UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS

Revista Científica

http://revistas.udistrital.edu.co/ojs/index.php/revcie/index

CIDC

Artículo de revisión

Deuda de la documentación en el desarrollo ágil de software: mapeo sistemático de la literatura

Documentation Debt in Agile Software Development: A Systematic Mapping of the Literature

Dívida documental no desenvolvimento ágil de software: um mapeamento sistemático da literatura

Juan-Carlos Narváez-Narváez[1]
César-Jesús Pardo-Calvache[2]
Carlos-Eduardo Orozco-Garcés[3]

Recibido: julio de 2022 **Aceptado:** octubre de 2022

Para citar este artículo: Narváez-Narváez, J. C., Pardo-Calvache, C. J. y Orozco-Garcés, C. E. (2023). Deuda de la documentación en el desarrollo ágil de software: mapeo sistemático de la literatura. *Revista Científica*, *46*(1), 107-121. https://doi.org/10.14483/23448350.19670

Resumen

En el desarrollo de software, la documentación es un proceso continuo en el cual se especifica qué hace el sistema, cómo lo hace y para quién lo hace, describiendo todas sus características desde diferentes perspectivas para facilitar la comprensión de todos los interesados. Sin embargo, esto no siempre se logra, debido a malas prácticas de documentación, lo cual lleva a un tipo de deuda técnica conocida como *deuda de la documentación*, relacionada con documentación faltante, inconsistente o incompleta que degrada la comprensión del sistema, afectando gravemente su desarrollo, mantenimiento y evolución. Este fenómeno parece agravarse en los enfoques ágiles, ya sea por una mala interpretación de los valores y principios ágiles, enfocándose en la entrega continua de software funcional más que en la documentación exhaustiva, o por desconocimiento de buenas prácticas de documentación. Si bien existen iniciativas que describen de forma general las causas, efectos y buenas prácticas para mitigar la deuda de documentación, aún hace falta comprender el impacto de este tipo de deuda para la industria de software y profundizar en una definición más completa en el desarrollo ágil de software. En ese sentido, se presentan los resultados de un mapeo sistemático que brinda una base sólida de conocimiento sobre la deuda de la documentación en el desarrollo ágil de software y permite identificar brechas u oportunidades de investigación en este tema.

Palabras clave: desarrollo ágil de software; deuda de la documentación; deuda técnica; documentación de software; gestión de la deuda de la documentación.

1. Ph. D. Universidad del Cauca (Popayán-Cauca, Colombia). juanarvaez@unicauca.edu.co.
2. Ph. D. Universidad del Cauca (Popayán-Cauca, Colombia). cpardo@unicauca.edu.co.
3. Ph. D. Universidad del Cauca (Popayán-Cauca, Colombia). carlosorozco@unicauca.edu.co.

Abstract

In software development, documentation is a continuous process that specifies what the system does, how it does it, and for whom it does it, describing all its characteristics from different perspectives in order to facilitate the understanding of all the interested parties. However, this is not always achieved due to poor documentation practices, leading to a type of technical debt known as documentation debt, which is related to missing, inconsistent, or incomplete documentation that degrades the understanding of the system, severely affecting its development, maintenance, and evolution. This phenomenon seems to be aggravated in agile approaches, either due to a misunderstanding of agile values and principles by focusing on the continuous delivery of functional software rather than comprehensive documentation, or a lack of good documentation practices. Although there are initiatives that generally describe the causes, effects, and good practices to mitigate documentation debt, it is still necessary to understand the impact of this type of debt on the software industry and to delve into a more complete definition in agile software development. In this sense, the results of a systematic mapping are presented, which provide a solid base of knowledge about documentation debt in agile software development and allow identifying gaps or research opportunities in this topic.

Keywords: agile software development; documentation debt; documentation debt management; software documentation; technical debt.

Resumo

No desenvolvimento de software, a documentação é um processo contínuo em que se especifica o que o sistema faz, como faz e para quem faz, descrevendo todas as suas características sob diferentes perspectivas para facilitar o entendimento de todos os interessados. No entanto, isso nem sempre é alcançado devido às más práticas de documentação, levando a um tipo de dívida técnica conhecida como dívida de documentação relacionada à documentação ausente, inconsistente ou incompleta que degrada a compreensão do sistema, afetando severamente seu desenvolvimento, manutenção e evolução. Esse fenômeno parece ser agravado nas abordagens ágeis, seja por uma incompreensão dos valores e princípios ágeis, com foco na entrega contínua de software funcional em vez de documentação abrangente, ou pela falta de boas práticas de documentação. Embora existam iniciativas que geralmente descrevem as causas, efeitos e boas práticas para mitigar a dívida de documentação, ainda é necessário entender o impacto desse tipo de dívida para a indústria de software e aprofundar uma definição mais completa em desenvolvimento ágil. . Nesse sentido, são apresentados os resultados de um mapeamento sistemático que fornece uma base sólida de conhecimento sobre a dívida de documentação no desenvolvimento ágil de software e permite identificar lacunas ou oportunidades de pesquisa neste tema.

Palavras-chaves: desenvolvimento ágil de software; dívida de documentação; dívida técnica; documentação de software; gestão de dívida de documentação.

Introducción

Actualmente, las empresas de software dedican grandes esfuerzos en la definición de buenas prácticas, procesos y técnicas que permitan mejorar las actividades ejecutadas durante el ciclo de vida de sus proyectos. En particular, la documentación del sistema es una de las más críticas ya que: (i) estandariza el conocimiento presente en los proyectos, (ii) alinea los objetivos estratégicos de la organización con los deseos del cliente, (iii) facilita que los equipos tengan una comprensión holística del sistema (Aghajani *et al.*, 2019; Matturro *et al.*, 2019), y (iv) establece una dirección clara hacia la transparencia con los clientes, lo que permite la implementación de soluciones acordes con sus necesidades (Shmerlin *et al.*, 2015).

En este sentido, la falta de documentación consistente y eficaz que permita describir de manera clara y unívoca las características de un sistema se ha convertido en una problemática de alto impacto para la industria del software en general (Rios *et al.*, 2020b). Las consecuencias de esta problemática están relacionadas con: (i) falta de conocimiento estandarizado en los proyectos, (ii) degradación

de la integridad del sistema en el tiempo, (iii) dificultad para mantener una trazabilidad de nuevos cambios en etapas futuras del proyecto y (iv) poca claridad de los deseos del cliente (McBurney *et al.*, 2018).

Además, se desconocen estándares y modelos de facto para documentar el proyecto en todas sus fases, lo que incrementa la incidencia de malas prácticas relacionadas con la definición, implementación, seguimiento y entrega de los artefactos de documentación necesarios para el proyecto (McBurney *et al.*, 2018).

