Proceso y evolución de los métodos formales en la ingeniería de requisitos*

Jorge Hernán Suaza Jiménez[**]
Gloria Amparo Lora Patiño[***]

Recibido: 01/07/2019 • Aceptado: 26/11/2019
https://doi.org/10.22395/rium.v19n37a6

Resumen

La ingeniería de requisitos se considera la fase más importante del ciclo de vida de los productos de *software,* porque en ella se especifican las necesidades de los clientes y es la base para la ejecución de las demás fases de la ingeniería de *software*. Los modelos que actualmente se utilizan para realizar la elicitación de requisitos se han propuesto y documentado de forma amplia, pero se centran solo en las técnicas para colectar información y descuidan la actividad de documentarla de manera adecuada. Además, para estructurar la especificación de requisitos sigue en uso el lenguaje natural como forma de comunicación y comprensión con el cliente. Debido a las ambigüedades que este causa, se dificulta la interpretación; esto conlleva reprocesos en las etapas posteriores del ciclo de vida del *software*. De acuerdo con ello, es necesario que las organizaciones desarrolladoras de *software* consideren formalizar el proceso de elicitación de requisitos si desean hacer más eficiente su proceso de desarrollo. En este artículo se hace una revisión de literatura para determinar el proceso y evolución de los métodos formales desde la perspectiva de la ingeniería de requisitos.

Palabras clave: métodos formales; ingeniería de requisitos; calidad del software; lenguaje matemático.

* Artículo derivado del proyecto de investigación "Diseño de un modelo semiformal para documentar la elicitación de requisitos", identificado con el código interno P14117 del Instituto Tecnológico Metropolitano, Medellín. Año de inicio: 2014. Año de finalización: 2015.

** Magíster en ingeniería de *software* e ingeniero informático. Profesor investigador del Instituto Tecnológico Metropolitano, Facultad de Ingeniería, Grupo de Investigación Automática, Electrónica y Ciencias Computacionales. Correo electrónico: jorgesuaza@itm.edu.co. Orcid: https://orcid.org/0000-0002-9669-8145.

*** Magister en seguridad informática. Instituto Tecnológico Metropolitano. Correo electrónico: glorialora231287@correo.itm.edu.co. Orcid: http://orcid.org/0000-0002-0635-7305.

Process and evolution of formal methods in requirements engineering

Abstract

Requirements Engineering is considered the most important phase of the life cycle of software products because it specifies the needs of the customers, and it is also the basis for the execution of the other phases of software engineering. The models currently used to perform the requirements elicitation have been proposed and widely documented, but they are focused only on the techniques to collect information, disregarding the activity of properly documenting this information. Moreover, to structure the requirements specification, natural language continues to be used as a means of communication and understanding with the customer. Due to the ambiguities caused by this language, its interpretation becomes difficult, and this leads to reprocesses in the later stages of the software life cycle. According to the above, it is necessary for software development organizations to consider formalizing the process of requirements elicitation if they wish to make their development process more efficient. A literature review is carried out in this paper to determine the process and evolution of the formal methods from the requirements engineering perspective.

Keywords: Formal methods; requirements engineering; software quality; mathematic language.

Processo e evolução dos métodos formais na engenharia de requisitos

Resumo

A engenharia de requisitos é considerada a fase mais importante do ciclo de vida dos produtos de software, porque nela se especificam as necessidades dos clientes e é a base para a execução das demais fases da engenharia de *software*. Os modelos que atualmente são utilizados para realizar a elicitação de requisitos foram propostos e documentados de forma ampla, mas se centram só nas técnicas para coletar informação e descuidam a atividade de documentar de maneira adequada. Além disso, para estruturar a especificação de requisitos segue em uso a linguagem natural como forma de comunicação e entendimento com o cliente. Devido às ambiguidades que este causa, dificulta-se a interpretação; isso implica reprocessos nas etapas posteriores do ciclo de vida do software. De acordo com isso, é necessário que as organizações desenvolvedoras de *software* considerem formalizar o processo de elicitação de requisitos se desejam tornar mais eficiente seu processo de desenvolvimento. Neste artigo, faz-se uma revisão de literatura para determinar o processo e a evolução dos métodos formais a partir da perspectiva da engenharia de requisitos.

Palavras-chave: métodos formais; engenharia de requisitos; qualidade do *software*; linguagem matemática.

INTRODUCCIÓN

El *software* se diseña y desarrolla a través de una serie de etapas conocidas como *ciclo de vida*, en el que se estructura una solución a las necesidades del cliente. Es una labor de trabajo en equipo entre los ingenieros de *software* y los interesados en el producto, y la mayor parte de los problemas en el desarrollo se relaciona con la ingeniería de requisitos —de modo específico, con la educción—. Si las necesidades del cliente no se elicitan correctamente, el desarrollo presentará problemas porque el producto no responderá a las necesidades de las partes, aunque se disponga de buenos lenguajes de especificación. Aquí radica la importancia de la educción de requisitos: en ella se genera la información necesaria para la especificación, que a su vez es la base para el diseño y el desarrollo de la solución.

Los modelos que se utilizan hoy para realizar la educción de requisitos se centran solo en las técnicas para acopiar información, pero descuidan la actividad de documentarla de forma adecuada. Además, persisten en usar, en primer orden, el lenguaje natural como forma de comunicación y comprensión con el cliente. Debido a las ambigüedades causadas por este último, la interpretación se dificulta, hecho que acarrea un incremento en costos y tiempos de entrega, además de procesos de reingeniería en el diseño y la arquitectura [1].

Desde que en la década de 1960 se promulgó la llamada "crisis del *software"* debido a los errores encontrados en la calidad, fiabilidad y seguridad de los productos de *software*, los investigadores y la industria han intentado solucionarla, hasta ahora sin éxito. Una manera de evitar estos inconvenientes radica en la formalización de requisitos: según Gore y Diallo [2], este es el proceso de describir un sistema y sus propiedades deseadas utilizando un lenguaje cuyas sintaxis y semántica se definan en términos matemáticos.

