Universidad de las Ciencias Informáticas

Dirección de Calidad de Software

Método para determinar la complejidad de los requisitos funcionales de software.

Trabajo final presentado en opción al título de

Máster en Informática Aplicada

Autor: Daimara Mustelier Sanchidrian

Tutora: Dr.C Lizandra Arza Pérez

Cotutora: Msc Karina Pérez Teruel

Ciudad de La Habana, Diciembre de 2014

AGRADECIMIENTOS

A mis padres por todo su amor, dedicación, ayuda, por ser ejemplos de sacrificio, de constancia, por no dejarme caer nunca, ustedes son mi orgullo y mi gran satisfacción.

A mi amor, por todo su cariño, paciencia, por estos años de felicidad, gracias por existir y por ser tan especial. Gracias por tu preocupación y ayuda en todo este tiempo de investigación y desvelo.

A mi abuela, tía, primo, hermano y sobrinas, por tenerme siempre en lo más presente de sus vidas, por su preocupación y ayuda a lo largo de mi vida.

A la familia de Ale que aunque en la distancia no han dejado de preocuparse por mi superación y me han acogido como uno más de ellos, gracias por ser tan buenos conmigo.

A todas mis amistades que me han dado tanto ánimo, que han estado conmigo en los momentos buenos y malos, gracias por cada día que han estado presentes.

A todos mis compañeros de la Dirección de Calidad de software y a los que forman parte junto a nosotros del Programa de Mejoras, ustedes son maravillosos y ha sido un placer para mí haber trabajado juntos.

A mis tutoras por su ayuda, por sus buenos consejos y revisiones. Liza gracias por estar y guiarme en los dos momentos más importantes de mi crecimiento como profesional.

A los profesores que he tenido y que contribuyeron en mi formación como Máster.

A la UCI por las oportunidades que me ha dado para realizar mi vida profesional.

Dedicatoria

A mis padres, a Ale y a mi hijo Lucas que está por nacer, ustedes son la razón que me lleva a sonreírle a la vida cada día.

DECLARACIÓN JURADA DE AUTORÍA

Declaro por este medio que yo Daimara Mustelier Sanchidrian, con carné de identidad 84070501131, soy el autor principal del trabajo final de maestría *Método para determinar la complejidad de los requisitos funcionales de software*, desarrollada como parte de la Maestría en Informática Aplicada y que autorizo a la Universidad de las Ciencias Informáticas a hacer uso de la misma en su beneficio, así como los derechos patrimoniales con carácter exclusivo.

Y para que así conste, firmo la presente declaración jurada de autoría en Ciudad de La Habana a los ____ días del mes de _____ del 2014.

Daimara Mustelier Sanchidrian

RESUMEN

Los requisitos son razones fundamentales tanto de éxito como de fracaso en el desarrollo de un software, es por ello que todo esfuerzo que permita reducir errores en los requisitos y prevenir este esfuerzo malgastado resulta una acertada decisión. A partir de la mejora continua de los procesos que se llevan a cabo en la Universidad de las Ciencias Informáticas, de la puesta en práctica y ejecución de las evaluaciones que se realizan para conocer la complejidad de los requisitos funcionales en los proyectos de desarrollo de software y de diagnósticos realizados, se identificaron un grupo de insuficiencias en la evaluación de la complejidad de los requisitos. El presente documento describe los resultados de la investigación orientada a la definición de un método que permita disminuir los errores en las estimaciones correspondientes a la complejidad de los requisitos funcionales de software. La complejidad de un Requisito funcional se plantea como un problema donde es adecuado su tratamiento bajo incertidumbre. El método propuesto emplea técnicas de soft computing para manejar la incertidumbre y agregar la información que se tiene de las variables que definen la complejidad de los requisitos a partir del análisis de las descripciones y especificaciones de los requisitos funcionales. Los componentes del método son la valoración de las variables que definen la complejidad de un Requisito funcional, determinar complejidad y transformar resultados en términos lingüísticos. Se diseña una Guía metodológica basada en el método que brinda un marco de actividades para realizar la determinación de la complejidad de los requisitos funcionales de software. La definición del método y la Guía metodológica permitieron el desarrollo de una herramienta para la determinación de la complejidad que ayuda a la realización de esta actividad en los proyectos de la Universidad. Para corroborar la validez del método y la Guía metodológica se aplicaron métodos, como son un cuasi experimento y la técnica de Iadov.

Palabras claves: ingeniería de requisitos, requisito funcional, complejidad, incertidumbre.

ABSTRACT

The requirements are fundamental reasons of both success and failure in the development of a software, which is why every effort that allows to reduce errors in the requirements and prevent this wasted effort results a wise decision. From the continuous improvement of the processes carried out at the University of Information Science, the implementation and execution of evaluations conducted to determine the complexity of the functional requirements in software development projects and from diagnoses, a group of deficiencies were identified in evaluating the complexity of the requirements. This paper describes the results of research aimed to define a method to reduce errors on the corresponding estimates for the complexity of the functional requirements of software. The complexity of a functional Requirement arises as a problem where treatment under uncertainty is appropriate. The proposed method employs soft computing techniques to manage uncertainty and add the information we have of the variables that define the complexity of the requirements from the analysis of the descriptions and specifications of the functional requirements. The components of the method are the valuation of the variables that define the complexity of a functional Requirement, to determine complexity and to transform results in linguistic terms. A methodology guide based on the method that provides a framework of activities for determining the complexity of the functional requirements of software is designed. The method definition and methodology guide allowed the development of a tool for determining the complexity that helps carry out this activity in the University projects. To corroborate the validity of the method and methodology guide where applied methods, such as a quasi-experiment and Iadov technique.

Keywords: requirements engineering, functional requirement, complexity, uncertainty.

ÍNDICE

INTRODUCCION

La Industria del Software continúa su auge y desarrollo en la sociedad, muchas son las aplicaciones que cada vez son más aplicables a cualquier otra rama de la ciencia o la economía de cualquier país. Sin embargo aunque la Industria del Software en el mundo se ha desarrollado considerablemente en los últimos años, con un desarrollo acelerado, los resultados alcanzados no cubren las expectativas inicialmente deseadas debido básicamente a que, la cantidad de recursos a consumir (en tiempo principalmente) es alta y el trabajo realizado casi nunca tiene la calidad requerida. Los proyectos se concluyen en fecha posterior a lo planificado y los problemas no se detectan a tiempo.

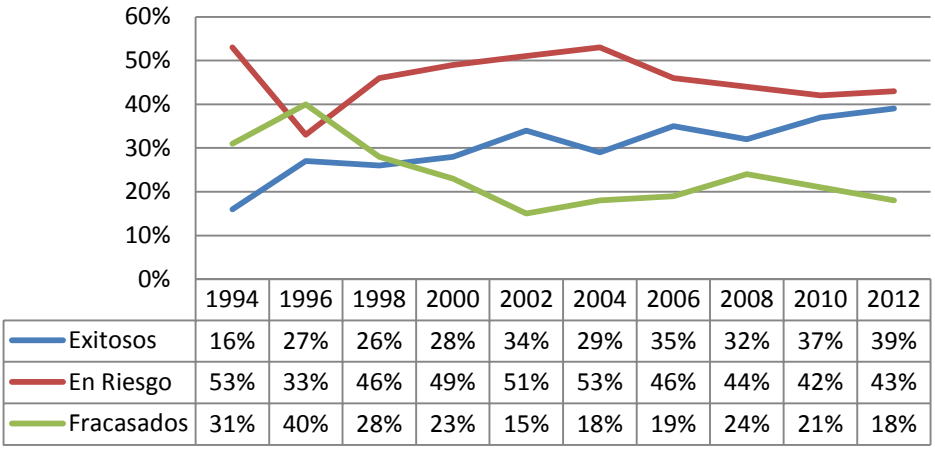
La Ingeniería de Requisitos (IR) cumple un papel fundamental en el proceso de desarrollo de software, refiriéndose a la rama de Ingeniería de Software relacionada con los objetivos reales y las funciones de los sistemas de software. La IR está relacionada con las especificaciones exactas del comportamiento del software y la evolución de los requisitos en el tiempo.

Dentro de los principales beneficios que se obtienen de la Ingeniería de Requisitos se pueden mencionar los siguientes [1]:

- Permite gestionar las necesidades del proyecto en forma estructurada.
- Mejora la capacidad de predecir cronogramas de proyectos, así como sus resultados.
- Disminuye los costos y retrasos del proyecto.
- Mejora la calidad del software.
- Mejora la comunicación entre equipos.
- Evita rechazos de usuarios finales.

El Standish Group da seguimiento a un estudio iniciado en el año 1994 en el que realiza un análisis del estado de la Industria del Software [2]. El informe CHAOS, que elabora el Standish Group, es el informe más famoso sobre el éxito y fracaso de los proyectos en el sector de las tecnologías de la información (TI), este suele realizarse cada dos años.

A continuación se muestra un resumen de ello a partir de reportes publicados hasta el año 2012 [3, 4].



	1994	1996	1998	2000	2002	2004	2006	2008	2010	2012
Exitosos	16%	27%	26%	28%	34%	29%	35%	32%	37%	39%
En Riesgo	53%	33%	46%	49%	51%	53%	46%	44%	42%	43%
Fracasados	31%	40%	28%	23%	15%	18%	19%	24%	21%	18%

Figura 1. Proyectos exitosos, en riesgo y fracasados (elaboración propia).

En el reporte anterior hay una marcada tendencia, a largo plazo, a aumentar los proyectos exitosos, a disminuir los proyectos fracasados, aunque desde el 2002 con un 15% estos han aumentado a un 24% y nuevamente en el 2010 han comenzado a disminuir y lo más significativo en los proyectos en riesgo es la estabilidad en un rango elevado (entre el 43% y el 53%) con una discreta disminución desde el 2004.

Los factores que inciden en los resultados de los proyectos exitosos son la participación del usuario, clara exposición de los requisitos y una planificación adecuada. Los proyectos en riesgo exceden el cronograma y presupuesto, no incluyen todos los requisitos planificados y sufren cambios en las especificaciones y en las necesidades inicialmente identificadas. Los proyectos fracasados presentan requisitos incompletos, escasa implicación del usuario y no cuentan con los recursos necesarios. En lo anteriormente planteado se constata que existe un elemento determinante que influye en el éxito o fracaso de un proyecto y es el asociado a la definición y entendimiento de los requisitos funcionales del software, lo cual ubica a la Ingeniería de Requisitos como un eslabón de suma importancia en el desarrollo de un proyecto de software.

En Cuba las empresas desarrolladoras de software no están al margen de la situación que se presenta en los estudios mundiales. En aras de aunar los esfuerzos individuales que han venido realizando diversas instituciones del país para alcanzar una fortaleza que permita incursionar, con más efectividad, en los mercados extranjeros fue creado Incusoft (Industria Cubana del Software).

La Universidad de las Ciencias Informáticas (UCI) es un centro docente-productivo, que a partir del año 2004 comienza a tener una presencia productiva en la Industria del Software en Cuba con el desarrollo de proyectos informáticos fundamentalmente relacionados con la informatización de la sociedad y la

exportación. En aras de aumentar la calidad de sus procesos y productos comienza a llevarse a cabo en la Universidad a partir del año 2008 el Proceso de Mejora CMMI nivel 2 para desarrollo, logrando en el 2011 la implementación de estos procesos en tres centros de desarrollo de la Universidad. A raíz de este proceso se logra establecer un expediente para proyectos de desarrollo [55], en el cual se formalizan dos métodos de suma importancia, siendo estos la “Evaluación de requisitos funcionales” el mismo corresponde al área de proceso de Administración de requisitos y permite la determinación de la complejidad de los requisitos funcionales y el “Método de estimación post arquitectura”, correspondiente al área de proceso de Planeación de proyectos, este método permite reajustar la estimación en cuanto a la planificación realizada al inicio del proyecto una vez que se define con mayor exactitud qué desarrollar y cómo, teniendo como base fundamental la estimación que se realiza de la complejidad de los requisitos funcionales.

Un diagnóstico realizado por Calisoft en el año 2012 con el propósito de determinar fortalezas y debilidades de la actividad productiva de la UCI arrojó dentro de sus principales resultados que las prácticas del área de proceso de Administración de requisitos tienen un bajo nivel de implementación en general, con una tendencia a aumentar el porciento de prácticas no implementadas en esta área, otro resultado importante lo constituyó los factores potenciales de éxito y de fracaso que más influyen en los proyectos de desarrollo, siendo la definición de los requisitos uno de los factores con mayor porciento que influye en ambos casos.

Una de las actividades a realizar dentro de la disciplina de IR descrita en el área de proceso de Administración de requisitos para el nivel 2 de CMMI para desarrollo [5], es la de Obtener y especificar requisitos, dentro de esta uno de los productos de trabajo a realizar por parte de los proyectos es el método de Evaluación de requisitos funcionales antes mencionado, este producto se basa en los siguientes elementos:

- Variables que definen la complejidad de un requisito y deben ser analizadas por cada requisito en cuestión.
- Métodos matemáticos que brindan una puntuación para cada requisito teniendo en cuenta las variables consideradas por los expertos.
- Hoja de cálculo del paquete Microsoft Office donde se llevan a cabo los cálculos por cada requisito, quedando registrado la clasificación final de la complejidad en cuanto a alta, media y baja por cada uno.

Este resultado constituye una base importante para las estimaciones relacionadas con el tiempo, esfuerzo, recursos y costo de un proyecto de desarrollo de software que se realizan mediante el Método de estimación post arquitectura, lo cual impacta en la planificación de los proyectos, de ahí que una buena razón para que la planificación se realice correctamente, es precisamente lograr una estimación de la complejidad de los requisitos funcionales lo más alejada posible de errores.

A partir de la mejora continua de los procesos que se llevan a cabo en la Universidad, de la puesta en práctica y ejecución de este método, y del diagnóstico realizado por Calisoft, se identificaron un grupo de insuficiencias:

Concepción del método:

- No tiene en cuenta que es un problema en el que pueden intervenir múltiples expertos que pueden tener diferentes criterios en cuanto a las variables y a la forma de valorarlas.
- Las variables en su mayoría corresponden precisamente a requisitos no funcionales que se pudieran asociar a un requisito funcional o al comportamiento o a lo que deseamos de un sistema en términos de calidad, por lo no están directamente relacionadas con un requisito funcional.
- No tiene en cuenta la incertidumbre como parte de la información cualitativa que se emite en las clasificaciones de alta, media y baja de complejidad.
- No es flexible en cuanto al peso o nivel de importancia de la variable, dado que se debe tener en cuenta que existen diferentes proyectos para los cuales la importancia de una variable no necesariamente tiene que ser la misma.

Determinación de la complejidad:

- El trabajo se puede tornar engorroso dado que no todas las celdas presentan la formulación, llevando al usuario a realizar los cálculos fuera de la herramienta o a dedicar tiempo a corregir estos errores en la propia herramienta.
- Si se deja una celda en blanco realiza una división por cero que no tiene definición, originando estimaciones erróneas.
- No permite la generación de reportes, lo que afecta la toma de decisiones en los proyectos relacionadas con la complejidad de los requisitos.

Estos elementos impiden realizar un correcto análisis de la complejidad de los requisitos funcionales de software impactando negativamente en las estimaciones que se realizan de los requisitos en cuanto a su complejidad.

A partir de lo anterior se ha identificado el siguiente **problema**:

Las insuficiencias en la determinación de la complejidad de los requisitos funcionales de software afectan la correcta estimación de la planificación en los proyectos de desarrollo de software.

El **objeto de estudio** de este trabajo lo constituye: Ingeniería de Requisitos.

Se define como **objetivo general** de la investigación: Definir un método para la determinación de la complejidad de los requisitos funcionales de software que disminuya los errores en la estimación de la planificación en los proyectos de desarrollo de software, teniendo en cuenta la incertidumbre de la información.

Para dar cumplimiento al objetivo general se definen los siguientes objetivos específicos:

- Elaborar el marco teórico acerca de la Ingeniería de Requisitos, métodos para determinar la complejidad de los requisitos funcionales y modelos computacionales para gestionar la incertidumbre.
- Definir variables para determinar la complejidad de los requisitos funcionales.
- Definir el método para la determinación de la complejidad de los requisitos funcionales de software, teniendo en cuenta la incertidumbre de la información.
- Definir una Guía metodológica que basada en el método brinde el marco de actividades para realizar la determinación de la complejidad de los requisitos funcionales de software.
- Desarrollar una herramienta informática basada en el método propuesto para la determinación de la complejidad de los requisitos funcionales de software.
- Validar el método propuesto a partir de su aplicación en diferentes proyectos de desarrollo.

Se identifica como **campo de acción:** Métodos para determinar la complejidad de los requisitos funcionales de software.

Se plantea como **hipótesis de investigación**: Si se define un método para la determinación de la complejidad de los requisitos funcionales de software que tenga

en cuenta la incertidumbre de la información disminuirán los errores en la estimación de la planificación en los proyectos de desarrollo de software.

Las **variables de la investigación** son:

- Variable Independiente: método para la determinación de la complejidad de los requisitos funcionales de software que tenga en cuenta la incertidumbre.
- Variable Dependiente: disminución de los errores en la estimación de la planificación en los proyectos de desarrollo de software.

