

**REFLECTION ON THE REQUIREMENTS SPECIFICATION IN A
UNIVERSITY CONTEXT****REFLEXIÓN SOBRE LA ESPECIFICACIÓN DE REQUERIMIENTOS
DENTRO DE UN CONTEXTO UNIVERSITARIO****MSc. Alveiro Alonso Rosado Gómez, Ing. Alejandra Verjel Ibañez**

Universidad Francisco de Paula Santander - Sede Ocaña, Facultad de ingenierías.
Vía Acolsure, Sede el Algodonal, Ocaña, Norte de Santander, Colombia.
Tel.: (+57-7) 569 0088, Fax: (+57-7) 569 0088, Ext. 413.
Email {aorosadog, averjeli}@ufpso.edu.co

Abstract: This paper is aiming to deepen the concepts of engineering requirements (ER), seeking to know what alternatives techniques or forms there are for specification requirements, a process that is part of the development of Software. The Guide to the software engineering body of knowledge (SWEBOK) states that there are two types of requirements specifications; the specification of system requirements and software requirements specification, the first talk of global solutions for the organization and the second speaks of the development of requirements-level features, this paper only the software of specification of requirements was addressed.

Keywords: Software development, requirements specification, specification tools, requirements engineering, technical specification.

Resumen: Este artículo, aborda la especificación de requerimientos, a partir del marco teórico que sobre el mismo hacen los autores propuestos en los currículos de las materias de Análisis y Diseño de Sistemas e Ingeniería de Software, las cuales hacen parte del programa de Ingeniería de Sistemas de la Universidad Francisco de Paula Santander Ocaña. Los conceptos expuestos son verificados mediante la revisión de los proyectos de grado, que tienen como objetivo principal la construcción de un sistema o software. Desde luego, la academia tiene la responsabilidad de seguir buenas prácticas, éstas no se evidencian dentro del material estudiado, al encontrar que se siguen cometiendo los errores comunes que se advierten en la literatura y que en ella misma se indica cómo evitarlos; esto se demuestra al encontrar que los detalles de caso de uso son la práctica predominante de especificar requerimientos, pero como se encuentran escritos no obedecen a una notación estricta o completa a pesar que la forma de representarlos está escrita en lenguaje natural estructurado.

Palabras clave: Desarrollo de software, especificación de requerimientos, herramientas de especificación, Ingeniería de requerimientos, técnicas de especificación.

1. INTRODUCCIÓN

Uno de los problemas que enfrentan quienes trabajan en proyectos de desarrollo de software, es el de traducir de forma correcta y precisa las ideas

suministradas por los usuarios sobre sus expectativas y necesidades, de manera que tengan un significado claro para el cliente y los miembros del equipo de desarrollo.

La definición clara y precisa de requerimientos o requisitos, ayuda a disminuir el grado de incertidumbre para alcanzar el éxito del proyecto (Larman, 2003); la Ingeniería de Requisitos (IR), define una serie de especificaciones que describen en forma consistente y compacta, las necesidades de los usuarios. La IR establece el proceso de requerimientos, que cuenta con actividades como: levantamiento, análisis, especificación y verificación de requerimientos.

Este artículo comienza con la definición inicial de los requisitos, posteriormente muestra con cifras la rigurosidad con que se llevan a cabo estas especificaciones dentro de los trabajos de grado de los estudiantes del programa de Ingeniería de Sistemas, de la Universidad Francisco de Paula Santander Ocaña, con las cifras expuestas se hace una reflexión a partir de las características encontradas con respecto a algunas de las técnicas o formas que existen para especificar requerimientos.

2. MARCO TEÓRICO

Para reconocer la importancia que dentro del proceso de desarrollo del Software, tiene la identificación y definición de requerimientos, se debe tener claro qué son y cómo se pueden especificar. A continuación se muestran algunas definiciones que sobre el tema hacen distintos autores incluyendo la de libros de referencia para las materias de Análisis y Diseño e Ingeniería de Software.

