Hacia el desarrollo de un modelo adaptativo para gestionar la deuda social en entornos de desarrollo de software ágil distribuido

Eydy del C. Suarez Brieva[1]

[1]Universidad Popular del Cesar
80, Valledupar – Cesar- Colombia

eydysuarez@unicesar.edu.co

Abstract. *The misalignment of socio-technical congruence in software development teams—understood as the disconnect between social and technical factors—can lead to suboptimal decisions and unforeseen costs, giving rise to the concept of social debt. Through a systematic literature review, various solutions for detecting social debt have been identified; however, these approaches operate in isolation. Therefore, this study proposes the development of an adaptive model to manage social debt in distributed agile teams, integrating its causes, effects, and mitigation strategies. The model is grounded in an Action Research approach structured into four iterative cycles.*

Resumen. *La desalineación de la congruencia sociotécnica en equipos de desarrollo de software, entendida como la desarticulación entre factores sociales y técnicos, puede generar decisiones subóptimas que conllevan a costos imprevistos dando origen al concepto de deuda social. Mediante una revisión sistemática de la literatura, se identificaron soluciones para detectar la deuda social; sin embargo, estas propuestas operan de forma independiente. Por ello, se propone el desarrollo de un modelo adaptativo para gestionarla en equipos ágiles distribuidos, integrando causas, efectos y estrategias. El modelo se sustenta en un enfoque de Investigación-Acción estructurado en cuatro ciclos.*

1. Introducción

La Ingeniería de Software es una disciplina centrada en actividades realizadas mayormente por personas [Pantaleo and Rinaudo, 2015]. Sin embargo, la IA generativa ha cobrado importancia al automatizar tareas como la escritura de código, la simulación de escenarios complejos y la detección y corrección de errores, lo que mejora la productividad y reduce los tiempos de respuesta [Terragni *et al*., 2025]. Tradicionalmente, estas tareas eran ejecutadas por grupos organizados de trabajo especializado y a pesar del surgimiento de la IA generativa y de sus beneficios, el factor humano en el desarrollo de software sigue siendo esencial en la planificación, asignación de recursos y consideraciones *éticas y sociales* en los proyectos de software. El desarrollo de software requiere profesionales con habilidades sociales y técnicas, capaces de compartir conocimiento, implementar buenas prácticas, definir roles y responsabilidades, y fomentar la transparencia en su trabajo. Aunque la IA generativa mejora la eficiencia de los equipos, no reemplazará por completo el talento humano, ya que ciertas actividades requieren pensamiento lógico complejo, creatividad en el diseño de software, interacción con clientes para comprender sus necesidades, resolución de problemas imprevistos y

consideraciones éticas y regulatorias [de Campos *et al*., 2024]. Además, la supervisión humana sigue siendo clave para mitigar sesgos en el desarrollo de software.

Cabe destacar que el Reporte del Estado de la Agilidad 2023 [State of Agile, 2023] destaca el crecimiento del trabajo remoto, impulsado por el Desarrollo Global de Software (DGS), este modelo plantea desafíos en la comunicación y colaboración entre equipos distribuidos. Factores como la diferencia horaria, el idioma y las costumbres pueden dificultar la coordinación en comparación con equipos co-localizados [García *et al*., 2022]. Para enfrentar estos retos, las empresas deben garantizar una comunicación efectiva entre sus miembros. En consecuencia, una comunicación, coordinación o cooperación deficiente y la desalineación de los factores sociales y técnicos entre los miembros del equipo, puede generar "costos adicionales en el estado del proyecto, originado por decisiones sociotécnicas subóptimas", lo cual da lugar a la deuda social en adelante DS [Tamburri *et al*. 2013]. La DS debe ser supervisada y mitigada oportunamente, ya que su prolongación puede afectar significativamente la eficiencia y productividad del equipo. Ignorar su importancia puede desencadenar en conflictos personales e interpersonales, barreras lingüísticas, culturales, desconfianza, irritabilidad, así como dificultades en la toma de decisiones, lo que impacta directamente en el bienestar emocional y social de los miembros del equipo [Lambiase, 2024]. Es de anotar, que la DS también se manifiesta en problemas técnicos visibles en artefactos de software, como documentación incomprensible y componentes de software incompletos [Caballero-Espinosa *et al*., 2023]. Además, la automatización impulsada por la IA generativa podría reducir la interacción y el rol de los desarrolladores, afectando su motivación y aumentando la incertidumbre laboral.