Para lograr el objetivo se deben realizar las siguientes **tareas de investigación**:

- Realización de búsquedas bibliográficas sobre la Ingeniería de Requisitos, métodos para determinar la complejidad de los requisitos funcionales y modelos computacionales para gestionar la incertidumbre.
- Estudio de los diferentes métodos basados en la complejidad de los requisitos.
- Análisis de las variables identificadas que pueden definir la complejidad de los requisitos funcionales.
- Análisis de las técnicas de soft computing que permiten el tratamiento de la incertidumbre.
- Realización de encuestas a analistas, arquitectos y desarrolladores de los proyectos productivos.
- Definición del método para determinar la complejidad de los requisitos funcionales de software en los proyectos productivos.
- Definición de una Guía metodológica basada en el método propuesto.
- Implementación de una herramienta que permita la determinación de la complejidad de los requisitos funcionales de software.
- Aplicación de la propuesta en una muestra de los proyectos de desarrollo de la Universidad de las Ciencias Informáticas.
- Evaluación del resultado de la aplicación.

Dentro de los métodos aplicados para llevar a cabo la investigación se encuentran los siguientes:

Métodos teóricos:

Analítico-Sintético: posibilita realizar el análisis de documentos, teorías, permitiendo de esta manera la obtención y descomposición de los elementos más importantes que se relacionan con la complejidad de los requisitos.

Histórico-Lógico: permite comprobar teóricamente el comportamiento del proceso de Ingeniería de Requisitos a lo largo de su desarrollo.

Métodos empíricos:

Encuesta: permite obtener información concerniente sobre las posibles variables que definen la complejidad de un requisito en los proyectos.

Observación y medición: para la evaluación de los resultados de la aplicación del método y la guía metodológica propuestos.

El **aporte teórico-práctico** de la investigación radica en:

Obtención de variables que definen la complejidad de los requisitos funcionales de software.

Obtención de una Guía metodológica para la determinación de la complejidad de los requisitos funcionales de software.

Obtención de una herramienta que permite determinar la complejidad de los requisitos funcionales de software.

La **novedad científica** de la investigación se basa en la determinación de la complejidad de los requisitos funcionales de software mediante técnicas de soft computing.

El presente trabajo está compuesto por introducción, tres capítulos, conclusiones, recomendaciones, referencias bibliográficas y anexos.

En el CAPÍTULO I. LA INGENIERIA DE REQUISITOS COMO PARTE DEL PROCESO DE DESARROLLO DE SOFTWARE:

- Se conceptualiza qué es la Ingeniería de Requisitos.
- Se clasifican los requisitos, haciendo énfasis fundamentalmente en los requisitos funcionales de software y en su clasificación por complejidad.
- Se tratan los métodos de estimación que tienen en cuenta los requisitos funcionales.
- Se analizan algunos modelos lingüísticos, seleccionando aquel que permita tratar la incertidumbre en la complejidad de los requisitos funcionales de software.

En el CAPÍTULO II. MÉTODO PARA DETERMINAR LA COMPLEJIDAD DE LOS REQUISITOS FUNCIONALES DE SOFTWARE:

- Se presenta el método para determinar la complejidad de los requisitos funcionales de software aplicando técnicas de soft computing, utilizando en esta propuesta el modelo lingüístico de 2-tuplas.
- Se definen y validan un grupo de variables que definen la complejidad de los requisitos funcionales de software.

- Se seleccionan las variables lingüísticas a tener en cuenta en la propuesta del método.
- Se describen cada uno de los componentes del método propuesto.
- Se define una Guía metodológica basada en el método propuesto.

En el CAPÍTULO III. APLICACIÓN DEL MÉTODO Y ANÁLISIS DE LOS RESULTADOS:

- Se describe la aplicación de la Guía metodológica mediante la herramienta desarrollada para la determinación de la complejidad de los requisitos funcionales de software.
- Se realiza una comparación mediante un cuasi experimento entre el método de Evaluación de requisitos por el cual trabajan actualmente los proyectos de desarrollo de la Universidad y el método propuesto en la presente investigación para validar que existe una disminución de errores al realizar la estimación de la complejidad de los requisitos funcionales de software.
- Se aplica la técnica de Iadov con el objetivo de medir el nivel de satisfacción de los usuarios respecto a los resultados arrojados por la herramienta.
- Se analizan los resultados y recomendaciones a partir de la validación del método y de la Guía metodológica propuestos.

CAPÍTULO I: LA INGENIERIA DE REQUISITOS COMO PARTE DEL PROCESO DE DESARROLLO DE SOFTWARE

1. Introducción

En la actualidad se cuenta con una mejor administración de los proyectos de software, los líderes de proyectos en su gestión deben balancear los requisitos de tiempo, calidad, costos y alcance, al constituir éstos, áreas fundamentales para la obtención de resultados satisfactorios dentro de un proyecto. En apoyo a obtener mejores resultados en las estimaciones de tiempo, recursos, costo y esfuerzo del proyecto, se debe lograr una correcta definición y un buen entendimiento de los requisitos funcionales, para ello se hace necesario desempeñar correctamente cada una de las actividades de la disciplina que permite llevar a cabo el proceso de administración de los requisitos, siendo esta la Ingeniería de Requisitos.

2. Ingeniería de Requisitos

La Ingeniería de Requisitos es una de las disciplinas más importantes en el proceso de desarrollo de software, se utiliza para definir todas las actividades involucradas en el descubrimiento, documentación y mantenimiento de los requisitos para un proyecto de software determinado. Sin unos requisitos bien descritos, el desarrollador no sabe qué construir, el usuario no sabe qué esperar, y no hay manera de validar que el sistema creado realmente satisface las necesidades originales del usuario. Varias han sido las definiciones referentes a esta disciplina [6, 7], coincidiendo de alguna manera con los siguientes elementos:

- Incluye el conjunto de tareas que conducen a comprender cuál será el impacto del software sobre el negocio, qué es lo que el cliente quiere y cómo interactuarán los usuarios finales con el software.
- Es el proceso de desarrollar una especificación de software completa, consistente y no ambigua.
- Área del conocimiento que explica cómo lograr la obtención, análisis, especificación y validación de los requisitos del software.
- El proceso que establece y mantiene acuerdos sobre los cambios de requerimientos, entre los clientes y el equipo del proyecto.

En este sentido la autora concuerda con cada uno de los elementos descritos anteriormente. Existe un elemento esencial en esta disciplina y sin el cual no existiría un producto de software, es el relacionado con los Requisitos funcionales.

2.1 Requisitos funcionales de software

Los errores introducidos en la etapa de requisitos son altamente perjudiciales fundamentalmente porque obligan al equipo de desarrollo a realizar un arduo y costoso trabajo de reprogramación para corregirlos. Resulta de 68 a 110 veces más costoso corregir un defecto de requisitos encontrado cuando el software ya está operando de lo que costaría si el mismo defecto fuera encontrado durante la etapa de desarrollo de requisitos [8]. Un error, omisión o mala interpretación en los requisitos obliga a los miembros del proyecto a rehacer todo el trabajo hecho sobre la base de requisitos incorrectos. Todo esfuerzo que permita reducir errores en los requisitos y prevenir este esfuerzo malgastado resulta una acertada decisión.

Los requisitos de software generalmente pueden clasificarse en dos grupos: los funcionales y los no funcionales. En la presente investigación se tendrán en cuenta los requisitos funcionales. Se estudiaron varias definiciones del tema, tales como:

- Una condición o capacidad que una aplicación debe cumplir o tener para resolver un problema o alcanzar un objetivo [9].
- Los requisitos son una especificación de lo que debe ser implementado. Son descripciones de cómo el sistema debe comportarse, o de un sistema de propiedad o atributo. Ellos pueden ser una limitación en el proceso de desarrollo del sistema [7].
- Capacidad, característica o factor de calidad de un sistema mediante el cual se pretende cumplir con determinadas necesidades o restricciones operativas, aportando un valor y una utilidad para un cliente o usuario dentro del marco de solución de un problema en un entorno real [10].

La autora asume la definición de la IEEE que enmarca a un requisito funcional como una condición que una aplicación debe cumplir, resumiendo en gran medida lo planteado por los diferentes autores citados.

A la hora de analizar un requisito es de suma importancia conocer cuáles son las características que este debe cumplir, las mismas radican en [1]:

- Especificado por escrito: Como todo contrato o acuerdo entre dos partes.
- Posible de probar o verificar. Si un requisito no se puede comprobar, entonces ¿cómo se sabe si se cumplió con él o no?
- Conciso: Un requerimiento es conciso si es fácil de leer y entender. Su redacción debe ser simple y clara para aquellos que vayan a consultarlo en un futuro.

- Completo: Un requisito está completo si no necesita ampliar detalles en su redacción, es decir, si se proporciona la información suficiente para su comprensión.
- Consistente: Un requisito es consistente si no es contradictorio con otro requisito.
- No ambiguo: Un requisito no es ambiguo cuando tiene una sola interpretación. El lenguaje usado en su definición, no debe causar confusiones al lector.

Las características antes mencionadas brindan la posibilidad de un tratamiento ideal durante el análisis de los requisitos funcionales por parte de todos los involucrados.

Otro aspecto importante a tener en cuenta durante el análisis de los requisitos es el proceso de priorización de estos, varios autores [11, 12], concuerdan en que es un proceso de decisión preferentemente consensuado, destinado a establecer el orden de implementación de una serie de requisitos de software.

Aspectos como el costo, tiempo, riesgo [12-15] implican un análisis de todos los participantes en un proceso de priorización, para ello se considera de suma importancia la información disponible de los requisitos implementables, reflejándose la complejidad de los mismos como un elemento esencial para el costo de implementación, tiempo necesario que se ha de incurrir al implementarlos en forma efectiva, así como en el grado de riesgo y probabilidad de cambio de un requisito.

La complejidad de los requisitos funcionales de software se puede tratar entonces como uno de los elementos que apoya las decisiones del proceso de priorización, así como las estimaciones de tiempo de desarrollo, recursos humanos necesarios, costo y alcance de un proyecto.

2.2 Complejidad de los requisitos funcionales de software

Para aumentar las posibilidades de éxito durante el desarrollo de un proyecto de software, se debe describir exactamente la funcionalidad que el requisito entregará. Luego se deben evaluar con el fin de determinar la factibilidad de su implementación, dado que las limitaciones de la tecnología y los costos pueden hacer inviable su cumplimiento. Se considera que la evaluación del requisito podría partir de un análisis exhaustivo de la complejidad del mismo.

La complejidad se puede definir como “el grado en que un sistema o componente tiene un diseño o aplicación que es difícil de entender y verificar” [9, 16].

Al referirse a la complejidad de un requisito, se está haciendo alusión a la característica que describe la dificultad de diseño e implementación de este dentro

del proceso de desarrollo de software. Es de gran importancia para los expertos clasificar los requisitos de software según su complejidad, dado que permite brindar un mejor tratamiento de los mismos durante el desarrollo de cualquier producto, sirviendo de soporte para la toma de decisiones de las fases posteriores.

Varios han sido los métodos de estimación propuestos en busca de cuantificar con exactitud el tamaño funcional del software, en algunos casos teniendo como base la complejidad de los requisitos funcionales de software y otros lo utilizan como un elemento adicional a tener en cuenta.

3. Métodos de estimación de software

La estimación de software proporciona la parte esencial de una buena gestión de proyectos. Se presenta en la actualidad como un medio esencial para realizar las estimaciones oportunas del tiempo y recursos necesarios para el desarrollo de proyectos de software [17].

3.1 Métodos para determinar la complejidad del software

3.1.1 Una medida de complejidad basada en el documento de ingeniería de requisito.

Ashish Sharma propone una medida basada en los factores derivados del documento de especificación de requisitos del software (SRS, por sus siglas en inglés), definido como “Documentación de los requisitos esenciales (funciones, rendimiento, restricciones de diseño, y atributos) del software y de sus interfaces externas” [9]. La ventaja de este enfoque es que es capaz de estimar la complejidad de software en las primeras fases del ciclo de vida del software, incluso antes de que se lleve a cabo el análisis y diseño [18].

Para generar esta medida de complejidad se han considerado los siguientes atributos [18]:

- **Complejidad de entradas y salidas (IOC)**

  Se refiere a la entrada y salida del sistema de software y las interfaces y los archivos adjuntos.

- **Requisito funcional (FR)**

  Define las acciones fundamentales que deben tener lugar en el software en la aceptación y el procesamiento de las entradas y salidas. Puede ser apropiado para dividir el requisito funcional en sub-funciones o sub-procesos.

- **Requisitos no funcionales (NFR)**

  Se refiere a los requisitos relacionados con la calidad del software, aparte de la funcionalidad.

- **Complejidad del requisito (RC)**

  Se refiere a la suma de todos los requisitos, es decir funcional y su descomposición en sub-funciones y requisitos no funcional.

- **Complejidad del producto (PC)**

  Se refiere a la complejidad general sobre la base de las funcionalidades del sistema.

- **Complejidad de atributos personales (PCA)**

  Para el desarrollo efectivo del software la experiencia técnica juega un papel muy importante.

- **Restricciones de diseño impuestas (DCI)**

  Se refiere al número de restricciones que han de tenerse en cuenta durante el desarrollo del software.

- **Complejidad de interfaces**

  Se utiliza para definir el número de integración exteriores / Interfaces para el módulo / programa / sistema propuesto. Estas interfaces pueden ser interfaz de hardware, la interfaz de comunicación y software de interfaz.

- **Complejidad de usuarios y ubicación (ULC)**

  Se refiere al número de usuarios para acceder al sistema y ubicaciones (simples o múltiples) en el que el sistema se va a desplegar.

- **Complejidad de las características del sistema (SFC)**

  Se refiere a las características que se añadirán al sistema con el fin de mejorar la apariencia.

Esta medida es computacionalmente simple y ayudará al desarrollador en la evaluación de la complejidad del software en fases tempranas, basándose en el SRS el cual debe tener todas las características, contenido y funcionalidad para hacer esta estimación de manera correcta.

Sin embargo para la presente investigación se considera lo siguiente:

- La medida propuesta determina la complejidad de los requisitos funcionales de forma empírica, sin declarar cuales son las variables que definen la complejidad de estos requisitos o un método cuyo objetivo principal consista en la determinación de la complejidad de los requisitos funcionales.
- No se tiene en cuenta técnicas que permitan el tratamiento de la incertidumbre al trabajar con información cualitativa en cada una de las definiciones.

- Se tendrán en cuenta los atributos interfaces, requisito funcional y restricciones en la presente investigación como parte del análisis de las variables que definen la complejidad de un requisito funcional.

3.2 Métodos de medición del tamaño funcional del software

3.2.1 Puntos de Función (IFPUG FPA)

Albrecht presentó por vez primera en Octubre de 1979, en Monterey (California) los resultados obtenidos en sus estudios [19] proponiendo el concepto de Puntos de Función (PF) como una nueva métrica de las aplicaciones de software, permitiendo de esta forma la medición previa del tamaño funcional del software. Para ello debían ser capaces de determinar el tamaño a partir de la especificación de requisitos, la especificación de diseño, el código fuente, o la aplicación en funcionamiento. Con esto pretendían resolver los problemas derivados del uso de la cantidad de líneas de código como medida del tamaño del software: por un lado, dependía del lenguaje de implementación; y por otro, era una medida “a posteriori”, pues se podía obtener una vez que el código era terminado [20].

En 1984 IBM mejoró aún más la definición para proporcionar complejidades individuales y un conjunto de características del sistema. En 1986 nació la IFPUG (Agrupación Internacional de Usuarios de Puntos de Función), cuyos objetivos eran hacerse cargo de la estandarización y promulgación de la métrica [21].

Desde entonces han sido propuestos múltiples extensiones y enfoques alternativos a los PF, principalmente orientados a extender la aplicabilidad más allá del dominio de los Sistemas de Información de Gestión, como fue propuesto originalmente por Albrecht. Esos métodos difieren en sus visiones y definiciones del tamaño funcional pero comparten el concepto central de medir la funcionalidad [20].

Actualmente hay cinco estándares para la medición del tamaño funcional del software reconocidos en la norma ISO/IEC 14143-1:2007 [22]:

- Método definido en la norma ISO/IEC 20926:2009 Software and systems engineering Software measurement - IFPUG functional size measurement method [23].
- Método definido en la norma ISO/IEC 20968:2002 Mk II Function Point Analysis – Counting Practices Manual [24].
- Método definido en la norma ISO/IEC 19761:2003 COSMIC-FFP - A Functional Size Measurement Method [25].
- Método definido en la norma, ISO/IEC 29881:2008 Information technology - Software and systems engineering - FiSMA 1.1 functional size measurement method [26].

- Método definido en la norma ISO/IEC 24570:2005 Software engineering - NESMA function size measurement method version 2.1 - Definitions and counting guidelines for the application of Function Point Analysis, [27].

El método Puntos de Función (IFPUG FPA) ha evolucionado, a partir de la propuesta original de Allan Albrecht en IBM. Según afirman varios autores [20, 21, 28] es el más conocido y más utilizado. Es por ello que el análisis detallado se centrará en el método definido por el IFPUG, teniendo en cuenta además que las variantes que surgen a partir de este método buscan fundamentalmente una mayor precisión en el conteo de Puntos de Función.

La métrica FPA tiene en cuenta un grupo de características generales del sistema (GSC) que tienen que ver con el entorno de la aplicación y fórmulas que permiten determinar el tamaño funcional del software basándose en los puntos de función no ajustados, el valor del factor de ajuste (VAF) y los puntos de función ajustados, autores como Mateo y Rodríguez [20, 28] , describen cada uno de estos elementos donde difieren uno de otros solo en notaciones conservando para cualquier caso el mismo significado.

- **Características generales del sistema para Puntos de Función**

1. Comunicación de datos	8. Actualización en línea
2. Procesamiento de datos distribuidos	9. Procesamiento complejo
3. Rendimiento	10. Reusabilidad
4. Configuración Altamente Usada	11. Facilidad de instalación
5. Tasa de la Transacción	12. Facilidad de uso
6. Entrada de datos en línea	13. Portabilidad
7. Eficiencia usuario final	14. Facilidad de cambio

Tabla 1. Características generales del sistema para PF (elaboración propia).