El concepto de *métodos formales* alude a técnicas y herramientas basadas en principios y postulados matemáticos que se utilizan, entre otros, para especificar, diseñar, validar y verificar sistemas de *software* y *hardware*. La especificación utilizada en los métodos formales está conformada por enunciados bien formados en una lógica matemática. La verificación formal por deducciones rigurosas sigue la misma lógica; es decir, cada paso sigue una regla de inferencia y, por lo tanto, se puede comprobar mediante un proceso mecánico [3]. Los métodos formales surgieron como puntos de vista analíticos con los que es posible verificar el desarrollo de sistemas mediante la lógica y las matemáticas, lo que aporta ventajas para mejorar la calidad de los programas y, por tanto, contribuye a la ingeniería de *software*. En la fase de la ingeniería de requisitos, especificar en términos formales es importante y requiere mayor cuidado, porque su objeto es garantizar que tanto el funcionamiento como el

desempeño del programa sean correctos bajo cualquier situación. En la educción se utilizan fórmulas matemáticas para documentar los requisitos, de tal forma que sean comprendidos por las partes involucradas y que en el diseño se armonicen de mejor forma en la estructura de la solución [4].

La importancia de los métodos formales yace en sistematizar e introducir rigor en todas las fases de desarrollo del *software*, en especial en la Ingeniería de Requisitos, con lo que sería posible evitar que se pasen por alto problemas críticos. Además, proporcionan un método estándar de trabajo a lo largo del proyecto, constituyen una base coherente entre las muchas actividades relacionadas y, al contar con mecanismos de descripción precisos y no ambiguos, proporcionan el conocimiento necesario para realizarlas con éxito [5].

Este artículo es el resultado de una revisión de literatura para determinar el proceso y evolución de los métodos formales en la elicitación de requisitos. En este sentido, se llevó a cabo una labor de consulta en diferentes bases de datos y bibliotecas digitales, encaminada a encontrar trabajos representativos que aportaran a la investigación.

1. MARCO TEÓRICO

Para Mathiassen y Munk-Madsen [6], las formalizaciones están relacionadas con los tipos de expresión y comportamiento del *software*. Estos autores discuten los límites a la aplicación de las formalizaciones en estos dos sentidos y los ilustran con ejemplos en el desarrollo de sistemas prácticos. También establecen que las formalizaciones son valiosas en algunas situaciones e insuficientes en otras, y plantean que la alternativa a la utilización acrítica de las formalizaciones es que los desarrolladores analicen las situaciones en las que se encuentran y, a partir de allí, elijan una combinación entre lo formal y lo informal hasta que sean capaces de comprenderlas y dominarlas.

Vilkomir *et al.* [7] afirman que los requisitos reglamentarios, a diferencia de aquellos para un sistema en particular, tienen un carácter genérico, son aplicables a una amplia gama de sistemas, y son la base para la certificación o el proceso de concesión de licencias. La formalización de los requisitos está resolviendo las inconsistencias entre ellos y la interpretación de los desarrolladores de sistemas críticos. Estos autores proponen un enfoque para hacerlo, que incluye la formalización de los requisitos reglamentarios como base para el desarrollo de métodos de evaluación del *software*. Ilustran su enfoque con ejemplos de requisitos formalizados para la protección de sistemas de control contra el acceso no autorizado y los fallos de *software* en modo común, usando la notación *Z* para la formalización.

Por su parte, Webel y Gotzhein [8] establecen que la necesidad de la formalización de los requisitos se debe al hecho de que se necesita una descripción precisa de ellos,

así como una amplia comprensión entre el usuario y el proveedor de los servicios de calidad. Para esto, los mecanismos que realizan estos procesos requieren descripciones más precisas y un alcance real de la formalización. Estos mecanismos suelen estar integrados en capas, por lo que se necesita más de un punto de vista sobre los requisitos. Actualmente, esta relación rigurosa de los puntos de vista no se está logrando y los métodos formales necesarios aún están por definirse. Se espera que en el futuro se involucren mejor y en mayor grado en los procesos de la ingeniería de requisitos, y que los desarrolladores tengan una mejor comprensión de la formalización.

La especificación formal es el proceso de describir un sistema y sus propiedades deseadas mediante un lenguaje con una sintaxis y una semántica definidas matemáticamente [9]. Por eso, el objetivo de Chatterjee y Johari [10] tiene que ver con la formalización automatizada de la especificación de requisitos, de forma que se puedan generar con facilidad los casos de prueba formales.

Cavada *et al.* [11] llegan a la conclusión que la ingeniería de requisitos es una de las fases más importantes del proceso de desarrollo de *software*, y que en aplicaciones críticas es importante apoyar la validación de los requisitos con técnicas formales para identificar y eliminar los defectos. Sin embargo, los requisitos se escriben a menudo en documentos textuales, y su formalización y validación no son adecuadas por la falta de conocimiento sobre los métodos formales.

Post y Hoenicke [12] evalúan una cadena de herramientas para analizar requisitos de tiempo real de modo algorítmico. De acuerdo con esta cadena de herramientas, los autores formalizan los requisitos de un sistema en lenguaje natural. Ellos experimentan la compilación automática desde fórmulas en una lógica en tiempo real. Las fórmulas se pueden comprobar de forma automática para las propiedades cuya violación indica un error en la especificación de requisitos. Presentan un estudio de viabilidad en el contexto de varios proyectos en la herramienta *Bosch*. Los resultados del estudio indican que el esfuerzo para formalizar los requisitos de tiempo real es aceptable, los algoritmos de análisis son computacionalmente factibles; y el beneficio (en cuanto a detección de errores de especificación) parece significativo.

Li *et al.* [13] concluyen que los requisitos son informales, vagos, ambiguos y, muchas veces, inalcanzables, por lo que el problema de la ingeniería de requisitos es formalizarlos y luego transformarlos a través de un proceso sistemático en una especificación formal que pueda entregarse a los diseñadores para el desarrollo. En su trabajo proponen un marco para la transformación de los requisitos informales a formales, y luego a una especificación. El marco consiste en una ontología de requisitos, un lenguaje formal de modelado de requisitos para la representación de los funcionales y no funcionales, así como un amplio conjunto de operadores de refinamiento por el

cual los requisitos se transforman de forma gradual en una especificación formal y medible. Esta propuesta incluye una metodología sistemática y herramientas de apoyo para realizar la transformación. Los resultados sugieren que la ontología y el lenguaje de modelado son adecuados para la captura de requisitos, y que la metodología es eficaz en el manejo de los requisitos en la práctica.

