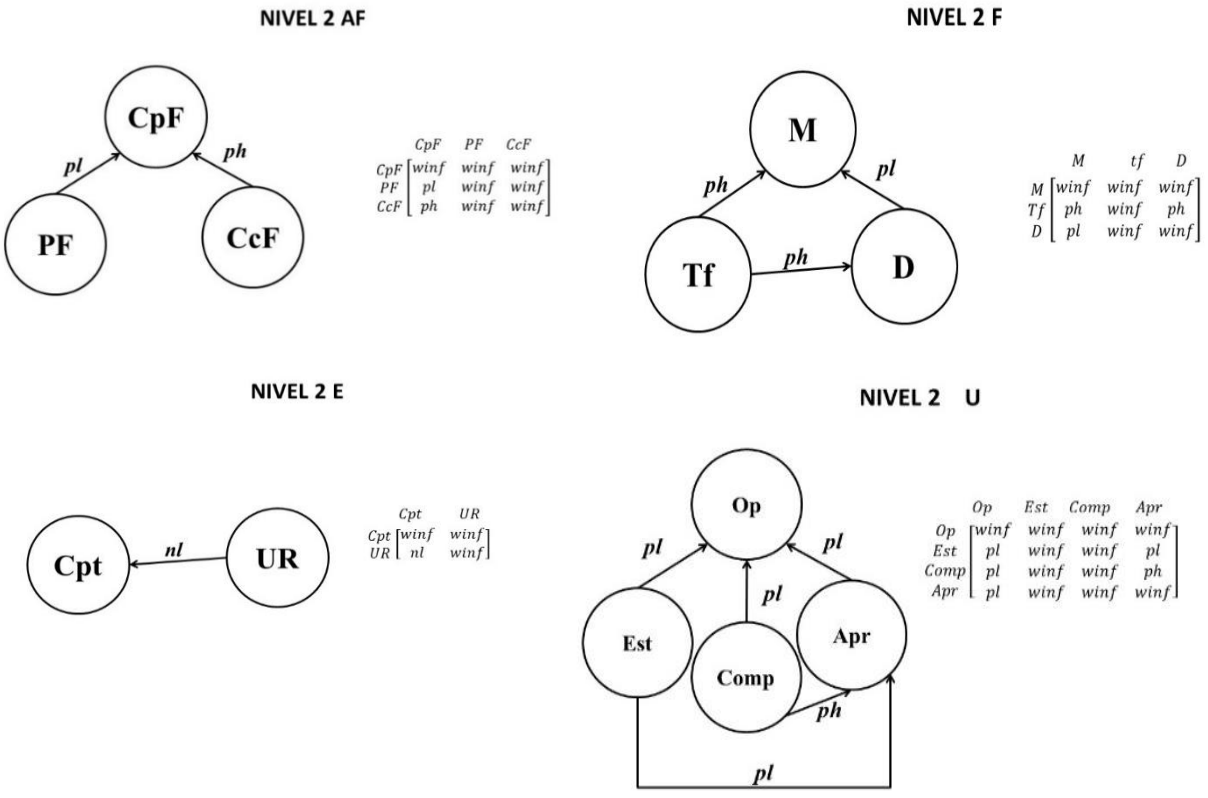

Figura III.12. Mapas Cognitivos Difusos del nivel 2 (Sub características)

Tabla III.5. Medidas de calidad para cada uno de los productos de software

Código	Adecuación funcional (AF)				Eficiencia	
	Completitud funcional (CpF)		Pertinencia funcional	Corrección funcional (CcF)	Comportamiento temporal	Uso de recursos
	Cobertura de aplicación funcional (CAF)	Idoneidad funcional *(IF)*	Estabilidad de la especificación funcional *(EEF)*	Corrección Computacional (CC)	Tareas por tiempo (TpT)	Uso de Memoria (MEM)
P1	0.62	0.76	0.64	0.38	1.47	2
P2	0.696	0	0.73	0.69620253	3.92	1
P3	0.58	0.57	0.58	0.34615385	1.75	3
P4	0.71	0.12	0.72	0.63855422	3.4	1
P5	0.43	0.86	0.47	0.33173077	3.08	3
P6	0.12	1.05	0.16	0.12244898	0.63	2
P7	0.65	0.666666667	0.65	0.31481481	0.45	2
P8	0.92	0.238095238	0.92	0.71	3	0.5
P9	0.68	0.39	0.68	0.44230769	0.94	1
P10	0.79	0.285714286	0.79	0.69767442	1.17	1
P11	0.38	0.125	0.48	0.26153846	1.78	1
P12	0.89	0.3	0.89	0.63513514	4	1

Código	Fiabilidad				Portabilidad		
	Tolerancia a fallos	Madurez		Disponibilidad	Capacidad para ser instalado	Adaptabilidad	
	Evitar el fracaso (EF)	Densidad de fracasos frente caso de pruebas (DF)	Cobertura de las pruebas (CP)	Tiempo de inactividad (Tina)	Facilidad de instalación (Fins)	Adaptabilidad de Hardware (Ahw)	Adaptabilidad de Software (Asw)
P1	0.4	0.0172	0.98	1.5789474	M	0.45	0.55
P2	0.382979	0.008148483	0.98	1.3636364	M	0.8	0.8
P3	0.714286	0.117346939	0.93	3	M	0.5	0.8
P4	0.941176	0.018838908	1	7.5	A	0.7	0.9
P5	0.743243	0.005021914	1	2	M	0.5	0.5
P6	0.454545	0.084710744	1	1.1428571	A	0.7	0.7
P7	0.833333	0.216049383	1	3	M	0.6	0.7
P8	1	0.06122449	0.95	10	MA	0.9	0.9
P9	0.208333	0.010864645	1	1.5	M	0.5	0.5
P10	0.714286	0.071428571	1	3	M	0.5	0.5
P11	0.9375	0.015869141	1	1.5	M	0.5	0.5
P12	0.78125	0.015136719	1	1.875	M	0.5	0.5

Código	Usabilidad			
	Capacidad para reconocer su adecuación	Capacidad de aprendizaje	Capacidad para operar	Estética
	Descripción Exhaustiva (DE)	Integridad de la documentación (IntDoc)	Satisfacción (Sat)	Apariencia (Apa)
P1	0.5	0.92	M	A
P2	0.5	0.87	M	A
P3	0.5	0.69	M	A
P4	1	0.86	M	A
P5	0.5	0.76	M	A
P6	0.5	0.92	A	A
P7	0.5	0.8	M	A
P8	1	0.96	MA	MA
P9	0.5	0.74	M	A
P10	0.5	0.88	M	A
P11	0	0.72	M	A
P12	1	0.95	M	A

Tabla III.6. Normalización de los datos para obtener la matriz de evaluación.

Código	Adecuación funcional (AF)				Eficiencia	
	Completitud funcional (CpF)		Pertinencia funcional	Corrección funcional (CcF)	Comportamiento temporal	Uso de recursos
	Cobertura de aplicación funcional (CAF)	Idoneidad funcional *(IF)*	Estabilidad de la especificación funcional *(EEF)*	Corrección Computacional (CC)	Tareas por tiempo (TpT)	Uso de Memoria (MEM)
P1	0.62	0.848	0.64	0.38	0.294	0.8
P2	0.696	1	0.73	0.696202532	0.784	0.9
P3	0.58	0.886	0.58	0.346153846	0.35	0.7
P4	0.71	0.976	0.72	0.638554217	0.68	0.9
P5	0.43	0.828	0.47	0.331730769	0.616	0.7
P6	0.12	0.79	0.16	0.12244898	0.126	0.8
P7	0.65	0.86666667	0.65	0.314814815	0.09	0.8
P8	0.92	0.95238095	0.92	0.71	0.6	0.95
P9	0.68	0.922	0.68	0.442307692	0.188	0.9
P10	0.79	0.94285714	0.79	0.697674419	0.234	0.9
P11	0.38	0.975	0.48	0.261538462	0.356	0.9
P12	0.89	0.94	0.89	0.635135135	0.8	0.9

Código	Fiabilidad				Portabilidad		
	Tolerancia a fallos	Madurez		Disponibilidad	Capacidad para ser instalado	Adaptabilidad	
	Evitar el fracaso (EF)	Densidad de fracasos frente caso de pruebas (DF)	Cobertura de las pruebas (CP)	Tiempo de inactividad (Tina)	Facilidad de instalación (Fins)	Adaptabilidad de Hardware (Ahw)	Adaptabilidad de Software (Asw)
P1	0.4	0.9828	0.98	0.921052632	M	0.45	0.55
P2	0.382978723	0.99185152	0.98	0.931818182	M	0.8	0.8
P3	0.714285714	0.88265306	0.93	0.85	M	0.5	0.8
P4	0.941176471	0.98116109	1	0.625	A	0.7	0.9
P5	0.743243243	0.99497809	1	0.9	M	0.5	0.5
P6	0.454545455	0.91528926	1	0.942857143	A	0.7	0.7
P7	0.833333333	0.78395062	1	0.85	M	0.6	0.7
P8	1	0.93877551	0.95	0.5	MA	0.9	0.9
P9	0.208333333	0.98913536	1	0.925	M	0.5	0.5
P10	0.714285714	0.92857143	1	0.85	M	0.5	0.5
P11	0.9375	0.98413086	1	0.925	M	0.5	0.5
P12	0.78125	0.98486328	1	0.90625	M	0.5	0.5

Código	Usabilidad			
	Capacidad para reconocer su adecuación	Capacidad de aprendizaje	Capacidad para operar	Estética
	Descripción Exhaustiva (DE)	Integridad de la documentación (IntDoc)	Satisfacción (Sat)	Apariencia (Apa)
P1	0.5	0.92	M	A
P2	0.5	0.87	M	A
P3	0.5	0.69	M	A
P4	1	0.86	M	A
P5	0.5	0.76	M	A
P6	0.5	0.92	A	A
P7	0.5	0.8	M	A
P8	1	0.96	MA	MA
P9	0.5	0.74	M	A
P10	0.5	0.88	M	A
P11	0	0.72	M	A
P12	1	0.95	M	A

Tabla III.7. Unificación de los datos para obtener la matriz de evaluación.