Con el objetivo de establecer estrategias que permitan estructurar de manera clara las actividades presentes durante el ciclo de vida de los proyectos existen enfoques tradicionales y ágiles que caracterizan las fases del ciclo de vida del software. Cada uno presenta ventajas y desventajas de acuerdo con las necesidades de documentación de la organización (Shylesh, 2017); por ejemplo: los enfoques tradicionales son rigurosos y recomiendan comenzar con el desarrollo del sistema solo después de haber documentado de forma exhaustiva las decisiones de diseño funcional y no funcional, lo cual establece un panorama claro desde etapas iniciales del proyecto, pero reduce la flexibilidad en etapas posteriores debido a que los cambios no contemplados que ocurran en el futuro no son bien recibidos (Shaydulin y Sybrandt, 2017; Islam y Ferworn, 2020).

Por otra parte, los enfoques ágiles proponen el principio "software funcional sobre documentación exhaustiva" (Hazzan y Dubinsky, 2014), priorizando la entrega de valor continuo en periodos cortos de tiempo (Alsaqqa *et al.*, 2020), no obstante, esto puede llevar a malas interpretaciones u omisiones y, como consecuencia, generar procesos subóptimos, inmaduros, poco claros y que son aplicados de manera incorrecta por los equipos (Martini *et al.*, 2020). Adicionalmente, la interpretación subjetiva de los principios ágiles puede llevar a malas prácticas para entregar prematuramente el sistema, resultando en una acumulación progresiva de deuda técnica que degrada el software con el paso del tiempo y que genera reprocesos, aumenta las brechas de mantenibilidad del sistema y baja la productividad de los equipos de trabajo (Rios *et al.*, 2020a).

Actualmente, los enfoques ágiles son muy utilizados en la industria de software y han apoyado procesos como: pruebas, gestión de proyectos, técnicas de programación o mantenimiento (State of Agile, 2021). No obstante, enfrentan desafíos para lograr que la documentación sea más comprensible, concreta, precisa y clara para todos los interesados sin descuidar la entrega de valor continuo (Rios *et al.*, 2020a) y garantizando una mayor transparencia de los artefactos de documentación (Shafiq y Waheed, 2018; Voigt *et al.*, 2016). Los procesos de negocio en las empresas deben documentarse para que sean fácilmente accesibles, usables, informativos, entendibles y auditables (Cappelli *et al.*, 2013), y así podrían ser más claros, coherentes, consistentes y evolucionarían correctamente (Martini *et al.*, 2020).

Ahora bien, la documentación es una de las actividades que menos interés despierta en los equipos (Rios *et al.*, 2020a), siendo evitada aproximadamente en el 60 % de los casos (Rios *et al.*, 2020b). Esto se debe principalmente a: (i) genera poco interés en los equipos porque puede tomar mucho tiempo, y (ii) la tendencia de los profesionales a pensar que otras actividades como el diseño, la implementación, pruebas o la arquitectura ofrecen un valor más visible para los interesados (Rios *et al.*, 2020a) dejando de lado la documentación. Por esto, además de la ejecución de procesos incompletos, ambiguos y poco claros, surge un fenómeno bastante nocivo para el software conocido como deuda de la documentación (en adelante DD), lo que a su vez conduce a otros tipos de deudas como: deuda de procesos (Martini *et al.*, 2019), deuda técnica relacionada con aspectos operativos (Behutiye *et al.*, 2017) y deuda social relacionada con la interacción entre los miembros del equipo.

En general, una deuda es una obligación adquirida de manera voluntaria para obtener un

beneficio a corto o mediano plazo, pero que debe pagarse en un tiempo determinado cumpliendo con los intereses que genera (Rios *et al.*, 2020a; Hazzan y Dubinsky, 2014). En el contexto de la Ingeniería de Software, la deuda se relaciona con aquellas tareas o responsabilidades que no fueron realizadas durante el ciclo de vida o que fueron deficientes, generando intereses como: dilatación de las entregas, disminución de la productividad, baja calidad o insatisfacción del cliente y que deben pagarse a través de reprocesos, sobrecostos, sobrecarga de trabajo (Rios *et al.*, 2020a), por ejemplo, el uso de malas prácticas de codificación puede llevar a futuras refactorizaciones y solución frecuente de incidentes que podrían ser evitados desde el principio mediante el uso de buenas prácticas de codificación, principios y patrones de diseño documentados correctamente (Li *et al.*, 2015; Alves *et al.*, 2016).

Por lo mencionado, y con el propósito de conocer el estado actual del conocimiento sobre deuda de la documentación y su impacto en el desarrollo ágil de software, se presentan los resultados de un mapeo sistemático de la literatura (en adelante MSL) que permitió recopilar información relevante sobre definiciones, validaciones, procesos, propuestas, estudios aplicados y métodos de investigación descritos en la literatura. Se evidencia una ambigüedad en la comprensión de la DD, en especial en el desarrollo ágil de software, debido a que es un tema poco explorado, imposibilitando abordar otras alternativas, como cuantificar el nivel de deuda en la documentación. Asimismo, fue posible encontrar aportes relacionados con la definición conceptual de la DD en el contexto de los enfoques tradicionales y ágiles, causas, efectos y estrategias para mitigarla o prevenirla aplicando buenas prácticas y herramientas tecnológicas que automatizan el proceso de documentación. Sin embargo, no se evidenciaron estudios que busquen medir, cuantificar o caracterizar la DD.

Finalmente, el resto del documento está organizado así: en la segunda sección se presenta la metodología de búsqueda. En la tercera sección se presenta el análisis de los resultados obtenidos en la búsqueda. Y en la cuarta sección se presentan las conclusiones y trabajos futuros.

Metodología

El MSL presentado sigue los lineamientos de Petersen *et al.* (2015); Kitchenham y Charters (2007) y Budgen *et al.* (2008), para las actividades: (i) definición de las preguntas de investigación; (ii) conducción de la búsqueda de artículos relevantes; (iii) selección de artículos primarios aplicando criterios de inclusión y exclusión; (iv) evaluación de la calidad de los artículos primarios; y (v) extracción de información. En la Figura 1 se presenta proceso, actividades, entradas y salidas obtenidas en la búsqueda.

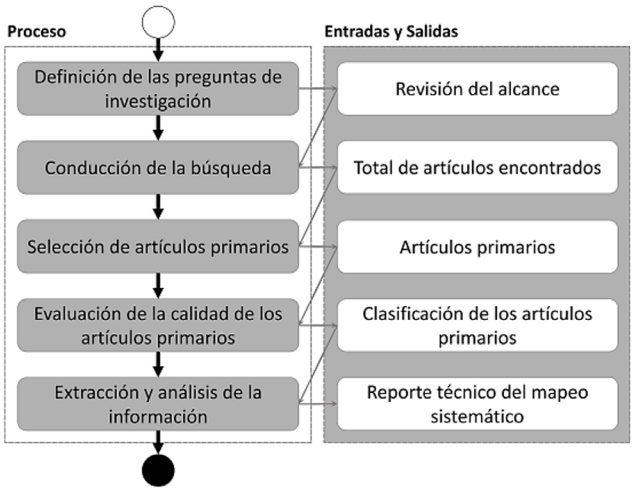

Figura 1. Proceso para el mapeo sistemático sugerido por Petersen *et al.* (2008).