Al utilizar notaciones y lenguajes formales es posible realizar transformaciones de modelos automáticos [14]. A diferencia de un modelo no formal, sus notaciones son intuitivas: permite una mejor abstracción de los detalles y aplica metodologías estandarizadas y bien definidas [15], lo que permite estructurar con claridad los requisitos del sistema y generar la formalización de requisitos.

Los requisitos se consideran la parte fundamental sobre la que se pueden desarrollar las soluciones de *software;* y en la actualidad existe una creciente demanda de enfoques más rigurosos y sistemáticos para formalizarlos [16]. Serna y Serna [17] argumentan que los métodos formales se utilizan hoy para modelar complejos sistemas críticos de seguridad, pero poco se trabaja en la formalización de los requisitos desde las primeras etapas de la ingeniería de requisitos.

La representación más utilizada para formalizar los requisitos son los lenguajes de especificación formal [16]. Con este tipo de especificación, los desarrolladores comprenden mejor el sistema y eluden las ambigüedades, los flujos, las omisiones y las inconsistencias de los requisitos. Además, la especificación es un importante mecanismo de comunicación entre los clientes y los diseñadores; entre los diseñadores y ejecutores; y entre los ejecutores y los probadores [18]. Pero las especificaciones formales no son documentos que se escriben una vez, y por lo general no se logran al principio del proceso de desarrollo de *software*. Se necesita tiempo para crear una primera versión de utilidad que, luego de mucho esfuerzo y revisiones, permita desarrollar una especificación cercana a las necesidades que los clientes tienen en mente [19]; al respecto, Bollin y Rauner [20] sostienen que un buen nivel de comprensión es una cuestión clave como atributo de calidad. Para Wolf [21], si se hace uso de un lenguaje de especificación formal, el sistema puede ser descrito con precisión en cuanto a funcionalidad, concurrencia, integridad y exactitud. Esto significa que las propiedades de un sistema se pueden analizar sin tener que ejecutarlo.

1.1 Trabajos relacionados

A continuación se detallan los 41 trabajos que compusieron la muestra final de la revisión.

Bhavsar [22] presenta un uso de los métodos formales en la formalización de requisitos centrados en el usuario. De acuerdo con este autor, los resultados son

prometedores en cuanto al mejoramiento de la calidad del producto de *software*. Por su parte, Serna [23] hace un recorrido histórico por los métodos formales y describe sus aplicaciones y beneficios en la formalización de requisitos. Propone que la investigación en esta área se debe orientar a aplicaciones prácticas y a buscar reducir los costos de su utilización.

Bishop *et al.* [24] integran los conceptos de los métodos formales desde el inicio de la programación y, mediante una didáctica simple, muestran cómo hacer una especificación formal y qué modelos se pueden utilizar. De Sousa *et al.* [25] analizan el método *B* para la especificación de requisitos y describen los casos de uso y las propiedades de seguridad. Aunque su aplicación es sencilla, es un aporte hacia la masificación de los métodos formales en la especificación de requisitos.

Gao y Wang [26] proponen una metodología para formalizar la ingeniería de requisitos, mediante el diseño de un modelo basado en lenguaje Z, orientado a analizar y gestionar requisitos (MZASR): describen la especificación de requisitos de *software* por herramientas y modelos UML, y la semántica individual se establece como especificación Z mediante un enfoque ascendente. Por su parte, Hassan *et al.* [27] plantean el análisis y el diseño formal de la seguridad de los requisitos mediante especificaciones en *B*. Los resultados demuestran efectividad y capacidad en la producción de *software* seguro. Van der Poll [28], a su turno, describe los métodos formales más comunes y la importancia de aplicar la especificación formal en los primeros pasos del ciclo de vida; enfrenta las posiciones de los críticos y defensores de los métodos formales y presenta ejemplos de historias de éxito en la industria. Singh y Patterh [29], a su vez, utilizan la notación *Z* para formalizar los requisitos en busca de protección y seguridad contra amenazas internas: un caso de estudio de éxito.

Cimatti *et al.* [30] desarrollan una metodología y técnicas para formalizar y validar requisitos en aplicaciones críticas de seguridad, y aplican el resultado en un caso de estudio con resultados mejorados. En cambio, Serna [5] describe la aplicación de la especificación formal de requisitos y el mejoramiento que se logra en la calidad del producto final. Este autor propone la incorporación de los métodos formales en los contenidos curriculares de los programas de ciencias computacionales.

Barringer *et al.* [31] presentan un método práctico para usar especificaciones formales en pruebas de registro en el Mars Science Laboratory. Demuestran, así, las ventajas de utilizar un lenguaje de especificación formal. Por otro lado, Fernández *et al.* [32] proponen una revisión a la aplicación de los métodos formales en proyectos reales en la industria: presentan los diversos casos de éxito y describen la situación actual en México en cuanto a la aplicación de métodos formales.

Ibrahim *et al.* [33] aplican una base formal para la especificación y verificación de requisitos, dentro de lo cual presentan una definición formal y una teoría de composición formal para la Ingeniería de Requisitos y describe los resultados. Además de esto, Yan [19] describe la aplicación de la especificación formal en *Communications-Based Train Control* (CBTC), con lo que obtiene una descripción profunda y exacta del sistema, y lenguaje *Z* en el desarrollo. El autor concluye que al usar métodos formales se gana confianza para asegurar el sistema. Incluso Ammar y Abdallah [34], presentan un nuevo enfoque para la especificación y verificación formal basado en la reescritura de la lógica; luego, lo aplican en un caso de estudio con resultados prometedores en cuanto a mejoramiento de la calidad. Por su parte, Kaur *et al.* [35] presentan un estudio comparativo entre los métodos de especificación formal y descripción de las razones de por qué usarlos en los procesos industriales.

Sumado este trabajo a lo anterior, You *et al.* [36] muestran los logros y problemas de los métodos formales y validan su aplicación para garantizar la seguridad y estabilidad del sistema. A su turno, Barlas *et al.* [37] presentan una comparación entre la especificación de requisitos tradicional y la formal; con la segunda se alcanza mayor comprensión, se eliminan ambigüedades y se obliga a una mejor precisión de la especificación.