Código	Adecuación funcional (AF)				Eficiencia	
	Completitud funcional (CpF)		Pertinencia funcional	Corrección funcional (CcF)	Comportamiento temporal	Uso de recursos
	Cobertura de aplicación funcional (CAF)	Idoneidad funcional *(IF)*	Estabilidad de la especificación funcional *(EEF)*	Corrección Computacional (CC)	Tareas por tiempo (TpT)	Uso de Memoria (MEM)
P1	M	A	A	M	B	A
P2	A	MA	A	A	A	MA
P3	M	MA	M	B	B	A
P4	A	MA	A	A	A	MA
P5	M	A	M	B	M	A
P6	MB	A	B	MB	B	A
P7	A	A	A	B	MB	A
P8	MA	MA	MA	A	M	MA
P9	A	MA	A	M	B	MA
P10	A	MA	A	A	B	MA
P11	M	MA	M	B	B	MA
P12	MA	MA	MA	A	A	MA

Código	Fiabilidad				Portabilidad		
	Tolerancia a fallos	Madurez		Disponibilidad	Capacidad para ser instalado	Adaptabilidad	
	Evitar el fracaso (EF)	Densidad de fracasos frente caso de pruebas (DF)	Cobertura de las pruebas (CP)	Tiempo de inactividad (Tina)	Facilidad de instalación (Fins)	Adaptabilidad de Hardware (Ahw)	Adaptabilidad de Software (Asw)
P1	M	MA	MA	MA	M	M	M
P2	M	MA	MA	MA	M	A	A
P3	A	MA	MA	A	M	M	A
P4	MA	MA	MA	M	A	A	MA
P5	A	MA	MA	MA	M	M	M
P6	M	MA	MA	MA	A	A	A
P7	A	A	MA	A	M	M	A
P8	MA	MA	MA	M	MA	MA	MA
P9	B	MA	MA	MA	M	M	M
P10	A	MA	MA	A	M	M	M
P11	MA	MA	MA	MA	M	M	M
P12	A	MA	MA	MA	M	M	M

Código	Usabilidad			
	Capacidad para reconocer su adecuación	Capacidad de aprendizaje	Capacidad para operar	Estética
	Descripción Exhaustiva (DE)	Integridad de la documentación (IntDoc)	Satisfacción (Sat)	Apariencia (Apa)
P1	M	MA	M	A
P2	M	A	M	A
P3	M	A	M	A
P4	MA	A	M	A
P5	M	A	M	A
P6	M	MA	A	A
P7	M	A	M	A
P8	MA	MA	MA	MA
P9	M	A	M	A
P10	M	MA	M	A
P11	MB	A	M	A
P12	MA	MA	M	A

Finalmente, se realizó el componente C3. Agregación y recomendación, para obtener el índice de calidad. Primero, se obtuvo la matriz G, contentiva del valor de los criterios con la influencia del resto de los mismos, según la Ec. *II.26)*, y se ponderó el valor de cada criterio con el vector peso *(Ver Ec. II.27).*

El próximo paso, fue la agregación de las medidas para la obtención de los valores de las sub-características y la penalización de las subcaracterísticas, utilizando la Ec. *II.28*. Los valores de las subcaracterísticas ya penalizadas, se muestran en la Tabla III.8. Estos últimos pasos, fueron ejecutados iterativamente hasta determinar el índice de calidad (IQ) (Ver Tabla III.9) y el nivel de calidad (QL).

Tabla III.8. Valores de las subcaracterísticas

Código	Adecuación funcional			Eficiencia			Fiabilidad		
P1	0.3522	0.4013	0.4055	0.0452	0.0449	0.0526	0.1604	0.1753	0.1876
P2	0.3532	0.4017	0.4058	0.0469	0.0478	0.0567	0.1604	0.1753	0.1876
P3	0.3518	0.4010	0.4054	0.0451	0.0448	0.0526	0.1607	0.1756	0.1879
P4	0.3540	0.4025	0.4061	0.0467	0.0476	0.0567	0.1610	0.1760	0.1879
P5	0.3517	0.4010	0.4054	0.0477	0.0476	0.0556	0.1608	0.1757	0.1879
P6	0.3389	0.3909	0.3946	0.0449	0.0446	0.0525	0.1604	0.1754	0.1877
P7	0.3520	0.4011	0.4056	0.0451	0.0448	0.0526	0.1605	0.1754	0.1879
P8	0.3550	0.4033	0.4062	0.0463	0.0472	0.0565	0.1610	0.1762	0.1880
P9	0.3524	0.4010	0.4053	0.0444	0.0450	0.0537	0.1600	0.1748	0.1872
P10	0.3535	0.4020	0.4061	0.0444	0.0450	0.0537	0.1607	0.1756	0.1879
P11	0.3520	0.4009	0.4049	0.0444	0.0450	0.0537	0.1612	0.1759	0.1876
P12	0.3543	0.4027	0.4063	0.0472	0.0480	0.0568	0.1608	0.1759	0.1880

Código	Portabilidad			Usabilidad		
P1	0.0401	0.0403	0.0405	0.0929	0.1015	0.1088
P2	0.0403	0.0404	0.0406	0.0910	0.1000	0.1088
P3	0.0402	0.0404	0.0406	0.0929	0.1015	0.1088
P4	0.0404	0.0406	0.0407	0.0913	0.1004	0.1090
P5	0.0401	0.0403	0.0405	0.0905	0.1002	0.1088
P6	0.0403	0.0405	0.0407	0.0931	0.1017	0.1089
P7	0.0402	0.0404	0.0406	0.0929	0.1015	0.1088
P8	0.0405	0.0407	0.0407	0.0922	0.1010	0.1091
P9	0.0401	0.0403	0.0405	0.0937	0.1014	0.1088
P10	0.0401	0.0403	0.0405	0.0937	0.1015	0.1088
P11	0.0401	0.0403	0.0405	0.0935	0.1010	0.1084
P12	0.0401	0.0403	0.0405	0.0914	0.1005	0.1090

Análisis entre métodos:

Se comparan los índices de calidad obtenidos por diferentes métodos (con o sin interdependencia), cuyo resultado se contrasta con un TOPSIS modificado, que no incluye la interdependencia. El índice de calidad según cada método, se muestra en la

Tabla III.10. La correlación entre los métodos, se ilustra en la Tabla III.11, usando una escala de tres colores, donde el verde muestra los valores similares y el rojo los diferentes.

Tabla III.9. Índices de calidad.

Código	Índice de calidad	Ranking
P1	0.7565	7
P2	0.7587	4
P3	0.7565	8
P4	0.7604	3
P5	0.7581	5
P6	0.7457	12
P7	0.7564	9
P8	0.7615	1
P9	0.7560	11
P10	0.7578	6
P11	0.7564	10
P12	0.7607	2

TOPSIS, no permite la interdependencia, y su concepción es diferente en el sentido de que trata los valores de distancia hacia un ideal positivo y uno negativo. El experimento corrobora que la incorporación de la interdependencia entre los criterios introduce cambios en la solución del problema, siendo este un elemento clave a tener en cuenta en la selección y evaluación de productos de software.

Tabla III.10. Índice de calidad por cada método.

Código	***TOPSIS***	***Con Interdependencia***	***Sin Interdependencia***
P1	0.5700522	0.756509321	0.57854811
P2	0.74010191	0.758711624	0.7594518
P3	0.54298263	0.756461074	0.55922716
P4	0.75571609	0.760370322	0.78822435
P5	0.54848154	0.758114103	0.58270977
P6	0.38963868	0.745713404	0.36062646
P7	0.59477069	0.756447202	0.58564576
P8	0.83639786	0.76145407	0.88474509
P9	0.65555625	0.755998396	0.64883098
P10	0.70789644	0.757847338	0.70400873
P11	0.54149367	0.756430424	0.57542599
P12	0.8192678	0.760704789	0.83879022

Tabla III.11 Correlación entre métodos por el índice de calidad.

	TOPSIS	***Con Interdependencia***	***Sin Interdependencia***
TOPSIS	1	0.848754459	0.9913218
Con Interdependencia	0.84875446	1	0.8912385
Sin Interdependencia	0.9913218	0.891238505	1

Experimento para validar la satisfacción del cliente y la facilidad en la toma de decisiones:

La satisfacción del cliente, se evalúa a través de dimensiones como la percepción de la calidad y organización del servicio (tiempo de respuesta, comunicación, eficacia en resolución de problemas), información resultante de la evaluación, software o herramienta de apoyo. Es importante tener en cuenta aspectos propios de la calidad del servicio, como son la información obtenida del proceso de evaluación, la evaluación integral de las características de calidad, el índice y la rapidez con que el laboratorio atiende sus solicitudes.

Para facilitar la toma de decisiones, es esencial crear una base lógica, racional y transparente para el análisis, logrando una mejor estructuración del problema y mayor participación e influencia de todos.

El modelo desarrollado ha logrado mejorar la satisfacción de los clientes del laboratorio de pruebas y facilitar la toma de decisiones de los directivos de las empresas productoras que evalúan sus productos. Dichas características, se convierten en indicadores para su validación.

Con el objetivo de validar en la práctica el modelo y la herramienta propuesta, se realizó un experimento, cuyo diseño consistió en la aplicación del modelo y la utilización de la herramienta automatizada, como muestra de las facilidades que posee la misma y la efectividad del modelo.

Se seleccionaron tres productos de software, pertenecientes a la evaluación de aplicaciones de Gestión de Entidades. Para preservar información confidencial, se sustituyen los nombres reales por otros identificadores.

La selección está basada en los criterios siguientes: a fines de 2015, los proyectos participaron en el piloto para alcanzar el nivel 2 de CMMi y se encuentran mejor organizados y centrados en lograr la calidad del producto final. De ellos, se tiene la documentación necesaria para realizar la evaluación. Son productos estables, que concluyeron su desarrollo en 2016, lo que ha favorecido poder aplicarles el modelo de evaluación en más de una ocasión al año y observar los avances experimentados. La Figura III.13, muestra los resultados.

En el Producto A, las No Conformidades (NC) detectadas en la primera evaluación, fueron erradicadas y en el segundo período de evaluación, alcanzaron el máximo valor, para la característica Portabilidad. En el caso del Producto B, se pudo determinar que continúa con iguales problemas en dicha característica o generaron otros al resolver las NC anteriores, pues en ambos períodos obtiene la misma

evaluación. Por el contrario, en el Producto C, en los dos períodos de evaluación, mantuvo la Eficiencia, con una evaluación satisfactoria; es decir, que continúa cumpliendo los requisitos de calidad para esta característica. Los jefes de proyecto deben enfocarse, por tanto, en dichos resultados, determinar planes de acciones correctivas y verificar su cumplimiento.