Considerando lo expuesto anteriormente, se realizó una Revisión Sistemática de la Literatura (RSL) para analizar las causas, efectos y estrategias de prevención de la DS en equipos ágiles distribuidos. A partir del análisis de los 47 artículos primarios, se identificaron herramientas, modelos y métricas; sin embargo, ninguna de las propuestas ofrece una solución integral que combine la identificación de la DS con estrategias específicas para su mitigación. Teniendo en cuenta esta brecha, se propone el desarrollo de un modelo adaptativo para gestionar la DS en equipos de desarrollo de software ágil distribuido. Para ello, se incorpora una ontología que representa formalmente el conocimiento asociado a la DS, esta permite estructurar, conceptualizar y relacionar de manera explícita los elementos como causas, efectos, riesgos, métricas y estrategias de mitigación. Al proporcionar una representación semántica, la ontología se convierte en un componente esencial del modelo adaptativo, ya que facilita la integración y análisis de datos mediante técnicas de analítica de datos. De este modo, la ontología no solo mejora la capacidad de detección y comprensión de la DS, sino que también permite la adaptación de estrategias de intervención personalizadas ajustadas al contexto especifico de cada equipo ágil.

Además de la introducción presentada, esta propuesta se divide en 8 secciones. La Sección 2 expone los objetivos de investigación. La Sección 3 presenta un resumen del conocimiento actual del dominio del problema. En la Sección 4 se presenta la metodología de investigación. La Sección 5 solución propuesta y resultados. La Sección 6 aborda la descripción de las actividades pendientes. La Sección 7 ilustra implicaciones y en la Sección 8 las referencias bibliográficas.

2. Objetivos del proyecto de investigación

A continuación, se presenta el objetivo general y los objetivos específicos definidos para la investigación.

2.1. Objetivo General

Construir un modelo adaptativo como apoyo a la gestión de la deuda social en equipos de desarrollo de software ágil distribuido.

2.2. Objetivos específicos

OE1. Caracterizar los elementos fundamentales que contribuyen a la deuda social en equipos de desarrollo de software ágil mediante el análisis de la literatura existente, con el fin de clarificar los conceptos, factores, métodos y relaciones que inciden en su manifestación y gestión.

OE2. Desarrollar una ontología de dominio específico con el propósito de organizar y utilizar el conocimiento relacionado con la gestión de la deuda social en equipos de desarrollo de software ágil distribuido, haciendo uso de los términos y relaciones identificadas en el OE1.

OE3. Construir un modelo adaptativo para identificar las causas y efectos que genera la DS en equipos de desarrollo de software ágil distribuido y, con base en los efectos detectados, establecer estrategias de mitigación para su gestión. Este modelo analizará en tiempo real la acumulación de deuda social, ajustando dinámicamente sus estrategias según el contexto organizacional. Se fundamentará en el análisis de literatura (OE1) y la creación de una ontología (OE2).

OE4. Evaluar el modelo adaptativo de gestión propuesto en el OE3 mediante la validación de juicio de expertos, que permita identificar áreas de mejora y proponer acciones correctivas en el modelo propuesto.