Cada una de estas características es evaluada en un rango desde 0 (Sin influencia) a 5 (Esencial), teniendo en cuenta un grupo de criterios pre-establecidos.

- **Valor del Factor de Ajuste (*VAF*)**

Una vez que se ha determinado la influencia de cada GSC en el sistema, se utiliza la siguiente fórmula para obtener el valor del VAF:

$$VAF = 0.65 + 0.01 \sum_{i=1}^{14} F_i \qquad (1)$$

siendo F_i el valor asignado a cada GSC.

- **Puntos de Función no ajustados (*PFsA*)**

Consiste en sumar el número de componentes de cada tipo conforme a la complejidad asignada, la cual se clasifica en Alta, Media y Baja, para ello se tiene en cuenta la tabla 2, donde aparecen los Puntos de Función para cada componente.

Según la teoría de Albrecht, los componentes o características principales del software se basaban en [28]:

- **Entradas externas**: Los datos provienen bien de una aplicación ajena al sistema, o bien del usuario, el cual los introduce a través de una pantalla de entrada de datos.
- **Salidas externas**: Los datos suelen ser los resultados derivados de la ejecución de algoritmos o la evaluación de fórmulas, y generan informes (reportes) o archivos de salida que sirven de entrada a otras aplicaciones.
- **Consultas Externas**: Consiste en la selección y recuperación de datos de uno o más Ficheros Lógicos Internos o de uno o más Ficheros Externos de Interfaz, y su posterior devolución al usuario o aplicación que los solicitó.
- **Ficheros Lógicos Internos**: Es un conjunto de datos definidos por el usuario y relacionados lógicamente, que residen en su totalidad dentro de la propia aplicación, y que son mantenidos a través de la Entradas Externas del sistema.
- **Ficheros Externos de Interfaz**: Es un conjunto de datos definidos por el usuario, que están relacionados lógicamente y que sólo son usados para propósitos de referencia. Los datos residen en su totalidad fuera de los límites de la aplicación y son mantenidos por otras aplicaciones.

Componente	**Complejidad del componente (factor de peso)**			**Total**
	Baja	**Media**	**Alta**	
Entradas externas	*3	*4	*6	
Salidas externas	*4	*5	*7	
Consultas Externas	*3	*4	*6	
Ficheros Lógicos Internos	*7	*10	*15	
Ficheros Externos de Interfaz	*5	*7	*10	
	Nº Total de Puntos Función sin Ajustar (PFsA):			_____

Tabla 2. Conteo de Puntos de Función (elaboración propia).

$$PFsA = \sum_{i=1}^{15} (n^{o} \text{ de componentes de tipo } i^{*} \text{ peso del componente}) \quad (2)$$

- **Puntos de Función ajustados (*PFA*)**

Para determinar el total de Puntos de Función ajustados se consideran los puntos función no ajustados por el factor de ajuste.

$$PFA = PFsA * VAF \qquad (3)$$

El método de estimación Puntos de Función es un método de gran utilidad para la estimación del tamaño del software, sin embargo se plantea lo siguiente:

- No determina la complejidad de los requisitos funcionales.
- No tiene en cuenta técnicas que permitan el tratamiento de la incertidumbre al manejar información cualitativa, dado que el análisis de las características generales del sistema se realiza mediante criterios subjetivos en base a la opinión de expertos.
- Se tendrá en cuenta para el análisis de las variables que definen la complejidad de los requisitos funcionales, las relacionadas con facilidad de cambio y transacciones.

3.3 Métodos para el cálculo de la complejidad de requisitos

3.3.1 Modelo matemático y procedimiento para evaluación por complejidad de los requisitos del software.

Karina Pérez Teruel junto a otros autores [29] proponen este modelo basándose en el producto de trabajo “Evaluación de requisitos” del área de proceso Administración de requisitos, que se lleva a cabo en los proyectos de desarrollo de la UCI a raíz de la implantación de los procesos del nivel 2 de CMMI para desarrollo.

Este modelo tiene como objetivo principal la determinación de la complejidad de los requisitos funcionales de software, siendo la determinación de la complejidad base fundamental para las estimaciones de tiempo, esfuerzo, recursos y costo que se proponen en el método post-arquitectura utilizado en la UCI en los proyectos de desarrollo de software.

El modelo en cuestión cuenta un grupo de variables y métodos matemáticos que permiten el tratamiento de las mismas, obteniéndose de esta forma la complejidad para cada requisito [29].

- **Variables que definen la complejidad de los requisitos funcionales de software**

  Complejidad por Interfaces: el componente interfaz se aplica a requisitos que presenten algún tipo de complejidad en su interacción con los siguientes elementos:

  - Humanas (formularios, informes).

- Equipo (Tomógrafos, Rayos X); Ejemplo: API, drivers.
- Programación (Programas externos necesarios para apoyar el producto); Ej. Adicionar un nuevo usuario, usar el LDAP).
- Comunicación (Protocolos de comunicación que serán utilizados).

Diferentes comportamientos: se refiere a que un mismo requisito se comporta de manera diferente ante determinadas situaciones; Ej. Admisión de un paciente, puede ser normal, o por emergencia, en el primer caso se recoge más información que en el segundo.

Formas de inicialización: se refiere a que un mismo requisito puede ser inicializado de diferentes formas, Ej. Ver detalles de una historia clínica, se inicializa cuando se crea la historia clínica en una vista previa, y cuando se selecciona mostrar historia clínica.

Consultas a fuentes de almacenamientos: los requisitos pueden presentar diversidad en la cantidad y complejidad de la interacción con las fuentes de datos.

- Base de Datos
- Ficheros
- Otros

Restricciones de validación: complejidad de todas las validaciones que lleve un requisito, tanto las validaciones de cara al cliente como en el servidor.

Grado de reutilización: complejidad de un requisito, para poder ser reutilizado por otros proyectos.

Lógica de negocio: los requisitos pueden presentar diferentes niveles de complejidad para la implementación de la lógica de negocio que contienen; Ej. Operaciones, métodos matemáticos.

- **Asignación y Normalización de Pesos de las Variables**

Para la valoración se utilizó un modelo de Scoring utilizando una expresión algebraica que produce una puntuación para cada requisito, teniendo en cuenta las n variables consideradas más importantes por parte de los expertos. Para obtener esa valoración, cada una de estas variables es ponderada en relación a su importancia relativa con respecto al resto de las variables. Se utiliza un modelo aditivo basado en la suma ponderada.

Se le asigna un peso a cada variable con un valor del 1 al 5 teniendo en cuenta la importancia de las variables al estar presente en el requisito, este peso aparece fijo para cada variable.

$$w_n = \frac{C_n}{\sum_{i=1}^{k} C_i} \quad (4)$$

C_n Peso asignado a la variable

w_n Peso normalizado de cada variable

k Total de variables

- **Valoración Final de la Complejidad de un Requisito**

Cada variable tiene una evaluación de complejidad alta, media o baja, según su incidencia en el requisito, teniendo en cuenta esto se le asigna una valoración (v) que toma valores de 1 a 3. Este valor es multiplicado por el peso normalizado de cada variable y se obtiene una valoración global (V) del requisito n como se describe en la ecuación 5.

En la valoración final de la complejidad se empleó el método matemático de suma ponderada. El mismo calcula la ponderación de las alternativas como resultado del sumatorio del producto del peso de cada variable por el valor que toma para esa alternativa la variable correspondiente.

$$V_i = \sum_{i=1}^{k} w_i v_i \qquad (5)$$

V_i Valoración final de la complejidad obtenida por el requisito i

v_i Valoración de cada variable del requisito i

w_n Peso normalizado de cada variable

Baja si, $0 < V_n \leq 1.4$

Media si, $1.4 < V_n \leq 2.4$

Alta si, $2.4 < V_n \leq 3$

El modelo propuesto, basado en el producto de trabajo “Evaluación de requisitos” es de gran utilidad para la determinación de la complejidad en los proyectos de desarrollo de software de la UCI.

Sin embargo se considera lo siguiente:

- Los métodos matemáticos utilizados en la evaluación por complejidad de requisitos de software son de gran utilidad en la información cualitativa que se persigue, pero no tienen en cuenta la incertidumbre de los datos con los que se trabaja. Se hace necesario realizar un estudio de técnicas que permitan representar la información teniendo en cuenta la incertidumbre de los datos en este tipo de problemas donde la decisión se basa en información cualitativa.
- No es flexible en cuanto al peso o nivel de importancia de las variables, dado que diferentes expertos pueden tener diferentes criterios a la hora de valorar una variable para un requisito funcional.

- Las variables en su mayoría corresponden a requisitos no funcionales, por lo no están directamente relacionadas con un requisito funcional.
- Se tendrán en cuenta para el análisis de las variables que definen la complejidad de un requisito funcional las relacionadas con: Interfaces, Diferentes comportamientos y Restricciones.

4. La incertidumbre en la complejidad de los requisitos funcionales de software

La complejidad de los requisitos funcionales es una información cualitativa, por lo que no puede ser evaluada de forma precisa. El modelado de la incertidumbre surge para representar la imprecisión en la información de determinados problemas, dado que la información que se trata puede tener diferentes rangos de valoración y los valores pueden tener distinta naturaleza. En ocasiones, la información que manipula un problema puede que no sea adecuado su tratamiento mediante un valor cuantitativo, sin embargo, esta puede ser fácilmente valorada en forma cualitativa. Técnicas como el soft computing y la computación con palabras, brindan un marco adecuado para una representación más cercana a la realidad del problema, que pueda representar el razonamiento humano y permita incorporar sus valoraciones con tratamiento de la incertidumbre.

4.1 Soft computing y la computación con palabras

Las técnicas de soft computing y la computación con palabras se han constituido a partir de formalizar prácticas asociadas al uso de técnicas de inteligencia artificial empleando la Lógica Borrosa. La computación con palabras es uno de los paradigmas actuales para el tratamiento de la incertidumbre y la información lingüística [30-32]. Zadeh introdujo el concepto de computación con palabras (CWW, por sus siglas en inglés), “es una metodología en la que los objetos de la computación son palabras y propuestas extraídas de un lenguaje natural”, [33].

La CWW atiende a tres objetivos fundamentales, [34]:

- Ofrecer una metodología para calcular y razonar cuando la información disponible no es suficientemente precisa como para justificar el empleo de números.
- Aprovechar la tolerancia de la imprecisión para alcanzar manejabilidad, robustez, bajo coste y mejor relación con la realidad.
- Proporcionar bases para el desarrollo de lenguajes de programación que pudieran aproximarse a los lenguajes naturales en apariencia y en capacidad de expresión.

Otros autores y el propio Zadeh [35-38] han hecho reflexiones o definiciones sobre soft computing que de una manera u otra redundan en los mismos conceptos. Se puede considerar soft computing como un enfoque multidisciplinario donde la teoría de conjuntos borrosos facilita la representación del razonamiento humano y su capacidad de aprender en un ambiente de incertidumbre e imprecisión.

Tanto la CWW como las técnicas de soft computing tienen su base en los conceptos asociados a la teoría de conjuntos borrosos, la lógica borrosa y el modelado lingüístico de la información. Por ello se hace necesaria una revisión de los principales conceptos y definiciones de estas áreas.

4.2 Elementos fundamentales de la lógica borrosa

La Lógica borrosa [39] fue diseñada para representar y razonar sobre conocimiento expresado de forma lingüística o verbal: Conocimientos "vagos", "borrosos". El aspecto central de los sistemas basados en la teoría de la lógica borrosa es que a diferencia de los que se basan en la lógica clásica tienen la capacidad de reproducir aceptablemente los modos usuales del razonamiento, considerando que la certeza de una proposición es una cuestión de grado.

Definición 1 [39]: Un ***conjunto borroso*** $\tilde{A}$ en X se caracteriza por una función de pertenencia $f_{\tilde{A}}(x)$ que asocia a cada elemento de X un número real del intervalo $[0,1]$, $f_{\tilde{A}}: X \rightarrow [0,1]$, donde el valor de $f_{\tilde{A}}(x)$ en x representa el "grado de pertenencia de x al conjunto $\tilde{A}$ ".

A diferencia de los conjuntos clásicos en los que un elemento pertenece o no a un conjunto, en los conjuntos borrosos los elementos pueden pertenecer con determinado grado. El elemento fundamental es la función de pertenencia. Se definen varios tipos de funciones de pertenencia [40-42], siendo las más empleadas las triangulares y las trapezoidales.

Estas representaciones de los conjuntos borrosos se les conoce como conjuntos borrosos de primer orden (de tipo 1), sin embargo en los últimos años se ha venido empleando la representación de los conjuntos de tipo 1 en conjuntos de 2 tuplas [30, 43] y los conjuntos borrosos de tipo 2 [32, 44, 45]. Varios autores [43, 46] plantean las limitaciones de los conjuntos de tipo 1 para representar las palabras, fundamentado en que utiliza funciones de pertenencia precisas y ellas guían a las palabras. Un elemento importante que se introduce a partir de estas teorías es el uso de un Modelo lingüístico borroso, que tiene como base teórica para su desarrollo la teoría de los conjuntos borrosos.

4.3 Modelado lingüístico borroso

El modelado lingüístico borroso es un enfoque aproximado que representa los aspectos cualitativos como valores lingüísticos mediante variables lingüísticas [47], [48].

Definición 2 [49] : Se entiende por ***variable lingüística*** una variable cuyos valores son palabras o sentencias en un lenguaje natural o artificial. En términos más específicos una variable lingüística es caracterizada por una quíntupla (N, T(N), U, G, M), donde N es el nombre de la variable, T(N) es el conjunto de términos, que son el conjunto de valores lingüísticos, U es el universo de discurso, G es la regla sintáctica para generar los términos en T(N), y M es la regla semántica para asociar a cada término lingüístico X su significado.

En la definición de variable lingüística es importante la selección de las etiquetas y los conjuntos borrosos asociados a ellas, en estos se concentra un peso significativo en la obtención del ordenamiento de las alternativas y la solución más adecuada. Es preciso buscar una expresión de la cardinalidad, que permita representar la información sin afectar la precisión y discriminación de algunos valores. La cardinalidad recomendada en los modelos lingüísticos es un valor impar entre 5 y 9, sin superar los 11 o 13 términos [50].

Una vez delimitados los términos lingüísticos se debe definir la semántica del conjunto de etiquetas y para ello existen varios enfoques. El más utilizado es el enfoque basado en funciones de pertenencia, que describe los conjuntos borrosos mediante funciones de pertenencia [41, 42, 50].

Para la representación de la información lingüística se han desarrollado varios modelos que independientemente de la operatoria y la representación de la información a utilizar se hace necesario el estudio de operadores que permitan agregar la información y ordenar las alternativas.

Definición 3 [41]: Un operador de agregación es una función $\mathrm{F}: [0,1]^n \rightarrow [0,1]$ que cumple las condiciones de contorno $\mathrm{F}(0,\ldots,0) = 0$ y $\mathrm{F}(1,\ldots,1) = 1$, debe ser continuo y monótono no decreciente.

Estos operadores tienen variantes que incorporan a su definición pesos, que facilitan el tratamiento diferenciado de cada uno de los valores a agregar. El operador media ponderada permite que diferentes valores tengan diferente importancia. Esto se realiza asignando a cada valor x_i un peso asociado w_i, que indica cuál es la importancia de ese valor.

Definición 4 [51] : Sea $x = \{x_i \ldots x_n\}$ un conjunto de valores numéricos y $W = \{w_i \ldots w_n\}$ un vector numérico con los pesos asociados a cada x_i, tal que, w_1 corresponde a x_1

y así sucesivamente. La media ponderada se obtiene como sigue:

$$\bar{x} = \frac{\sum_{i=1}^{n} x_i w_i}{\sum_{i=1}^{n} w_i} \quad (5)$$

La aplicación de operadores de agregación es importante pues permiten integrar la información de los expertos y deben ser considerados para formar los criterios para evaluar y ordenar las alternativas. Se considera adecuado, el uso de estos operadores en la investigación propuesta.

4.4 Modelado lingüístico borroso basado en 2-tuplas

Como soporte a la solución de problemas de decisión se han desarrollado modelos lingüísticos computacionales que facilitan el tratamiento de la incertidumbre mediante el uso de la Lógica Borrosa, las variables lingüísticas y la CWW. Estos modelos clásicos definen su funcionamiento en dos enfoques fundamentales: modelo basado en el Principio de Extensión y el Modelo Simbólico [52].

El uso de la modelación lingüística supone realizar operaciones con palabras, por lo que [51, 52] coinciden en que estos modelos computacionales presentan una serie de limitaciones desde el punto de vista de la pérdida de información debido a las operaciones de aproximación, lo cual afecta la precisión en los resultados finales.

Con el objetivo de mejorar la precisión en los procesos de la CWW y resolver las limitaciones anteriormente expuestas se han propuesto distintos modelos simbólicos basados en el modelo lingüístico borroso, tales como:

- Modelo Lingüístico de 2-tuplas [51].
- Modelo lingüístico 2-tupla proporcional [53].
- Modelo Lingüístico Virtual [54].

Rodríguez [52] realiza una comparación entre estos modelos, afirmando lo siguiente:

El modelo lingüístico virtual presenta operaciones cuyos resultados son valores numéricos que pueden estar fuera del universo del discurso, por lo que no pueden ser representados lingüísticamente. Obtiene valores difíciles de entender, su única utilidad es la ordenación.

El modelo 2-tupla proporcional propone operaciones simbólicas y funciones de transformación, sin embargo sus resultados sólo tienen asignada una sintaxis, porque su semántica no está claramente definida. Utiliza cuatro valores para representar una única valoración.