Definición de las preguntas de investigación

Se definieron seis preguntas de investigación mediante el enfoque Goal-Question-Metric (GQM) (Basili y Caldiera, 1994), sin embargo, solo se consideraron el nivel conceptual, para establecer los objetivos de la revisión y el nivel operativo, para definir las preguntas que darán cumplimiento a los objetivos. A continuación, se presentan los

objetivos de búsqueda (en adelante OB) definidos en el primer nivel:

- **OB1.** Categorizar las evidencias para dimensionar el interés científico e investigativo de la DD durante el desarrollo ágil de software.
- **OB2.** Conocer las principales iniciativas científicas sobre DD en el desarrollo ágil de software y clasificarlas según el esquema propuesto por Wieringa *et al.* (2006) y la definición de criterios de calidad para establecer su relevancia.
- **OB3.** Recopilar información sobre definiciones conceptuales, propuestas, validaciones, procesos y métodos de investigación para identificar el grado de desarrollo de las iniciativas.

En la Tabla 1 se presentan las preguntas de investigación y el OB asociado para categorizar la información relevante sobre la DD en el desarrollo ágil de software.

Búsqueda de artículos relevantes

El proceso fue llevado a cabo mediante la siguiente cadena de búsqueda: *"documentation debt" OR "process debt" AND ("agile software projects" OR "agile software development" OR "agile development" OR "agile approach")*, la cual fue adaptada y ejecutada en siete bases de datos científicas: IEEE Xplore, Science Direct, Scopus, Google Scholar, Springer Link, ACM y Web Of Science. Se consideró una ventana de tiempo entre 2015 y mayo de 2022. La búsqueda se inició en 2015 debido a que fue el año cuando se evidencian los primeros trabajos enfocados en DD.

Criterios de inclusión y exclusión

La selección de artículos relevantes se basó en una revisión de tres niveles: (nivel 1) revisión del título, (nivel 2) revisión del resumen o abstract, introducción y conclusiones, y (nivel 3) revisión sobre el texto completo para determinar si el artículo cumplía con al menos uno de los siguientes criterios de inclusión (CI):

- **CI1.** Artículos cuyo tema principal sea la DD en el desarrollo ágil de software.
- **CI2.** Artículos cuyo tema se relacione con la documentación de software.

Tabla 1. Preguntas de investigación.

Preguntas de investigación	OB
P1: ¿Cuál es la distribución temporal de los artículos primarios?	OB1
Motivación: presentar una tendencia macro de la literatura entre 2015 y mayo de 2022.	
P2: ¿Cuáles son los tipos de investigación desarrollada?	OB2
Motivación: clasificar los diferentes tipos de investigación que se evidencian según el esquema de clasificación propuesto por Wieringa *et al.* (2006), p. ej.: investigación de validación, investigación de evaluación, propuesta de solución, artículos filosóficos, documentos de opinión o artículos de experiencia personal.	
P3: ¿Cuál es la calidad de los artículos primarios?	
Motivación: determinar la calidad de cada artículo seleccionado de acuerdo con la tabla II: Criterios para la evaluación de la calidad de artículos primarios.	
P4: ¿Qué tipos de soluciones se han propuesto?	OB3
Motivación: identificar el tipo de contribución en una o varias de las siguientes categorías: (i) definición conceptual, (ii) causas, efectos, impactos y limitaciones, (iii) métodos o técnicas de evaluación, (iv) herramientas tecnológicas, (v) validación en la industria, (vi) metodologías de documentación, (vii) otros.	
P5: ¿Qué resultados se han alcanzado con las propuestas realizadas?	
Motivación: identificar el impacto de las propuestas realizadas con base en los resultados obtenidos durante su validación en la industria del software.	
P6: ¿Qué beneficios y desafíos conlleva la investigación en el tema?	
Motivación: identificar la incidencia de la DD en el desarrollo ágil de software.	

- **CI3.** Artículos que traten temas relacionados con la deuda técnica.
- **CI4.** Artículos que hayan sido publicados en revistas, congresos o conferencias de prestigio con revisión por pares.

Posteriormente, para la selección de artículos primarios, se descartaron aquellos artículos que cumplieran con al menos uno de los siguientes criterios de exclusión (CE):

- **CE1.** Artículos duplicados (considerando solo el más completo y reciente).
- **CE2.** Artículos donde se aborde superficialmente el tema de investigación.
- **CE3.** Artículos donde no se aborden problemas relacionados con la deuda de la documentación de software durante el desarrollo ágil de software.
- **CE4.** Artículos de tipo debate o disponibles solo en forma de presentaciones o resumen.

Criterios de evaluación de la calidad

Con el propósito de medir la calidad de los artículos primarios y determinar su grado de pertinencia con respecto a la DD en el desarrollo ágil de software, se diseñó un método de evaluación basado en doce criterios con una puntuación de tres valores: no cumple (-1), cumple parcialmente (0) y sí cumple (+1) (Kitchenham y Charters, 2007). Así cada artículo puede obtener una calificación entre -12 y +12. No obstante, el objetivo de evaluar la calidad de los estudios es identificar aquellos que son más pertinentes para la investigación, por lo que una puntuación baja no es un criterio de exclusión. Los criterios de evaluación de la calidad se presentan en la Tabla 2.

Tabla 2. Criterios de evaluación de la calidad.

No	Criterio de calidad	Puntuación		
		+1	0	-1
1	El artículo se enfoca en investigar el fenómeno de la DD en el contexto del desarrollo ágil de software.	Sí	Parcialmente	No
2	En el artículo se ofrece una descripción clara del problema de investigación abordado.	Sí	Parcialmente	No
3	El artículo sigue un proceso de investigación estructurado y fundamentado.	Sí	Parcialmente	No
4	El artículo proporciona una definición clara sobre DD.	Sí	Parcialmente	No
5	El artículo propone un conjunto de elementos a considerar para evaluar la deuda de los artefactos de documentación en el ciclo de vida del desarrollo ágil de software.	Sí	Parcialmente	No
6	El artículo propone una forma de evaluar la DD en entornos de desarrollo ágil de software.	Sí	Parcialmente	No
7	El artículo expone de manera clara y detallada los resultados obtenidos tras validar su propuesta.	Sí	Parcialmente	No
8	El artículo presenta claramente las contribuciones de investigación hacia la industria o la academia.	Sí	Parcialmente	No
9	El artículo describe claramente la discusión de las limitaciones del proceso de investigación realizado y del análisis de los resultados obtenidos.	Sí	Parcialmente	No
10	El artículo describe claramente los trabajos futuros o alternativas de investigación.	Sí	Parcialmente	No
11	El artículo ha sido publicado en una revista, conferencia o congreso relevante. Se utilizó la clasificación de cuartiles propuestos por SCImago (s.f.) para clasificar las revistas y el ranking del Computing Research & Education (CORE) (s.f.) para congresos y conferencias.	Muy relevante	Relevante	No relevante
12	El artículo ha sido citado por otros autores (según el índice de citas de Google Scholar).	$n > 10$	$1 \leq n \leq 10$	$n = 0$

Acrónimos: Muy Relevante: cuartil Q1 para revistas y A* para congresos y conferencias. **Relevante:** cuartiles Q2 y Q3 para revistas y A o B para congresos y conferencias. **No Relevante:** cuartil Q4 para revistas y C o no clasificados para congresos y conferencias. **n:** cantidad de citas.