3. Resumen del conocimiento actual del dominio del problema

La DS se define como un "costo adicional en el estado del proyecto, derivado de decisiones sociotécnicas subóptimas, que resultan de la ruptura de la congruencia sociotécnica (STC)" [Tamburri *et al*., 2015]. La falta de identificación, evaluación y adopción de medidas para controlar o minimizar los costos generados por la DS, puede afectar negativamente el bienestar mental y emocional de los trabajadores [Saeeda *et al*., 2024], dando origen a lo que en el entorno laboral se conoce como: factores de riesgo psicosocial. Según la [Organización Internacional del Trabajo, 2013], estos riesgos surgen de "las interacciones entre el trabajo, su entorno, la satisfacción laboral y las condiciones organizativas", además de factores como las "capacidades del trabajador, necesidades, cultura y situación personal, que pueden influir en su salud, rendimiento y satisfacción laboral".

Desde esta perspectiva, la DS se refiere a la ausencia de acciones y servicios que afectan el bienestar de los equipos, sea co-localizados o distribuidos. A partir de la RSL, se identificó herramientas para prevenir y gestionar la DS, se destacó la importancia del seguimiento periódico a las interacciones humanas y al desarrollo de habilidades blandas y técnicas de los individuos. Se definieron 2 ejes de estudio: (i) tipos de solución, que incluyen: (a) (14) modelos conceptuales y estadísticos [Yilmaz, O'Connor, and Clarke 2016], [Anwar *et al*., 2019], (b) (3) marco de trabajo [Catolino *et al*., 2021], (c) (3) herramientas automatizadas, (d) (4) modelos predictivos [Almarimi *et al*., 2022) y (e) (2)

otros modelos [Khan *et al*., 2019], y (ii) las categorías de análisis abarcan: (a) (11) métricas para determinar la DS, (b) (9) causas de la DS, (c) (2) efectos de la DS y (d) (6) gestión de la DS. Se puede señalar que predomina el uso de modelos conceptuales y estadísticos, evidenciando un enfoque basado en métricas para detectar la DS en equipos de desarrollo de software ágil distribuido.

4. Esbozo de la metodología de investigación

Para esta propuesta, se adopta la metodología Investigación-Acción, debido a su enfoque práctico y participativo, lo que permite ajustar y refinar el *modelo adaptativo* en función de la problemática planteada. Esta metodología consta de cuatro ciclos, con una duración total de *54 meses*, cada uno guiado por las etapas de *planeación, ejecución, desarrollo y reflexión*, como se ilustra en la Figura 1. Cada ciclo, contempla actividades específicas, seguidas de una evaluación para identificar áreas de mejora y optimizar el modelo. Estas actividades se desarrollarán de la siguiente manera: (i) primer ciclo (20 meses), Se realizará la RSL y, posteriormente, se llevará a cabo la caracterización de la información a partir de los estudios primarios, (ii) segundo ciclo (12 meses), se elaborará una *ontología de dominio específico*, denominada OntoSocialDebt, para gestionar el conocimiento sobre la deuda social, (iii) tercer ciclo (14 meses), se construirá el *modelo adaptativo V 1.0* y (iv) cuarto ciclo (8 meses), se realizará una validación por juicio de expertos. A continuación, se presenta un resumen de las actividades de investigación realizadas.

