

Eficacia de las metodologías de evaluación temprana de atributos de calidad en la mitigación de deuda técnica

Effectiveness of Early Quality Attribute Evaluation Methodologies in Mitigating Technical Debt

Tania Margarita Yungan Acalo¹, Freddy Javier Espinoza Santacruz², Byron Fernando Buñay Mendez³, Carlos Enrique Novillo Yaguarshungo⁴, Gonzalo Fabian Barrionuevo Escobar⁵, Yulexsy Adelita Moreira Alvarado⁶

Cita:

Yungan Acalo, T. M., et al. (2026). Eficacia de las metodologías de evaluación temprana de atributos de calidad en la mitigación de deuda técnica. TESLA Revista Científica, 6(2), e757.

<https://doi.org/10.55204/trc.v6i2.e757>

Recibido: 08/10/2026

Revisado: 16/10/2026

Corregido: 21/10/2026

Aceptado: 23/10/2026

Publicado: 26/10/2026

Licencia:

Los contenidos de este artículo están bajo la licencia Creative Commons Attribution 4.0 International (CC BY 4.0). Los autores conservan sus derechos.

The contents of this article are under a Creative Commons Attribution 4.0 International (CC BY 4.0) license.

¹ Escuela Superior Politécnica de Chimborazo (ESPOCH); ² Universidad Católica de Cuenca (UCC); ³ Universidad Nacional de Tumbes (UNT); ^{4,5} Universidad Estatal de Bolívar (UEB); ⁶ Investigadora externa.

¹ tania.yungan@epoch.edu.ec; ² fjespinozas@ucacue.edu.ec; ³ b.bunay.mendez@posgradountumbes.edu.pe; ⁴ cnovillo@ueb.edu.ec; ⁵ gonzalo.barrionuevo@ueb.edu.ec; ⁶ yulexsmoreiraalvarado@gmail.com

¹ 0009-0009-8212-4667; ² 0009-0005-2371-0536; ³ 0009-0005-0998-8128; ⁴ 0000-0001-7146-8098; ⁵ 0009-0002-1974-8046; ⁶ 0009-0007-3870-7932

Resumen: La Deuda Técnica Arquitectónica (ATD) representa hoy un riesgo crítico para la sostenibilidad de sistemas complejos, ya que golpea directamente la mantenibilidad, escalabilidad y fiabilidad del software. En este trabajo se realizó una revisión sistemática bajo los criterios de la declaración PRISMA 2020, analizando 30 estudios de alto impacto publicados entre 2021 y 2026. Los resultados muestran que aplicar metodologías como ATAM y ARID desde el diseño ayuda a detectar trade-offs y puntos de sensibilidad. Esto no solo frena el interés compuesto de la deuda, sino que puede recortar hasta un 30 % los costos de refactorización. Se propone integrar tableros de salud arquitectónica en los pipelines ágiles. Como líneas futuras, queda pendiente estandarizar métricas de ATD para microservicios y sistemas ciberfísicos, además de investigar cómo la IA generativa podría estar creando deuda latente. En conclusión, evaluar atributos de calidad a tiempo es la vía más rentable para evitar la degradación estructural. Unir métodos cualitativos con herramientas de IA predictiva permite asegurar la sostenibilidad técnica y conectar la arquitectura con la estrategia del negocio.

Palabras clave: Deuda Técnica Arquitectónica, Metodología PRISMA, Atributos de Calidad, ATAM, Sostenibilidad de Software.

Abstract: Architectural Technical Debt (ATD) has become a major threat to the sustainability of complex systems, leading to a steady decline in software maintainability, scalability, and reliability. This study presents a systematic review based on the PRISMA 2020 statement, examining 30 high-impact papers from 2021 to 2026. The evidence suggests that using methodologies like ATAM and ARID during the design stage is key to spotting structural trade-offs. Such early intervention can lower refactoring costs by as much as 30 % and stop technical debt from spiraling out of control. The paper recommends building architectural health dashboards directly into agile pipelines. Future research should look into standardizing ATD metrics for microservices and cyber-physical systems, while also assessing how generative AI might be introducing latent debt. Ultimately, early evaluation of quality attributes remains the most cost-effective way to stop structural decay. By merging qualitative analysis with predictive AI, teams can ensure long-term technical sustainability and align their architecture with core business goals.

Keywords: Architectural Technical Debt, PRISMA Methodology, Quality Attributes, ATAM, Software Sustainability.