Estrategia de extracción de datos

Para realizar una extracción uniforme de la información relevante de los estudios primarios, se definió una ficha con los elementos necesarios para el análisis de los artículos. En la Tabla 3 se presenta la ficha definida para apoyar la estrategia de extracción de datos. Además, esta plantilla facilitó el proceso de análisis en etapas posteriores.

Ejecución de la búsqueda

Se realizó una iteración por cada base de datos: Google Scholar, Scopus, Science Direct, Springer Link, IEEE Xplore, ACM y Web Of Science. Sin embargo, cada motor de búsqueda cuenta con su propia parametrización, por lo cual fue necesario adaptar la cadena de búsqueda de acuerdo con estas configuraciones. En la Tabla 4 se presenta la cadena adaptada a cada base de datos.

En la Tabla 5 se presenta el resumen de los artículos encontrados. Tras aplicar la cadena de búsqueda, se encontraron 366 artículos; posteriormente, se aplicaron los criterios de inclusión, lo cual redujo el número total a 155 artículos considerados relevantes; finalmente, se aplicaron los criterios de exclusión, de donde resultaron 27 artículos primarios (en adelante AP). Adicionalmente, se aplicó un proceso de revisión hacia atrás (backward snowballing) (Jalali y Wohlin, 2012) a los AP con el propósito de identificar estudios que fueron omitidos en los resultados de la búsqueda inicial, como resultado, se identificaron cuatro estudios adicionales que fueron clasificados como AP. Y se incluyeron ocho estudios más como literatura gris. El detalle de los artículos primarios puede ser consultado en el siguiente enlace: https://bit.ly/3WAM4NT.

Tabla 3. Ficha de información relevante.

Identificación			
Título:			
DOI:		Número de citaciones:	
Fecha de publicación:		Conferencia o revista:	
Autores:			
Nombre	País	Universidad	Grupo de investigación
Resumen			
Descripción:			
Tipo de investigación [Clasificación de Wieringa *et al.* (2006)]: ☐ Investigación de validación ☐ Propuesta de solución ☐ Documento de opinión ☐ Investigación de evaluación ☐ Artículo filosófico o similar ☐ Experiencia personal			
Metodología de investigación:			
Tipo de solución(es) ofrecida(s): ☐ Definición conceptual ☐ Métodos de evaluación ☐ Validación en la industria ☐ Otros ☐ Causas, efectos y limitaciones ☐ Herramientas tecnológicas ☐ Métodos de documentación			
Propuesta:			
Evaluación de la propuesta:			
Problema abordado			
Elementos para la justificación			
Aspectos para destacar			

Tabla 4. Adaptación de la cadena de búsqueda.

Cadena de búsqueda ejecutada	Base de datos
"documentation debt" OR "process debt" AND ("agile software projects" OR "agile software development" OR "agile development" OR "agile approach")	Google Scholar
"documentation debt" OR "process debt" AND ("agile software projects" OR "agile software development" OR "agile development" OR "agile approach") AND (LIMIT-TO (PUBYEAR, 2022) OR LIMIT-TO (PUBYEAR, 2021) OR LIMIT-TO (PUBYEAR, 2020) OR LIMIT-TO (PUBYEAR, 2019) OR LIMIT-TO (PUBYEAR, 2018) OR LIMIT-TO (PUBYEAR, 2017) OR LIMIT-TO (PUBYEAR, 2016) OR LIMIT-TO (PUBYEAR, 2015)) AND (LIMIT-TO (SUBJAREA, "COMP")) AND (LIMIT-TO (LANGUAGE, "English"))	Scopus, Science Direct, Springer Link
("Full Text & Metadata":"documentation debt" OR "Full Text & Metadata":"process debt" AND ("Full Text & Metadata":"agile software projects" OR "Full Text & Metadata":"agile software development" OR "Full Text & Metadata":"agile development" OR "Full Text & Metadata":"agile approach"))	IEEE Xplore
[All: "documentation debt"] OR [All: "process debt"] AND [[All: "agile software projects"] OR [All: "agile software development"] OR [All: "agile development"] OR [All: "agile approach"]]] AND [Publication Date: (01/01/2015 TO 05/31/2022)]	ACM
ALL=(documentation debt OR process debt AND (agile software projects OR agile software development OR agile development OR agile approach)) Refinado por: CATEGORÍAS DE WEB OF SCIENCE: (COMPUTER SCIENCE SOFTWARE ENGINEERING) AND AÑOS DE PUBLICACIÓN: (2022 OR 2019 OR 2018 OR 2017 OR 2020)	Web Of Science

Tabla 5. Resultados de la búsqueda

Fuente	E	R	Re	P
Google Scholar	189	61	0	20
Scopus	44	13	13	0
Science Direct	18	12	7	1
Springer Link	13	8	7	0
IEEE Xplore	56	37	10	4
ACM	33	21	15	1
Web Of Science	13	5	2	1
Total	366	155	54	27

Acrónimos: E: encontrados, R: relevantes, Re: repetidos, P: primarios

Resultados

A continuación, se presenta el análisis de las respuestas a cada una de las preguntas de investigación. El análisis del aporte de cada artículo a las preguntas planteadas se presenta en el siguiente enlace: https://bit.ly/3YZYuRh.

P1: ¿Cuál es la distribución temporal de los artículos primarios?

De acuerdo con los resultados presentados en la Figura 2, se puede observar que la producción científica en el año 2015 fue del 12,9 % (A2, A10, A25, A28), en 2016 del 29 % (A4, A9, A11, A12, A24, A27, A29, A31), en 2017 del 6,45 % (A5, A15), en 2018 del 22,58 % (A3, A6, A7, A18, A23, A26, A30), en 2019 del 3,22 % (A14), en 2020 del 22,58 % (A1, A8, A13, A16, A17, A21, A22) y en el 2021 del 6,45 % (A19, A20).

En particular, se observó que en el 51,61 % de los AP se investiga la deuda de la documentación y su relación con la deuda técnica generada en requisitos, diseño, arquitectura, pruebas y despliegue; el 22,58 % de los estudios menciona aspectos de la documentación en el contexto del desarrollo de software, pero no describe su aplicación a través del uso de enfoques tradicionales o ágiles; finalmente, en el 12,9 % restante se proponen soluciones para gestionar la DD en entornos de trabajo que aplican enfoques ágiles.

P2: ¿Cuáles son los tipos de investigación desarrollada?