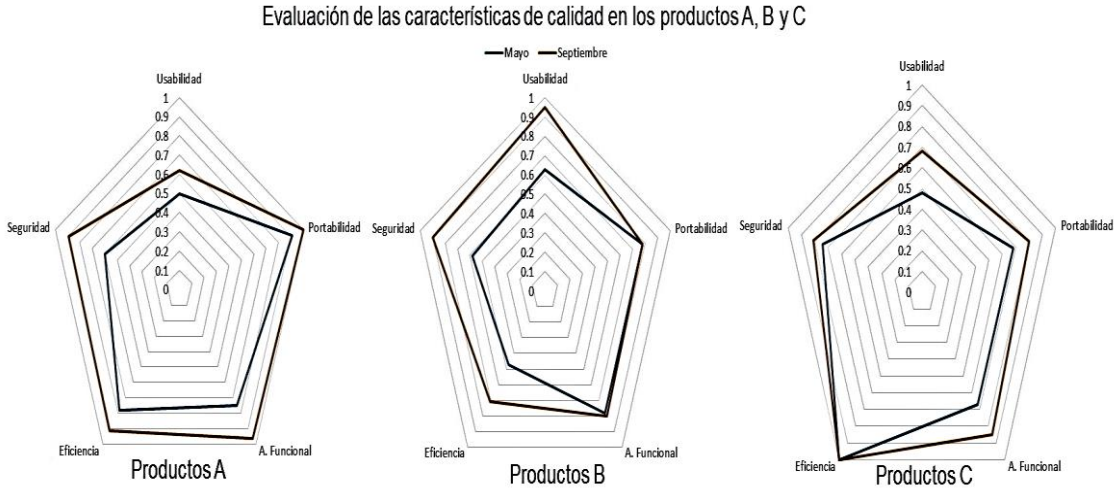

Figura III.13 Evaluaciones de los productos de software (mayo-septiembre/2017).

Los resultados, evidencian el análisis que permite el modelo de evaluación propuesto y la calidad en las decisiones tomadas. Los mismos influyen en la rapidez para la toma de decisiones, pues brinda la información necesaria de la evaluación. La evaluación integral y escalonada de los criterios de la calidad, ofrece información cualitativa y cuantitativa a los clientes.

La herramienta automatizada, muestra los valores desglosados en características, subcaracterísticas y medidas; además, permite agilizar el proceso y alcanzar rapidez en el servicio, lo que se traduce como satisfacción del cliente.

La Figura III.14, muestra los resultados de la evaluación del índice de calidad.

De manera general, los productos A, B y C, en la primera evaluación (mayo) obtuvieron un índice de calidad más bajos que en la segunda iteración (septiembre). Dichos resultados, permitieron a la alta gerencia definir en qué equipo de proyecto debían reforzar las actividades de gestión de la calidad del producto de software y definir una estrategia para motivar a los equipos de proyectos más rezagados.

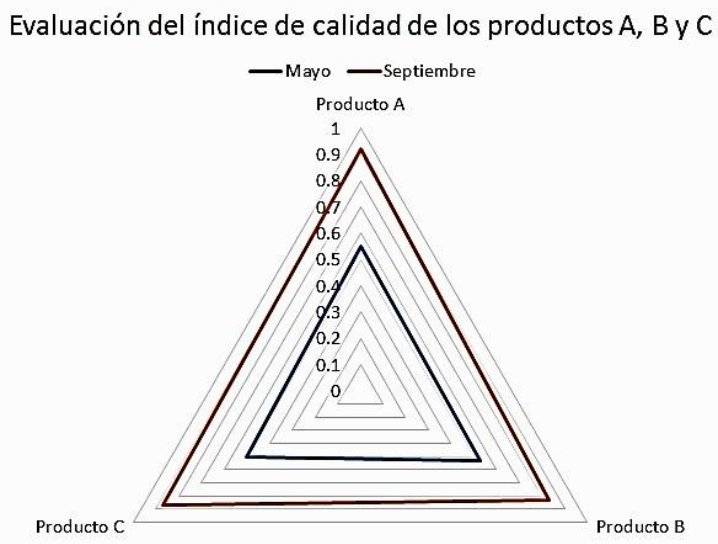

Figura III.14 Evaluación del índice de calidad de los productos A, B y C.

El incremento demostró la factibilidad de las acciones tomadas por la alta gerencia y la calidad de la toma de decisiones, en cuanto a la definición de los objetivos de calidad para las nuevas versiones y futuros desarrollos. Dichos análisis valieron para demostrar con datos reales las tendencias de incumplimiento de los requisitos de calidad y a su vez proponer acciones para revertir la situación.

De acuerdo a los indicadores planteados para la validación, obtuvo el siguiente resultado:

- Toma de decisiones: Los resultados del experimento validaron el método y la herramienta, demostrando la facilidad y ayuda que brinda a los jefes para tomar decisiones, pues los datos cuantitativos que ofrece, permiten crear una base lógica, racional y transparente para el análisis, logrando una mejor estructuración del problema y trazar acciones para elevar la calidad de las nuevas versiones o futuros productos.
- Satisfacción del cliente: Con los resultados escalonados, a nivel de características, subcaracterísticas y por último el índice de calidad, visualizados en la herramienta, los clientes se sienten con información rápida, entendible y útil para determinar si adquieren o no el producto o si han desarrollado un producto con un índice de calidad aceptable. La misma favorece la organización del servicio y el tiempo de respuesta. La evaluación integral, ofrece mayor confianza y efectividad del servicio brindado por el LIPS.

CONCLUSIONES Y TRABAJOS FUTUROS

A continuación se presentan las conclusiones derivadas de esta investigación, enfatizando los resultados alcanzados. Además, se comentan las líneas de investigación propuestas a desarrollar para dar continuidad a la misma en un futuro.

Conclusiones

La relevancia de la evaluación de la calidad de los productos de software, tanto como la selección de estos para su uso bajo condiciones específicas y las limitaciones de los métodos tradicionales para llevar a cabo este proceso, que no permiten modelar adecuadamente la naturaleza del mismo, motivaron el desarrollo de la investigación, guiada por los objetivos trazados, enunciados en la introducción, como:

1. Construir el marco teórico referencial relacionado con el análisis de los elementos fundamentales del proceso de evaluación de la calidad y la selección de los productos de software, que facilite la toma de decisiones bajo incertidumbre, con el propósito de deslindar e identificar las limitaciones existentes. Además, analizar en profundidad los métodos más importantes de solución, con énfasis en las técnicas de *Soft Computing* más utilizadas.
2. Considerar la evaluación y selección de productos de software como un problema de toma de decisiones, utilizando las bases teóricas de la lógica difusa y la modelación lingüística, lo que implica el desarrollo de un modelo que formalice el proceso de evaluación y selección bajo incertidumbre, trate la información heterogénea, la interdependencia entre los criterios y los criterios esenciales, haciendo uso de técnicas de *Soft Computing*.
3. Desarrollar una herramienta para soportar la utilización del modelo propuesto, ya sea para la experimentación o el desarrollo de aplicaciones reales, tratando de forma sencilla y flexible los parámetros del problema y métodos de solución.
4. Contrastar la viabilidad, operatividad y eficacia de la propuesta, a partir de su aplicación en casos reales en la industria cubana de software, especialmente en el Centro Nacional de Calidad de Software cubano (CALISOFT) facilitando la toma de decisiones y la satisfacción del cliente.

Como consecuencia de los retos planteados en dichos objetivos, y el trabajo realizado a lo largo de la investigación, descrito en esta memoria, se pudieron alcanzar las siguientes conclusiones:

Con respecto al primer objetivo, se realizó un estudio de los fundamentos teóricos de la evaluación y selección de los productos de software, según su calidad, y se constató la pertinencia del tema en la actualidad, a partir de la presencia del software en todos

los ámbitos de la sociedad. A dicho problema se le ha dedicado atención desde diferentes esferas, como la computacional, la estratégica o la de la mercadotecnia.

Al analizar los diferentes modelos y métodos existentes, se determinó que presentan características comunes, como la definición de un modelo de calidad que estructura de manera arbórea los criterios y medidas a tener en cuenta en la evaluación. Además, se pudo identificar y confirmar las limitaciones que estos presentan para modelar la complejidad en el contexto real de heterogeneidad de los datos, información imprecisa, interdependencia entre los criterios y restricciones de criterios esenciales.

Tal situación, provoca que no se refleje la naturaleza del problema y se obtengan soluciones con una inadecuada correlación respecto a la calidad del producto en su operatividad. El análisis indujo al uso de técnicas de *Soft Computing*.

Por esa razón, el objetivo 2 de la tesis se logró con la definición de un modelo de selección y evaluación de productos de software, basado en la teoría de conjuntos difusos, modelación lingüística y MCD.

La solución modela adecuadamente la incertidumbre relacionada con el proceso y posibilita el tratamiento de la información heterogénea. También permite la valoración de la penalización debido a restricciones determinadas por la utilización de los productos en condiciones específicas.

Todo lo anterior, conduce a la obtención de resultados más precisos, facilitando la toma de decisiones sobre la adquisición y puesta en el mercado de un producto como corrección del proceso de desarrollo entre otras ventajas.

El modelo, permitió estructurar este tipo de problema de manera clara y lógica, precisando el marco de evaluación, la recopilación de los datos necesarios de diversas fuentes, el trabajo con diferentes dominios de datos y escalas, flexibilizando el proceso.

Para garantizar el uso práctico del modelo propuesto en la industria del software, este debe implementarse a través de un sistema automatizado, que abarque todo el proceso y facilite la toma de decisiones. Dicha afirmación nos conduce al tercer objetivo, para cuya consecución se ha desarrollado una herramienta de software libre (SoftQuality) que implementa el modelo propuesto y sirve de ayuda a los participantes en el proceso de evaluación y selección. Además, incorpora otras formas de agregación de la información, que permiten realizar comparaciones entre soluciones.

La herramienta y el modelo propuesto, han sido avalados y utilizados en la práctica por CALISOFT y otras empresas cubanas, lo que demuestra que se ha obtenido una solución sólida y útil.

Además, se analizó un caso práctico real, asociado a un problema de evaluación y selección de productos de software en CALISOFT, que evidencia la aplicabilidad de la

propuesta y su influencia en el proceso de toma de decisiones, dando cumplimiento al objetivo cuatro. Los resultados obtenidos en este caso de estudio, demuestran la utilidad, eficacia y operatividad del modelo en un contexto real.
Por último, se realizó un experimento con dos productos para validar la facilidad en la toma de decisiones y la satisfacción del cliente. Los resultados, demostraron que la aplicación del modelo favorece la organización, confianza, efectividad del servicio brindado y la disminución del tiempo de respuesta, mejorando así los niveles de satisfacción del cliente. La propuesta permite, asimismo, una mejor estructuración del problema; crea una base lógica, racional y transparente para el análisis y traza acciones con el objetivo de elevar la calidad de las nuevas versiones o futuros productos. A partir de estos, los clientes disponen de información rápida, entendible y útil, todo lo cual favorece la toma de decisiones.