Al contrario de Barlas *et al.* [37], Wolff [21] utiliza la experiencia en la industria para combinar el desarrollo ágil *Scrum* con los métodos formales, pero desde una conceptualización teórica; presenta una evaluación y discusión acerca de los beneficios de utilizarlos para formalizar los requisitos. Bollin [38], desde un marco teórico, concluye que el proceso de la formalización de los requisitos es un problema de adaptación transcultural: analiza sus pros y contras, y da a conocer un modelo refinado para un proceso de desarrollo de *software* formal. Desde la experiencia, Serna y Serna [39] presentan una investigación teórica en la que se concluye que el método formal ha alcanzado significativos avances, lo que vislumbra aplicaciones ambiciosas en el futuro para apoyar rigurosos y coherentes planes de estudios en ciencias computacionales.

Atlee *et al.* [40] ofrecen una serie de recomendaciones para formalizar los requisitos, aportan al logro práctico de la selección de una técnica específica. Su objetivo es mejorar la aceptación de los métodos formales en el modelado de requisitos.

Por su parte, Serna y Serna [41] sostienen que la especificación formal todavía tiene usos limitados; la comunidad tiene una comprensión diferente acerca de su utilidad y necesidad. Estos autores sostienen que los investigadores se focalizan en la especificación escrita durante el diseño del modelo funcional y dejan a un lado cualquier tratamiento formal a los no funcionales. Chen y Ouyang [42], a su turno, analizan el dominio de los métodos formales para identificar defectos y disminuir fracasos. Aplican

la formalización en la especificación, el modelado y la verificación. De igual manera, Chan *et al.* [43] proponen una metodología de modelado de requisitos para convertir los informales en formales; con ella se busca eliminar ambigüedades y defectos, refinar y perfeccionar el sistema, y componer un medio de comunicación entre las partes interesadas. Gore y Diallo [2], por su parte, hacen un estudio de los lenguajes de especificación formal; describen los modelos que se pueden aplicar y realizan un análisis general, y realizan una experimentación con estadística de depuración en un estudio de caso.

Noaman *et al.* [44] aplican la especificación formal por medio de *Z* para potencializar la seguridad del sistema y reducir las amenazas. Los resultados son prometedores para ampliaciones y elaboraciones futuras. Lockhart *et al.* [45] demuestran la utilidad de los métodos formales para la especificación de requisitos en sistemas de seguridad críticos. La especificación formal reduce la ambigüedad del diseño, ayuda a probar la consistencia, e incrementa la confianza y el rendimiento. Por su parte, Wu *et al.* [46] proponen reglas de transformación desde el modelo *B* y describen los requisitos con máquinas abstractas. Para los autores, la especificación formal es un proceso intermedio entre la especificación de requisitos y la escritura del código.

Serna y Serna [17] realizan una revisión de la literatura, hacen un recorrido por la esencia, la función, el uso y los inconvenientes de las técnicas de especificación formal, y analizan criterios de valoración y evaluación a sus debilidades. De igual manera, Bollin y Rauner-Reithmayer [20] muestran una serie de recomendaciones para realizar especificaciones formales, e identifican que la facilidad de lectura de una especificación es un factor clave para mejorar la calidad del *software*.

Schraps y Peters [9] aplican gramática formal con anotaciones semánticas para formalizar requisitos, de tal forma que puedan ser analizados y procesados por el computador. Los requisitos formulados pueden ser analizados y procesados en las primeras etapas de desarrollo.

Serna y Serna [47] afirman que muchos de los problemas desafiantes en la construcción de sistemas requieren apoyo formal para su modelado y análisis. También sostienen que en los procesos de investigación y aplicación de las ciencias computacionales existe un número creciente de aplicaciones, pero todavía no constituyen una parte integral de los procesos formativos en pregrado y posgrado.

Azeem *et al.* [48] utilizan especificación formal *Z* para mejorar la calidad y confianza del sistema: muestran trabajos relacionados y aplican una propuesta en un estudio de caso en salud. Esto lo confirman Li *et al.* [49], quienes aplican O*bject-Z* para describir sistemas complejos y demuestran su utilidad en un caso de estudio de

un sistema de suministro de gasolina. Los resultados obtenidos alientan el uso de la especificación formal.

Serna y Serna [50] describen aspectos relevantes de los métodos formales y realizan un marco del futuro de la investigación en el área. Los resultados invitan a la reflexión sobre este componente de las ciencias computacionales y respecto a la necesidad de tener una comunidad más amplia y sólida.

Jeffery *et al.* [51] realizan un estudio empírico para encontrar la productividad de los proyectos que utilizan métodos formales, específicamente al formalizar los requisitos. Estos autores identifican una serie de preguntas sobre los resultados en productividad de estos proyectos. A su turno, Tamrakar y Sharma [52] comparan tres métodos para especificar formalmente: *Z*, *B* y *VDM*. Usar especificaciones formales no asegura un sistema completamente correcto, pero se aumenta la confianza en él. De igual manera, Singh y Yadav [53] utilizan el método *Event-B* para escribir especificaciones formales, con el fin de garantizar la compresión de los límites en los que un algoritmo puede ser usado. Para los autores, la especificación, la validación y la verificación formales son la clave para obtener un mejor diseño. Walter *et al.* [54], por su parte, realizan una representación de formalización de requisitos por medio de diagramas *SysML* a través de la semántica *RSL*; formalizan únicamente los requisitos no funcionales y aplican ese diseño a un caso de estudio en el área automotriz.

2. ANÁLISIS DE RESULTADOS

Al utilizar la especificación formal, las propiedades del sistema se describen mediante un lenguaje con una sintaxis y una semántica definidas en términos matemáticos [7]. Algunos lenguajes formales, tales como *Z*, *B* y *VDM*, se centran en especificar el comportamiento de los sistemas secuenciales, y en ellos los estados se describen en estructuras matemáticas como conjuntos, relaciones y funciones [36]; mientras que métodos como *CSP*, *CCS*, *Statecharts*, lógica temporal y autómatas se centran en la especificación de los comportamientos del sistema en términos de secuencias, árboles u órdenes parciales de eventos [9]. El uso de lenguajes formales como *Z* y *B* puede mejorar la confianza del usuario en el sistema y los impactos se esta última en su uso [48].