El modelo 2-tupla mantiene una representación difusa de la información lingüística. Sólo puede obtener valores en el universo del discurso de la variable y garantiza precisión cuando el conjunto de etiquetas es simétrico y uniformemente distribuido.

Para definir el modelo de representación basado en 2-tuplas se parte del concepto básico de traslación simbólica. Siendo $S = \{S_0, \ldots, S_g\}$ un conjunto de términos lingüísticos, y $\beta \in [0, g]$ un valor obtenido por un método simbólico operando con información lingüística [51]

Definición 5 [51]: La traslación simbólica de un término lingüístico S_i es un número valorado en el intervalo $[-0{,}5, 0{,}5]$ que expresa la "diferencia de información" entre una cantidad de información expresada por el valor $\beta \in [0, g]$ obtenido en una operación simbólica y el valor entero más próximo, $i \in \{0, \ldots, g\}$, que indica el índice de la etiqueta lingüística (S_i) más cercana en S.

A partir de este concepto, [51] desarrolla un modelo de representación para la información lingüística que utiliza como base la representación 2-tuplas, $(r_i, \alpha_i), r_i \in S \; y \; \alpha_i \in [-0{,}5, 0{,}5)$, donde: r_i representa la etiqueta lingüística, y α_i es un número que expresa el valor de la distancia desde el resultado original al índice de la etiqueta lingüística r_i más cercana en el conjunto de términos lingüísticos S, es decir, su traslación simbólica.

En la representación basada en 2-tuplas, se ha desarrollado toda una operatoria asociada a ella. Para operar con 2-tuplas se ha adaptado el operador media ponderada definido en el epígrafe anterior como sigue:

Definición 6 [51]: Sea $x = \{(r_1, a_1), \ldots, (r_m, a_m)\}$ un conjunto de 2-tuplas y $W = \{w_i \ldots w_n\}$ un vector numérico con los pesos asociados a cada 2-tupla. La media ponderada extendida $\bar{x}^e$ se define como:

$$\bar{x}^e = \Delta\left(\frac{\sum_{i=1}^{m} \Delta^{-1}(r_i, a_i).w_i}{\sum_{i=1}^{m} w_i}\right) = \Delta\left(\frac{\sum_{i=1}^{m} \beta_i.w_i}{\sum_{i=1}^{m} w_i}\right) \quad (6)$$

Para la determinación de la complejidad de los requisitos funcionales de software se hará uso del Modelo lingüístico de 2-tuplas, por las siguientes razones:

- Se basa en el Modelado lingüístico Borroso, pues mantiene una sintaxis y semántica difusa al representar y operar con términos lingüísticos.
- La valoración final de la complejidad de los requisitos funcionales de software solo tomará un valor del propio universo de términos lingüísticos inicial.

- Es adecuado para el tratamiento de la incertidumbre y es cercano al modelo cognitivo de los humanos.
- No hay pérdida de información, por lo que los resultados de las operaciones brindan información precisa.
- Ofrece resultados cualitativos fáciles de entender.
- Facilita la implementación de la computación con palabras.

5. Conclusiones del capítulo

- Las variables identificadas en los métodos de estimación estudiados se consideran de gran importancia para el análisis de la complejidad en los requisitos funcionales de software.
- La complejidad de un requisito funcional se plantea como un problema donde es adecuado su tratamiento bajo incertidumbre.
- Para la agregación de la información existen operadores de agregación. Se considera adecuado el uso del operador media ponderada.
- La modelación lingüística, los modelos lingüísticos computacionales y la computación con palabras permiten integrar las técnicas adecuadas para crear una solución que trate la complejidad de los requisitos funcionales de software.
- El uso del modelado lingüístico empleando la representación en 2-tuplas se considera la más acertada, a partir de que minimiza la pérdida de información asociada al proceso de agregación de información y aproximación.

CAPITULO II: MÉTODO PARA DETERMINAR LA COMPLEJIDAD DE LOS REQUISITOS FUNCIONALES DE SOFTWARE

1. Introducción

El presente capítulo tiene como objetivo, fundamentar y describir el método y la guía metodológica propuestos, teniendo en cuenta la aplicación de técnicas de soft computing. Este método brinda la complejidad de los requisitos funcionales de software basándose en un conjunto de variables que son analizadas a partir de productos de trabajo elaborados por los proyectos de desarrollo y el criterio de los expertos, transformando esta información mediante operaciones que se fundamentan en este capítulo. Se propone una guía metodológica basada en el método, la cual brinda un marco de actividades para realizar la determinación de la complejidad de los requisitos funcionales de software.

2. Descripción general del método

El **objetivo principal** del método es brindar la complejidad de los requisitos funcionales de software a partir de representar el criterio de los expertos en la valoración de las variables que definen la complejidad de un Requisito funcional.

Las **funciones principales** están relacionadas con ofrecer un marco de trabajo que permita el análisis de la información que aparece como resultado de las descripciones y especificaciones de los requisitos funcionales que se realizan como parte de los productos de trabajo de un proyecto de desarrollo de software en el área de Administración de requisitos.

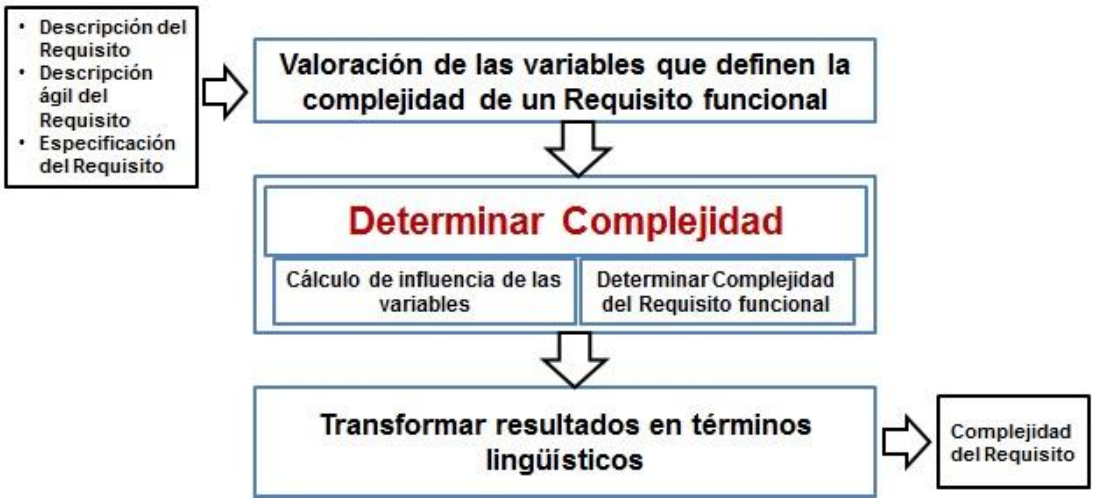


Figura 2. Método para la determinación de la complejidad de los requisitos funcionales de software (elaboración propia).

Principios del método

La determinación de la complejidad de los requisitos funcionales de software es una actividad donde es importante el tratamiento de la incertidumbre en la solución a considerar de acuerdo a lo identificado en los estudios realizados por la autora.

Del estudio realizado en el primer capítulo sobre las técnicas de soft computing y el modelado lingüístico se arribó a conclusiones que conllevan a establecer los fundamentos teóricos del método propuesto sobre la base del empleo de la modelación lingüística.

El método para la determinación de la complejidad de los requisitos funcionales del software que se presenta se sustenta sobre los siguientes principios:

1. **Modelación lingüística de la información**: Permite tratar la información cualitativa y la incertidumbre que representa la presencia de criterios de expertos.
2. **Utilización de operadores de agregación**: Permiten integrar la información de diferentes fuentes con que se toma la decisión para obtener los criterios.
3. **La complejidad se expresa con palabras**: El resultado del método está expresado en palabras o términos lingüísticos que pueden ser interpretados de manera natural por los expertos.

Definiciones y bases teóricas del método

A partir del estudio realizado del modelado de la incertidumbre, así como de los conceptos fundamentales para el procesamiento de esta información, se revisaron algunas de las definiciones más importantes y las facilidades de su uso. Se formalizan como base teórica del método que se plantea, las siguientes definiciones.

- **Para el modelado de la información:** Definición de Conjuntos Borrosos (Definición 1), definición de Variable Lingüística (Definición 2), las técnicas de soft computing y la computación con palabras tratado en el epígrafe 4.1 y el modelado lingüístico tratado en los epígrafes 4.3 y 4.4 del capítulo 1.
- **Para la operatoria:** Definición de traslación simbólica (Definición 5) y definición de Operador media ponderada (Definición 6) del capítulo 1.

La información de entrada y salida del método

Las **entradas** del método lo constituyen los productos de trabajo relacionados con las descripciones y especificaciones de un requisito funcional correspondientes al expediente de proyectos para desarrollo [55], los cuales deben ser analizados por los expertos para realizar la valoración de las variables, para el caso de los

proyectos que usen metodologías ágiles tendrán como entrada “Descripción ágil del Requisito” y para los proyectos que usen metodologías robustas “Descripción del Requisito” y “Especificación del requisito”.

Como **salida** se obtiene la Complejidad del Requisito y se hace corresponder con los términos lingüísticos asociados a la variable lingüística correspondiente, teniendo en cuenta los cálculos realizados al aplicar el operador de agregación de media ponderada y la transformación de los resultados en términos lingüísticos.

La relación entre los diferentes componentes del método se establece a partir de la definición de dos elementos fundamentales:

- Variables que definen la complejidad de un requisito funcional.
- Variables lingüísticas asociadas a la valoración y a la complejidad del requisito.

2.1 Variables que definen la complejidad de los requisitos funcionales de software

En el capítulo 1 se realizó un estudio de métodos que de alguna forma tratan la complejidad de los requisitos funcionales de software, de este estudió se arribó a conclusiones que permitieron identificar un grupo de variables importantes que pueden definir la complejidad de un Requisito funcional. Con el objetivo de aprobar la propuesta de estas variables para el análisis de la complejidad de requisitos funcionales en los proyectos de desarrollo de la UCI, se aplicó una encuesta donde su representatividad debía contar con tres aspectos fundamentales:

- Metodologías usadas en los centros de desarrollo (metodologías robustas y ágiles).
- Tipos de proyectos de desarrollo (Portales, Sistemas Operativos, Sistemas de Gestión, Componentes y Tecnologías base).
- Representación de los catorce centros de desarrollo de la universidad.

La población a estudiar fue el personal involucrado en los diferentes centros de desarrollo, dígase, los especialistas implicados en los temas de análisis y como unidad de estudio el área de proceso de Administración de requisitos. La selección se realizó con la técnica de muestreo no probabilística, muestreo intencional para poder obtener la mayor representatividad e información posible, de acuerdo con los intereses de la investigación, encuestando la mayor cantidad posible de especialistas que tienen experiencia en la administración de requisitos, que se han enfrentado a diferentes problemáticas durante la ejecución de las actividades relacionadas con los requisitos funcionales de software, las cuáles le han permitido madurar, adquirir más experiencia y conocimiento como experto de esta área.

La encuesta cuenta con 3 preguntas aplicadas a 114 especialistas, de ellos 77 analistas, 20 arquitectos, 17 desarrolladores, (en el Anexo 1 se puede ver el diseño de la encuesta). La misma se realizó con el objetivo de validar por parte de los expertos la propuesta del conjunto de variables que influyen en la complejidad de los requisitos funcionales de software.

Análisis de los resultados de la encuesta aplicada

En la pregunta # 1 se les propuso seleccionar de un conjunto de variables las que en su consideración influían en la complejidad de los requisitos funcionales de software.

Los resultados derivados de esta pregunta en cuanto a la selección de variables arrojaron un valor de 95% de aceptación por cada variable. Estas valoraciones reflejan la aprobación de los especialistas de las variables propuestas.

En la pregunta # 2 se les propuso seleccionar si consideraban que la determinación de la complejidad de los requisitos funcionales de software constituye un elemento importante a tener en cuenta para la planificación de los proyectos de desarrollo de software.

Los resultados fueron muy satisfactorios, dado que el 100% de los encuestados plantearon estar de acuerdo con la afirmación de esta pregunta.

En la pregunta # 3 se puso a consideración de los especialistas la inclusión de otras variables a tener en cuenta en la determinación de la complejidad de los requisitos funcionales de software independientemente de la selección realizada en la pregunta # 1.

En este caso no se identificaron nuevas variables a tener en cuenta.

A partir de los resultados de la encuesta aplicada, se relacionan a continuación las variables a tener en cuenta para la determinación de la complejidad de los requisitos funcionales de software:

- **Interfaces**: Se aplica a requisitos que presenten algún tipo de complejidad en su interacción con los siguientes elementos, considerados subcomponentes.
  Humanas (formularios, informes)
  Equipo (Tomógrafos, Rayos X)
  Programación (Programas externos necesarios para apoyar el producto)
  Comunicación (Protocolos de comunicación que serán utilizados)
  Atributos (Se refiere a la complejidad que puede agregarle la cantidad de atributos contenidos en una interfaz)

- **Facilidad de cambio**: Esfuerzo específico de diseño e implementación del requisito para facilitar cambios futuros durante el desarrollo del producto.
- **Dependencia con otros requisitos**: Describe el grado de relación de un requisito con otros, según la cantidad de requisitos con los que guarde algún tipo de dependencia para su implementación.
- **Requisitos asociados**: Se refiere a la descomposición en sub-funciones de un requisito más general y que no guardan dependencia entre sí.
- **Transacciones**: Es la interacción con una estructura de datos compleja, compuesta por varios procesos que se han de aplicar uno después del otro.
- **Diferentes comportamientos**: Se refiere al comportamiento que puede tener el requisito ante determinadas situaciones, en dependencia de ello se recogerá más información.
- **Restricciones**: Se refiere a los requisitos asociados a las restricciones de diseño, implementación, interfaces, físicas y reglas de negocio.
    - **Restricciones de diseño**: limitar el diseño y declarar los requisitos sobre el enfoque que debe tenerse en cuenta en el desarrollo del sistema.
    - **Restricciones de implementación**: poner límites al proceso de generación de código o de construcción (estándares requeridos, lenguajes, herramientas o plataforma)
    - **Restricciones de interfaz**: son requerimientos para interactuar con sistemas externos, describiendo los protocolos o la naturaleza de la información que debe ser transferida a través de la interfaz.
    - **Restricciones físicas**: afectan el hardware o el empaquetado del sistema (forma, tamaño y peso)
    - **Reglas del negocio**: son las políticas, normas, estatutos, acuerdos, resoluciones o cualquier tipo de decisión que gobierna la forma en que la institución opera. Ellas restringirán los pasos descritos en el flujo del caso de uso.

2.2 Selección de las variables lingüísticas y del conjunto de términos lingüísticos

Teniendo en cuenta el estudio realizado en el capítulo 1 se utilizará una función de pertenencia de tipo triangular con cinco términos lingüísticos, considerando que la cardinalidad recomendada en los modelos lingüísticos es un valor impar entre 5 y 9, sin superar los 11 o 13 términos.

La influencia de las variables sobre los requisitos funcionales estará representada por la variable lingüística "*Influencia de la variable"* y el conjunto de términos lingüísticos: No influye, Muy baja, Baja, Media, Alta. Por lo tanto la Influencia de las variables se delimita por un término lingüístico del conjunto S= {s_0= No influye, s_1= Muy Baja, s_2=Baja, s_3=Media, s_4=Alta}.

La Figura 3 muestra gráficamente la distribución uniforme de los cinco términos lingüísticos asociados a las variables lingüísticas en un intervalo de [0, 1].

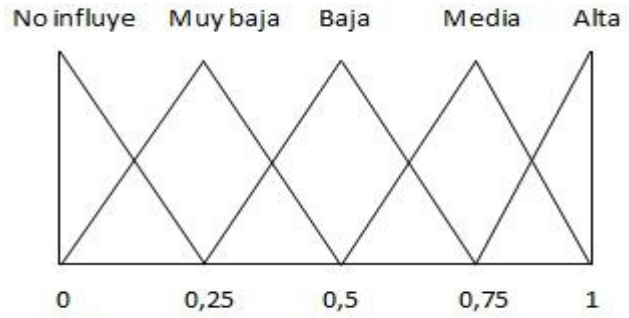


Figura 3. Sintáctica y semántica del conjunto de términos lingüísticos (elaboración propia).

La complejidad de los requisitos funcionales de software estará dada por la variable lingüística "*Complejidad del requisito funcional"* y el conjunto de términos lingüísticos: Muy baja, Baja, Media, Alta, Muy Alta. Por lo tanto la Complejidad de los requisitos se delimita por un término lingüístico del conjunto S= {s_0= Muy baja, s_1= Baja, s_2=Media, s_3=Alta, s_4= Muy Alta}.

La Figura 4 muestra gráficamente la distribución uniforme de los cinco términos lingüísticos asociados a las variables lingüísticas en un intervalo de [0, 1].

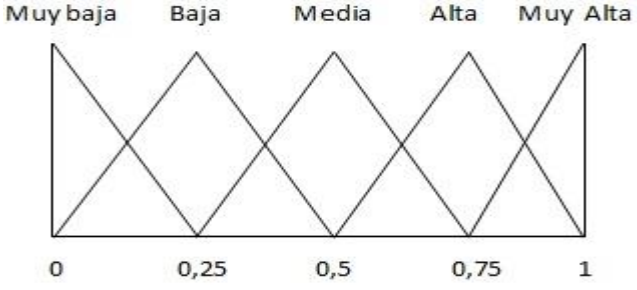


Figura 4. Sintáctica y semántica del conjunto de términos lingüísticos (elaboración propia).