De esta manera, han sido cumplidos satisfactoriamente todos los objetivos trazados al inicio de la investigación, quedando evidenciado en los resultados presentados en esta memoria.

Trabajos futuros

Dada la relevancia del tema abordado, la complejidad en ascenso del software y la necesidad de avanzar hacia el logro de la excelencia en los productos, se justifica la continuidad de la investigación, encaminándose hacia las siguientes líneas:

- Extender el modelo propuesto, incorporando la modelación de la naturaleza dinámica de la evaluación, ya que los parámetros cambian con el tiempo y produce un impacto en la evaluación final del producto.
- El proceso de evaluación de la calidad, incluye procesos como el de las pruebas de software, medición y análisis. Se ha determinado que con frecuencia presentan inconsistencias entre los diferentes términos en estas áreas del conocimiento. Por otra parte, si analizamos los estándares internacionales desarrollados por las principales organizaciones e instituciones de estandarización, como IEEE e ISO, la problemática se mantiene. Todo esto conlleva a la búsqueda de técnicas y herramientas que proporcionen un vocabulario común para resolver el problema de integridad e inconsistencia identificado y reducir la pérdida o desaprovechamiento del conocimiento. Dicha necesidad conduce al diseño de una ontología para la evaluación de la calidad de productos de software, actualmente en fase de desarrollo.
- Es necesario no solo evaluar sino también predecir la evaluación de la calidad de productos intermedios del proceso de desarrollo. A partir de los datos

almacenados de diversas evaluaciones, pudieran incorporarse técnicas o algoritmos para el aprendizaje automático al modelo propuesto. Los pesos de los mecanismos de agregación pueden modificarse en función del contexto, aprender los pesos de la función de agregación a partir del comportamiento histórico.

- Extender y potenciar la plataforma SoftQuality, en cuanto a:
    - la incorporación de otros métodos de agregación para comparar soluciones,
    - mejorar la amigabilidad de las interfaces, hasta convertirla en un sistema de ayuda a la decisión, fácilmente manejable por decisores y expertos en esta problemática, y
    - personalizar de acuerdo a los tipos de productos y usos, debido a que existen referencias genéricas de criterios para algunos de ellos.
- Analizar y experimentar si el modelo propuesto pudiera utilizarse en otros campos del conocimiento para la toma de decisiones, siempre y cuando existan limitaciones semejantes.

REFERENCIAS BIBLIOGRÁFICAS

[1] E. C. Decision, “Horizon 2020 Work Programme 2018-2020 5 . i . Information and Communication Technologies,” 2018.

[2] Oficina Nacional de Normalización (NC), “NC ISO/IEC 25000: 2011 Ingeniería de software — Requisitos de calidad y evaluación de productos software (SQuaRE) -Guía para SQuaRE.,” 2011.

[3] L. A. Zadeh, “Soft computing and fuzzy logic,” *IEEE Softw.*, vol. 11, no. 6, pp. 48–56, 1994.

[4] J. L. Verdegay, R. R. Yager, and P. P. Bonissone, “On heuristics as a fundamental constituent of soft computing,” *Fuzzy Sets Syst.*, vol. 159, no. 7, pp. 846–855, 2008.

[5] J. Kacprzyk and R. R. . Yager, “Linguistic summaries of data using fuzzy logic.,” *Int. J. Gen. Syst.*, vol. 30, no. (2), p. 133–154., 2001.

[6] E. Cables, M. S. García-cascales, and M. T. Lamata, “The LTOPSIS : An alternative to TOPSIS decision-making approach for linguistic variables,” *Expert Syst. Appl.*, vol. 39, no. 2, pp. 2119–2126, 2012.

[7] E. Cables, M. T. Lamata, and J. L. Verdegay, “RIM-reference ideal method in multicriteria decision making,” *Inf. Sci. (Ny).*, no. 337–338, pp. 1–10, 2016.

[8] V. Pasrija, S. Kumar, and P. R. Srivastava, “Assessment of Software Quality: Choquet Integral Approach,” *Procedia Technol.*, vol. 6, pp. 153–162, Jan. 2012.

[9] B. Gong and L. Chen, “The Comprehensive Evaluation of the Unascertained Measure for Software Quality,” *Phys. Procedia*, vol. 25, pp. 1621–1626, Jan. 2012.

[10] E. S. Benavides, C. G. Aros, and A. P. G. Henao, “Calidad y métricas de software: una propuesta desde la lógica difusa,” *PERFILES Lib.*, vol. 7, pp. 51–59, 2011.

[11] Real Academia Española, “Diccionario de la lengua española.,” *2018*. [Online]. Available: http://dle.rae.es/.

[12] R. Moen and C. Norman, “Evolution of the PDCA Cycle,” *Society*, pp. 1–11, 2009.

[13] E. E. Adam, L. M. Corbett, B. E. Flores, N. J. Harrison, T. S. Lee, B. Rho, and J. Ribera, “An international study of quality improvement approach and firm performance,” *Survey*, 2006.

[14] W. E. Deming, *Out of the Crisis. (Fuera de la Crisis)*, Press, MIT. 1986.

[15] C. J. Robinson and M. K. Malhotra, “Defining the concept of supply chain quality management and its relevance to academic and industrial practice,” *Int. J. Prod. Econ.*, vol. 96, no. 3, pp. 315–337, 2005.

[16] J. M. Juran and A. B. Godfrey, *Juran’s Quality Control Handbook*. 1998.

[17] K. Ishikawa, “Que Es El Control Total De La Calidad.” p. 261, 1997.

[18] P. Bozzelli and P. Lago, “A systematic literature review on green software

metrics," 2014.

[19] "Sitio oficial de la ISO." [Online]. Available: http://www.iso.org/iso/home.htm.

[20] "IEEE Publications & Standards." [Online]. Available: https://www.ieee.org/publications_standards/index.html. [Accessed: 20-Jun-2006].

[21] I. Sommerville, *Software Engineering. Eighth Edition, Addison- Wesley Publishers.* 2007.

[22] R. Pressman, *Software Engineering: A practitioner´s approach Fifth Edition. McGraw Hill.* 2001.

[23] I. STANDARD 610-1990, "IEEE Standard Computer Dictionary: Compilation of IEEE Standard Computer Glossaries," 1990.

[24] E. Diez, "Aseguramiento de la calidad en la construcción de sistemas basados en el conocimiento. Un enfoque práctico," Instituto Tecnológico Buenos Aires, 2003.

[25] M. Rodriguez and M. Piattini, "Experiencias en la Industria del Software : Certificación del Producto con ISO / IEC 25000," in *XVIII Congreso Iberoamericano en Ingeniería de Software CIbSE 2015*, 2015.

[26] M. B. Chrissis, M. Konrad, and S. Shrum, *CMMI ® Guía para la integración de procesos y la mejora de productos*, Segunda ed. 2009.

[27] J. A. McCall and J. P. Cavano, "A framework for the measurement of software quality," pp. 133–139, 1978.

[28] B. W. Boehm, J. R. Brown, H. Kaspar, M. Lipow, G. McLeod, and M. Merritt, "Characteristics of Software Quality." North Holland, 1978.

[29] R. B. Grady and D. L. Caswell, *Software Metrics: Establishing a Company-Wide Program*. Prentice-Hall, 1987.

[30] R. G. Dromey, "A Model for Software Product Quality," vol. 21, no. 2, 1995.

[31] ISO/IEC, "International Standard. ISO/IEC 9126-1 (2001), Part 1, 2, 3: Quality Model." Institute of Electrical and Electronics Engineering, p. 82, 2001.

[32] F. Scalone, "Estudio comparativo de los modelos y estandares de calidad del software," Universidad Tecnológica Nacional. Facultad Regional, Buenos Aires, 2006.

[33] B. W. Boehm, J. R. Brown, M. Lipow, and E. Group, "Quantitative evaluation of software quality," *Proc. 2nd Int'l. Conf. Softw. Eng. IEEE Comput. Soc.*, pp. 592–605, 1976.

[34] R. Grady and D. Caswell, *Software Metrics: Establishing a Company-Wide Program*. 1987.

[35] ISO/IEC-9126, "ISO/IEC-9126:1991. Information Technology – Software Product Evaluation – Quality Characteristics and Guideline for their Use," 1991.

[36] H. Solano and I. Torres, "Análisis de frameworks para el desarrollo de aplicaciones móviles en la plataforma Android," Universidad del Azuay, 2013.

[37] W. Perry, *Effective methods for EDI quality assurance*. Prentice-Hall, New Jersey, USA, 1987.

[38] A. Rawashdeh and B. Matalkah, “A New Software Quality Model for Evaluating COTS Components,” *J. Comput. Sci.*, vol. 2, no. 4, pp. 373–381, 2006.

[39] A. Álvarez, “Los Atributos de Calidad del Software,” *QA news*, no. 10, p. 37, 2012.

[40] L. Dávila Nicanor and P. Mejía Álvarez, “Evaluación de la Calidad de Software en Sistemas de Información en Internet.” México, 2012.

[41] H. Zulzalil, A. Ghani, M. Selamat, and R. Mahmod, “A case study to identify quality attributes relationships for web-based applications,” *Ijcsns*, vol. 8, no. 11, pp. 215–220, 2008.

[42] H. Zulzalil and A. Ghani, “Using fuzzy integral to evaluate the web-based applications,” *Malaysian Softw. Eng. Interes. Group(MSEIG)*, 2011.

[43] F. Radulovic, *A software quality model for the evaluation. Thesis of master in artificial intelligence research.* Universidad Politécnica de Madrid, Facultad de Informática, 2011.

[44] A. K. Pandey, “Analytical Network Process based Model to Estimate the Quality of Software Components,” pp. 678–682, 2014.

[45] A. H. Odeh, “SMSCQA : System for Measuring Source Code Quality Assurance,” *Int. J. Comput. Appl.*, vol. 60, no. 8, pp. 35–39, 2012.

[46] H. Al-Kilidar, K. Cox, and B. Kitchenham, “The use and usefulness of the ISO/IEC 9126 quality standard,” *2005 Int. Symp. Empir. Softw. Eng. 2005.*, pp. 126–132, 2005.

[47] D. Type, “ISO / IEC JTC1 / SC7 / N3617 Abstract , DIS 25020 - Software and system engineering – Software reference model and guide,” 2006.