Z trabaja en alta abstracción a nivel del sistema y proporciona una sólida base para su diseño. Con el uso de este lenguaje se descubren errores en la especificación y en las fases de pruebas y mantenimiento. Esto es conveniente porque, con la ingeniería tradicional, corregir errores en etapas posteriores incrementa los costos [48]; además, *Z* es una manera de descomponer una especificación en pequeñas partes, llamadas *esquemas*. Por su parte, *B* es uno de los métodos formales más conocidos. Se basa en la lógica de primer orden, la teoría de conjuntos, la aritmética de enteros y las sustituciones

generalizadas; se utiliza para el diseño de *software* desde los requisitos funcionales y permite producir casos de prueba que demuestran la exactitud y la consistencia del modelo *software* aplicado. El objetivo de *B* es obtener un producto probado y fiable [25]; mientras que *VDM* se utiliza para demostrar la equivalencia de los conceptos del lenguaje de programación [52].

De acuerdo con Pandey y Batra [16], las empresas se enfrentan en la actual era digital al desafío de liberar proyectos de *software* de calidad a tiempo y ajustados al presupuesto; pero la realidad es que se entrega con errores, falta de funcionalidad y, a veces, con sobrecostos. Estos ocurren debido a errores en la especificación de requisitos, cuya corrección puede acarrear fuertes cargas en tiempo y dinero cuando se detectan en las fases tardías del ciclo de vida. Por lo tanto, en la especificación de requisitos se debe seleccionar una metodología formal para abordar la fiabilidad durante la ingeniería de requisitos y el diseño, porque los métodos formales permiten desarrollar los errores antes de la liberación del producto [45].

Los errores de especificación pueden ser reducidos drásticamente mediante el uso de métodos formales y, en consecuencia, el ingeniero de *software* puede crear una especificación más completa, coherente e inequívoca que con los métodos convencionales [35] [48]. Por su parte, Bollin y Rauner-Reithmayer [20] manifiestan que una buena especificación formal es sintáctica y semánticamente correcta, y permite un mapeo sin pérdidas entre todos los conceptos de la especificación y el modelo mental del sistema especificado; también agregan que debe ser completa, coherente y adecuada, y tener en cuenta que la facilidad de comprensión es un requisito esencial para decidir sobre su corrección semántica.

En resumen, para conocer qué se publica en la literatura en relación con la formalización de requisitos, en la tabla 1 se presenta el resumen de los resultados de esta investigación. El inicio de la línea de tiempo de observación se estableció en 2010. Aunque es conocido que la mayor parte del trabajo en esta área se presentó en la segunda mitad del siglo pasado, el objetivo de esta investigación fue verificar su progreso actual. Los trabajos de la muestra final se clasificaron de acuerdo con el enfoque primario presentado. Se aclara aquí que muchos autores presentan resultados en más de una tipología, es decir, su investigación puede ser teórica y, al mismo tiempo, experimental.

- *Investigación teórica:* artículos que contienen definiciones o descripciones acerca de la formalización de requisitos.
- *Investigación experimental:* demuestra resultados a partir de experimentos en laboratorio.

- *Trabajos de aplicación práctica:* describen y aplican métodos, técnicas o procedimientos de formalización de requisitos en casos de estudio.
- *Trabajos de aplicación industrial:* aquellos en los que la formalización de requisitos se aplica en casos reales.

Tabla 1. Formalización de requisitos según la literatura

Medio		Aplicación			
		Teórica	Experimental	Práctica	Industrial
Revista	66 %	63 %	34 %	20 %	23 %
Conferencia	34 %				

Fuente: elaboración propia.

3. PROGRESO DEL TRABAJO EN LA FORMALIZACIÓN DE REQUISITOS

El progreso de la formalización de requisitos será definido como *acelerado* o *lento* en función del porcentaje de aplicaciones en la industria. En la figura 1 se clasifican los resultados de la investigación y se muestra su participación porcentual.

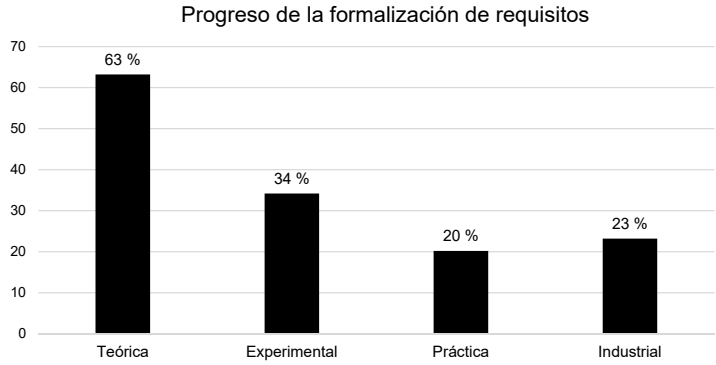

Figura 1. Clasificación y participación porcentual de los resultados de la investigación

Fuente: elaboración propia.

De acuerdo con la revisión de literatura en esta investigación, el trabajo en formalización de requisitos ha tenido un progreso lento. En la figura 1 se muestra que la mayoría de los trabajos tomados en cuenta para la presente investigación se enfocan más en definiciones teóricas, que son representadas por el 63 %, y solo el 23 % de estos en aplicación en la industria. Algunos autores han analizado este hecho y presentan sus conclusiones. Entre las causas se destacan falta de formación en la academia; falta de aceptación en la industria; ausencia de demostraciones realmente importantes de sus ventajas; costos; falta de personal capacitado; fobia de los ingenieros a las matemáticas; ausencia de deseo en la industria por experimentar con principios

que no tienen comprobación suficiente; y pérdida de interés y patrocinio respecto de la investigación en métodos formales durante la última década [23] [55] [40] [17].