Según las características de la investigación desarrollada en los AP se realizó una clasificación basada en seis categorías: validaciones, evaluaciones, propuestas de solución, artículos filosóficos, artículos de opinión y artículos de experiencia personal (Wieringa *et al.*, 2006). De acuerdo con esto, los resultados obtenidos son los siguientes: 21 artículos (67,74 %) ejecutan métodos de evaluación

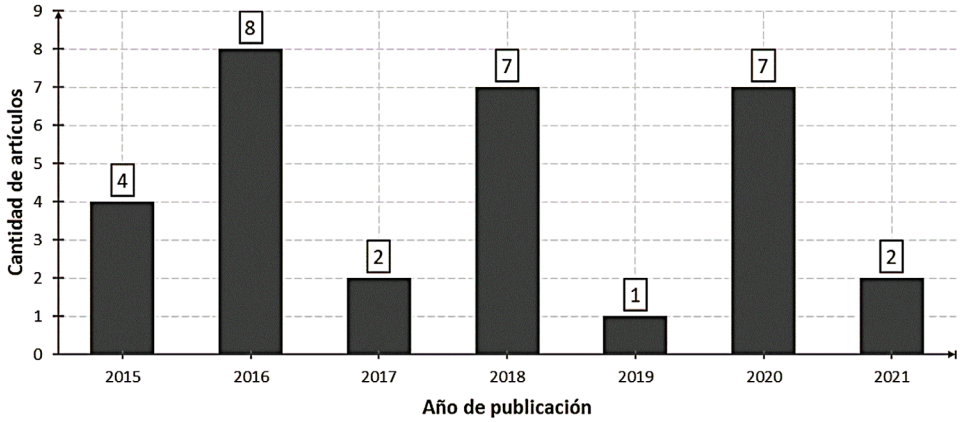

Figura 2. Cantidad de artículos por año.

a través de estudios de campo, estudios de caso, encuestas o combinaciones de ellos; 7 artículos (22,58 %) realizan procesos de validación a través de mecanismos como revisiones de la literatura y experimentos; 4 artículos (12,90 %) describen la experiencia personal presentada por los autores luego de realizar procesos de investigación en empresas de software; 2 artículos (6,45 %) realizan propuestas de solución en las cuales se proponen ontologías y definiciones conceptuales enfocadas en la documentación de software y su relación con la deuda técnica. Finalmente, no se identificaron estudios de tipo filosófico o de opinión. Lo mencionado se muestra en la Figura 3, pero los detalles de la clasificación de los artículos pueden ser consultados en el siguiente enlace: https://bit.ly/3Z5tmzV.

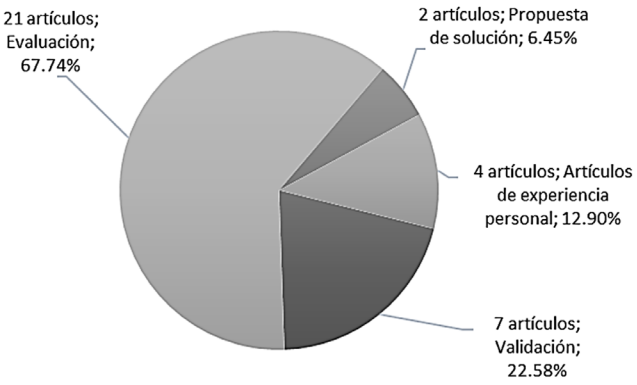

Figura 3. Distribución porcentual y cantidad de artículos por tipo de investigación

P3: ¿Cuál es la calidad de los artículos primarios?

La Figura 4 presenta la agrupación de artículos según la calificación obtenida tras la suma de los valores individuales propuestos para llevar a cabo la evaluación de la calidad de los estudios. De acuerdo con los resultados, se puede observar que: el 6,45 % de los artículos (A11, A4) lograron la calificación más alta con 9 y 8 puntos respectivamente, el 3,23 % (A28) con 7 puntos, el 16,13 % (A1, A2, A20, A21, A27) con 6 puntos, el 6,45 % (A24, A25) con 5 puntos, el 6,45 % (A23, A29) con cuatro puntos, el 16,13 % (A3, A14, A17, A22, A26) con 3 puntos, el 9,68 % (A8, A12, A30) con 2 puntos, el 6,45 % (A13, A18) con 1 punto, el 16,13 % (A5, A10, A15, A16, A31) con 0 puntos y el 3,23 % (A19, A7, A9, A6) con -2, -3, -5 y -6 puntos respectivamente.

Los detalles de la clasificación obtenida por los artículos primarios pueden ser consultados en el siguiente enlace: https://bit.ly/3PPxQpN.

P4: ¿Qué tipos de soluciones se han propuesto?

De acuerdo con los resultados presentados en la Figura 5, fue posible observar que el 38,71 % de los estudios (A1, A2, A3, A4, A6, A10, A16, A19, A20, A28, A29, A31) proponen definiciones conceptuales que buscan comprender, estructurar y estandarizar el conocimiento en torno a la DD en el desarrollo ágil de software. Por otro lado, el 45,16 % de los estudios (A1, A2, A3, A5, A7, A8,

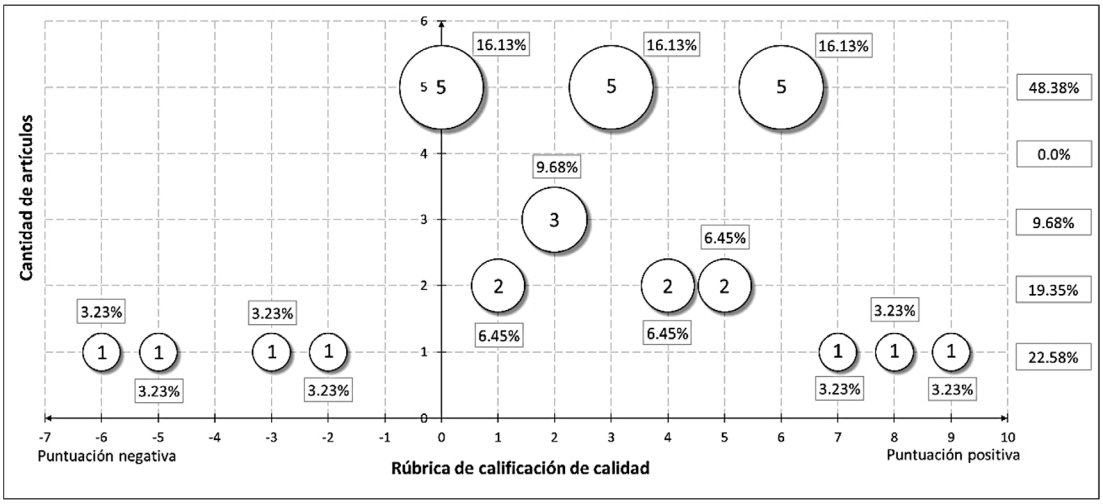

Figura 4. Agrupación de artículos según su puntuación de calidad.