[48] M. Morisio and A. Tsoukias, “IusWare: a methodology for the evaluation and selection of software products,” *Softw. Eng. IEE Proc.*, vol. 144, no. 3, pp. 162–174, 1997.

[49] I. Stamelos, I. Vlahavas, I. Refanidis, and A. Tsoukia, “Knowledge based evaluation of software systems : a case study,” *Inf. Softw. Technol.*, vol. 42, pp. 333–345, 2000.

[50] M. Azuma, “Software products evaluation system : quality models , metrics and processes- International Standards and Japanese Practice,” *Inf. Softw. Technol.*, vol. 38, pp. 145–154, 1996.

[51] A. Dasso and A. Funes, “Software Quality Metrics Aggregation,” *13th Argentine Symp. Softw. Eng. ASSE 2012 ISSN 1850-2792*, pp. 312–323, 2012.

[52] X. Liu and J. Pang, “A Fuzzy Synthetic Evaluation Method for Software Quality,” *2010 2nd Int. Conf. E-bus. Inf. Syst. Secur.*, no. m, pp. 1–4, May 2010.

[53] J. S. Challa, A. Paul, Y. Dada, and V. Nerella, “Integrated Software Quality Evaluation : A Fuzzy Multi-Criteria Approach,” *J. Inf. Process. Syst.*, vol. 7, no. 3, pp. 473–518, 2011.

[54] S. K. Dubey, A. Gulati, and A. Rana, “Usability Evaluation of Software Systems using Fuzzy Multi- Criteria Approach,” *IJCSI Int. J. Comput. Sci. Issues ISSN 1694-0814*, vol. 9, no. 3, pp. 404–409, 2012.

[55] J. S. Challa, A. Paul, Y. Dada, V. Nerella, and P. R. Srivastava, “Quantification of Software Quality Parameters Using Fuzzy Multi Criteria Approach,” *2011 Int. Conf. Process Autom. Control Comput.*, pp. 1–6, Jul. 2011.

[56] A. Singh and S. K. Dubey, “Evaluation of Usability Using Soft Computing Technique,” *Int. J. Sci. Eng. Res.*, vol. 4, no. 12, pp. 162–166, 2013.

[57] B. Furer, F. Ruggeri, S. M. Voci, C. A. Borges, and B. Slater, “Avaliação da usabilidade de um sistema computadorizado de epidemiologia nutricional,” *Bras Epidemiol*, vol. 16, no. 4, pp. 966–975, 2013.

[58] A. Nuraini and Y. Widyani, “Software with service oriented architecture quality assessment,” *Proc. 2014 Int. Conf. Data Softw. Eng. ICODSE 2014*, 2014.

[59] A. Koscianski and J. C. B. Costa, “Combining analytical hierarchical analysis with ISO/IEC 9126 for a complete quality evaluation framework,” *Proc. 4th IEEE Int. Softw. Eng. Stand. Symp. Forum (ISESS’99). “Best Softw. Pract. Internet Age,”* pp. 218–226, 1999.

[60] K. Lee, “A software product evaluation process for supporting the component based development,” *Third ACIS Int’l Conf. Softw. Eng. Res. Manag. Appl.*, pp. 56–63, 2005.

[61] O. Gi oug, K. Doo yeon, K. Sang il, and R. Sung yul, “A Quality Evaluation Technique of RFID Middleware in Ubiquitous Computing,” *2006 Int. Conf. Hybrid Inf. Technol.*, pp. 6–11, 2006.

[62] C.-W. Chang, C.-R. Wu, and H.-L. Lin, “Integrating fuzzy theory and hierarchy concepts to evaluate software quality,” *Softw. Qual. J.*, vol. 16, no. 2, pp. 263–276, Nov. 2007.

[63] H. Wan-jiang and L. Tian-bo, “Study On Quality Evaluation Model Of Communication System,” pp. 278–281, 2012.

[64] R. Lamichhane and P. Meesad, “A Usability Evaluation for Government Websites of Nepal Using Fuzzy AHP,” *7th Int. Conf. Comput. Inf. Technol. IC2IT2011*, pp. 99–104, 2011.

[65] S. Bhatnagar, S. K. Dubey, and A. Rana, “Quantifying Website Usability using Fuzzy Approach,” *Int. J. Soft Comput. Eng. ISSN 2231-2307*, vol. 2, no. 2, pp. 424–428, 2012.

[66] S. K. Dubey, A. Mittal, and A. Rana, “Measurement of Object Oriented Software Usability using Fuzzy AHP,” *Int. J. Comput. Sci. Telecommun.*, vol. 3, no. 5, pp. 98–104, 2012.

[67] M. L. Etaati, S. Sadi-Nezhad, A, “Using Fuzzy Analytical Network Process and ISO 9126 Quality Model in Software Selection: A case study in E-learnig Systems”,” *J. Appl. Sci.*, vol. 11, no. 1, pp. 96–103, 2011.

[68] Sugiyanto and S. Rochimah, “Integration of DEMATEL and ANP Methods for calculate The weight of characteristics software quality based Model ISO 9126,” in *International Conference on Information Technology and Electrical Engineering (ICITEE)*, 2013.

[69] M.-J. Blin and A. Tsoukiàs, “Multi-Criteria Methodology Contribution to the Software Quality Evaluation,” *Softw. Qual. Journal,* vol. 9, no. 1, pp. 113–132, 2001.

[70] Gh. A. Montazer and H. Q. Saremi, “An Application of Type-2 Fuzzy Notions in Website Structures Selection : Utilizing Extended TOPSIS Method,” *WSEAS Trans. Comput.*, vol. 7, no. 1, pp. 8–15, 2008.

[71] C. Wei, “Research on the Software Quality Evaluation with 2-tuple Linguistic Information,” *Int. J. Digit. Content Technol. its Appl.*, vol. 6, no. 9, pp. 329–335, May 2012.

[72] Q. Li, X. Zhao, R. Lin, and B. Chen, “Relative entropy method for fuzzy multiple attribute decision making and its application to software quality evaluation,” *J. Intell. Fuzzy Syst.*, vol. 26, no. 4, pp. 1687–1693, 2014.

[73] Q. Li, X. Zhao, and G. Wei, “Model for software quality evaluation with hesitant fuzzy uncertain linguistic information,” *J. Intell. Fuzzy Syst.*, vol. 26, no. 6, pp. 2639–2647, 2014.

[74] X. Huang, “Study on the Software Quality Evaluation with Hesitant Fuzzy Information,” *Int. J. Digit. Content Technol. its Appl.*, vol. 6, no. September, pp. 317–322, 2012.

[75] S. Xiao, “Induced interval-valued intuitionistic fuzzy Hamacher ordered weighted geometric operator and their application to multiple attribute decision making,” *J. Intell. Fuzzy Syst.*, vol. 27, no. 1, pp. 527–534, 2014.

[76] T. L. Saaty, *The Analytic Hierarchy Process: Planning, Priority Setting, Resource Allocation*. New York: McGraw-Hill International Book Company, 1980.

[77] C. L. . Hwang and K. Yoon, *Multiple Attribute Decision Making: Methods and Applications.* New York: Springer-Verlag, 1981.

[78] B. Roy, “Classement et choix en présence de points de vue multiples: La méthode ELECTRE,” *Rev. française d´automatique, D’informatique Rech. opérationnelle.*, vol. 8, pp. 57–75, 1968.

[79] T. Miyoshi and M. Azuma, “An Empirical Study of Evaluating Software development environment quality,” *IEEE Trans. Softw. Eng.*, vol. 19, no. 5, pp. 425–435, 1993.

[80] K. Lee and S. J. Lee, “A Quantitative Evaluation Model Using the ISO / IEC 9126 Quality Model in the Component Based Development Process,” *Comput. Sci. Its Appl. - ICCSA 2006 LNCS 3983*, pp. 917–926, 2006.

[81] R. Takahashi, Y. Muraoka, and Y. Nakamura, “Building Software Quality Classification Trees : Approach, Experimentation, Evaluation,” *Proc. Eighth Int. Symp. Softw. Reliab. Eng.*, pp. 222–233, 1997.

[82] R. Kumar, S. Rai, and J. L. Trahan, “Neural-network techniques for software-quality evaluation,” *Annu. Reliab. Maintainab. Symp. Proceedings. Int. Symp. Prod. Qual. Integr.*, pp. 155–161, 1998.

[83] W. Pedrycz, L. Han, J. F. Peters, S. Ramanna, and R. Zhai, “Calibration of software quality: Fuzzy neural and rough neural computing approaches,” *Neurocomputing*, vol. 36, no. 1–4, pp. 149–170, Feb. 2001.

[84] T. M. Khoshgoftaar and N. Seliya, “Unsupervised learning for expert-based software quality estimation,” *Eighth IEEE Int. Symp. High Assur. Syst. Eng. Proceedings.*, pp. 149–155, 2004.

[85] A. D. Belchior and R. P. S. Simao, “Quality Characteristics for Software Components : Hierarchy and Quality Guides,” *LNCS 2693*, vol. 2693, pp. 184–206, 2003.

[86] T. Rosqvist, M. Koskela, and H. Harju, “Software Quality Evaluation Based on Expert Judgement,” *Softw. Qual. J.*, vol. 11, no. 1, pp. 39–55, 2003.

[87] W. Li, “An approach to software quality assessment with interval intuitionistic trapezoidal fuzzy information,” *Advances in Information Sciences and Service Sciences*, vol. 4, no. 7. pp. 240–247, 2012.

[88] H. Yang, “Measuring Software Product Quality with ISO Standards Base on Fuzzy Logic Technique,” pp. 59–67, 2012.

[89] D. Nabil, A. Mosad, and H. A. Hefny, “Web-Based Applications quality factors : A survey and a proposed conceptual model,” *Egypt. Informatics J.*, vol. 12, no. 3, pp. 211–217, 2011.

[90] H. Sohn, M. Lee, B. Seong, and J. Kim, “Quality Evaluation Criteria Based on Open Source Mobile HTML5 UI Framework for Development of Cross-Platform,” vol. 9, no. 6, pp. 1–12, 2015.

[91] I. Samoladas, G. Gousios, and I. Spinellis, D. and Stamelos, “The SQO-OSS Quality Model : Measurement Based Open Source Software Evaluation,” *IFIP Int. Fed. Inf. Process.*, vol. 275, no. Open Source Development, Communities and Quality, pp. 237–248, 2008.

[92] T. E. Vollman, “Software quality assessment and standards,” *Computer (Long. Beach. Calif).*, vol. 26, no. 6, pp. 118–120, 1993.