2.1 Requerimientos o requisitos

Son capacidades y condiciones con las cuales debe cumplir un sistema (Jacobson *et al.*, 2000). La palabra requisito se utiliza en un sentido general, refiriéndose a “necesidades” del cliente (Pressman, 1997) colocados en un producto de software que contribuye a la solución de cierto problema (Abrañán, 2001).

Los requisitos del software son el origen de 20% de todos los errores de código o defectos en software y también de más de 30% de defectos realmente severos, difíciles de reparar. La tasa a la que cambian los requisitos después de la definición inicial varía entre 1% y más de 3% por mes (Capers, 2008). Éstos problemas, son independientes del modelo de proceso o ciclo de vida que se lleve (Saujanya *et al.*, 2014), este comportamiento no sólo pasa con los métodos tradicionales de desarrollo de software, algunos

procesos ágiles, que tienen como característica iniciar con la escritura de código tan pronto se entienden un conjunto mínimo de requisitos, tienen una tasa promedio de incremento de estos del 12% por mes calendario (Yu y Sharp, 2011).

2.2 Ingeniería de requisitos

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

La IR define siete tareas diferentes: concepción, indagación, elaboración, negociación, especificación, validación y administración de requerimientos. Estas tareas proporcionan un marco apropiado para entender lo que desea el cliente, analizar las necesidades, evaluar la factibilidad, negociar una solución razonable, especificar la solución sin ambigüedades, validar la especificación y administrar los requisitos (Maiden y Robertson, 2005).

2.3 Especificación de requerimientos

La *Guide to the software engineering body of knowledge* (Swebok), divide la especificación de requerimientos en dos, una es la especificación de requerimientos de sistema, que define los requerimientos desde el punto de vista de la organización como una solución total y no dividida por funcionalidades; por otra parte está la especificación de requerimientos de software, que establece la base para el acuerdo entre los clientes y los contratistas o los proveedores que deben hacer el producto de software, así como lo que no se espera que haga. Si bien el Swebok, es un referente mundial para el tema que se está estudiando, la literatura de referencia de este artículo, no aborda el tema de los requisitos del sistema.

Los requisitos del software se escriben a menudo en lenguaje natural, con una formalidad que va desde lo general a lo específico, identificándose como breve, informal o completa, dependiendo del nivel de detalle con que se redacten los requisitos. La selección de notaciones apropiadas permite que tanto requisitos como los aspectos particulares de la arquitectura del software se escriban más ajustadamente que en el lenguaje natural (Gervasi y Zowghi, 2005).

(Pressman, 2010) al igual que (Zave & Jackson, 1997), definen una Especificación de Requerimientos de Software (ERS), como un documento que contiene una descripción detallada de todos los aspectos del software que se van a elaborar, antes que el proyecto comience. Por otro lado, (Sommerville, 2005), propone varias formas diferentes de redactar los requerimientos del sistema en unas notaciones especializadas, como son: el lenguaje natural estructurado, lenguajes de descripción de diseño, notaciones gráficas y especificaciones matemáticas.

Un estudio realizado en el 2006 por el Carnegie Mellon Software Engineering Institute, en donde se compara métodos de especificación de requerimientos desde una perspectiva de la arquitectura, define que existen cinco (5) métodos diferentes que sirven para el levantamiento y especificación de requerimientos (Bass *et al.*, 2006). Estos métodos son:

Especificación de requisitos uso del lenguaje natural: La especificación de requisitos, con frecuencia se realiza utilizando lenguaje natural. Una ventaja principal de utilizar lenguaje natural para especificar requisitos es que son comprensibles por los *stakeholders*¹ (Kamalrudin *et al.*, 2011).

Análisis de casos de uso: es utilizado para hacer el análisis de los actores, los cuales representan a personas y dispositivos que usen el sistema, el objetivo que se persigue es describir la interacción de ese actor con el sistema (Microsoft, 2013). Los casos de uso, también llamados fragmentos de funcionalidad, representan partes del sistema que aportan un resultado de valor para sus actores (Gelperin, 2005). Este análisis produce dos resultados, uno de ellos es el diagrama de casos de uso, el cual es una representación gráfica del software en lenguaje unificado de modelado (Unified Modeling Language, UML) (Fuentes y Vallecillo, 2004); el otro corresponde a un conjunto de especificaciones estructuradas llamadas detalle de caso de uso, el cual describe la interacción entre el usuario y el sistema dentro de unos escenarios que representan secuencias de pasos que lleva a cabo el usuario dentro de la funcionalidad (Arias, 2005). Estos artefactos permiten que los desarrolladores de software y los clientes lleguen a un acuerdo sobre las condiciones

y posibilidades que debe cumplir el sistema (Perdita y Pooley, 2007);