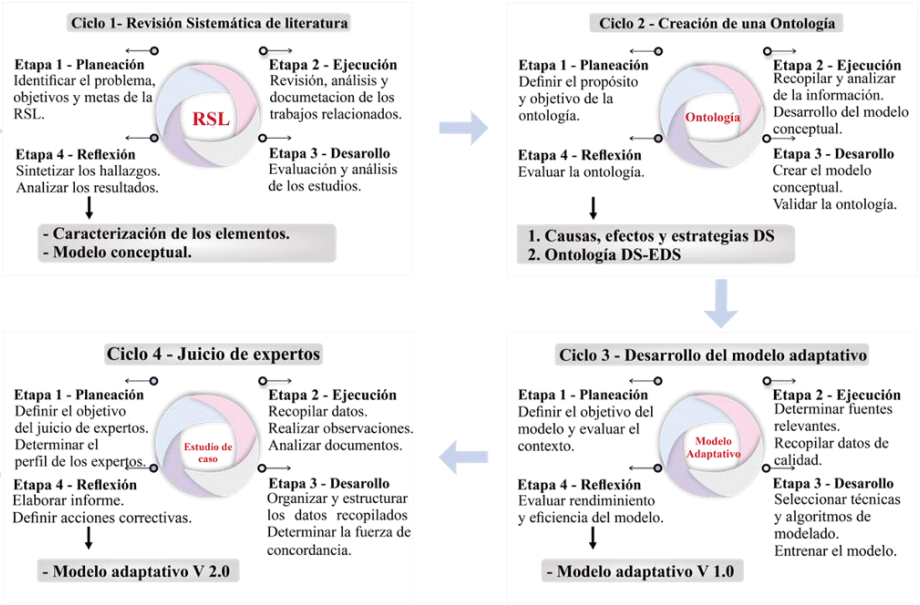

Figura 1. Proceso metodología Investigación – Acción

En el primer ciclo, se desarrolló la caracterización de los elementos mediante una RSL, identificando 47 artículos publicados entre 2013 y 2024. Estos abordan las causas que generan la deuda social y su relación con deficiencias en factores como la comunicación, coordinación y colaboración. Asimismo, se analizaron factores que contribuyen a la acumulación de la DS y estrategias para mitigar sus efectos. Además, se profundizó en categorías clave relacionadas con el fenómeno de la deuda social, abordando aspectos como causas, métricas para su identificación, efectos generados, estrategias de mitigación y enfoques para su gestión. Por otro lado, se identificaron

iniciativas de solución, que centraron en determinar la presencia de cuatro olores comunitarios: silo organizacional, nube negra, cuello de botella y lobo solitario. A continuación, se presentan los objetivos de búsqueda y preguntas de investigación definidas para la realización de la RSL.

Objetivos de búsqueda:

OB1. Categorizar los estudios e investigaciones teniendo en cuenta criterios demográficos, con el propósito de evaluar el interés científico de la comunidad académica con relación al tema de DS, causas, efectos y procedimientos empleados para mitigar y controlar su presencia en el contexto de desarrollo ágil de software.

OB2. Analizar la diversidad de estudios identificados y relacionados con el tema DS, basándose en el esquema de clasificación propuesto por [Wieringa *et al*., 2006] mediante la definición de criterios de calidad, seleccionar los artículos primarios teniendo en cuenta factores como claridad, suficiencia, relevancia y contribución investigativa.

OB3. Recopilar información actualizada, relevante y validada sobre el tema en estudio. Este proceso permitirá identificar las tendencias actuales, así como anticipar retos y desafíos futuros relacionados con la DS en el contexto de desarrollo ágil de software.

Preguntas de investigación:

RQ1. ¿Cuáles son los tipos de investigación identificados y las contribuciones reportadas sobre la DS en el contexto de desarrollo de software ágil distribuido?

RQ2. ¿Cuáles son los dominios en los que se puede generar e impactar la DS en el contexto de desarrollo de software ágil distribuido?

RQ3. ¿Existe algún tipo o propuesta de solución definida para prevenir, identificar y gestionar la DS en equipos de desarrollo de software ágil distribuido?

RQ4. ¿Cuáles son las características de las empresas en las cuales se han realizado estudios para determinar la DS en equipos de desarrollo de software ágil distribuido?

RQ5. ¿Qué patrones permiten identificar y gestionar la DS en equipos de desarrollo de software ágil distribuido?

RQ6. ¿Cuáles son los desafíos y/o trabajos futuros identificados que contribuyen a determinar la deuda social en equipos de desarrollo de software ágil distribuido?