2.3 Descripción de los componentes del método

Para lograr el correcto funcionamiento del método, se hace necesario realizar cada uno de los componentes propuestos en el mismo, estos se describen a continuación:

- **Valoración de las variables que definen la complejidad de un requisito funcional**: Se valoran las variables en función de los términos lingüísticos y el peso asociado para cada una según el criterio de los expertos.
- **Determinar complejidad**: Se determina la complejidad para cada Requisito funcional, teniendo en cuenta la influencia de las variables y los pesos asignados por todos los expertos.
- **Transformar resultados en términos lingüísticos**: Se realiza una transformación del resultado de la complejidad en 2-tuplas, basándose en la traslación simbólica del resultado obtenido.

La **valoración de las variables que definen la complejidad de un requisito funcional** se realiza con el objetivo de llevar a cada una de las variables a una misma variable lingüística "*Influencia de la variable".*

La valoración cuenta de tres pasos fundamentales:

1. Criterio de los expertos, basado en las entradas del método propuesto, así como en su conocimiento y experiencia en el área de administración de requisitos.
2. Asociar a partir del criterio de los expertos las variables al término lingüístico de la variable lingüística "*Influencia de la variable".*

   $S= \{s_0= \text{No influye}, s_1= \text{Muy baja}, s_2=\text{Baja}, s_3=\text{Media}, s_4=\text{Alta}\}$
3. Valorar a partir del criterio de los expertos los pesos asociados para cada una de las variables.

   El peso estará dado en una escala del 1 al 5, el mismo radicará en la importancia que le atribuye el experto a la presencia de la variable en el Requisito funcional.

La **determinación de la complejidad** se realiza a partir de haber relacionado para cada variable los términos lingüísticos y los pesos asignados por los expertos, mediante el operador de agregación de media ponderada adaptado a 2-tuplas.

- **Cálculo de influencia de las variables**: para cada variable se operará teniendo en cuenta los siguientes elementos:

1. Un valor $\beta \in [0, g]$ partiendo de que el conjunto de términos lingüísticos está representado por $S = \{S_0, \ldots, S_g\}$, utilizando para el método propuesto un total de cinco términos $S = \{S_0, S_1, S_2, S_3, S_4\}$, por lo que el valor $\beta \in [0,4]$.
2. Pesos asociados w_i para cada variable, los cuales se asocian a los términos lingüísticos seleccionados en dependencia del criterio de los expertos.

Teniendo cada uno de estos elementos el cálculo de influencia de cada variable se realizaría mediante:

$$I_v = \sum_{i=1}^{n} \beta_i \cdot w_i \quad (7)$$

Donde I_v representa la influencia de la variable que se esté analizando, β_i es el valor del término lingüístico y $w_i$es el peso asignado por cada experto, donde *n* representa la cantidad de expertos.

- **Determinar complejidad del Requisito funcional**: es un paso clave para lograr la salida del método propuesto en la investigación, para ello se tendrá en cuenta la influencia de cada variable.

  Para determinar la complejidad de modo que los cálculos se realicen en una sola operación se utilizará el operador de media ponderada adaptado para 2-tuplas.

$$C_{rf} = \Delta\left(\frac{\sum_{i=1}^{m} \Delta^{-1}(r_i, a_i) w_i}{\sum_{i=1}^{m} w_i}\right) = \Delta\left(\frac{\sum_{i=1}^{m} I_{vi}}{\sum_{i=1}^{m} w_i}\right) \quad (8)$$

Donde C_{rf} es el valor de la complejidad del Requisito funcional a partir del análisis de cada una de las variables, $\sum_{i=1}^{m} I_{v_i}$ el total de la suma de la influencia de cada una de las variables, $\sum_{i=1}^{m} w_i$ representa el peso total asignado a cada variable, *m* representa la cantidad de variables.

Transformar resultados en términos lingüísticos: La complejidad es una información cualitativa que se clasificará teniendo en cuenta el conjunto de términos lingüísticos S= {s_0= Muy baja, s_1=Baja, s_2=Media, s_3=Alta, s_4= Muy Alta}.

Para transformar los resultados en términos lingüísticos se hará uso de la representación en 2-tuplas (s_i, α), donde: s_i representa el término lingüístico, y α_i es un número que expresa el valor de la distancia desde el resultado original al índice del término lingüístico s_i más cercana en el conjunto de términos lingüísticos S, es decir, su traslación simbólica.

3. Guía metodológica para la determinación de la complejidad de los requisitos funcionales de software

A partir del método se elaboró una Guía metodológica para la determinación de la complejidad de los requisitos funcionales de software que define un conjunto de fases y actividades que están en correspondencia con las entradas, salidas y componentes del método. Esta guía brinda el marco para la realización de la actividad relacionada con la evaluación de requisitos en el área de proceso de Administración de requisitos, la cual consiste en la determinación de la complejidad de los requisitos funcionales de software.

3.1 Requisitos y descripción general de la Guía metodológica

Para la aplicación de la Guía metodológica es necesario que se cumplan con los siguientes **requisitos**:

- Que exista un proyecto de desarrollo de software donde se determine la complejidad de los requisitos funcionales de software.
- Contar con los productos de trabajo relacionados con las descripciones y especificaciones de los requisitos funcionales de software.
- Tener conocimiento de las definiciones que se plantean como parte del método propuesto.
- Contar con el grupo de expertos que brinde los criterios para determinar la influencia de cada variable.
- Contar con la aplicación informática que implemente la formulación definida en el método.

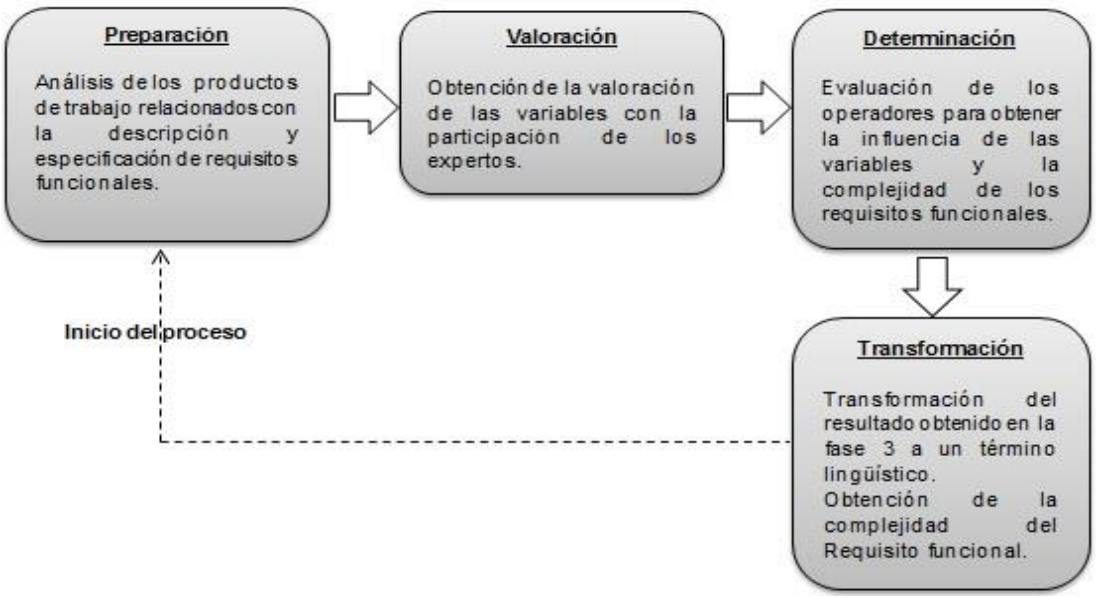


Figura 5. Fases de la Guía metodológica (elaboración propia).

Las fases de la Guía metodológica son: **Preparación**, **Valoración**, **Determinación, Transformación**. Estas fases se definen en función de realizar las actividades del proceso de Administración de requisitos que utiliza el método para la determinación de la complejidad de los requisitos funcionales de software.

3.2 Descripción de las fases

Fase 1: Preparación

Entradas: Productos de trabajo: Descripción de requisitos, Descripción ágil de requisitos y Especificación de requisitos.

Salidas: Variables que definen la complejidad del Requisito funcional.

Descripción general: Para la realización de esta fase los proyectos deben contar con los productos de trabajo relacionados con la descripción y especificación de requisitos, estos deben ser analizados exhaustivamente por parte de los expertos, en función de garantizar que la información a tener en cuenta para la determinación de la complejidad sea suficiente. Los expertos deben tener conocimiento de las definiciones que se plantean como parte del método propuesto. La calidad con la que se realice esta fase garantizará la correcta ejecución de las fases posteriores.

Representación:

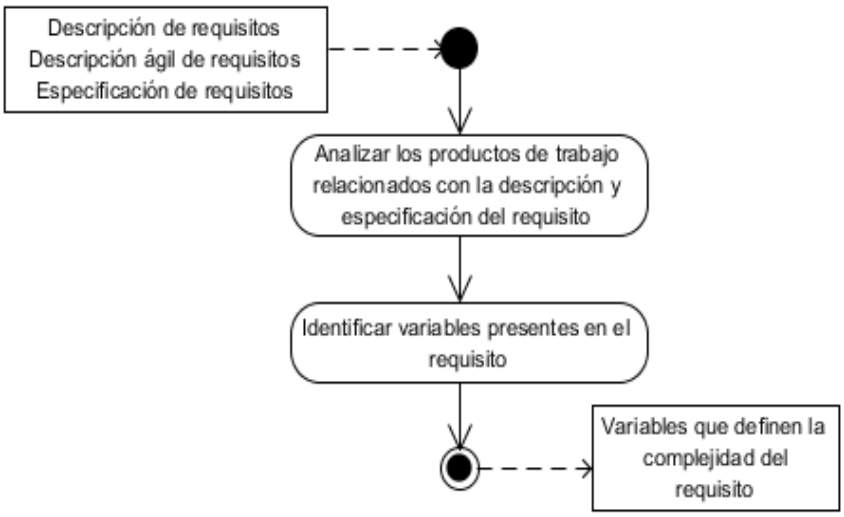


Figura 6. Representación de la fase 1: Preparación (elaboración propia).

Descripción de las actividades:

Inicio de la fase: Para el inicio de esta fase se deben haber realizado los productos de trabajo relacionados con la descripción y especificación del requisito que se esté

analizando, de forma tal que se garantice la información que permitirá a los expertos determinar la complejidad del Requisito funcional en cuestión.

Analizar los productos de trabajo del requisito: Una vez realizados los productos de trabajo relacionados con la descripción y especificación de requisitos, se analizan en función de buscar los elementos asociados a las variables que definen la complejidad de los requisitos funcionales y el nivel de importancia que representa la presencia de cada variable en el Requisito funcional.

Identificar variables presentes en el requisito: En función del análisis realizado en la actividad anterior se identifican las variables presentes en el Requisito funcional y el peso asociado para cada una, desde este momento los expertos deben tener criterios sobre la influencia de la variable en el Requisito funcional y el peso que tiene para el mismo.

Cierre de la fase: Con el cierre de esta fase se tienen las variables que definen la complejidad de un Requisito funcional y se está en condiciones de realizar la valoración de las variables por parte de los expertos.

Fase 2: Valoración

Entradas: Variables que definen la complejidad del Requisito funcional.

Salidas: Valoración de las variables teniendo en cuenta el criterios de los expertos, términos lingüísticos y pesos asociados para cada variable.

Descripción general: Esta fase es automatizada. En esta fase se realizan las actividades relacionadas con el componente de valoración de las variables que definen la complejidad de un Requisito funcional definido en el método. Se obtiene la información relacionada con los términos lingüísticos y los pesos asociados para cada variable.

Representación

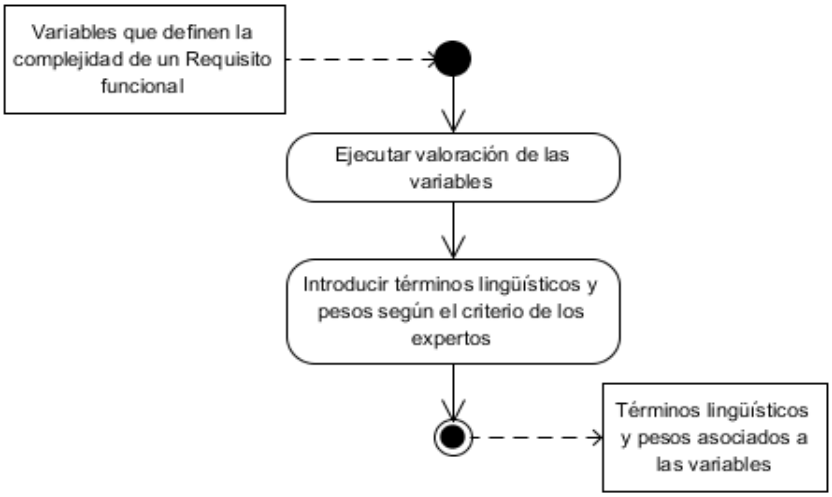


Figura 7. Representación de la fase 2: Valoración (elaboración propia).

Descripción de las actividades

Inicio de la fase: En el inicio de esta fase cada experto tiene conocimiento de las variables presentes en el Requisito funcional y se comienza a interactuar con la herramienta informática que permitirá el registro del criterio de los expertos para cada una de las variables.

Ejecutar valoración de las variables: Primeramente se introducen los requisitos en la herramienta mediante la importación del excel “Criterios para validar requisitos del producto” correspondiente al expediente de proyectos de desarrollo [55], una vez introducidos los requisitos se ejecuta la opción de realizar valoración.

Introducir términos lingüísticos y pesos según el criterio de los expertos: Se introducen en la herramienta los criterios de los expertos teniendo en cuenta los términos lingüísticos asociados a la variable lingüística “*Influencia de la variable”* y el peso en una escala del 1 al 5, estos términos, así como los valores del peso estarán en correspondencia con lo definido en el componente valoración de las variables que definen la complejidad de un Requisito funcional descrito en el método propuesto.

Cierre de la fase: Con esta fase se obtienen los términos lingüísticos y pesos asociados para cada una de las variables presentes en el Requisito funcional, teniendo en cuenta el criterio de los expertos.

Fase 3: Determinación

Entradas: Términos lingüísticos y pesos para cada variable.

Salidas: Influencia de las variables y complejidad del Requisito funcional

Descripción general: Esta fase es automatizada. En esta fase se realizan las actividades relacionadas con el componente determinar complejidad definido en el método. Se obtiene la información relacionada con la influencia de las variables y la complejidad del Requisito funcional, teniendo en cuenta el operador de agregación media ponderada.

Representación

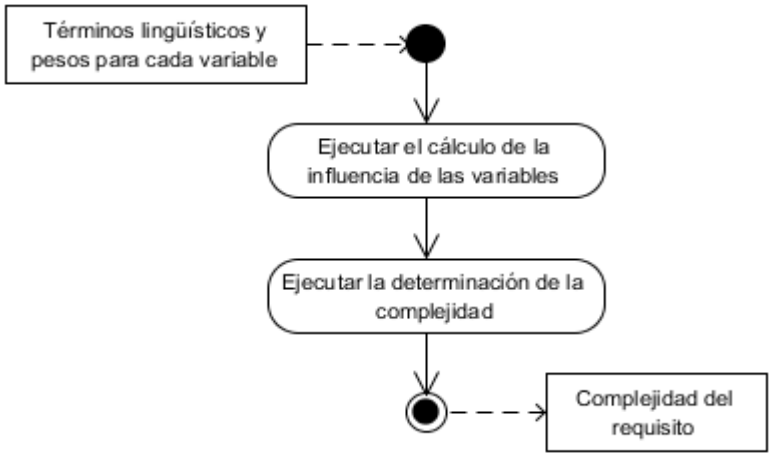


Figura 8. Representación de la fase 3: Determinación (elaboración propia).

Descripción de las actividades

Inicio de la fase: Con el inicio de esta fase se cuenta con los términos lingüísticos y pesos para cada una de las variables del Requisito funcional que se esté analizando.

Ejecutar el cálculo de la influencia de las variables: Una vez que se haya introducido cada criterio por variables como se explica en la fase anterior se determina la influencia de las variables, según lo definido en el componente determinar complejidad en la actividad relacionada con el cálculo de la influencia de las variables.

Ejecutar la determinación de la complejidad: Para el cálculo de la complejidad se recomienda que el analista principal del proyecto de desarrollo al cual se le esté determinando la complejidad de sus requisitos funcionales, tenga la responsabilidad

de determinar la complejidad del requisito en la herramienta, una vez que haya sido informado de que todos los expertos han introducido sus criterios, lo cual podrá comprobar además al mostrarse en la herramienta los criterios dados para cada una de las variables por parte de los expertos. Para la determinación de la complejidad se tendrá en cuenta lo definido en el componente determinar complejidad en la actividad relacionada con la determinación de la complejidad.

Cierre de la fase: Con el cierre de esta fase se obtiene la complejidad del requisito, la cual se muestra al usuario una vez que se ejecute la fase siguiente.

Fase 4: Transformación

Entradas: Resultado de la complejidad del Requisito funcional.

Salidas: Complejidad del Requisito funcional expresada en un término lingüístico.

Descripción general: Esta fase es automatizada. En esta fase se realizan las actividades relacionadas con el componente transformar resultados en términos lingüísticos. Se obtiene la información relacionada con la complejidad del Requisito funcional mediante la transformación en 2-tuplas.