Los investigadores de mediados del siglo pasado estaban convencidos de que la única manera de mejorar la calidad de los productos de *software* radicaba en la matematización de la misma ingeniería de *software* [28]; empero, conforme sus resultados comenzaron a publicarse, la industria comenzó a encontrar y poner barreras al progreso de este trabajo. Solo el *software* crítico los arropó como tabla de salvación para solucionar sus problemas de fiabilidad y seguridad, y son diversos los casos de éxito en estos desarrollos. El problema surge cuando las empresas de desarrollo de *software* comercial se enfrentan a situaciones de incumplimiento, que incrementan los costos del proceso y las obligan a dedicar menos tiempo a la estructura misma de las actividades. La decisión es disminuir la mayor cantidad de tiempo posible y, como la forma tradicional de realizar la verificación y validación es una barrera de contención al final del ciclo de vida, esta fase se recorta para encontrar algún tiempo extra [36].

El estudio sobre la formalización de requisitos se ha relegado a unos pocos investigadores que convencen a sus estudiantes de posgrado para que trabajen con ellos. Visto el mencionado inconveniente de que los profesionales temen a todo lo que tenga que ver con matemáticas, se necesita con premura que la formalización empiece a permear los procesos formativos en la academia y que los estudiantes conozcan sus ventajas y potencialidades, lo mismo que sus desventajas y problemas, de tal forma que se interesen y profundicen en su asimilación y comprensión.

De acuerdo con los autores citados en esta investigación, la formalización de requisitos se investiga de forma mayoritariamente teórica y de esta manera no se logrará convencer a la industria de su aplicación. Puede ser que la etapa de la teorización ya haya sido superada en los muchos trabajos que se presentaron el siglo pasado. La necesidad actual es masificar los métodos formales y encontrar la forma de aplicarlos en la matematización de la ingeniería de *software*, pero con costos que estén al alcance de las empresas que lo desarrollen. De esta manera será posible reavivar el interés en esta área de trabajo y encontrar el patrocinio necesario para ejecutar la investigación necesaria. Si el objetivo es superar la crisis del *software*, los métodos formales deben ser el centro de desarrollo porque sus bondades ya han sido demostradas con suficiencia, y porque el lenguaje con el que funciona el computador es matemático.

4. CONCLUSIONES

Los resultados demuestran que el progreso y desarrollo de la formalización de requisitos son lentos y que, en los últimos años, el interés por darle continuidad ha disminuido. Aunque se acepta que los métodos formales son una herramienta útil y necesaria para

superar la crisis de la calidad del *software*, muy pocos investigadores quieren abordar la formalización de requisitos.

Dentro de las investigaciones obtenidas se puede evidenciar la importancia de la aplicación de los métodos formales durante el desarrollo de un producto de *software*, ya que evita ambigüedades y permite obtener productos más precisos y seguros. No obstante, cabe resaltar que en los últimos 5 años la aplicación de la formalización de requisitos en la industria ha sido muy poca por factores como desconocimiento de la formalización de requisitos por parte de los desarrolladores; búsqueda de desarrollos de *software* ágiles o hechos en el menor tiempo posible; y falta de publicaciones por parte de las empresas que aplican la formalización de requisitos en sus desarrollos.

La industria necesita demostraciones de los beneficios que representa la formalización, pero hasta ahora no tiene inclinación directa por apoyar las iniciativas que en este sentido propone la comunidad. Solo en el desarrollo de sistemas críticos se aprecia una amplia participación de las empresas de desarrollo y la industria porque su objetivo es que funcionen sin poner en riesgo la vida humana o las inversiones económicas.

Se necesita mayor diligencia de los métodos formales en los procesos de ingeniería de requisitos, pero, sobre todo, es urgente contar con el personal capacitado para aplicarla y darle continuidad.

Esta es una sociedad dependiente del *software* porque muy pocas de sus actividades están por fuera del ámbito de este desarrollo tecnológico; empero, los productos que se liberan o entregan a los usuarios aún no resultan satisfactorios en aspectos como fiabilidad y seguridad. Las matemáticas ofrecen la posibilidad de superar este problema y su representación en los métodos formales es una alternativa prometedora.

REFERENCIAS

[1] Å. Dahlstedt, y A. Persson, "Requirements interdependencies - Moulding the state of research into a research agenda", en *Ninth International Workshop on Requirements Engineering Foundation for Software Quality (REFSQ 2003),* Klagenfurt, 2003.

[2] R. Gore, y S. Diallo, "The need for usable formal methods in verification and validation", en *2013 Winter Simulation Conference (WSC),* Washington, 2013. DOI: https://doi.org/10.1109/WSC.2013.6721513.

[3] C. Burgess, *The Role of Formal Methods in Software Engineering Education and Industry*, Bristol: University of Bristol, 1995.

[4] P. Zave, "Classification of research efforts in requirements engineering", *ACM Computing Surveys*, vol. 29, n.° 4, pp. 315-321, 1997. DOI: https://doi.org/10.1109/isre.1995.512563.

[5] E. Serna, "Métodos formales e Ingeniería de Software", *Revista Virtual Universidad Católica del Norte*, n.° 30, pp. 158-184, 2010, Disponible en: http://revistavirtual.ucn.edu.co/index.php/RevistaUCN/article/viewFile/62/129.

[6] L. Mathiassen y A. Munk-Madsen, "Formalization in systems development", en *International Joint Conference on Theory and Practice of Software Development on Formal Methods and Software Development*, Berlín, 1985. DOI: https://10.1007/3-540-15199-0_7.

[7] S. Vilkomir, J. Bowen y A. Ghose, "Formalization and assessment of regulatory requirements for safety-critical software", *Innovations in Systems and Software Engineering*, vol. 2, n.° 3-4, pp. 165-178, 2006. DOI: https://10.1007/s11334-006-0006-8.

[8] C. Webel y R. Gotzhein, "Formalization of Network Quality-of-Service Requirements", *Lecture Notes in Computer Science*, vol. 4574, pp. 309-324, 2007. DOI: https://10.1007/978-3-540-73196-2_20.

[9] M. Schraps y M. Peters, "Semantic annotation of a formal grammar by Semantic Patterns", en *2014 IEEE 4th International Workshop on Requirements Patterns (RePa),* Karlskrona, 2014. DOI: https://10.1109/repa.2014.6894838.