En el segundo ciclo, se definió el propósito y alcance de la ontología para identificar y gestionar la DS en equipos de desarrollo de software. Se clasificaron sus causas según factores sociales, culturales, organizacionales, individuales y de equipo, permitiendo analizar su origen, efectos negativos y su impacto en la productividad, calidad técnica, gestión de proyectos y cultura organizacional. Además, se avanzó en el diseño preliminar de la ontología, formulando preguntas de competencia. La evaluación de la ontología, realizada en cada fase, busca identificar errores, garantizar la calidad del diseño y documentar el progreso. Es de anotar que, para los ciclos 3 y 4, aún no se ha avanzado en el desarrollo de las actividades. Por lo tanto, estas serán relacionadas en la Sección 6. Declaración de las actividades pendientes.

5. Introducción de la solución propuesta y resultados obtenidos

El análisis realizado evidencia la existencia de herramientas para detectar la DS mediante la identificación de causas, tales como: *silo organizacional*, *lobo solitario*, *cuello de botella* y *nube negra* [Caballero-Espinosa *et al*., 2023]. Sin embargo, aún es necesario explorar nuevas causas que puedan impactar el bienestar social de los miembros del equipo y la empresa, y que aún no estén incluidos en estas herramientas. Si bien

existen múltiples métricas para evaluar la comunicación, colaboración y coordinación en equipos de desarrollo de software, la literatura revisada sugiere la necesidad de definir y profundizar en nuevas métricas que permitan determinar con mayor precisión la presencia de DS en equipos de desarrollo de software ágil distribuidos geográficamente [Ahmad 2023], [Dreesen *et al*., 2021]. Además, se destaca la importancia de implementar estrategias efectivas de mitigación, considerando intervenciones culturales, organizacionales y técnicas para reducir su impacto. Es importante destacar que todas estas soluciones operan de manera independiente, y hasta el desarrollo de esta propuesta, no se evidencia en la literatura un modelo adaptativo que integre de manera sistemática la identificación y gestión de la DS [Palomba *et al*., 2021], [Tamburri *et al*., 2019] . Por ello, se plantea el desarrollo de un modelo adaptativo o herramienta automatizada que apoye a las empresas en la gestión la DS, recopilando datos sobre sus causas subyacentes y permitiendo la identificación de áreas críticas que requieren intervención. Con base en los datos recopilados, el modelo facilitará la implementación de acciones y estrategias personalizadas para abordar los problemas de la DS en equipos de desarrollo de software, mejorando la productividad de los miembros del equipo, la eficiencia y calidad del software. Dado que la dinámica de los equipos es cambiante, el modelo debe ser un proceso continuo y adaptable a las necesidades emergentes del entorno.

6. Declaración de las actividades pendientes

Para el desarrollo final de esta propuesta está pendiente el desarrollo de las siguientes actividades.

Ciclo 2 – Construcción de la ontología: Se desarrollará siguiendo la metodología REFSENO , estructurada en 6 etapas de las cuales están pendientes: (i) reutilización de ontologías y metadatos, se aprovechará la información disponible en la web para extraer conocimientos relevantes; (ii) elaboración del modelo conceptual, creación de un glosario de términos, identificación y jerarquización de clases y definición de atributos y relaciones; (iii) implementación: selección de la herramienta para el desarrollo de la ontología, (iv) evaluación: validación con la metodología FOCA para identificar errores y documentar el progreso en cada fase y (v) con base en los resultados de la evaluación, se realizan los ajustes pertinentes a la ontología, con el fin de mejorar su precisión, coherencia y capacidad de representación del dominio.

Ciclo 3- Construcción del Modelo adaptativo: las actividades pendientes incluyen: (i) definición del modelo, detallará los métodos y estrategias para la identificación y gestión de la DS, (ii) diseño del modelo, establecerá los métodos para la recolección y limpieza de los datos, (iii) desarrollo del modelo, se seleccionarán los algoritmos, técnicas de modelado y datos más adecuados para su entrenamiento y (iv) evaluación del modelo, abordará su aplicación en un contexto real para determinar su eficacia en la resolución del problema.