Representación

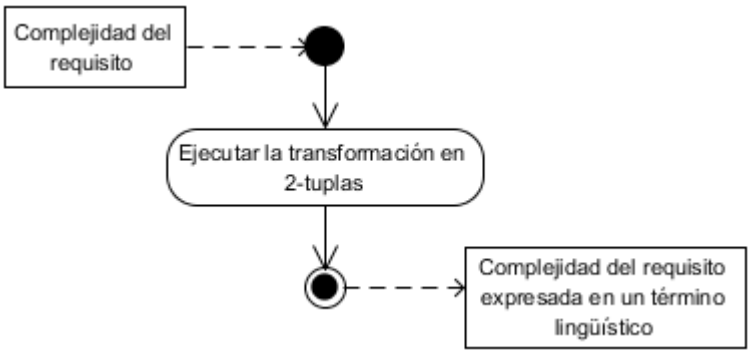


Figura 9. Representación de la fase 4: Transformación (elaboración propia).

Descripción de las actividades

Inicio de la fase: Esta fase se inicia con la complejidad del requisito, lo cual permitirá realizar las transformaciones necesarias para ser mostrada al usuario.

Ejecutar la transformación en 2-tuplas: Una vez que se haya determinado la complejidad del requisito se obtiene un valor numérico el cual debe ser transformado en función de mostrar al usuario un resultado final, pero expresado en uno de los términos lingüísticos de la variable lingüística "*Complejidad del requisito*

funcional" definidos en el componente transformar resultados en términos lingüísticos. Para la ejecución de esta transformación se utilizará lo definido en el componente mencionado anteriormente del método propuesto.

Cierre de la fase: Con el cierre de esta fase el usuario conoce la complejidad del requisito, mostrándose la misma en la herramienta.

Cada una de las fases y actividades descritas en la Guía metodológica deberán ser ejecutadas para cada Requisito funcional al cual se le desee determinar la complejidad.

4. Conclusiones del capítulo

- Se define el método que permite la determinación de la complejidad de los requisitos funcionales de software. Se definen las variables para el análisis de la complejidad de un requisito y las variables lingüísticas para la representación de la información. Se utiliza el modelado lingüístico para el tratamiento adecuado de la información, operadores de agregación que permiten la integración y transformación de la información.
- A partir del método se diseña una Guía metodológica que define cuatro fases estrechamente relacionadas con los componentes del método. Esta Guía metodológica permite establecer un proceso para la determinación de la complejidad de los requisitos funcionales de software.

CAPITULO III: APLICACIÓN DEL MÉTODO Y ANÁLISIS DE LOS RESULTADOS

1. Introducción

En el presente capítulo se describen los resultados de la validación del método y de la Guía metodológica a partir de métodos y de la aplicación de la propuesta. Se muestran los resultados de un cuasi experimento al aplicar la guía metodológica en dos momentos. Estos resultados fueron validados por personas involucradas en el proceso mediante la aplicación de la técnica de Iadov.

2. Diseño de la validación

La complejidad de un requisito funcional se plantea como un problema donde es adecuado su tratamiento bajo incertidumbre y la solución dada se concentra en un método que permite brindar los resultados teniendo en cuenta el criterio de múltiples expertos. Estos resultados deben corresponderse en gran medida con el juicio humano de quienes realizan esta actividad, lo cual ayudaría favorablemente a las actividades posteriores que dependen del buen análisis y ejecución que se realice del problema planteado. Siendo considerado como indicador fundamental para la validación de este método la disminución de errores en la estimación de la planificación en los proyectos de desarrollo de software.

Un elemento de gran importancia es la comprensión del método propuesto, de manera que durante el proceso de Administración de requisitos se entienda por parte de todos los involucrados las actividades que deben ser realizadas. Al implementar la guía metodológica que permite la aplicación del método se da solución a los problemas relacionados con la falta de entendimiento de la actividad relacionada con la determinación de la complejidad de los requisitos funcionales en el área de proceso de Administración de requisitos.

A partir de lo expresado anteriormente se propone para la validación, métodos experimentales y cuantitativos, de manera que se valide lo relacionado con las consecuencias del problema que se plantea y la satisfacción en cuanto a la solución propuesta. Para la validación se emplearon el siguiente método y técnica.

- Cuasi-experimento: Se realiza a partir de los resultados de la aplicación práctica de la Guía metodológica. Se emplea el método como un clasificador y se valida el indicador propuesto mediante la medida de efectividad, resultado de su comportamiento.
- Técnica de Iadov para medir la satisfacción de los expertos sobre los resultados obtenidos.

3. Aplicación de la Guía metodológica para la determinación de la complejidad de requisitos funcionales

En la aplicación de la Guía metodológica se ejecutaron las fases y actividades según lo descrito en el epígrafe 3 del capítulo 2. En la definición de esta guía se proponen algunas fases que deben ser automatizadas, debido a la cantidad de información a procesar, que depende de la cantidad de requisitos funcionales a los que se les desee determinar la complejidad y la cantidad de criterios a procesar según los expertos involucrados en esta actividad.

Se desarrolló una herramienta informática llamada COMREQ que implementa las ecuaciones y transformaciones necesarias para la automatización de las fases y actividades que la Guía metodológica sugiere. A continuación se tratan los elementos fundamentales de la aplicación de la misma, así como de la herramienta informática desarrollada.

En la fase de preparación: Se cuenta con los productos de trabajo relacionados con la descripción y especificación de los requisitos. Para el análisis de los mismos se debe tener conocimiento de las definiciones que garantizan el funcionamiento del método propuesto para el cual se aplica esta Guía metodológica, de forma tal que exista un conocimiento previo de las variables que definen la complejidad de un Requisito funcional y de las variables lingüísticas, lo cual permitirá una correcta interacción con la herramienta desarrollada.

Las variables lingüísticas involucradas en este proceso son: Influencia de las variables (I_v) y complejidad del requisito funcional (C_{rf}). La definición de estas variables se muestra en el Anexo 2.

Las variables que definen la complejidad de un Requisito funcional fueron identificadas a partir del estudio de varios métodos de estimación de software que tuviesen en cuenta la complejidad de los requisitos funcionales de software[18, 19, 29]. Las variables que han sido definidas para el análisis de la complejidad de los requisitos funcionales se encuentran en el Anexo 3. Las mismas se muestran en la herramienta desarrollada para todos los proyectos de desarrollo de software.

En la fase de valoración: Se automatizan todas sus actividades en la herramienta COMREQ, permitiendo la importación de los requisitos funcionales, estos requisitos son los correspondientes al excel “Criterios para validar requisitos del producto”, del mismo se escogerá la columna que lleva por nombre Requisito, correspondiente a la tabla de Requisitos Aceptados, para ello se especificará la fila y columna donde comienza la información del primer requisito. Una vez que se hayan importado los requisitos funcionales se brinda la posibilidad de introducir el criterio de cada

experto involucrado en la actividad de determinar la complejidad de estos requisitos funcionales, tal y como se muestra en el Anexo 4.

En la fase de determinación: Se automatizan todas sus actividades en la herramienta COMREQ, para ello se utilizan las ecuaciones relacionadas con el cálculo de la influencia de las variables y la determinación de la complejidad de los requisitos funcionales, así como los términos lingüísticos y la escala de pesos a tener en cuenta para el cálculo de la influencia de las variables y la determinación de la complejidad, tal y como se muestra en el Anexo 2. La ejecución de estas actividades se realiza internamente en la herramienta. La información de los criterios introducidos por cada experto para cada variable del requisito se muestran al presionar la opción de calcular complejidad, tal y como se muestra en el Anexo 4.

En la fase de transformación: Se automatizan todas sus actividades en la herramienta COMREQ, se muestra al usuario el resultado final de la complejidad del requisito, mediante la transformación del resultado obtenido en la fase de determinación a un término lingüístico, este estará basado en uno de los términos lingüísticos definidos para la variable lingüística "*Complejidad del requisito funcional",* tal y como se muestra en el Anexo 2*.*

Para la aplicación de la Guía metodológica se seleccionaron un grupo de profesionales relacionados con el área de Administración de requisitos y encargados por su rol de realizar la actividad de determinar la complejidad de los requisitos funcionales de software, esto se tuvo en cuenta como parte de la técnica de Iadov para valorar la satisfacción de los usuarios involucrados en este proceso.

3.1 COMREQ: Herramienta para la determinación de la complejidad de los requisitos funcionales de software

La herramienta desarrollada posibilita la determinación de la complejidad de los requisitos funcionales de software en los proyectos de desarrollo de la UCI, teniendo como entrada la información relacionada con las descripciones y especificaciones de los requisitos funcionales. Se obtiene como salida la complejidad de cada uno de los requisitos funcionales de software. Con el desarrollo de esta herramienta se logra el procesamiento de todos los criterios introducidos por parte de los expertos, así como la normalización de los pesos asignados a las variables y las transformaciones necesarias para dar el resultado final en un término lingüístico. En el Anexo 4 se muestran imágenes de la herramienta.

La herramienta se ejecutó procesando la información de 4 proyectos cerrados, además se ejecutó con la información de un grupo de requisitos de proyectos

de desarrollo actualmente en ejecución, para los cuales se tuvo en cuenta la complejidad a partir de los resultados obtenidos en la herramienta desarrollada, en el Anexo 5 se muestra un fragmento de los datos procesados.

4. Cuasi experimento

Se escoge este tipo de experimento puesto que no es posible realizar un experimento puro, los cuasi experimentos se diferencian de los experimentos en que la asignación de participantes a los grupos no se hace aleatoriamente, ni por emparejamiento. Ocurre cuando los grupos están previamente confeccionados (grupos intactos) [56]. Para el diseño cuasi experimental en la presente investigación se tendrá en cuenta un estudio del antes y el después en proyectos de desarrollos previamente seleccionados, este estudio establece una comparación previa a la creación del método y a la Guía metodológica y otra posterior aplicando estos últimos.

Figura 10. Información de la evaluación de requisitos y aplicación de la Guía metodológica (elaboración propia).

Como resultado de la herramienta informática se obtuvieron los resultados en cuanto a la complejidad de los requisitos funcionales de software de 4 proyectos de desarrollo previamente seleccionados, teniendo en cuenta los siguientes elementos:

- Proyectos cerrados.
- Representación de los diferentes tipos de proyectos que se desarrollan en la universidad (Portales, Sistemas Operativos, Sistemas de Gestión, Componentes y Tecnologías base).
- Información relacionada con los productos de trabajo: (Descripción de requisitos, Descripción ágil del requisito, Especificación de requisitos y Evaluación de requisitos).
- Información de la complejidad de los requisitos funcionales mediante el producto de trabajo Evaluación de requisitos.
- Información de la complejidad real de los requisitos funcionales una vez terminado el desarrollo de estos proyectos.

Al contar con lo anteriormente expuesto se plantea para el cuasi experimento el uso del método propuesto como un clasificador y comparar los resultados utilizando la medida de efectividad. Los clasificadores son sistemas que tienen como objetivo la clasificación de objetos de acuerdo a un grupo de clases definidas a partir de información que tienen almacenada.

El método definido puede funcionar como un clasificador considerando que su objetivo es brindar la complejidad de los requisitos funcionales, a partir de buscar el nivel de cercanía respecto a los términos lingüísticos de la variable lingüística "*Complejidad del requisito funcional",* empleando para ello los resultados obtenidos en la herramienta desarrollada. En este sentido se pueden homologar los componentes de los sistemas de clasificación con los elementos que definen a un clasificador:

- Las clasificaciones en términos lingüísticos de la complejidad constituyen las clases en las que se clasifica al Requisito funcional.
- Los requisitos constituyen los casos u objetos a clasificar.
- Para buscar la cercanía en el método se utiliza la transformación en 2-tuplas mediante la definición de traslación simbólica, por lo que pudiera suponer que es el motor de inferencia del clasificador.

Para la evaluación y validación de un clasificador se utilizan casos cuya clase es conocida, se establece la medida de efectividad a partir de los casos clasificados correctamente:

$$Efectividad = \frac{CC}{Total} \times 100 \qquad (9)$$

donde CC: son los casos clasificados correctamente por el clasificador, empleando los resultados de la herramienta COMREQ (CU), *Total* es el total de casos. Se determina la efectividad del clasificador y se analizan los resultados en función de validar el indicador definido para el método propuesto.

4.1 Descripción del diseño para el cuasi experimento

Como primer paso se analizan los requisitos que serán clasificados teniendo en cuenta los 4 proyectos de desarrollo seleccionados para el experimento, estos proyectos corresponden a cuatro centros de desarrollo. Los centros de desarrollo se diferencian en el campo de aplicación de los proyectos que desarrollan, es por ello que los proyectos seleccionados tributan a los diferentes tipos de proyectos, representando de esta forma al resto de los centros que tienen igual tipo de desarrollo y no se encuentran en la muestra. Considerando estos elementos, se valida que el comportamiento del método como resultado del cuasi experimento en

un proyecto de desarrollo de un tipo específico puede ser considerado similar en el resto de los proyectos de igual desarrollo.

Se tiene la información de los requisitos funcionales que fueron clasificados en cuanto a la complejidad según lo descrito en el modelo para la evaluación por complejidad de los requisitos funcionales de software que se aplica en el producto de trabajo Evaluación de requisitos correspondiente al expediente de proyectos de desarrollo [29, 55], además se cuenta con la información de la complejidad real de los requisitos funcionales una vez terminado el desarrollo de los proyectos seleccionados, en la tabla 3 aparece la información por proyectos de los requisitos y la clasificación por complejidad teniendo en cuenta lo anteriormente planteado.

Centro	**Proyecto**	**Requisitos funcionales**	**Complejidad "Evaluación de requisitos"**			**Complejidad real**		
			Alta	**Media**	**Baja**	**Alta**	**Media**	**Baja**
CESOL	NovaDesk: Sistema para la Gestión de Soporte	69	32	28	9	42	18	9
CIDI	Portal Web Empresa Ómnibus Nacionales	63	4	56	3	7	51	5
DATEC	Mercado de Datos-SAPMI	9	3	6		2	7	
CEIGE	Sistema automatizado de la Gestión Bancaria (Quarxo) Fase 2	46	23	12	11	28	10	8

Tabla 3. Información de la complejidad de los requisitos por proyectos de desarrollo seleccionados para el cuasi experimento (elaboración propia).

Para el desarrollo del cuasi experimento, en función de validar la efectividad del clasificador se procesaron los datos en la herramienta de los cuatro proyectos, con el objetivo de obtener las clasificaciones en cuanto a la complejidad de los requisitos funcionales, los resultados obtenidos se muestran en la tabla 4.

Centro	Proyecto	Requisitos funcionales	Complejidad				
			Muy baja	Baja	Media	Alta	Muy alta
CESOL	NovaDesk: Sistema para la Gestión de Soporte	69		8	16	41	5
CIDI	Portal Web Empresa Ómnibus Nacionales	63	2	4	50	7	
DATEC	Mercado de Datos-SAPMI	9			7	2	
CEIGE	Sistema automatizado de la Gestión Bancaria (Quarxo) Fase 2	46	2	8	9	23	4

Tabla 4. Información de los requisitos por proyectos de desarrollo teniendo en cuenta la aplicación de la Guía metodológica de la presente investigación (elaboración propia).

Teniendo en cuenta estos elementos se determina la efectividad del CU en busca de la cercanía del método propuesto como clasificador con respecto a los resultados reales de los proyectos al concluir su desarrollo en cada una de las clasificaciones en cuanto a la complejidad de los requisitos funcionales. Para el cálculo de la efectividad se tomarán los resultados de las clasificaciones de Alta, Media y Baja, partiendo de que los resultados obtenidos en las clasificaciones Muy baja y Muy alta para los proyectos escogidos no son de gran impacto con relación al total de requisitos, por lo que se considera que la no elección de estas clasificaciones no afecta el objetivo que tiene el cálculo de la efectividad.

A continuación se muestran los resultados del cálculo de la efectividad y la comparación mediante gráficas de los resultados alcanzados al determinar la complejidad de los requisitos funcionales mediante el producto de trabajo Evaluación de requisitos y el método propuesto, con respecto al real.

Cálculo de la efectividad de CU

Proyecto: NovaDesk: Sistema para la Gestión de Soporte.

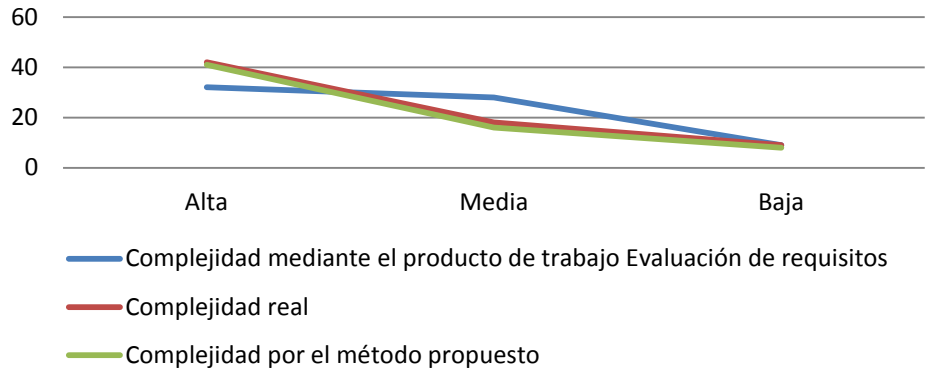


Figura 11. Resultados alcanzados mediante la Evaluación de requisitos y el método propuesto respecto al resultado real de la complejidad (elaboración propia).