La calidad SEI Quality Attribute Workshop (QAW): busca el descubrimiento de atributos de calidad críticos del software en las fases iniciales de su desarrollo, esto se logra con talleres que relacionan a los *stakeholders* de un sistema de manera temprana en el ciclo de vida, para descubrir los atributos de calidad. Los talleres tienen una duración definida, y al final del taller ya se puede tener un refinamiento de los escenarios concretos del software (Sarkar *et al.*, 2012).

Análisis global: el propósito de este método es identificar y analizar los factores tecnológicos, organizacionales y de productos que tienen influencia en el entorno de la arquitectura.

Método de O'Brien: un planteamiento que vincula fuertemente las decisiones arquitectónicas con la calidad de los atributos que se manifiestan cuando los objetivos empresariales son capturados explícitamente.

Son diversas las formas de especificar requerimientos, a pesar de que existen métodos definidos y aceptados, no son del dominio general, teniendo como característica el país, el área geográfica o el sector de la organización en donde se esté llevando a cabo el desarrollo.

3. METODOLOGÍA

Para poder tener un contexto de la especificación de requerimientos dentro de la Universidad, se decidió trabajar de manera cuantitativa sobre los proyectos de grado de los estudiantes de Ingeniería de Sistemas en donde el objetivo principal del proyecto fuera el desarrollo de software. La selección de esta muestra se hace porque son los estudiantes de este programa los que de acuerdo a su plan de estudios desarrollan estos temas, dentro de los contenidos programáticos de las materias de análisis y diseño de sistemas e ingeniería de software.

Se tomó una muestra de 60 proyectos, en los cuales el resultado final fuera un sistema o un software (Gervasi y Zowghi, 2005). Por cada aplicación se evaluaron los conceptos teóricos que indican cómo debería ser la especificación de los requerimientos cuando se está desarrollando una aplicación; Se indagó por la forma de especificar los requerimientos; si fue en plantilla, detalles de caso

¹*Stakeholder* es un término inglés utilizado por primera vez por R. E. Freeman en su obra: "Strategic Management: A Stakeholder Approach", para referirse a quienes pueden afectar o son afectados por las actividades de una empresa.

de uso, historia de usuario, algún otro tipo o no se utilizó. También se evaluó la formalidad con que estaban escritos; si fueron breves, informales o completos (Larman, 2003).

Por otro lado se tuvo en cuenta la cantidad de especificaciones; entiéndase éstas como el número de descripciones que representan funcionalidades dentro del aplicativo, independiente de la forma en que se encuentre escrito; con éste número se pretendió mirar si existía una relación entre la formalidad aplicada y el número de especificaciones. Otro aspecto que también se observó, fue el del tipo de ciclo de vida del desarrollo, metodología o método que se utilizó para construir la aplicación (Capers, 2008); se miró si las fases que se llevaron corresponden a un método tradicional, evolutivo, ágil, algún otro o simplemente no se sigue ninguna referencia de este tipo (Yu y Sharp, 2011). Se indagó si existe una relación entre la especificación, su formalidad, el tipo de desarrollo y el tipo de concepción (Sistema o Software); es decir, si tuvieron en cuenta aspectos organizacionales o solo se limitaron a la aplicación (Gervasi y Zowghi, 2005). Por último cuando existían o no existían requerimientos definidos, cuál era la forma de presentarlo en el documento (Sommerville, 2005).