Ciclo 4- Validación juicio de expertos: se definirán objetivos, se seleccionarán expertos según su experiencia, y se precisará el problema a abordar. Se prepararán materiales, se diseñará un instrumento de evaluación considerando eficiencia, eficacia, rendimiento y usabilidad, y se definirán unidades de análisis de la DS. Posteriormente, se socializará el cuestionario, se recopilarán datos y se aplicarán las mejoras sugeridas

7. Implicaciones esperadas

- La RSL evidencia un creciente interés en la investigación sobre la DS, pero persisten vacíos en su comprensión y gestión. Se requiere profundizar en la identificación de

decisiones subóptimas, sus criterios de evaluación, actores involucrados y métricas de impacto. Además, es crucial desarrollar mecanismos de seguimiento y mitigación para evitar su acumulación.

- El modelo adaptativo mejorará la gestión de la DS en equipos de desarrollo de software ágil distribuido, ayudando a reducir su impacto negativo en la productividad, motivación y colaboración. Asimismo, proporcionará acciones adaptables al contexto organizacional, facilitará la identificación temprana de los factores que generan la DS y permitirá mitigar sus efectos antes que afecten el desempeño y la calidad del software. Además, fomentará un entorno colaborativo y saludable, y mediante métricas e indicadores evaluar su estado y evolución a lo largo del tiempo.
- Fortalecimiento de la cultura organizacional enfocándose en el bienestar físico, social y emocional de los miembros del equipo de desarrollo de software ágil.

8. Referencias

Almarimi, Nuri, Ali Ouni, Moataz Chouchen, and Mohamed Wiem Mkaouer. 2022. 'Improving the Detection of Community Smells through Socio-technical and Sentiment Analysis'. *Journal of Software: Evolution and Process* 35. doi: 10.1002/smr.2505.

Anwar, Rayhab, Mobashar Rehman, Khor Siak Wang, Manzoor Ahmed Hashmani, and Amjad Shamim. 2019. 'Investigation of Knowledge Sharing Behavior in Global Software Development Organizations Using Social Cognitive Theory'. *IEEE Access* 7:71286–98. doi: 10.1109/ACCESS.2019.2912657.

Caballero-Espinosa, Eduardo, Jeffrey C. Carver, and Kimberly Stowers. 2023. 'Community Smells—The Sources of Social Debt: A Systematic Literature Review'. *Information and Software Technology* 153. doi: 10.1016/J.INFSOF.2022.107078.

de Campos, Aline, Jorge Melegati, Nicolas Nascimento, Rafael Chanin, Afonso Sales, and Igor Wiese. 2024. 'Some Things Never Change: How Far Generative AI Can Really Change Software Engineering Practice'.

Catolino, Gemma, Fabio Palomba, Damian A. Tamburri, and Alexander Serebrenik. 2021. 'Understanding Community Smells Variability: A Statistical Approach'. *Proceedings - International Conference on Software Engineering* 2021-May:77–86. doi: 10.1109/ICSE-SEIS52602.2021.00017.

Dreesen, Tim, Phil Hennel, Christoph Rosenkranz, and Thomas Kude. 2021. 'The Second Vice Is Lying, the First Is Running into Debt. Antecedents and Mitigating Practices of Social Debt: An Exploratory Study in Distributed Software Development Teams'. *Proceedings of the Annual Hawaii International Conference on System Sciences* 2020-Janua:6826–35. doi: 10.24251/hicss.2021.818.

García, German David, Cesar Jesus Pardo Calvache, and Francisco Javier Alvárez Rodríguez. 2022. 'Society 5.0 and Soft Skills in Agile Global Software Development'. *Revista Iberoamericana de Tecnologías Del Aprendizaje* 17(2):197–207. doi: 10.1109/RITA.2022.3166966.