[10] R. Chatterjee y K. Johari, "A Simplified and Corroborative Approach towards Formalization of Requirements", *Communications in Computer and Information Science*, vol. 94, pp. 486-496, 2010. DOI: https://10.1007/978-3-642-14834-7_46.

[11] R. Cavada, A. Cimatti, A. Micheli, M. Roveri, A. Susi y S. Tonetta, "OthelloPlay – A Plug-in based tool for requirement formalization and validation", en *The 1st workshop on Developing tools as plug-ins – TOPI'11,* Honolulu, 2011. DOI: https://10.1145/1984708.1984728.

[12] A. Post y J. Hoenicke, "Formalization and Analysis of Real-Time Requirements: A Feasibility Study at BOSCH", *Lecture Notes in Computer Science*, vol. 7152, pp. 225-240, 2012. DOI: https://10.1007/978-3-642-27705-4_18.

[13] F. Li, J. Horkoff, A. Borgida, G. Guizzardi, L. Liu y J. Mylopoulos, "From Stakeholder Requirements to Formal Specifications Through Refinement", *Requirements Engineering: Foundation for Software Quality*, vol. 9013, pp. 164-180, 2015. DOI: https://10.1007/978-3-319-16101-3_11.

[14] H. Ehrig, G. Engels, F. Orejas y M. Wirsing, "Semi-Formal and Formal Specification Techniques for Software Systems" [internet], 2000. Disponible en: https://pdfs.semanticscholar.org/3323/1736efa0f613ba416094e1a59262b23f8a2d.pdf.

[15] C. Snook y M. Butler, "UML-B: Formal Modeling and Design Aided by UML", *ACM Transactions on Software Engineering and Methodology (TOSEM)*, vol. 15, n.° 1, pp. 92-122, 2006.

[16] S. Pandey y M. Batra, "Formal methods in requirements phase of SDLC", *International Journal of Computer Applications (0975 – 8887),* vol. 70, n.° 13, pp. 7-14, 2013.

[17] E. Serna y A. Serna, "La especificación formal en contexto: actual y futuro", *Ingeniare – Revista Chilena de Ingeniería*, vol. 22, n.° 2, pp. 243-256, 2014. DOI: https://10.4067/S0718-33052014000200010.

[18] H. Krad, "Formal methods and automation for system verification", *en 2011 Fourth International Conference on Modeling, Simulation and Applied Optimization (ICMSAO 2011)*, Kuala Lumpur, 2011. DOI: https://10.1109/icmsao.2011.5775479.

[19] F. Yan, "Studying Formal Methods Applications in CBTC", *en 2011 International Conference on Management and Service Science (MASS2011),* Bangkok, 2011. DOI: https://10.1109/icmss.2011.5999325.

[20] A. Bollin y D. Rauner-Reithmayer, "Formal specification comprehension: the art of reading and writing z", en *The 2nd FME Workshop on Formal Methods in Software Engineering - FormaliSE 2014,* Hyderabad, 2014. DOI: https://10.1145/2593489.2593491.

[21] S. Wolff, "Scrum goes formal: Agile methods for safety-critical systems", en *2012 First International Workshop on Formal Methods in Software Engineering: Rigorous and Agile Approaches (FormSERA),* Zurich, 2012. DOI: https://10.1109/formsera.2012.6229784.

[22] M. Bhavsar, "Analysis of Multiagent Based Interactive Grid Using Formal Methods - A Reliable Approach", en *2010 3rd International Conference on Emerging Trends in Engineering and Technology*, Goa, 2010. DOI: https://10.1109/icetet.2010.5.

[23] E. Serna, "Métodos formales: Perspectiva y aplicación futura", en *III Jornadas de Investigación de la Facultad de Ingenierías*, pp. 64-68, Medellín, 2011.

[24] M. Bishop, B. Hay y K. Nance, "Applying Formal Methods Informally", en *2011 44th Hawaii International Conference on System Sciences*, Kauai, 2011. DOI: https://10.1109/hicss.2011.71.

[25] T. de Sousa, J. Almeida, S. Viana y J. Pavón, "Automatic analysis of requirements consistency with the B method", *ACM SIGSOFT Software Engineering Notes*, vol. 35, n.° 2, pp. 1-4, 2010. DOI: https://10.1145/1734103.1734114.

[26] H. Gao y S. Wang, "Based on Formal Methods in Trustable Software Requirements Engineering", en *2010 International Conference on Internet Technology and Applications,* Wuhan, 2010. DOI: https://10.1109/itapp.2010.5566647.

[27] R. Hassan, M. Eltoweissy, S. Bohner y S. El-Kassas, "Formal analysis and design for engineering security automated derivation of formal software security specifications from goal-oriented security requirements", *IET Software*, vol. 4, n.° 2, pp. 149-160, 2010. DOI: https://10.1049/iet-sen.2009.0059.

[28] J. Van der Poll, "Formal methods in software development: A road less travelled", *South African Computer Journal*, vol. 45, pp. 165-178, 2010. DOI: https://10.18489/sacj.v45i0.33.

[29] M. Singh y M. Patterh, "Formal Specification of Common Criteria Based Access Control Policy Model", *International Journal of Network Security*, vol. 11, n.° 3, pp. 139-148, 2010.

[30] A. Cimatti, M. Roveri, A. Susi y S. Tonetta, "Formalization and Validation of Safety-Critical Requirements", *Electronic Proceedings in Theoretical Computer Science*, vol. 20, pp. 68-75, 2010. DOI: https://10.4204/eptcs.20.

[31] H. Barringer, A. Groce, K. Havelund y M. Smith, "An Entry Point for Formal Methods: Specification and Analysis of Event Logs", *Electronic Proceedings in Theoretical Computer Science*, vol. 20, pp. 16-21, 2010. DOI: https://10.4204/eptcs.20.2.

[32] C. Fernández *et al*., "Métodos formales aplicados en la industria del software", *Temas de Ciencia y Tecnología*, vol. 5, n.° 43, pp. 3-12, 2011. Disponible en: http://www.utm.mx/edi_anteriores/temas43/1ENSAYO_43_1-R.pdf.