- Como resultado para CU se tiene según la clasificación de Baja:

  CC=8 para un Total = 9 y una ***Efectividad = 88,8***

- Como resultado para CU se tiene según la clasificación de Media:

  CC=16 para un Total = 18 y una ***Efectividad = 88,8***

- Como resultado para CU se tiene según las clasificación Alta:

  CC=41 para un Total = 42 y una ***Efectividad = 97,6***

Proyecto: Portal Web Empresa Ómnibus Nacionales

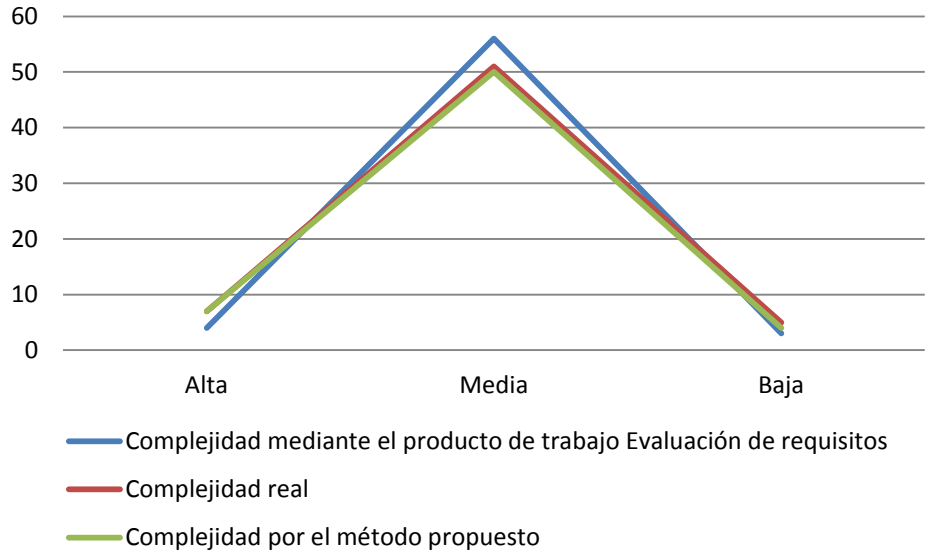


Figura 12. Resultados alcanzados mediante la Evaluación de requisitos y el método propuesto respecto al resultado real de la complejidad (elaboración propia).

- Como resultado para CU se tiene según la clasificación de Baja:

  CC=4 para un Total = 5 y una ***Efectividad = 80***

- Como resultado para CU se tiene según la clasificación de Media:

  CC=50 para un Total = 51 y una ***Efectividad = 98***

- Como resultado para CU se tiene según las clasificación Alta:

  CC=7 para un Total = 7 y una ***Efectividad = 100***

Proyecto: Mercado de Datos-SAPMI

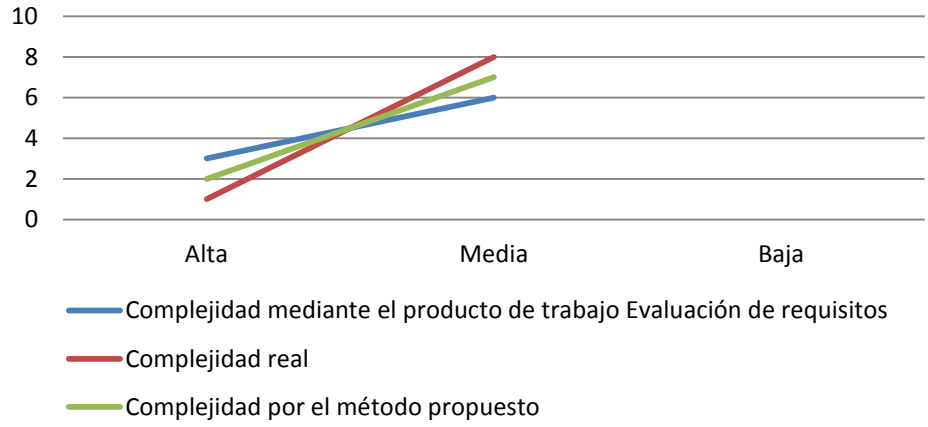


Figura 13. Resultados alcanzados mediante la Evaluación de requisitos y el método propuesto respecto al resultado real de la complejidad (elaboración propia).

- Como resultado para CU se tiene según la clasificación de Media:

  CC=7 para un Total = 7 y una ***Efectividad = 100***

- Como resultado para CU se tiene según la clasificación de Alta:

  CC=2 para un Total = 2 y una ***Efectividad = 100***

Proyecto: Sistema automatizado de la Gestión Bancaria (Quarxo) Fase 2

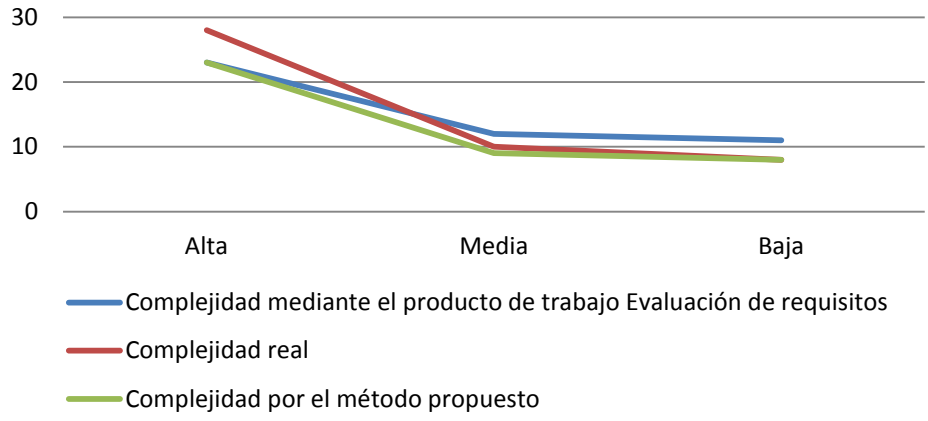


Figura 14. Resultados alcanzados mediante la Evaluación de requisitos y el método propuesto respecto al resultado real de la complejidad (elaboración propia).

- Como resultado para CU se tiene según la clasificación de Baja:

  CC=8 para un Total = 8 y una ***Efectividad = 100***

- Como resultado para CU se tiene según la clasificación de Media:

  CC=9 para un Total = 10 y una ***Efectividad = 90***

- Como resultado para CU se tiene según las clasificación Alta:

  CC=23 para un Total = 28 y una ***Efectividad = 82***

4.1.1 Análisis de los resultados

A partir de los resultados obtenidos se realiza el análisis del comportamiento del método mediante la efectividad del clasificador CU y la comparación de los resultados alcanzados en dos momentos diferentes:

- La efectividad CU se comportó en todos los casos por encima de un 80 % para cada una de las clasificaciones, lo que evidencia una coincidencia entre los resultados reales del proyecto y el método propuesto en la presente investigación, lo cual se considera un comportamiento satisfactorio. Este comportamiento se debe en parte a la capacidad del método de procesar varios criterios de expertos y de poder ejecutarse por diferentes tipos de proyectos.
- Los resultados alcanzados al comparar los dos momentos en los que fue determinada la complejidad demostraron que el método y la Guía metodológica propuestos dieron resultados más próximos a la realidad de los proyectos.
- Se logra la disminución de errores en la estimación de la planificación en los proyectos de desarrollo de software al obtener resultados de la complejidad de los requisitos funcionales más próximos a la realidad de los proyectos.

5. Validación de satisfacción del usuario. Aplicación de la Técnica de Iadov.

Conocer el estado de satisfacción del usuario respecto al método y a la Guía metodológica propuestos es de gran utilidad para la validación de la presente investigación. Iadov es una técnica que permite el estudio del grado de satisfacción de los involucrados en un proceso o actividad objeto de análisis. Esta técnica ha sido ampliamente utilizada por su carácter genérico [57]. La técnica está conformada por cinco preguntas: tres cerradas y dos abiertas las cuales son reformuladas en la presente investigación para evaluar la satisfacción de los expertos sobre el método y la Guía metodológica. A partir de las preguntas se

conforma el “Cuadro Lógico de Iadov” que establece la relación entre las preguntas cerradas, indicando la posición de cada persona en la escala de satisfacción.

¿Las salidas del método, complejidad del requisito satisface sus necesidades relacionadas con los tema de planificación?	¿Considera usted que se debe continuar realizando la determinación de la complejidad de los requisitos funcionales por el método anterior?								
	No			No sé			Sí		
	¿Utilizaría usted el método y la Guía metodológica propuestos para la determinación de la complejidad de los requisitos funcionales en los proyectos de desarrollo de software?								
	Si	No Sé	No	Si	No Sé	No	Si	No Sé	No
Me gusta mucho	1	2	6	2	2	6	6	6	6
No me gusta mucho	2	2	3	2	3	3	6	3	6
Me da lo mismo	3	3	3	3	3	3	3	3	3
Me disgusta más de lo que me gusta	6	3	6	3	4	4	3	4	4
No me gusta nada	6	6	6	6	4	4	6	4	5
No sé qué decir	2	3	6	3	3	3	6	3	4

Tabla 5. Cuadro Lógico de Iadov. Modificado por la autora.

El número resultante de la interrelación de las tres preguntas indica la posición de cada cual en la escala de satisfacción siguiente:

1. Clara satisfacción
2. Más satisfecho que insatisfecho
3. No definida
4. Más insatisfecho que satisfecho
5. Clara insatisfacción
6. Contradictoria

La información relacionada con 40 requisitos funcionales correspondientes a diferentes tipos de proyectos se introdujo en la aplicación para obtener la complejidad de los mismos. Estos resultados se mostraron a un grupo de usuarios responsables de determinar la complejidad de estos requisitos funcionales. La complejidad de estos requisitos se definió tomando los resultados del método como criterio y siguiendo las actividades de la Guía metodológica.

Para medir el grado de satisfacción se tomó una muestra de 32 usuarios, teniendo en cuenta los años de experiencia en la producción, en el área de Administración de requisitos y como analista. A partir de esto se aplicó la técnica de Iadov para medir el nivel de satisfacción de los usuarios respecto a los resultados arrojados por

la herramienta. El resultado de la evaluación de la satisfacción individual fue el siguiente, según la escala de satisfacción:

Nivel de satisfacción	Cantidad
Clara satisfacción	25
Más satisfecho que insatisfecho	4
No definida	2
Más insatisfecho que satisfecho	1

Tabla 6. Resultados de la aplicación de la técnica de Iadov (elaboración propia).

Para obtener el índice de satisfacción grupal (ISG) se procesan los criterios de las personas de acuerdo a los niveles de satisfacción que se expresan en la escala numérica que oscila entre +1 y – 1 de la siguiente forma:

- + 1 Máximo de satisfacción
- 0,5 Más satisfecho que insatisfecho
- 0 No definido y contradictorio
- - 0,5 Más insatisfecho que satisfecho
- - 1 Máxima insatisfacción

La ISG se calcula por la siguiente fórmula:

$$ISG = \frac{A\,(+1) + B\,(+0{,}5) + C\,(0) + D\,(-0{,}5) + E\,(-1)}{N} \quad (10)$$

En esta fórmula *A, B, C, D, E,* representan el número de sujetos con índice individual 1; 2; 3 ó 6; 4; 5 y donde N representa el número total de sujetos del grupo.

Los valores que se encuentran comprendidos entre - 1 y - 0,5 indican insatisfacción; los comprendidos entre - 0,49 y + 0,49 evidencian contradicción y los que están entre 0,5 y 1 indican que existe satisfacción.

En este caso ISG fue de 0,83 lo que representa una satisfacción de los participantes con el método y la Guía metodológica propuestos.

El Iadov contempla además dos preguntas complementarias de carácter abierto.

1. ¿Considera útil la posibilidad de determinar la complejidad de los requisitos a partir de sus descripciones y especificaciones como criterio de entrada?
2. ¿Qué elemento usted adicionaría al método o a la Guía metodológica que se propone?

La principal recomendación de los usuarios radicó en la posibilidad de introducir nuevas variables que definan la complejidad del requisito a partir del análisis que se haga del requisito. Se considera de muy útil la posibilidad de valorar los diferentes criterios de expertos, así como la existencia de la Guía metodológica al brindar la posibilidad de procesar la información que se obtiene a partir de las descripciones y especificaciones de los requisitos funcionales, permitiendo estimar la complejidad de los requisitos funcionales de software con un mayor grado de cercanía al juicio humano.

La aplicación de la técnica de Iadov arrojó resultados satisfactorios que validan la propuesta realizada de acuerdo a la satisfacción de los usuarios, además de que fueron considerados los criterios expresados para introducir mejoras al método y a la Guía metodológica.

6. Conclusiones del capítulo

- Se aplicó la Guía metodológica, validada mediante el desarrollo de una herramienta informática que permitió probar los operadores y transformaciones definidos, desarrollar un cuasi experimento con los resultados, así como determinar el nivel de satisfacción de los usuarios mediante la técnica de Iadov.
- Se valida la disminución de errores a partir del comportamiento de la efectividad de CU y la comparación de los resultados alcanzados en dos momentos diferentes respecto a los resultados reales en los proyectos de desarrollo seleccionados para el cuasi experimento.
- Se valoran positivamente los resultados que brinda el método en función de lograr una correcta estimación en cuanto a la complejidad de los requisitos funcionales de software.

CONCLUSIONES GENERALES

- El estudio de la bibliografía consultada arrojó que la principal insuficiencia de los métodos de estimación de software que tienen en cuenta la complejidad de los requisitos funcionales radica en que no proponen técnicas que gestionen la incertidumbre de la información para realizar el análisis de la complejidad de los mismos.
- Las variables que definen la complejidad de los requisitos funcionales de software resultaron de gran utilidad para la elaboración del método propuesto.
- Las variables lingüísticas, los operadores y las transformaciones propuestas, teniendo en cuenta el modelo lingüístico 2-tuplas permiten el trabajo de la incertidumbre en la complejidad de los requisitos funcionales y arrojan una interpretación cualitativa de los resultados de las operaciones.
- El método propuesto permite la determinación de la complejidad de los requisitos funcionales de software a partir de operadores de agregación y transformaciones en términos lingüísticos que permiten la integración de las variables lingüísticas, variables que definen la complejidad de los requisitos funcionales y pesos asociados al nivel de importancia de estas últimas.
- La Guía metodológica obtenida brinda el marco para la correcta interpretación del método propuesto a partir de la implementación de sus fases y actividades.
- Se valida el método y la Guía metodológica propuestos mediante la aplicación de métodos que corroboran la disminución de errores en la estimación de la planificación al realizar correctamente la determinación de la complejidad de los requisitos funcionales de software.
- La herramienta informática desarrollada apoya la determinación de la complejidad de los requisitos funcionales, permitiendo la implementación de los operadores y las transformaciones propuestas en el método y las actividades y fases que fueron definidas en la Guía metodológica.

RECOMENDACIONES

Se recomienda para la continuidad de la investigación y particularmente para la mejora del método propuesto:

- Incorporar al área de proceso de Administración de requisitos el método y la Guía metodológica definidos en la presente investigación, permitiendo la extensión de estos a todos los centros de desarrollo de la UCI.
- Brindar la posibilidad de introducir nuevas variables que definan la complejidad del requisito funcional a partir del análisis que se haga del mismo.
- Continuar el estudio del método en función de evaluar otros modelos para determinar la complejidad de los requisitos funcionales de software.

REFERENCIAS BIBLIOGRÁFICAS

1. Michael, A.C., "*La ingeniería de requerimientos y su importancia en el desarrollo de proyectos de software".* InterSedes, 2006. VI: p. 4-5.
2. Boston, M. *"New Standish Group report shows more project failing and less successful projects"*. 2009.
3. Dominguez, J., *"The curious case of the CHAOS Report 2009"*. 2009: Project Smart.co.uk.
4. Group, T.S., "*CHAOS MANIFESTO 2013, think big, act small".* 2013.
5. SEI, "*CMMI para Desarrollo, Versión 1.3"*. 2010.
6. Pressman, R., S. , "*Ingeniería del Software: Un enfoque práctico"*, ed. S.e.E.M. Hill. 2006, México DF.
7. Sommerville, "*Ingeniería del Software"*, ed. S. Edición. 2009, Madrid, España.
8. Wiegers, K.E. "*More About Software Requirements".* Thorny Issues and Practical Advice 2006.
9. Engineers, T.I.o.E.a.E., "*IEEE Standard Glossary of Software Engineering Terminology"*. 1990, lEEE Std 610.121990: 345 East 47th Street, New York, NY 10017, USA.
10. Young, R., "*The Requirements Engineering Handbook"*. 2004, Boston, London. 275.
11. Durrillo, J.Z., Y, *"A study of Multi-Objetive Next Release Problem".* In Proceedings of the 2009 1st International Symposium on Search Based Software Engineering. IEEE Computer Society, 2009: p. 48-58.
12. Wiegers, K., "*Software Requirements".* Microsoft Press, 1999.
13. Aurum, A.W., C, *"The fundamentals nature of requirements engineering activities as a decision making-making process".* Information and Software Technology, 2003: p. 945-954.
14. Berander, P., *"Prioritization of Stakeholder Need in Software Engineering"*, in *Department of Systems and Software Engineering*. 2004, Blekinge Institute of Technology.
15. Motupally, P., *"Using Satisfaction Arguments and Rich Traceability in Requirement Priorization"*. 2008, Auckland University of Technology.
16. Kushwaha, D.S.a.M., A.K., "*A Complexity Measure based on Information Contained in Software".* Proceedings of the 5th WSEAS Int. Conf. on Software Engineering, Parallel and Distributed Systems, 2006: p. 187-195.