[93] K. Lee and S. J. Lee, “A Quantitative Software Quality Evaluation Model for the Artifacts of Component Based Development,” in *Proceedings of the Sixth International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing and First ACIS International Workshop on Self-Assembling Wireless Networks*, 2005.

[94] M. F. Bertoa, J. M. Troya, and A. Vallecillo, “Measuring the usability of software components,” *J. Syst. Softw.*, vol. 79, no. 3, pp. 427–439, Mar. 2006.

[95] I. Biscoglio and E. Marchetti, “An experiment of software quality evaluation in the audio-visual media preservation context,” *Proc. - 2014 9th Int. Conf. Qual. Inf. Commun. Technol. QUATIC 2014*, pp. 118–123, 2014.

[96] V. Bogado, S. Gonnet, and H. Leone, “Modeling and simulation of software architecture in discrete event system specification for quality evaluation,” *Simulation*, vol. 90, no. 3, pp. 290–319, 2014.

[97] M. Ortega and T. Rojas, “Construction of a Systemic Quality Model for,” pp. 219–242, 2003.

[98] M.-A. Côté, W. Suryn, C. Y. Laporte, and R. a. Martin, “The Evolution Path for Industrial Software Quality Evaluation Methods Applying ISO/IEC 9126:2001 Quality Model: Example of MITRE?s SQAE Method,” *Softw. Qual. J.*, vol. 13, no. 1, pp. 17–30, Mar. 2005.

[99] J. Dong, N. Shi, and W. Hu, “A New Approach of Software Quality Metrics: Fuzzy Extension Algorithm and Its Application,” *J. Softw.*, vol. 6, no. 10, pp. 1881–1889, Oct. 2011.

[100] D. L. Moody, “Theoretical and practical issues in evaluating the quality of conceptual models : current state and future directions,” *Data Knowl. Eng.*, vol. 55, pp. 243–276, 2005.

[101] S. Ahn, S. M. Huff, Y. Kim, and D. Kalra, “Quality metrics for detailed clinical models,” *Int. J. Med. Inform.*, vol. 82, no. 5, pp. 408–417, 2013.

[102] W. Han, X. Zhang, H. Jiang, and W. Li, “Study on Project Quality Assessment,” *2014 7th Int. Conf. Control Autom.*, pp. 56–59, 2014.

[103] I. Biscoglio and E. Marchetti, “Software Technologies,” vol. 555, pp. 63–80, 2015.

[104] L. Mich, M. Franch, and L. Gaio, “Evaluating and Designing Web Site Quality,” *MultiMedia, IEEE*, vol. 10, no. 1, pp. 34–43, 2003.

[105] B. D. Harper and K. L. Norman, “Improving user satisfaction: the questionnaire for user interaction satisfaction version 5.5,” *Proc. 1st Annu. Mid-Atlantic Hum. Factors Conf.*, p. 224–228., 1993.

[106] N. Bevan and M. Macleod, “Usability measurement in context,” *Behav. Inf. Technol.*, vol. 13, no. 1–2, pp. 132–145, Jan. 1994.

[107] Z. Mansor, Z. M. Kasirun, S. Yahya, and N. H. Arshad, “The Evaluation of WebCost Using Software Usability Measurement Inventory (SUMI),” *Int. J. Digit. Inf. Wirel. Commun.*, vol. 2, no. 2, pp. 197–201, 2012.

[108] V. Kumar, A. Sharma, R. Kumar, and P. S. Grover, “Quality aspects for component-based systems : A metrics based approach,” *Softw. – Pract. Exp.*, vol. 42, no. January, pp. 1531–1548, 2012.

[109] E. J. Chang, T. S. Dillon, and D. Cook, “An Intelligent System Based Usability Evaluation Metric,” *Intell. Inf. Syst. IASTED Int. Conf. on. IEEE Comput. Soc.*, pp. 218–226, 1997.

[110] A. R. Gray and S. G. MacDonell, “Fuzzy Logic for Software Metric Models Throughout the Development Life-Cycle,” *Inf. Sci. Discuss. Pap. Ser.*, vol. 99, no. 20, 1999.

[111] W. Siu Keung, T. Thi Nguyen, E. Chang, and N. Jayaratna, “Usability Metrics for E-learning,” *Move to Meaningful Internet Syst. 2003 OTM 2003 Work. Springer Berlin Heidelb.*, pp. 235–252, 2003.

[112] E. Chang and T. S. Dillon, “A usability-evaluation metric based on a soft-computing approach,” *IEEE Trans. Syst. Man, Cybern. - Part A Syst. Humans*, vol. 36, no. 2, pp. 356–372, Mar. 2006.

[113] M. Hub and M. Zatloukal, “Model of Usability Evaluation of Web Portals Based on the Fuzzy Logic,” *WSEAS Trans. Inf. Sci. Appl.*, vol. 7, no. 4, pp. 522–531, 2010.

[114] P. Jain, S. K. Dubey, and A. Rana, “Software usability evaluation method,” *Int. J. Adv. Res. Comput. Eng. Technol.*, vol. 1, no. 2, pp. 28–33, 2012.

[115] H. A. Al-Jamimi and M. Ahmed, “Prediction of Software Maintainability Using Fuzzy Logic,” *Softw. Eng. Serv. Sci. (ICSESS), IEEE 3rd Int. Conf.*, pp. 702–705, 2012.

[116] R. Nagpal, D. Mehrotra, A. Sharma, and P. Bhatia, “ANFIS Method for Usability Assessment of Website of an Educational Institute,” *World Appl. Sci. J.*, vol. 23, no. 11, pp. 1489–1498, 2013.

[117] V. Sharma and A. Sharma, “Software Usability Assessment Models and Metrics : A Survey,” *Int. KIET J. Softw. Commun. Technol.*, vol. 1, no. 1, pp. 7–15, 2013.

[118] L. Kumar and S. K. Rath, “Hybrid functional link artificial neural network approach for predicting maintainability of object-oriented software,” *J. Syst. Softw.*, vol. 0, pp. 1–21, 2016.

[119] M.-ángel Sicilia, E. G. Barriocanal, and T. Calvo, “An inquiry-based method for Choquet integral-based aggregation of interface usability parameters,” *Kybernetika*, vol. 39, no. 5, pp. 601–614, 2003.

[120] M. Sicilia and E. García, “Modelling Interacting Web Usability Criteria through Fuzzy Measures,” *Web Eng. Springer Berlin Heidelb.*, pp. 182–185, 2003.

[121] X. Wang, A. F. G. Contreras, M. Ceberio, C. Del Hoyo, L. C. Gutierrez, and S. Virane, “Interval-based algorithms to extract fuzzy measures for software quality assessment,” *2012 Annu. Meet. North Am. Fuzzy Inf. Process. Soc. NAFIPS 2012*, 2012.

[122] C. Ruan and J. Yang, “Software quality evaluation model based on weighted mutation rate correction incompletion G1 combination weights,” *Math. Probl. Eng.*, vol. 2014, 2014.

[123] L. Zadeh, “Fuzzy Sets,” *Information and Control*, vol. 8. pp. 338–353, 1965.

[124] H. J. Zimmermann, “Fuzzy set theory,” *Wiley Interdiscip. Rev. Comput. Stat.*, vol. 2, no. 3, pp. 317–332, 2010.

[125] H. J. Zimmermann, *Fuzzy Set Theory and Its Applications. 2ed*. 1992.

[126] B. Klir, George J., Yuan, *Fuzzy Sets and Fuzzy Logic: Theory and Applications*. 1995.

[127] P. J. Villacorta, A. D. Masegosa, D. Castellanos, and M. T. Lamata, “A new fuzzy linguistic approach to qualitative Cross Impact Analysis,” *Appl. Soft Comput.*, vol. 24, no. 0, pp. 19–30, 2014.

[128] L. A. Zadeh, “The Concept of a Linguistic Variable and its Application to Approximate Reasoning-II*,” *Inf. Sci. (Ny).*, vol. 8, pp. 301–357, 1975.

[129] R. M. Tong and P. P. Bonissone, “A Linguistic Approach to Decision making with Fuzzy Sets,” *Syst. Man Cybern. IEEE Trans.*, vol. 10, no. 11, pp. 716–723, 1980.

[130] P. P. Bonissone, “a Fuzzy Sets Based Linguistic Approach: Theory and Applications,” no. Zadeh 1978, 1980.

[131] B. Kosko, “Fuzzy Cognitive Maps,” *Int. J. Man -Machine Stud.*, vol. 24, no. April 1985, pp. 65–75, 1986.

[132] B. Kosko, “Hidden patterns in combined and adaptive knowledge networks,” *Int.*

J. Approx. Reason., vol. 2, no. 4, pp. 377–393, 1988.

[133] E. I. Papageorgiou and C. D. Stylios, “Fuzzy Cognitive Maps,” *Handb. Granul. Comput.*, pp. 755–774, 2008.

[134] E. I. Papageorgiou and J. L. Salmeron, “A review of fuzzy cognitive maps research during the last decade,” *IEEE Trans. Fuzzy Syst.*, vol. 21, no. 1, pp. 66–79, 2013.

[135] P. P. Groumpos, “Fuzzy Cognitive Maps: Basic theories and their application to complex systems,” *Fuzzy Cogn. Maps*, vol. 247, pp. 1–22, 2010.

[136] W. Stach, L. Kurgan, and W. Pedrycz, “A divide and conquer method for learning large fuzzy cognitive maps,” *Fuzzy Sets Syst.*, vol. 161, no. 19, pp. 2515–2532, 2010.

[137] C. Enriquepelaez, “Using fuzzy cognitive maps as a system model for failure modes and effects analysis,” *Inf. Sci. (Ny).*, vol. 88, no. 1–4, pp. 177–199, 1996.

[138] M. S. Khan and M. Quaddus, “Group decision support using fuzzy cognitive maps for causal reasoning,” *Gr. Decis. Negot.*, vol. 13, no. 5, pp. 463–480, 2004.

[139] A. Mourhir, T. Rachidi, E. I. Papageorgiou, M. Karim, and F. S. Alaoui, “A cognitive map framework to support integrated environmental assessment,” *Environ. Model. Softw.*, vol. 77, pp. 81–94, 2016.

[140] A. Baykasoğlu and İ. Gölcük, “Development of a novel multiple-attribute decision making model via fuzzy cognitive maps and hierarchical fuzzy TOPSIS,” *Inf. Sci. (Ny).*, vol. 301, pp. 75–98, 2015.

[141] Z. Kiszová and J. Mazurek, “Modeling dependence and feedback in ANP with fuzzy cognitive maps,” *Proc. 30th Int. Conf. Math. Methods Econ. Model.*, pp. 558–563, 2012.