[33] N. Ibrahim, V. Alagar y M. Mohammad, "Specification and Verification of Context-dependent Services", *Electronic Proceedings in Theoretical Computer Science*, vol. 61, pp. 17-33, 2011. DOI: https://10.4204/eptcs.61.2.

[34] B. Ammar y K. Abdallah, "Towards the formal specification and verification of multi-agent-based systems", *International Journal of Computer Science Issues*, vol. 8, n° 4, pp. 200-210, 2011.

[35] A. Kaur, S. Gulati y S. Singh, "A comparative study of two formal specification languages: Z-notation & B-method", en *The Second International Conference on Computational Science, Engineering and Information Technology - CCSEIT '12*, Coimbatore, 2012. DOI: https://10.1145/2393216.2393304.

[36] J. You, S. Xia y J. Li, "A survey on formal methods using in software development", *en IET International Conference on Information Science and Control Engineering 2012 (ICISCE 2012),* Shenzhen, 2012. DOI: https://10.1049/cp.2012.2353.

[37] K. Barlas, G. Koletsos y P. Stefaneas, "Extending standards with formal methods: Open Document Architecture", en *2012 International Symposium on Innovations in Intelligent Systems and Applications,* Trabzon, 2012. DOI: https://10.1109/INISTA.2012.6246931.

[38] A. Bollin, "Do you speak Z? Formal methods under the perspective of a cross-cultural adaptation problem", en *2013 1st FME Workshop on Formal Methods in Software Engineering (FormaliSE),* San Francisco, 2013. DOI: https://10.1109/formalise.2013.6612271.

[39] E. Serna y A. Serna, "Desafíos y Oportunidades de la Investigación en Métodos Formales", en *XII Conferencia Iberoamericana en Sistemas, Cibernética e Informática*, Orlando, 2011.

[40] J. Atlee, S. Beidu, N. Day, F. Faghih y P. Shaker, "Recommendations for improving the usability of formal methods for product lines", en *2013 1st FME Workshop on Formal Methods in Software Engineering (FormaliSE),* San Francisco, 2013. DOI: https://10.1109/formalise.2013.6612276.

[41] E. Serna y A. Serna, "Especificación formal - Presente y Futuro", en *XV International Convention and Fair Informatica,* La Habana, 2013.

[42] X. Chen y D. Ouyang, "Research on the Implementation of Internal Control in Enterprise Information System by Domain Analysis and Formal Methods: A Case Study of Sales Activities Internal Control Under Chinese Enterprise Environment", en *2013 Fourth World Congress on Software Engineering (WCSE 2013),* Hong Kong, 2013. DOI: https://10.1109/wcse.2013.31.

[43] L. Chan, R. Hexel y L. Wen, "Rule-Based Behaviour Engineering: Integrated, Intuitive Formal Rule Modelling", en *2013 22nd Australian Software Engineering Conference (ASWEC' 2013)*, Melbourne, 2013. DOI: https://10.1109/aswec.2013.13.

[44] M. Noaman, I. Alsmadi y A. Jaradat, "The specifications of E-Commerce Secure System using Z language", *The Research Bulletin of Jordan ACM*, vol. II, n.° III, pp. 127-131, 2013.

[45] J. Lockhart, C. Purdy y P. Wilsey, "Formal methods for safety critical system specification", en *2014 IEEE 57th International Midwest Symposium on Circuits and Systems (MWSCAS),* College Station, 2014. DOI: https://10.1109/mwscas.2014.6908387.

[46] T. Wu, Y. Dong y N. Hu, "Formal Specification and Transformation Method of System Requirements from B Method to AADL Model", en *2014 IEEE 17th International Conference on Computational Science and Engineering (CSE 2014),* Chengdu, 2014. DOI: https://10.1109/cse.2014.299.

[47] E. Serna y A. Serna, "Los Métodos Formales en Contexto", en *XIII Conferencia Iberoamericana en Sistemas, Cibernética e Informática (CISCI 2014)*, Orlando, 2014.

[48] M. Azeem, M. Ahsan, N. Minhas y K. Noreen, "Specification of e-Health system using Z: A motivation to formal methods", en *International Conference for Convergence for Technology - 2014,* Pune, 2014. DOI: https://10.1109/i2ct.2014.7092123.

[49] Y. Li, X. Pan, T. Hu, S. Sung y H. Yuan, "Specifying Complex Systems in Object-Z: A Case Study of Petrol Supply Systems", *Journal of Software*, vol. 9, n.° 7, pp. 1707-1717, 2014. DOI: https://10.4304/jsw.9.7.1707-1717.

[50] E. Serna y A. Serna, "Perspectiva y Aplicación de los Métodos Formales", en *XIII Conferencia Iberoamericana en Sistemas, Cibernética e Informática,* Orlando, 2014.

[51] R. Jeffery, M. Staples, J. Andronick, G. Klein y T. Murray, "An empirical research agenda for understanding formal methods productivity", *Information and Software Technology*, vol. 60, pp. 102-112, 2015. DOI: https://10.1016/j.infsof.2014.11.005.

[52] S. Tamrakar y A. Sharma, "Comparative Study and Performance Evaluation of Formal Specification Language based on Z, B and VDM Tools", *International Journal of Scientific & Engineering Research*, vol. 6, n.° 9, pp. 1540-1543, 2015.

[53] A. Singh y D. Yadav, "Formal Specification and Verification of Total Order Broadcast through Destination Agreement Using Event-B", *International Journal of Computer Science and Information Technology*, vol. 7, n.° 5, pp. 85-95, 2015. DOI: https://10.5121/ijcsit.2015.7506.

[54] S. Walter, A. Rettberg y M. Kreutz, "Towards Formalized Model-Based Requirements for a Seamless Design Approach in Safety-Critical Systems Development", en *2015 IEEE International Symposium on Object/Component/Service-Oriented Real-Time Distributed Computing Workshops*, Auckland, 2015. DOI: https://10.1109/isorcw.2015.51.

[55] A. Serna, "Los Métodos Formales en la Industria", *Revista Antioqueña de las Ciencias Computacionales y la Ingeniería de Software (RACCIS)*, vol. 2, n.° 2, pp. 44-51, 2012, Disponible en: http://fundacioniai.org/raccis/v2n2/n3a7.pdf.