Khan, Rafiq Ahmad, Mohd Yazid Idris, Siffat Ullah Khan, Muhammad Ilyas, Sikandar Ali, Aziz Ud Din, Ghulam Murtaza, Abdul Wahid Khan, and Saeed Ullah Jan. 2019. 'An Evaluation Framework for Communication and Coordination Processes in Offshore Software Development Outsourcing Relationship: Using Fuzzy Methods'. *IEEE Access* 7:112879–906. doi: 10.1109/ACCESS.2019.2924404.

Lambiase, Stefano. 2024. ‘Cultural and Socio-Technical Aspects in Software Development’. Pp. 482–87 in *ACM International Conference Proceeding Series*. Association for Computing Machinery.

Lambiase, Stefano, Gemma Catolino, Damian A. Tamburri, Alexander Serebrenik, Fabio Palomba, and Filomena Ferrucci. 2022. ‘Good Fences Make Good Neighbours?’ Pp. 67–78 in. Association for Computing Machinery (ACM).

Organización Internacional del Trabajo. 2013. *La Organización Del Trabajo y Los Riesgos Psicosociales: Una Mirada de Género*. San José.

Palomba, Fabio, Damian Andrew Tamburri, Francesca Arcelli Fontana, Rocco Oliveto, Andy Zaidman, and Alexander Serebrenik. 2021. ‘Beyond Technical Aspects: How Do Community Smells Influence the Intensity of Code Smells?’ *IEEE Transactions on Software Engineering* 47(1):108–29. doi: 10.1109/TSE.2018.2883603.

Pantaleo, Guillermo., and Ludmila Rinaudo. 2015. *Ingeniería de Software*. 1st ed. edited by Alfaomega. Ciudad Autónoma de Buenos Aires: Alfaomega Grupo Editor Argentino.

Saeeda, Hina, Muhammad Ovais Ahamd, and Tomas Gustavsson. 2024. ‘A Multivocal Literature Review on Non-Technical Debt in Software Development: An Insight into Process, Social, People, Organizational, and Culture Debt’. *E-Informatica Software Engineering Journal* 18(1):240101. doi: 10.37190/e-inf240101.

State of Agile. 2023. *The 17th State of Agile Report 17th State of Agile*. Raleigh, NC.

Tamburri, Damian A., Philippe Kruchten, Patricia Lago, and Hans Van Vliet. 2013. ‘What Is Social Debt in Software Engineering?’ Pp. 93–96 in *2013 6th International Workshop on Cooperative and Human Aspects of Software Engineering, CHASE 2013 - Proceedings*.

Tamburri, Damian A., Philippe Kruchten, Patricia Lago, and Hans van Vliet. 2015. ‘Social Debt in Software Engineering: Insights from Industry’. *Journal of Internet Services and Applications* 6(1). doi: 10.1186/s13174-015-0024-6.

Tamburri, Damian A., Fabio Palomba, Alexander Serebrenik, and Andy Zaidman. 2019. ‘Discovering Community Patterns in Open-Source: A Systematic Approach and Its Evaluation’. *Empirical Software Engineering* 24(3):1369–1417. doi: 10.1007/s10664-018-9659-9.

Terragni, Valerio, Annie Vella, Partha Roop, and Kelly Blincoe. 2025. ‘The Future of AI-Driven Software Engineering’. *ACM Transactions on Software Engineering and Methodology*. doi: 10.1145/3715003.

Wieringa, Roel, Neil Maiden, Nancy Mead, and Colette Rolland. 2006. ‘Requirements Engineering Paper Classification and Evaluation Criteria: A Proposal and a Discussion’. *Requirements Engineering* 11(1):102–7. doi: 10.1007/s00766-005-0021-6.

Yilmaz, Murat, Rory V. O’Connor, and Paul Clarke. 2016. ‘Effective Social Productivity Measurements during Software Development - An Empirical Study’. *International Journal of Software Engineering and Knowledge Engineering* 26(3):457–90. doi: 10.1142/S0218194016500194.