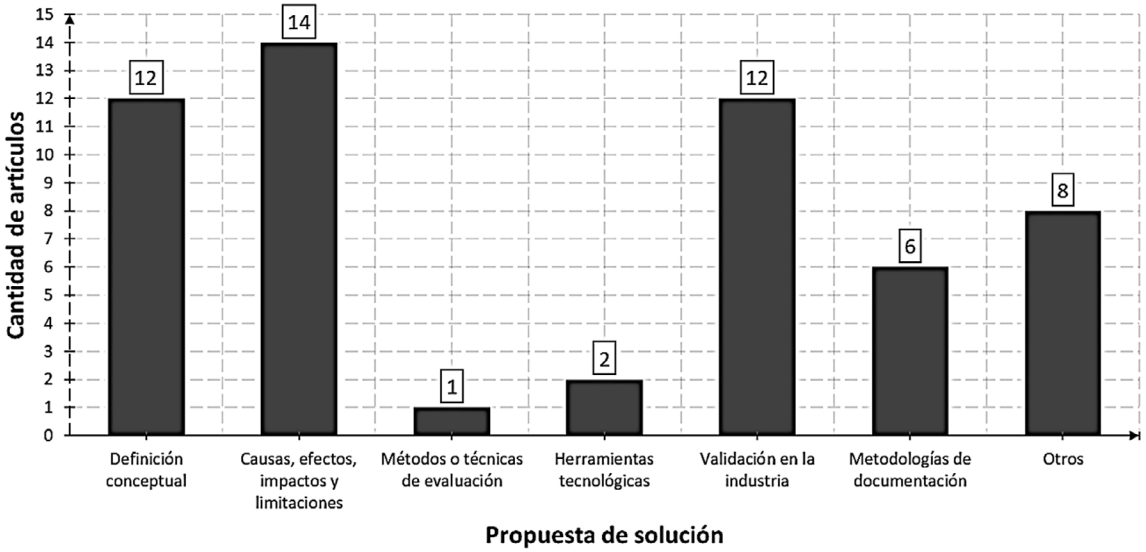

Figura 5. Distribución de estudios por tipo de solución

A9, A10, A12, A15, A17, A18, A20, A23) se enfoca en identificar causas, efectos, limitaciones e impactos que resultan como consecuencia de la DD en el desarrollo ágil de software, por ejemplo: en A1 y A17 se identifican 23 causas que contribuyen a la DD en el desarrollo ágil de software. De esta manera se han identificado un total de 45 causas que contribuyen a la DD y 42 efectos derivados como: código incorrecto, baja calidad del software, retraso en la entrega, entre otros. El detalle de las causas y efectos puede consultarse en https://bit.ly/3WBY5mz y https://bit.ly/3WZGYdU, respectivamente; por otro lado, en A4 y A12 se presentan los resultados después de analizar el impacto de la DD en los proyectos software en términos de costo, tiempo, productividad y degradación del sistema.

Con relación a la definición de procesos, técnicas, métodos y soluciones, se identificó que: 1 estudio (3,23 %) (A7) presenta un conjunto de métricas para medir la DD, sin embargo, no se presenta el detalle de las métricas. Por otro lado; 2

estudios (6,45 %) (A3, A7) proponen herramientas para la prevención de la DD durante la etapa de especificación de requisitos ágiles y para permitir la visualización y accesibilidad de la documentación en los proyectos, respectivamente; 12 estudios (38,71 %) (A3, A5, A7, A14, A15, A17, A20, A21, A22, A25, A26, A27) realizan estudios aplicados en la industria a través de cuestionarios que permiten identificar aspectos que generan deuda de proceso y deuda de la documentación en empresas desarrolladoras de software; 6 estudios (19,35 %) (A5, A9, A12, A13, A24, A25) proponen metodologías para llevar a cabo la gestión de la documentación a través de la identificación de artefactos que generan deuda técnica y el uso de lenguaje natural; finalmente, 8 estudios (25,81 %) (A1, A5, A6, A11, A19, A21, A24, A31) presentan otros enfoques de solución, tales como: prácticas de documentación para mitigar la DD (A1), estudio de la DD a nivel de código fuente a través de comentarios de código (A10, A22, A23, A27), antipatrones de documentación (A19) y un modelo de vocabulario para contextualizar los conceptos de DD (A21).

P5: ¿Qué resultados se han alcanzado con las propuestas realizadas?

De acuerdo con los resultados, el 38,71 % de los artículos primarios proponen algún tipo de definición sobre DD en el desarrollo de software. En particular, los artículos A1, A2, A3, A19, A20 y A28 relacionan la DD con documentación faltante, inconsistente, desactualizada o incompleta; en A1, A28 y A29 consideran que la DD se refiere a las actividades pendientes en cualquier etapa del ciclo de vida del software; y en A3 se afirma que la deuda conlleva a un esfuerzo adicional para mantener y evolucionar el software. Por otra parte, en A4 y A16 se plantea cómo una mala interpretación e implementación de los enfoques ágiles puede crear el escenario propicio para la DD, resultando en documentación de baja calidad o carencia absoluta de documentación en los proyectos, no obstante, los estudios concluyen que una interpretación correcta de la filosofía propuesta por las metodologías ágiles permite escribir documentación eficaz, de alto valor, precisa, comprensible y capaz de reflejar completamente el software que se construye, evitando, como se afirma en A6, A10 y A31, documentación pesada, estresante, estricta, rígida, extensa y compleja.

No obstante, de acuerdo con los resultados presentados en A6, las problemáticas relacionadas con la DD no afectan solo a los enfoques ágiles. De acuerdo con los resultados presentados por los autores, los enfoques tradicionales son poco flexibles debido a que el proceso de desarrollo debe comenzar si y solo si toda la documentación está completa, lo cual incurre en altos impactos cada vez que se requiera un cambio durante etapas posteriores. Los autores concluyen que es necesario lograr un equilibrio entre las necesidades de documentación y el enfoque utilizado para evitar escenarios relacionados con documentación escasa y desactualizada que terminan generando DD, la cual se manifestará en reprocesos a mediano y largo plazo.

Por otro lado, se identificaron 45 factores que causan la DD en el desarrollo ágil de software, por ejemplo: no utilizar estándares o buenas prácticas de documentación, requisitos poco detallados e insuficientes del sistema, documentación inadecuada, documentación incompleta, entre otros; el detalle de las causas puede consultarse en el siguiente enlace: https://bit.ly/3WBY5mz. Además, se identificaron 42 efectos que son el resultado de malas prácticas asociadas con la documentación en el desarrollo ágil de software y que generan DD, entre las cuales se destacan: código incorrecto y sin comentarios, reducción de la capacidad del software para ser mantenible, pérdida de conocimiento, baja calidad del software, sobrecosto en tiempo, dinero y esfuerzo, falta de entendimiento y claridad sobre los requisitos del sistema. El detalle de los efectos puede ser consultado en https://bit.ly/3WZGYdU.

Finalmente, se identificaron 54 buenas prácticas que buscan prevenir, gestionar o mitigar la DD a través de estrategias como: comentar el código,

definir procesos y buenas prácticas de documentación, documentar requisitos funcionales y no funcionales. En particular, 47 prácticas (87,03 %) se enfocan en la prevención de la DD, y 7 prácticas (12,96 %) en gestionar la DD cuando ya existe en los proyectos. El detalle de las prácticas identificadas se puede consultar en el siguiente enlace: https://bit.ly/3Z575Ck.