4. RESULTADOS

Con el análisis de los resultados obtenidos con la investigación de las tesis de grado de la Universidad Francisco de Paula Santander Ocaña, se puede concluir que de las formas de especificación de requerimientos que más se utiliza son el lenguaje natural estructurado dentro de los casos de uso, con un 91% y el 9% no utiliza ninguna forma de especificación; en cuanto a las otras formas de especificación como notaciones gráficas y especificaciones matemáticas se evidencia que no se utilizan en ningún caso, teniendo en cuenta que los tipos de desarrollo implementados el 88% son orientados a la web, frente a un 12% desarrollos de escritorio. Con respecto a la forma de especificación de requerimientos el 26% implementó un número de especificaciones mayor a 50, así mismo, la mitad utilizó detalles de caso de uso informal, el 34% formalidad completa y sólo el 18% una formalidad breve. En cuanto a la metodología, la más utilizada es la tradicional con un porcentaje de 86%, luego le siguen la metodología evolutiva con un 3% y la metodología ágil un 2%.

5. DISCUSIÓN

Con los resultados que se mostraron se puede afirmar que la forma de especificar requerimientos dentro del Universidad es el caso de uso, teniendo una notación informal o breve, estos resultados coinciden con los expuestos por (Yue *et al.*, 2013), quienes afirman que los casos de uso sufren de imprecisión, independientemente de seguir o no una plantilla; las cuales proporcionan notaciones en lenguaje natural estructurado coincidiendo con los resultados de (Parachuri *et al.*, 2014), dado que la especificación debe ser restringida mediante una serie de directrices, como lo afirma (Yue *et al.*, 2013), que disminuyan la ambigüedad para lograr especificaciones adecuadas y útiles.

El número de aplicaciones desarrolladas, están orientadas a plataforma web, pero al utilizar en el 34% de los proyectos la especificación completa, coincide con un número de especificaciones menores o iguales a 40; es posible que el autor buscara con esto acercarse a la programación y ejecución, con el fin de lograr un sistema agradable para el usuario final, cumpliendo sus expectativas (Hirschfeld *et al.*, 2012).

Con la información analizada se puede sugerir una falta de claridad sobre el tema de la especificación de requerimientos, es decir, no existe una uniformidad que certifique unos buenos resultados en el desarrollo de un sistema (Escalona y Koch, 2002); así mismo, se establece como de gran importancia un análisis de requisitos al momento de iniciar el software, ya que es fundamental una buena especificación de requerimientos para conseguir un producto de buena calidad (Orjuela *et al.*, 2011), debido a que un software de alta calidad es un software económico, fiable y eficiente (Barranco, 2001). Al no tener un método específico se debe saber elegir la forma más adecuada para realizar este análisis que es vital para las etapas de desarrollo (Tangarife *et al.*, 2014), para esto se realizó una comparación entre las diversas opciones que se encuentran disponibles.

6. CONCLUSIONES

Dentro de la literatura, no se hace diferencia entre especificación y levantamiento de requisitos. En libros impresos y digitales abordan este tema de forma diferente mezclando los requisitos con los casos de uso, lo cual produce una tendencia a creer que la única forma que existe para especificar

requerimientos es ésta, dejando de lado los documentos de requerimientos que son necesarios para delimitar de forma temprana el alcance del sistema o *Software*.

Entre las diferentes formas de especificación de requerimientos, el uso del lenguaje natural y el análisis de casos de uso, pueden ser una opción factible debido a que son más comprensibles por los *stakeholders*; en cambio, si se trata de no tener ambigüedad en la información, es recomendable utilizar especificaciones matemáticas; es decir, las formas de especificación de requerimientos deben ser elegidas según el diálogo entre analistas y clientes; al no existir un modelo definido para cada tipo de software, ya que las diversas formas disponibles tienen ventajas y desventajas, la decisión final la deben tomar los miembros del equipo de desarrollo y los *stakeholders*.

A pesar que los resultados confirman la tendencia generalizada de utilizar detalles de casos de uso para la especificación de requerimientos, la formalidad con que están escritos puede producir ambigüedad; dado que no reflejan por completo el alcance de la funcionalidad, esto puede producir subestimación en la complejidad del desarrollo. Para reducir esta ambigüedad, es necesario que el estudiante sea sensible de la importancia que tiene la formalidad en la especificación y de cómo puede afectar la calidad y los tiempos del desarrollo de sistemas o de software.