[142] M. Fan, Y. Lou, G. Wu, and X. Fu, “An improved analytic hierarchy process model on Software Quality Evaluation,” *2nd Int. Conf. Inf. Sci. Eng.*, pp. 1838–1842, Dec. 2010.

[143] H. J. Shyur, “COTS evaluation using modified TOPSIS and ANP,” *Appl. Math. Comput.*, vol. 177, no. 1, pp. 251–259, 2006.

[144] L. Stergioulas, M. Abassi, G. Xydopoulos, M. Fakhimi, R. Margineanu, L. A. Rifon, and M. J. F. Iglesias, “Evaluating E-learning platforms for schools: Use and usability, user acceptance, and impact on learning,” *Proc. - IEEE 14th Int. Conf. Adv. Learn. Technol. ICALT 2014*, pp. 19–21, 2014.

[145] A. Oztekin, Z. J. Kong, and O. Uysal, “UseLearn: A novel checklist and usability evaluation method for eLearning systems by criticality metric analysis,” *Int. J. Ind. Ergon.*, vol. 40, no. 4, pp. 455–469, Jul. 2010.

[146] ISO/IEC, “ISO/IEC 25010:2011 Systems and software engineering -- Systems and software Quality Requirements and Evaluation (SQuaRE) -- System and software quality models.” 2011.

[147] E. H. Cables Pérez and M. T. Lamata Jiménez, “Tesis Doctoral: Selección de personal con técnicas de Soft Computing. Propuesta de Desarrollo y de Software.,” Universidad de Granada, 2011.

[148] R. Yu and G. H. Tzeng, “A soft computing method for multi-criteria decision making with dependence and feedback,” *Appl. Math. Comput.*, vol. 180, no. 1, pp. 63–75, 2006.

[149] T. S. Community, “NumPy v1.14 Manual,” *2018*. [Online]. Available: https://docs.scipy.org/doc/numpy/.

[150] S. Van Der Walt, S. C. Colbert, G. Varoquaux, S. Van Der Walt, S. C. Colbert, G. Varoquaux, and T. Numpy, “The NumPy array : a structure for efficient numerical computation To cite this version : HAL Id : inria-00564007 The NumPy array : a structure for efficient numerical computation,” 2011.

[151] S. Developers, “SciPy.org Official web site,” 2018. [Online]. Available: https://www.scipy.org/.

[152] F. Pedregosa, R. Weiss, and M. Brucher, “Scikit-learn : Machine Learning in Python,” vol. 12, pp. 2825–2830, 2011.

[153] J. Bautista and C. National, “Scikit-Criteria : Colección de métodos de análisis multi-criterio integrado al stack científico de Python,” 2017.

[154] M. Vuorre and J. P. Curley, “Curating Research Assets : A Tutorial on the Git Version Control System,” pp. 1–23, 2016.

[155] A. Holovaty and J. Kaplan-Moss, *La guía definitiva de Django. Desarrolla aplicaciones web de forma rápida y sencilla*. 2015.

[156] Mesa Redonda, “Inquietudes y respuestas sobre la producción de software en Cuba,” 2017. [Online]. Available: http://mesaredonda.cubadebate.cu/mesa-redonda/2017/05/05/inquietudes-y-respuestas-sobre-la-produccion-de-software-en-cuba/.

[157] CALISOFT 2016, “Sitio oficial CALISOFT,” 2016. [Online]. Available: www.calisoft.cu.

[158] K. Pérez-Teruel, M. Leyva-vázquez, and V. Estrada Sentí, “Mental Models Consensus Process Using Fuzzy Cognitive Maps and Computing,” *Ing. Univ.*, vol. 19, no. 1, pp. 173–188, 2015.

PUBLICACIONES OBTENIDAS CON LA REALIZACIÓN DE LA TESIS

Revistas:

- **Y. Fernández Pérez,** C. Cruz Corona, J. L. Verdegay**.** "A New Model Based on Soft Computing for Evaluation and Selection of Software Products." en *IEEE Latin America Transactions*. 2018. (Aceptado)
- **Y. Fernández Pérez,** C. A. Cruz Corona, J. L. Verdegay. "Modeling the interrelation among software quality criteria using Computational Intelligence techniques". *International Journal of Computational Intelligence*. (Aceptado).

Capítulos de libros:

- **Y. Fernández Pérez,** A. Estrada Sentí, J. L. Verdegay, C. A. Cruz Corona. "Análisis de los modelos para la evaluación de la calidad de los productos de software: tendencias actuales y futuras" En Pelta et al. (Eds.) Contribuciones en Sof Computing, 156-175, ISBN: 978-84-338-5672-2. Editorial Universidad de Granada. 2014.
- **Y. Fernández Pérez,** A. Estrada Sentí, C. Cruz Corona, J. L. Verdegay. "Fuzzy Multicriteria Decision Making Methods Applied to Usability Software Assessment: An Annotated Bibliography". En Berger-Vachon et al. (Eds.) Complex Systems: Solutions and Challenges in Economics, Management and Engineering. pp 165-189**.** Springer 2017**.** https://link.springer.com/chapter/10.1007/978-3-319-69989-9_11..

Congresos:

- **Y. Fernández Pérez,** C. A. Cruz Corona, J. L. Verdegay. **"**Método multicriterio basado en Soft Computing para evaluar la calidad de software". En el XVIII Congreso Español sobre Tecnologías y Lógica Fuzzy - ESTYLF 2016.
- **Y. Fernández Pérez,** A. Estrada Sentí, J. L. Verdegay, C. A. Cruz Corona**. "**Un nuevo método para evaluar la calidad de productos de software basado en métodos de decisión multicriterio." En la II Conferencia Científica Internacional UCIENCIA 2016. 2016
- **Y. Fernández Pérez,** A. Estrada Sentí, C. Cruz Corona, J. L. Verdegay. **"**Fuzzy Cognitive Map for modeling the interrelationship between software quality criteria" En "2nd International Symposium on Fuzzy and Rough Sets" (ISFUROS '17)
- A Vázquez Abascal, **Y. Fernández Pérez,** A. Estrada Sentí, C. Cruz Corona, J. L. Verdegay**. "**Método para la evaluación de las características de calidad según la ISO/IEC 25000". En el VII Taller de Atención Ciudadana Comunicación Institucional, protección al consumidor y calidad del Ministerio de Comunicaciones, CALIPROT 2017. (Trabajo galardonado con Mención)
- A. Castañeda Martínez, **Y. Fernández Pérez,** A. Estrada Sentí, C. Cruz Corona, J. L. Verdegay**.** "Diseño de una ontología para la evaluación de productos de software en la UCI", En el VII Taller de Atención Ciudadana Comunicación Institucional, protección al consumidor y calidad del Ministerio de Comunicaciones, CALIPROT 2017. (Trabajo galardonado con Mención).

ANEXOS

Anexo I. Algunos términos de la calidad según la ISO 9000. (ISO)

Términos	Descripción
Producto	Es el resultado de un proceso.
Proceso	Conjunto de actividades mutuamente relacionadas o que interactúan, las cuales transforman elementos de entrada en resultados.
Requisito	Necesidad o expectativa establecida, generalmente implícita u obligatoria.
Calidad	Grado en el que un conjunto de características inherentes cumple con los requisitos.
Política de la calidad	Intenciones globales y orientación de una organización, relativas a la calidad, tal como se expresan formalmente por la alta dirección. Es la base para la implementación de un sistema de gestión de la calidad (SGC).
Sistema de gestión de la calidad	Sistema para establecer la política y objetivos, y para lograr dichos objetivos
Gestión de la calidad	Actividades coordinadas para dirigir y controlar la calidad de una organización. La dirección y control, en lo relativo a la calidad, generalmente incluye el establecimiento de la política, objetivos, planificación, control, aseguramiento y mejora de la calidad.
Sistema de gestión de la calidad	Sistema de gestión para dirigir y controlar la calidad en una organización.
Planificación de la calidad	Parte de la gestión enfocada al establecimiento de los objetivos y a la especificación de los procesos operativos necesarios y recursos relacionados para cumplir los objetivos de la calidad. El establecimiento de planes es parte de la planificación de la calidad.
Control de la calidad	Parte de la gestión de la calidad orientada al cumplimiento de los requisitos de esta
Aseguramiento de la calidad	Parte de la gestión orientada a proporcionar confianza en que se cumplirán los requisitos de la calidad
Mejora de la calidad	Parte de la gestión orientada a aumentar la capacidad de cumplimiento de los requisitos de calidad

Anexo II. Algunos estándares de la IEEE relacionados a calidad

Estándar	Breve explicación
730-2014 - IEEE Standard for Software Quality Assurance Processes	Establece los requisitos para iniciar, planificar, controlar y ejecutar los procesos de aseguramiento de la calidad del software. Se relaciona con la norma ISO/IEC/IEEE 12207: 2008, que explica el proceso del ciclo de vida del software, y con la ISO/IEC/IEEE 15289: 2015, que expresa los requisitos de la documentación a elaborar durante el desarrollo de software.
983-1986 - IEEE Guide for Software Quality Assurance Planning	Guía cuyo propósito es explicar el contenido de cada sección del Plan de Aseguramiento de la Calidad, que satisfaga los requisitos de la norma IEEE Std 730.
1061-1998 - IEEE Standard for a Software Quality Metrics Methodology.	Define una metodología para el establecimiento de requisitos de calidad, además para la definición, ejecución, análisis y validación de los indicadores de calidad de proceso y producto de software. La metodología, abarca todo el ciclo de vida del software. Sufrió una revisión, en 2009.
1298-1992 - IEEE Standard Software Quality Management System Part 1: Requirements	Establece los requisitos para el sistema de gestión de la calidad de software. Identifica cada uno de los elementos de un sistema de gestión de calidad con el objetivo de garantizar que el software cumpla con los requisitos de un contrato, orden de compra u otro acuerdo.
1012-2012 - IEEE Standard for System and Software Verification and Validation	Detalla el proceso de Verificación y validación (V&V) y utiliza para determinar si los productos desarrollados en una determinada actividad, se ajustan a los requisitos y cumplen las necesidades de uso previstos por los usuarios. Se aplica a los sistemas, software y hardware en desarrollo, mantenimiento o reutilización. Los procesos de V&V incluyen análisis, revisión, inspección, evaluación y pruebas de productos.
1465-1998 - IEEE Standard - Adoption of International Standard ISO/IEC 12119:1994(E) - Information Technology - Software Packages - Quality Requirements and Testing	Establece los requisitos de calidad para los paquetes de software e instruye cómo probarlos. Sufrió una revisión, en 2004.
1062-2015 - IEEE Recommended Practice for Software Acquisition	Describe un conjunto de consideraciones útiles que pueden ser seleccionadas y aplicadas durante una o más etapas del proceso de adquisición del software. Las prácticas recomendadas, pueden aplicarse al software ejecutado en cualquier sistema informático, independientemente del tamaño, complejidad y criticidad del mismo.