17. Labdelaoui, H., "*Métodos de Estimación de Tamaño Funcional del Software Aplicados a Enfoques de desarrollo".* 2001.
18. Ashish Sharma, D.S.K., "*A Complexity measure based on Requirement Engineering Document".* COMPUTER SCIENCE AND ENGINEERING, 2010. 1(1): p. 112-116.
19. Albrecht, A.J., *"Measuring Applications Development Productivity".* Proceedings of IBM Application Development Joint. SHARE/GUIDE Symposium, Monterey, CA, 1979.
20. Mateo, D.A.G., *"Estimación Temprana mediante Puntos de Función IFPUG" E_2IFP*. 2012, Universidad de Alcalá.
21. Busquelle, J.E., "*Análisis de Puntos de Función".* Lámpsakos, 2010. 4: p. 59-61.
22. ISO/IEC 14143-1, I.I., "*Information technology. Software measurement. Functional size measurement"*, in *Part 1: Definition of concepts*. 2007.
23. ISO/IEC 20926, I.I., "*IFPUG FSM Method. IFPUG functional size measurement method"*, in *Software and systems engineering - Software measurement. Functional Sizing Methods*. 2009.
24. ISO/IEC 20968, I.I., "*Mk II. Function Point Analysis"* in *Software engineering - Mk II Function Point Analysis - Counting Practices Manual*. 2002.
25. ISO/IEC 19761, I.I., "*COSMIC-FFP*, in *Software engineering. A functional size measurement method"*. 2003.
26. ISO/IEC 29881, I.I., "*Information technology - Software and systems engineering"*, in *FiSMA 1.1 functional size measurement method*. 2008.
27. ISO/IEC 24570, I.I., "*Software engineering - NESMA function size measurement method" version 2.1* in *Definitions and counting guidelines for the application of Function Point Analysis*. 2005.
28. Rodríguez, F.S., "*Planificación y Gestión de Sistemas de Información"*, in *Medida del Tamaño Funcional de Aplicaciones Software Autor*. 1999.
29. Pérez Teruel, K., Leyva Vásquez, Maikel Y., *"Modelo matemático y procedimiento para evaluación por complejidad de los requisitos software".* WERpapers, 2012.
30. Jin-Hsien, W. and H. Jongyun, "*A new version of 2-tuple fuzzy linguistic representation model for computing with words".* Fuzzy Systems, IEEE Transactions on, 2006. 14(3): p. 435-445.
31. Mendel, J.M., "*An architecture for making judgments using computing with words.* International Journal of Applied Mathematics and Computer Science", 2002. 12: p. 325-335.

32. Safarzadegan Gilan, S., M. Hassan Sebt, and V. Shahhosseini, "*Computing with words for hierarchical competency based selection of personnel in construction companies".* Applied Soft Computing, 2012: p. 860-871.
33. Zadeh, L.A., "*From computing with numbers to computing with words. From manipulation of measurements to manipulation of perceptions.* Circuits and Systems I: Fundamental Theory and Applications", IEEE Transactions on, 1999. 46(1): p. 105-119.
34. Zadeh, L.A., "*Nacimiento y evolución de la Lógica Borrosa, el soft computing y la computación con palabras: un punto de vista personal".* Psicothema, 1996. 8(2): p. 421-429.
35. Keukelaar, J., "*Topics in soft computing".* Royal Institute of Technology, Stockholm, 2002.
36. Verdegay, J.L., *De los conjuntos borrosos a la" Soft Computing".* 2005.
37. Zadeh, L.A., "*Soft computing and fuzzy logic".* Software, IEEE, 1994. 11(6): p. 48-56.
38. Zadeh, L.A., "*Some reflections on soft computing, granular computing and their roles in the conception, design and utilization of information/intelligent systems".* Soft Computing, 1998. 2(1): p. 23-25.
39. Zadeh, L.A., *Fuzzy sets.* "Information and control", 1965. 8(3): p. 338-353.
40. Dubois, D.J., *Fuzzy sets and systems: theory and applications*. Vol. 144. 1980: Academic press.
41. Merigó Lindahl, J.M., "*Nuevas extensiones a los operadores OWA y su aplicación en los métodos de decisión"*. 2009: Universitat de Barcelona.
42. Wierman, M.J., *An Introduction to the Mathematics of Uncertainty*. 2010, Centre for the Mathematics of Uncertainty.
43. Martinez, L., D. Ruan, and F. Herrera, "*Computing with words in decision support systems: an overview on models and applications".* International Journal of Computational Intelligence Systems, 2010. 3(4): p. 382-395.
44. Mendel, J.M., "*Type-2 fuzzy sets and systems: an overview".* Computational Intelligence Magazine, IEEE, 2007. 2(1): p. 20-29.
45. Türkşen, I.B., "*Type 2 representation and reasoning for CWW".* Fuzzy Sets and Systems, 2002. 127(1): p. 17-36.
46. Mendel, J.M., et al., "*What computing with words means to me".* IEEE Computational Intelligence Magazine, 2010. 5(1): p. 20-26.
47. Zadeh, L.A., *The Concept of a Linguistic Variable and Its Applications to Approximate Reasoning.* Part I, Information Sciences 8 (1975) 199-249, Part

II, Information Sciences 8 (1975) 301-357, Part III, Information Sciences 9, 1975.

48. Martínez López, L., "*Un nuevo modelo de representación de información linguistica basado en 2 tuplas para la agregración de preferencias linguisticas"*, in *Departamento de Ciencias de la Computación e Inteligencia Artificial*. 1999, Universidad de Granada: Granada.
49. Zadeh, L.A., "*The concept of a linguistic variable and its application to approximate reasoning—I".* Information sciences, 1975. 8(3): p. 199-249.
50. Bonissone, P.P. and K.S. Decker, "*Selecting uncertainty calculi and granularity: An experiment in trading-off precision and complexity".* arXiv preprint arXiv:1304.3425, 2013.
51. Martínez López, L., "*UN NUEVO MODELO DE REPRESENTACION DE INFORMACION LINGÜISTICA BASADO EN 2-TUPLAS PARA LA AGREGACION DE PREFERENCIAS LINGÜISTICAS"*, in *Departamento de Ciencias de la Computación e Inteligencia Artificial*. 1999, Universidad de Granada: Granada, España.
52. Rodríguez Dominguez, R.M., "*Un Nuevo Modelo para Procesos de Computación con Palabras en Toma de Decisión Lingüística"*, in *Departamento de Informática*. 2010, Universidad de Jaén: Escuela Politécnica Superior de Jaén, España.
53. Wang, J.-H. and J. Hao, "*A new version of 2-tuple fuzzy linguistic representation model for computing with words".* Fuzzy Systems, IEEE Transactions on, 2006. 14(3): p. 435-445.
54. Xu, Z., "*Interactive group decision making procedure based on uncertain multiplicative linguistic preference relations".* Systems Engineering and Electronics, Journal of, 2010. 21(3): p. 408-415.
55. EXCRIBA. *XABAL EXCRIBA Gestor de Documentos Administrativos v 3.0*. Available from: http://excriba.prod.uci.cu:8080/excriba/page/.
56. Grau, R., Correa, C., Rojas M., "*Metodología de la investigación" (Segunda Edición).* UNIVERSIDAD DE IBAGUÉ CORUNIVERSITARIA., 2004.
57. Rodríguez, D.A.L. and D.V.G. Maura, "*La técnica de Iadov: Una aplicación para el estudio de la satisfacción de los alumnos por las clases de educación física"* Revista Digital - Buenos Aires 2002. N° 47.

ANEXOS

Anexo 1: Encuesta para la validación por parte de los expertos de un conjunto de variables que se consideran a partir de estudios relacionados definen la complejidad de los requisitos funcionales de software.

Datos de interés

Centro _____

Cargo (si está ocupando alguno)_____

Proyecto_____

Rol_____

Años de experiencia en la producción _____

1. Seleccione de los elementos que se relacionan a continuación las que usted considere que influyen en la complejidad de los requisitos funcionales de software.
   La descripción de cada uno aparece al final de la encuesta.

   ___ Interfaces
   ___ Facilidad de cambio
   ___ Dependencia con otros requisitos
   ___ Requisitos asociados
   ___ Transacciones
   ___ Diferentes comportamientos
   ___ Restricciones

2. Considera usted que la determinación de la complejidad de los requisitos funcionales de software es un elemento importante a tener en cuenta para la planificación de los proyectos de desarrollo de software.

   Sí ___ No ___ No sé ___

3. Diga si considera que se deban tener en cuenta otras variables, en caso de ser positiva su respuesta.

   _____
   _____
   _____
   _____

Anexo 2: Elementos a tener en cuenta por la herramienta desarrollada a partir de la aplicación de la Guía metodológica.

Definición de las variables lingüísticas

Variable lingüísticas	**Descripción**	**Términos lingüísticos**
Influencia de la variable ***(I_v)***	Es la clasificación de los valores que puede tomar la influencia de las variables en los requisitos funcionales	S= {s_0= No influye, s_1= Muy Baja, s_2=Baja, s_3=Media, s_4=Alta}.
Complejidad del requisito funcional ***(C_{rf})***	Es la clasificación de los valores que puede tomar la complejidad de los requisitos funcionales	S= {s_0= Muy baja, s_1= Baja, s_2=Media, s_3=Alta, s_4= Muy Alta}.

Pesos asociados al nivel de importancia de las variables que definen la complejidad de un requisito funcional.

$$W = \{w_i ... w_n\} \quad W = \{1,2,3,4,5\}$$

Formulación a tener en cuenta en la herramienta desarrollada

- Cálculo de la influencia de una variable

$$I_v = \sum_{i=1}^{n} \beta_i \cdot w_i$$

Donde I_v representa la influencia de la variable que se esté analizando, β_i es el valor del término lingüístico y w_i es el peso asignado por cada experto, donde *n* representa la cantidad de expertos.

- Cálculo de la complejidad del requisito funcional

$$C_{rf} = \Delta\left(\frac{\sum_{i=1}^{m} \Delta^{-1}(r_i, a_i).w_i}{\sum_{i=1}^{m} w_i}\right) = \Delta\left(\frac{\sum_{i=1}^{m} I_{vi}}{\sum_{i=1}^{m} w_i}\right)$$

Donde C_{rf} es el valor de la complejidad del Requisito funcional a partir del análisis de cada una de las variables, $\sum_{i=1}^{m} I_{v_i}$ el total de la suma de la influencia de cada

una de las variables, $\sum_{i=1}^{m} w_i$ representa el peso total asignado a cada variable, *m* representa la cantidad de variables.

- Transformar resultados en términos lingüísticos

La complejidad es una información cualitativa que se clasificará teniendo en cuenta el conjunto de términos lingüísticos definidos.

Para transformar los resultados en términos lingüísticos se hará uso de la representación en 2-tuplas $(s_i, \alpha,)$ donde:

- s_i representa el término lingüístico.
- α_i es un número que expresa el valor de la distancia desde el resultado original al índice del término lingüístico s_i más cercana en el conjunto de términos lingüísticos S, es decir, su traslación simbólica.

Anexo 3: Variables que definen la complejidad de los requisitos funcionales de software.

Interfaces: Se aplica a requisitos que presenten algún tipo de complejidad en su interacción con los siguientes elementos, considerados subcomponentes. Humanas(Formularios,informes)
Equipo(Tomógrafos,RayosX)
Programación(Programas externos necesarios para apoyar el producto) Comunicación(Protocolos de comunicación que serán utilizados) Atributos(Se refiere a la complejidad que puede agregarle la cantidad de atributos contenidos en una interfaz)

Facilidad de cambio: Esfuerzo específico de diseño e implementación del requisito para facilitar cambios futuros durante el desarrollo del producto.

Dependencia con otros requisitos: Describe el grado de relación de un requisito con otros, según la cantidad de requisitos con los que guarde algún tipo de dependencia para su implementación.

Requisitos asociados: Se refiere a la descomposición en sub-funciones de un requisito más general y que no guardan dependencia entre sí.

Transacciones: Es la interacción con una estructura de datos compleja, compuesta por varios procesos que se han de aplicar uno después del otro.

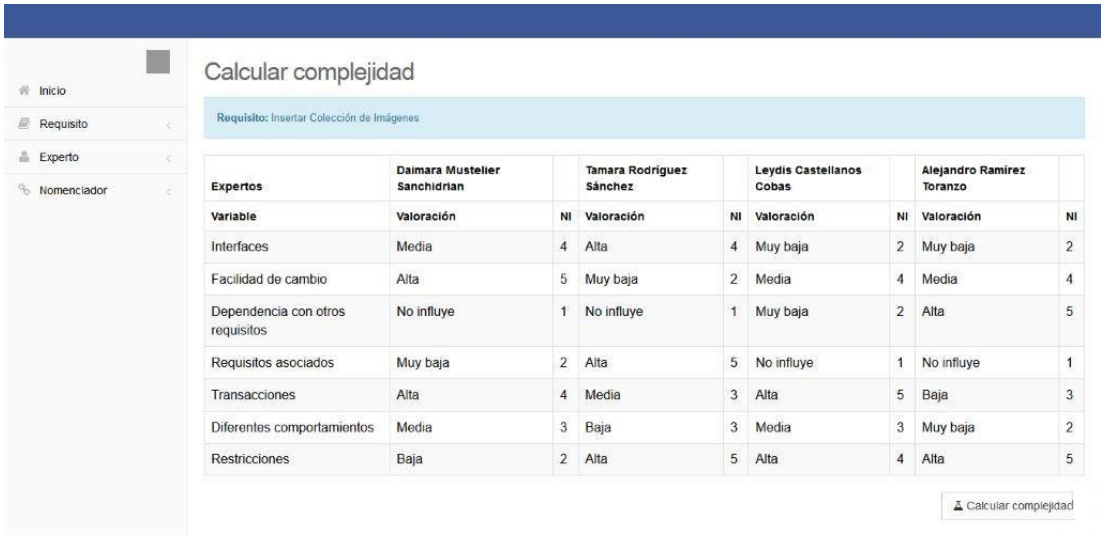
Diferentes comportamientos: Se refiere al comportamiento que puede tener el requisito ante determinadas situaciones, en dependencia de ello se recogerá más información.

Restricciones: Se refiere a los requisitos asociados a las restricciones de diseño, implementación, interfaces, físicas y reglas de negocio.

> **Restricciones de diseño**: limitar el diseño y declarar los requisitos sobre el enfoque que debe tenerse en cuenta en el desarrollo del sistema.
>
> **Restricciones de implementación**: poner límites al proceso de generación de código o de construcción (estándares requeridos, lenguajes, herramientas o plataforma)
>
> **Restricciones de interfaz**: son requerimientos para interaccionar con sistemas externos, describiendo los protocolos o la naturaleza de la información que debe ser transferida a través de la interfaz.
>
> **Restricciones físicas**: afectan el hardware o el empaquetado del sistema (forma, tamaño y peso)
>
> **Reglas del negocio**: son las políticas, normas, estatutos, acuerdos, resoluciones o cualquier tipo de decisión que gobierna la forma en que la institución opera. Ellas restringirán los pasos descritos en el flujo del caso de uso.

Anexo 4: COMREQ: Herramienta informática desarrollada.

Figura 15. Imagen de la herramienta una vez importados los requisitos.



Expertos	Daimara Mustelier Sanchidrian		Tamara Rodriguez Sánchez		Leydis Castellanos Cobas		Alejandro Ramirez Toranzo	
Variable	**Valoración**	**NI**	**Valoración**	**NI**	**Valoración**	**NI**	**Valoración**	**NI**
Interfaces	Media	4	Alta	4	Muy baja	2	Muy baja	2
Facilidad de cambio	Alta	5	Muy baja	2	Media	4	Media	4
Dependencia con otros requisitos	No influye	1	No influye	1	Muy baja	2	Alta	5
Requisitos asociados	Muy baja	2	Alta	5	No influye	1	No influye	1
Transacciones	Alta	4	Media	3	Alta	5	Baja	3
Diferentes comportamientos	Media	3	Baja	3	Media	3	Muy baja	2
Restricciones	Baja	2	Alta	5	Alta	4	Alta	5

Figura 16. Imagen con cada criterio dado por los expertos para el requisito "Insertar Colección de Imágenes".

Anexo 5: Procesamiento de datos en la herramienta.

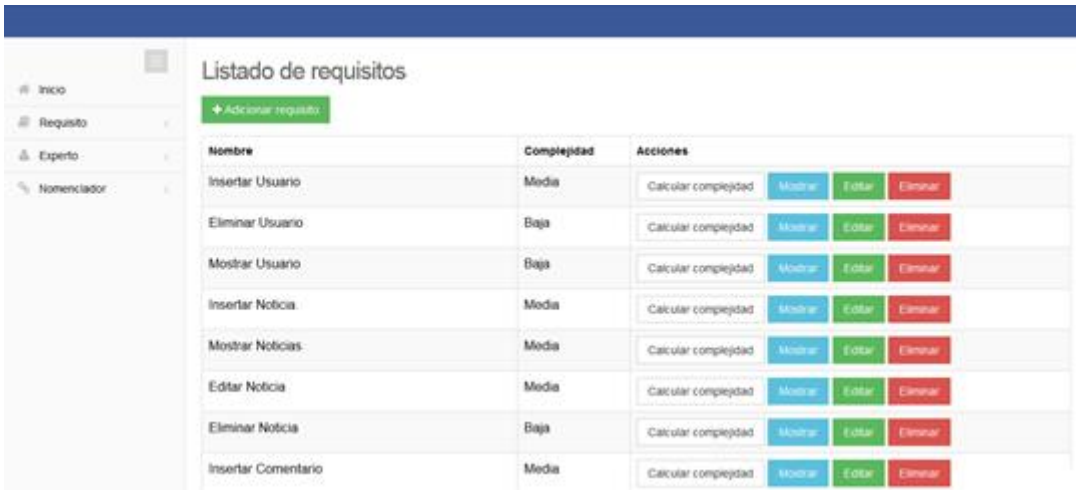


Figura 17. Fragmento de los datos procesados del proyecto "Portal Web Empresa Ómnibus Nacionales".

Figura 18. Fragmento de los datos procesados del proyecto "Sistema automatizado de la Gestión Bancaria (Quarxo) Fase 2".

Figura 19. Fragmento de los datos procesados de uno de los proyectos en ejecución "Sistema de Gestión de Importación y Exportación Bk Import/Export".