REFERENCIAS

- Abran, A. (2001). *Guide to the software engineering body of knowledge-SWEBOK*. IEEE Pres.
- Arias, M. (2005). La ingeniería de requerimientos y su importancia en el desarrollo de proyectos de software. *InterSedes: Revista de las Sedes Regionales*.
- Barranco, J. (2001). *Metodología del análisis estructurado de sistemas*. Univ Pontificia Comillas.
- Bass, L., et al. (2006). *A Comparison of Requirements Specification Methods from a Software Architecture Perspective*. <http://www.cmu.edu/index.shtml>: <http://repository.cmu.edu/sei/389/> (Consultado: 18 de noviembre 2014)
- Capers, J. (2008). *Estimación de costos y administración de proyectos de Software*. McGraw-Hill Interamericana.
- Escalona, M., y Koch, N. (2002). Ingeniería de Requisitos en Aplicaciones para la Web—Un estudio comparativo. *Universidad de Sevilla*.
- Fuentes, L., y Vallecillo, A. (2004). Una Introducción a los Perfiles UML . *Revista Novatica—Asociación de Técnicos de Informática-España*.
- Gervasi, V., y Zowghi, D. (2005). Reasoning about inconsistencies in natural language requirements. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, 277-330.
- Hirschfeld, R., et al. (2012). *Explicit use-case representation in object-oriented programming languages*. *ACM*, 51-60.
- Jacobson, I., et al. (2000). *El proceso unificado de desarrollo de software*. Reading: Addison Wesley.
- Kamalrudin, M., et al. (2011). Improving requirements quality using essential use case interaction patterns. *ICSE '11 Proceedings of the 33rd International Conference on Software Engineering* (pp. 531-540). ACM.
- Larman, C. (2003). *UML y patrones: una introducción al análisis y diseño orientado a objetos y al proceso unificado*. Madrid: Prentice Hall.
- Maiden, N., y Robertson, S. (2005). Developing use cases and scenarios in the requirements process. *ICSE '05 Proceedings of the 27th international conference on Software engineering* (pp. 561-570). New York: ACM.
- Microsoft. (2013). <http://msdn.microsoft.com/>. Retrieved Noviembre 19, 2014, from <http://msdn.microsoft.com/es-es/library/dd409432.aspx> (Consultado: 19 de noviembre 2014)
- Orjuela, A., et al. (2011). Planificación del alcance en proyectos de software. *Revista Colombiana de Tecnologías de Avanzada*.
- Parachuri, D., et al. (2014). *An empirical study of structural defects in industrial use-cases*. *ACM*, 14-23.
- Perdita, S., y Pooley, R. (2007). *Utilización de UML en ingeniería del software con objetos y componentes*. Madrid: Pearson Educación.
- Pressman, R. (1997). *Ingeniería del Software: Un enfoque práctico*. Mikel Angoar.
- Pressman, R. (2010). *Ingeniería del Software un Enfoque Practico*. Mexico, D.F: McGraw-Hill.
- Sarkar, S., et al. (2012). Creating design from requirements and use cases: bridging the gap between requirement and detailed design. *ISEC '12 Proceedings of the 5th India Software Engineering Conference*, pp. 3-12.

- Saujanya, V., et al. (2014). A framework for identifying and analyzing non-functional requirements from text. *TwinPeaks 2014 Proceedings of the 4th International Workshop on Twin Peaks of Requirements and Architecture*, 1-8.
- Sommerville, I. (2005). *Ingeniería del software*. Pearson Educación.
- Tangarife, L., et al. (2014). Modelo de interventoría de tecnologías de información en el área de conocimiento de la gestión del alcance de pmbok® y alineado con iso 21500 y cobit®. *Revista Colombiana de Tecnologías de Avanzada*.
- Yu, Y., y Sharp, H. (2011). Analysing requirements in a case study of pairing. *ACM New York, NY, USA, 2011*.
- Yue, T., et al. (2013). *Facilitating the transition from use case models to analysis models: Approach and experiments*. *ACM*, 5.
- Zave, P., y Jackson, M. (1997). Four dark corners of requirements engineering. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, 1-30.