Algunos estándares de la IEEE relacionados a calidad *(continuación)*

Estándar	Breve explicación
90003-2015 - IEEE Standard Adoption of ISO/IEC 90003:2014, Software Engineering -- Guidelines for the Application of ISO 9001:2008 to Computer Software	Proporciona orientación para las organizaciones en la aplicación de la ISO 9003 para la adquisición, suministro, desarrollo, operación y mantenimiento de software y servicios de apoyo relacionados. No añade ni cambia los requisitos de la ISO 9001: 2008.
12207-2008 - Systems and software engineering -- Software life cycle processes	Establece un marco común para los procesos del ciclo de vida del software, con una terminología bien definida, que puede ser referenciada por la industria. En esta revisión se integra la norma ISO/IEC 12207: 1995, con sus dos enmiendas y coordinó con la revisión paralela de ISO/IEC 15288: 2002 (procesos del ciclo de vida del sistema).
1045-1992 - IEEE Standard for Software Productivity Metrics, 2002	Define una manera consistente de medir los elementos que intervienen en la productividad del software. Las terminologías métricas de productividad, se dan para asegurar la comprensión de los datos de medición, tanto para el código fuente como para la documentación. Aunque prescribe mediciones para caracterizar el proceso de software, no establece normas de productividad, ni recomienda medidas como un método para evaluar los proyectos o desarrolladores de software. No mide la calidad del software, ni pretende mejorar la productividad, sino medirla. Su objetivo es lograr una mejor comprensión del proceso de software, que puede dar ideas para mejorarlo.
829-2008 - IEEE Standard for Software and System Test Documentation	Guía para la documentación del proceso de prueba y determinar si los productos de desarrollo de una determinada actividad se ajustan a los requerimientos y satisfacen las necesidades de uso previstas por los usuarios.
1028-2008 IEEE Standard for Software Reviews and Audits	Define cinco tipos de revisiones del software y auditorías, así como los procedimientos necesarios para la ejecución de cada tipo. Se refiere únicamente a las revisiones y auditorías; los procedimientos para determinarlas no están definidos, ni la disposición de los resultados. Los tipos incluidos son revisiones de la dirección, revisiones técnicas, inspecciones, tutoriales y auditorías.

Anexo III. Medidas de calidad de software

Medidas para evaluar la Adecuación Funcional.

Nombre	Función	Elemento de medida de Calidad (EMC)	Intervalo	Óptima
Completitud funcional				
Cobertura de aplicación funcional *(CAF)*	$X = 1 - \left(\frac{A}{B}\right)$	A: Número de funciones que no se ejecutan B: Número total de funciones	0,1	1
Idoneidad funcional *(IF)*	$X = \frac{A}{B}$	A: No conformidad funcional, B: Número de Casos de Prueba	0,m	0
Corrección funcional.				
Adecuación de escenarios con datos no válidos *(AEDNV)*	$X = \frac{A}{B}$	A: No conformidad validación, B: Cantidad de REQ	0, m	0
Corrección computacional *(CC)*	$X = 1 - \left(\frac{A}{B}\right)$	A: Funciones que no devuelven valores exactos y precisos B: Cantidad de funciones que necesitan corrección	0,1	1
Pertinencia funcional.				
Estabilidad de la especificación funcional *(EEF)*	$X = 1 - \left(\frac{A}{B}\right)$	A: Número de funciones que cambian B: Número total de funciones	0, 1	1

Medidas para evaluar la Eficiencia.

Nombre	Función	EMC	Intervalo	Óptima
Comportamiento temporal.				
Tiempo de respuesta media *(TRM)*	$X = \frac{1}{N}\sum_{i=1}^{N}\frac{(TF_i - TI_i)}{TM_i}$	TI_i Tiempo de comenzar la tarea i TF_i Tiempo en que se completa la tarea i TM_i Tiempo máximo estimado por la duración de la tarea i N Cantidad de tareas ejecutadas	0,1	0
Tareas por tiempo *(TpT)*	$X = \frac{A}{B}$	A: Número de tareas realizadas B: Tiempo observado	0, m	m
Uso de recursos				
Uso de CPU *(CPU)*	$X = \frac{A}{B}$	A: Tiempo de uso del CPU B: Tiempo máximo de uso del CPU	0, m	0
Uso de memoria *(MEM)*	$X = \frac{A}{B}$	A: Espacio de memoria usado B:Espacio máximo esperado de memoria	0, m	0
Uso E/S (*UES*)	$X = \frac{A}{B}$	A: Tiempo de respuesta de ES B: Tiempo de respuesta máximo de ES	0, m	0
Espacio en disco (*SHD*)	$X = \frac{A}{B}$	A: Espacio de disco usado B: Espacio máximo de disco a utilizar	0, m	0
Capacidad				
Accesos simultáneos (*AS*)	$X = \frac{\left(\frac{A}{C}\right)}{B}$	A: Número de accesos simultáneos C: Tiempo de operación B: Número máximo de accesos especificados	0,1	1
Solicitudes on line (*SS)*	$X = \frac{\left(\frac{A}{C}\right)}{B}$	A: Número de solicitudes procesadas C: Tiempo de operación B: Número máximo de solicitudes especificado	0, 1	1

Medidas para evaluar la Fiabilidad.

Nombre	Función	EMC	Intervalo	Óptima
Madurez				
Tiempo medio entre fallos *(MTBF)*	$X = \frac{A}{B}$	A: Tiempo operación B: Números de fallos ocurridos	0, m	m
Densidad de fracaso frente Caso de pruebas (*DF*)	$X = \frac{1}{N}\sum_{i=1}^{N}\frac{A_i}{B_i}$	A_i Número de fallos detectados por caso de prueba (CP) i B Número máximo de fallos detectables por CP i N Cantidad de CP ejecutados	0,1	0
Cobertura de las pruebas (*CP*)	$X = \frac{A}{B}$	A: Casos de prueba realizados B: Casos de prueba totales	0,1	1
Disponibilidad				
Proporción del tiempo de servicio (*PTS*)	$X = \frac{A}{B}$	A: Tiempo de servicio real B: Tiempo de servicio regulado	0, 1	1
Tiempo de inactividad (Tina)	$X = \frac{A}{B}$	A Tiempo total en que el sistema está caído B Número de interrupciones observadas	0, m	0
Tolerancia a fallos				
Evitar el fracaso (*EF*)	$X = \frac{A}{B}$	A Total de fallos evitados B Número de Casos de prueba con patrón de falla	0,m	m
Tiempo notificación de un fallo(*TpoNF*)	$X = \frac{(A-B)}{C}$	A Tiempo notifica el fallo B Tiempo que detecta el fallo, C Tiempo límite para notificar el fallo	0, m	0
Capacidad de recuperación				
Tiempo medio de recuperación (*TRec*)	$X = \frac{T}{C}$	T: Tiempo de recuperación C: Cantidad de veces que cae en recuperación	0, m	0

Medidas para evaluar la Portabilidad.

Nombre	Función	EMC	Intervalo	Óptima
Capacidad para ser instalado				
Eficiencia de instalación *(Efins*)	$X = \frac{T}{C}$	X=Tiempo de instalación C: Números de intento fallidos	0, m	0
Facilidad de instalación (*Fins*)	Etiquetas	Muy Bajo, Bajo, Medio, Alto, Muy Alto		Fácil
Adaptabilidad				
Adaptabilidad de Hardware (*AHw*)	$X = 1 - \frac{A}{B}$	A: Número de funciones que no se ejecutaron B: Número de funciones probadas en diferentes hardware	0,1	1
Adaptabilidad de Software (*ASw*)	$X = 1 - \frac{A}{B}$	A: Número de funciones que no se ejecutaron B: Número de funciones probadas en diferentes software	0,1	1
Capacidad para ser reemplazado				
Inclusividad funcional (*IFun*)	$X = \frac{A}{B}$	A: Número de funciones que son similares a la existentes y no cambian B: Número de funciones probadas que son similares a las existentes	0,1	1
Consistencia funcional (*CFun*)	$X = 1 - \frac{A}{B}$	A: Números de funciones nuevas que no se consideran consistentes B: Total de funciones nuevas	0,1	1

Medidas para evaluar la Usabilidad

Nombre	Función	EMC	Intervalo	Óptima
Capacidad para reconocer su adecuación				
Descripción exhaustividad (*DE*)	$X = \frac{A}{B}$	A Número de funciones bien descritas B Total de funciones	0,1	1
Comprensión de E/S (*CES*)	$X = \frac{A}{B}$	A: número de elementos de E/S que el usuario comprende en corto tiempo B número de elementos E/S que el usuario intenta comprender	0,1	1
Capacidad de aprendizaje				
Integridad de la documentación (IntDoc)	$X = \frac{A}{B}$	A: Número de funciones descrita correctamente B: Número total de funciones	0,1	1
Facilidad de acceso a la ayuda (Fhelp)	$X = \frac{A}{B}$	A Número de tareas que acceden a la ayuda B Número de tareas que necesitan ayuda	0,1	1
Capacidad para operar				
Consistencia operacional (ConO)	$X = 1 - \left(\frac{A}{B}\right)$	A Número de operaciones que se comportan inconsistente B Total de operaciones que tienen comportamiento similar	0,1	1
Satisfacción del software (Satisf)	Grado de conformidad	Muy Bajo, Bajo, Medio, Alto, Muy Alto		
Protección de errores de usuarios				
Evitar la operación incorrecta	$X = \frac{A}{B}$	A Patrones implementados B Patrones que se puede implementar	0,1	1
Comprobar validez de entrada	$X = \frac{A}{B}$	A Cantidad de datos validados B Cantidad de datos de entrada	0,1	1
Estética				
Cumplimiento estándar de interfaz por experto	Cumple con el estándar de interfaz	Muy Bajo, Bajo, Medio, Alto, Muy Alto	0,1	1
Valoración usuario	Apariencia del software	No Agradable, Poco Agradable, Agradable, Muy Agradable, Altamente Agradable		

Anexo IV. Código fuente de la plataforma SoftQuality.

El código fuente de SoftQuality se encuentra en el CD adjunto y también se puede descargar en http://modo.ugr.es/SoftQQuality/.