P6: ¿Qué beneficios y desafíos conlleva la investigación en el tema?

En la literatura analizada se evidencian múltiples beneficios y desafíos relacionados con la DD en el desarrollo ágil de software. Inicialmente, se observó que la DD trae beneficios a corto plazo para las organizaciones, ya que permite reducir los tiempos de entrega de manera considerable; no obstante, el beneficio inicial se ve afectado en el futuro debido a que se deben realizar reprocesos y sobreesfuerzos al momento de integrar nuevas funcionalidades al sistema, comprometiendo su capacidad de escalar y evolucionar con el paso del tiempo (Charalampidou *et al.*, 2018). En este sentido, a medida que se agregan nuevos defectos y malas prácticas, se presenta un fenómeno conocido como "acumulación de la deuda", es decir, los intereses generados como resultado de las actividades que no se hicieron o que están incompletas y que con el paso del tiempo acumulan niveles de deuda nocivos para las empresas que pueden llegar a ser imposibles de pagar. Sin embargo, la evidencia encontrada en la literatura permite concluir que el concepto de deuda de la documentación es relativamente nuevo (Rios *et al.*, 2020a) y es abordado como una metáfora del concepto de deuda presente en el aspecto jurídico y económico.

De acuerdo con lo anterior, se hace necesario investigar nuevas formas o prácticas de documentación para reducir la DD (Shmerlin *et al.*, 2015), sobre todo en escenarios como el desarrollo global de software (DGS), en donde las distancias geográficas, las diferencias de horario y costumbres culturales pueden reducir la formalidad en la documentación, o incluso generar una escasa o nula visibilidad del proyecto y requisitos poco claros (Portela y Borrego, 2016). Por otro lado, la falta de documentación o la documentación inadecuada durante el proceso de desarrollo ágil (Kamal *et al.*, 2020) resulta en un aumento de la complejidad en el proceso de gestión del cambio presente en el ciclo de vida de los proyectos, afectando la capacidad del sistema para evolucionar y escalar con el paso del tiempo.

Finalmente, como lo afirman Sinha *et al.* (2020), aunque la correcta documentación del software se ha convertido en uno de los factores de éxito más importantes para que las organizaciones logren escalar sus desarrollos de manera global cuando se involucran enfoques ágiles, esto representa un desafío muy grande para las empresas debido a que les exige documentar solo aquello que les sea necesario, útil y que explique adecuadamente el sistema, sin perder de vista los principios ágiles que priorizan la entrega de valor por encima de la documentación excesiva.

Conclusiones

Los hallazgos sugieren un interés de la comunidad científica en esta temática porque impacta en el éxito del software, especialmente en su mantenimiento y evolución.

Fue posible evidenciar diferentes definiciones de DD relacionadas con documentación faltante, inconsistente, desactualizada o incompleta. Sin embargo, carecen de profundidad ya que no han considerado otros aspectos que podrían ser importantes como usabilidad, accesibilidad, entendibilidad, comprensibilidad y auditabilidad.

Los enfoques ágiles son una alternativa flexible que permite actualizar la documentación ante los cambios que puedan surgir durante el ciclo de vida del software. Por lo tanto, es necesario que las investigaciones sobre DD en el desarrollo ágil de software permitan establecer un equilibrio entre la documentación realmente necesaria e importante y los enfoques ágiles, manteniendo la coherencia con uno de los valores del manifiesto ágil, el cual

establece "software funcional sobre documentación exhaustiva".

Por otro lado, se han identificado causas, efectos y buenas prácticas relacionadas con la DD en el desarrollo ágil de software; sin embargo, no se evidenciaron iniciativas enfocadas a medición, gestión o automatización de las actividades relacionadas con la documentación que permitan cuantificar, estimar o predecir la DD.

Es importante considerar que la prevención de la DD es más económica que pagar la deuda una vez sus intereses se manifiestan. Esta prevención puede llevarse a cabo gracias al seguimiento adecuado del proyecto, la adopción de buenas prácticas de documentación, requisitos bien definidos, procesos óptimos que se revisen y actualicen periódicamente, un diseño de pruebas adecuado y una capacitación constante del personal involucrado en las tareas relacionadas con la documentación del sistema mientras se construye. Asimismo, se podría contemplar la inclusión de soluciones basadas en herramientas de software o metodologías que posibiliten la detección y corrección de procesos subóptimos de documentación, malas prácticas, problemas de interacción entre los miembros del equipo.

Finalmente, como trabajo futuro se propone: (i) ampliar la definición conceptual sobre DD considerando otras características como usabilidad, informatividad, accesibilidad, auditabilidad y entendibilidad y con esto precisar en los aspectos que podrían causarla, (ii) caracterizar los atributos necesarios que se deben considerar para cuantificar la DD en proyectos de software ágil y (iii) llevar a cabo la definición de mecanismos que permitan predecir y cuantificar el nivel de deuda de la documentación que se generan durante el desarrollo ágil de software.

Agradecimientos

Los autores agradecen el apoyo recibido por parte de la Universidad del Cauca (Popayán), donde actualmente se desempeñan como Profesor de Cátedra y Profesor Titular, respectivamente.

Contribución de autores:

Juan-Carlos Narváez-Narváez: Curación de datos, Análsis formal, Investigación, Metodología, Visualización, Escritura – revisión y edición.
César-Jesús Pardo-Calvache: Conceptualización, Metodología, Administrador de proyectos, Supervisión, Validación.
Carlos-Eduardo Orozco-Garcés: Escritura – Borrador original

Referencias

Aghajani, E., Nagy, C., Vega-Márquez, O. L., Linares-Vásquez, M., Moreno, L., Bavota, G., Lanza, M. (2019). Software documentation issues unveiled. En *IEEE/ACM 41st International Conference on Software Engineering (ICSE)*, 1199-1210. https://doi.org/10.1109/ICSE.2019.00122

Alsaqqa, S., Sawalha, S., Abdel-Nabi, H. (2020). Agile software development: Methodologies and trends. *International Journal of Interactive Mobile Technologies*, *14*(11), 246-270. https://doi.org/10.3991/ijim.v14i11.13269

Alves, N., Mendes, T., de Mendonça, M., Spinola, R., Shull, F., Seaman, C. (2016). Identification and management of technical debt: A systematic mapping study. *Information and Software Technology*, *70*, 100-121. https://doi.org/10.1016/j.infsof.2015.10.008

Basili, V. R., Caldiera, G. (1994). Goal Question Metric Paradigm. *Encyclopedia of Software Engineering - 2 Volume Set*, 528-532. https://www.cs.umd.edu/~basili/publications/technical/T89.pdf

Behutiye, W. N., Rodríguez, P., Oivo, M., Tosun, A. (2017). Analyzing the concept of technical debt in the context of agile software development: A systematic literature review. *Information and Software Technology*, *82*, 139-158. https://doi.org/10.1016/j.infsof.2016.10.004

Budgen, D., Turner, M., Brereton, P., Kitchenham, B. (2008). Using Mapping Studies in Software Engineering. En *Proceedings of the 20th Annual Workshop of the Psychology of Programming Interest Group*.

Cappelli, C., Engiel, P., Araujo, R. de, do Prado Leite, J. (2013). Managing transparency guided by a maturity model. En *3rd Global Conference on Transparency Research HEC*.

Charalampidou, S., Ampatzoglou, A., Chatzigeorgiou, A., Tsiridis, N. (2018). Integrating traceability within the IDE to prevent requirements documentation debt. En *44th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, 421-428. https://doi.org/10.1109/SEAA.2018.00075

CORE Inc. (s.f.). CORE Conference Ranks. http://portal.core.edu.au/conf-ranks/

Hazzan, O., Dubinsky, Y. (2014). The agile manifesto. In: *Agile Anywhere: Essays on Agile Projects and Beyond* (pp. 9-14). Springer. https://doi.org/10.1007/978-3-319-10157-6_3

Islam, Z., Ferworn, A. (2020). A Comparison between agile and traditional software development methodologies. *Global Journal of Computer Science and Technology*, *20*(2), 7-42. https://doi.org/10.34257/gjcstcvol20is2pg7

Jalali, S., Wohlin, C. (2012). Systematic literature studies: Database searches vs. backward snowballing. En *ESEM '12: Proceedings of the ACM-IEEE International Symposium on Empirical Software Engineering and Measurement*, 29-38. https://doi.org/10.1145/2372251.2372257

Kamal, T., Zhang, Q., Akbar, M. A., Shafiq, M., Gumaei, A., Alsanad, A. (2020). Identification and prioritization of agile requirements change management success factors in the domain of global software development. *IEEE Access*, *8*, 44714-44726. https://doi.org/10.1109/ACCESS.2020.2976723

Kitchenham, B., Charters, S. (2007). *Guidelines for performing Systematic Literature Reviews in Software Engineering, Version 2.3*. EBSE Technical Report, Keele University, & University of Durham, UK

Li, Z., Avgeriou, P., Liang, P. (2015). A systematic mapping study on technical debt and its management. *Journal of Systems and Software*, *101*, 193-220. https://doi.org/10.1016/j.jss.2014.12.027

Martini, A., Besker, T., Bosch, J. (2020). Process debt: A first exploration. En *27th Asia-Pacific Software Engineering Conference (APSEC)*, 316-325. https://doi.org/10.1109/APSEC51365.2020.00040

Martini, A., Stray, V., Moe, N. B. (2019). Technical-, social- and process debt in large-scale agile: An exploratory case-study. In: R. Hoda (Ed.), *Agile Processes in Software Engineering and Extreme Programming - Workshops* (pp. 112-119). Springer. https://doi.org/10.1007/978-3-030-30126-2_14

Matturro, G., Raschetti, F., Fontán, C. (2019). A systematic mapping study on soft skills in software engineering. *Journal of Universal Computer Science*, *25*(1), 16-41. https://doi.org/10.3217/jucs-025-01-0016

McBurney, P. W., Jiang, S., Kessentini, M., Kraft, N. A., Armaly, A., Mkaouer, M. W., McMillan, C. (2018). Towards prioritizing documentation effort. *IEEE Transactions on Software Engineering*, *44*(9), 897-913. https://doi.org/10.1109/TSE.2017.2716950

Petersen, K., Feldt, R., Mujtaba, S., Mattsson, M. (2008). Systematic mapping studies in software engineering. En *EASE'08: Proceedings of the 12th international conference on Evaluation and Assessment in Software Engineering*, 68-77

Petersen, K., Vakkalanka, S., Kuzniarz, L. (2015). Guidelines for conducting systematic mapping studies in software engineering: An update. *Information and Software Technology*, *64*, 1-18. https://doi.org/10.1016/j.infsof.2015.03.007

Portela, L. T., Borrego, G. (2016). Scrumconix: Agile and documented method to AGSD. En *IEEE 11th International Conference on Global Software Engineering (ICGSE)*, 195-196. https://doi.org/10.1109/ICGSE.2016.39

Rios, N., Mendes, L., Cerdeiral, C., Magalhães, A. P. F., Perez, B., Correal, D., Astudillo, H., Seaman, C., Izurieta, C., Santos, G., Oliveira Spínola, R. (2020a). Hearing the voice of software practitioners on causes, effects, and practices to deal with documentation debt. In: N. Madhavji, L. Pasquale, A.

Ferrari, S. Gnesi (Eds.), *Requirements Engineering: Foundation for Software Quality* (pp. 55-70). Springer. https://doi.org/10.1007/978-3-030-44429-7_4

Rios, N., Spínola, R. O., Mendonça, M., Seaman, C. (2020b). The practitioners' point of view on the concept of technical debt and its causes and consequences: a design for a global family of industrial surveys and its first results from Brazil. *Empirical Software Engineering*, *25*(5), 3216-3287. https://doi.org/10.1007/s10664-020-09832-9

SCImago. (s.f.). SJR - SCImago Journal & Country Rank. https://www.scimagojr.com/

Shafiq, M., Waheed, U. (2018). Documentation in agile development: A comparative analysis. En *IEEE 21st International Multi-Topic Conference (INMIC)*, 1-8. https://doi.org/10.1109/INMIC.2018.8595625

Shaydulin, R., Sybrandt, J. (2017). To agile, or not to agile: A comparison of software development methodologies. *arXiv:1704.07469v1*. https://doi.org/10.48550/arXiv.1704.07469

Shmerlin, Y., Hadar, I., Kliger, D., Makabee, H. (2015). To document or not to document? An exploratory study on developers' motivation to document code. In: A. Persson, J. Stirna (Eds.), *Advanced Information Systems Engineering Workshops* (pp. 100-106). Springer. https://doi.org/10.1007/978-3-319-19243-7_10

Shylesh, S. (2017). A study of software development life cycle process models. *SSRN*. https://doi.org/10.2139/ssrn.2988291

Sinha, R., Shameem, M., Kumar, C. (2020). SWOT: Strength, weaknesses, opportunities, and threats for scaling agile methods in global software development. En *ISEC 2020: Proceedings of the 13th Innovations in Software Engineering Conference on Formerly known as India Software Engineering Conference*. https://doi.org/10.1145/3385032.3385037

State of Agile. (2021). *State of Agile Report*. https://stateofagile.com/#ufh-c-7027494-state-of-agile

Voigt, S., von Garrel, J., Müller, J., Wirth, D. (2016). A study of documentation in agile software projects. En *ESEM'16: Proceedings of the 10th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement*, 1-6. https://doi.org/10.1145/2961111.2962616

Wieringa, R., Maiden, N., Mead, N., Rolland, C. (2006). Requirements engineering paper classification and evaluation criteria: A proposal and a discussion. *Requirements Engineering*, *11*(1), 102-107. https://doi.org/10.1007/s00766-005-0021-6