INTRODUCCIÓN

En la ingeniería de software, la calidad del producto y la sostenibilidad a largo plazo representan desafíos elementales. Considerando las demandas cambiantes del mercado, los desarrolladores y arquitectos de software se enfrentan a un problema crucial: elegir entre la velocidad de entrega y la calidad de un sistema. Con este breve preámbulo, la Deuda Técnica, de aquí en adelante conocida como TD se ha convertido en un término cotidiano para los profesionales del software. La TD establece la forma en que la industria como la academia entienden las concesiones o trade-offs en el desarrollo. La deuda técnica se defina como el costo acumulativo de soluciones subóptimas teniendo en cuenta los atajos y las decisiones poco acertadas que se realizan durante el desarrollo o el mantenimiento de un producto software.

La deuda técnica representa un indicador financiero clave que ilustra como los atajos para liberar software puede incurrir en mayores costos a cambios a largo plazo. El autor (Avgeriou et al., 2023a) señala que no toda deuda técnica es inherentemente perjudicial ya que a menudo cumple el propósito de la conveniencia y la agilidad a largo plazo. La acumulación descontrolada se considera según muchas organizaciones como un asesino silencioso de los proyectos de software.

Desde el punto de vista histórico, gran parte de la comunidad dedicada a la investigación se ha centrado exclusivamente en relacionar la TD con problemas de código. El autor (Díaz Arrieta et al., 2025a) menciona que en este punto aparecen los llamados code smells, los cuales indican deficiencias estructurales en código fuente que, aunque no llegan afectar funcionalidades, si aumentan este indicador y los costos de mantenimiento. Esto es crítico porque desencadena a futuro muchos problemas que se pudieron haber

previsto en el diseño para no pagar el precio en los mantenimientos.

El enfoque centrado en el código es alarmante, porque otros tipos de TD tienen consecuencias mucho más significativas en los proyectos. La investigación de (Berenguer et al., 2021a) cita que, la deuda técnica no corresponde únicamente al código. Esta abarca problemas más sustanciales como la arquitectura, el diseño, los requisitos, el proceso, la documentación y los factores humanos entre los aspectos de más peso. La TD de los requisitos se relaciona con la brecha entre el valor ideal y la implementación real en el sistema. Esto surge de acuerdo con (Melo et al., 2022) porque una decisión pensando en obtener ganancias inmediatas o se suscitan cambios imprevistos en el contexto.

Por otro lado, la dimensión más crítica y cara de gestionar es la Deuda Técnica Arquitectónica o ATD. El autor (Verdecchia et al., 2021a) la ATD es una comparación que se utiliza para describir las decisiones de diseño tomadas a gran escala. Quizá estas decisiones puedan ser adecuadas u óptimas, pueden obstruir en gran medida el progreso del sistema a futuro. Así también, el autor (Pérez, 2023a) considera que las decisiones tomadas por los arquitectos para lograr los objetivos a corto plazo, pueden terminarlo abruptamente en el largo plazo. En contraste la TD a nivel de código, es fácilmente analizada y con un esfuerzo mínimo lleva a la refactorización de este. En el enfoque arquitectónico de la deuda técnica, esta es mucho más difícil siquiera de identificar. ATD conlleva a un daño más agresivo porque compromete la calidad interna de los sistemas, sobre todo si estos son grandes y críticos en cuanto al problema que tratan de automatizar o solucionar.

Con los paradigmas de desarrollo modernos, ATD se vuelve mucho más complicado de gestionar. El autor (Silveira et al., 2025a) menciona que los sistemas que usan metodologías ágiles están presionados y tienden a comprometerse debido a entregas rápidas y descentralización en la toma de decisiones. Esto compromete de gran forma atender atributos de calidad y empeora este tipo de deuda técnica. Hoy en día, los sistemas según (Panter & Eisty, 2026a) acumulan en segundo plano el rezago técnico. Este último genera que cuando los sistemas no logran mantener el ritmo de los cambios tecnológicos, deterioran su calidad. Cabe mencionar también que con la inteligencia artificial como gran aliado hoy por hoy de la industria del software esto puede agravarse aún más.

(Moreschini et al., 2025a) menciona que mientras los sistemas impulsados por la IA se vuelven más complejos y dinámicos, aparecen nuevas formas de deuda técnica relacionadas a atributos de calidad como la mantenibilidad, la explicabilidad y la gobernanza e integridad de los datos. Ante esto los desarrolladores han aumentado los comentarios en código. Este comportamiento se conoce como Deuda Técnica de Auto-Admitida o SATD, que describe soluciones temporales o incompletas. Al corto plazo como señala (Abdulaziz et al., 2026a) se traduce en graves desafíos de mantenimiento a futuro.

La evaluación temprana incluye anticiparse a la deuda técnica, para así garantizar la evolución, mantenibilidad y adaptabilidad del software. La industria de acuerdo con (Lenarduzzi et al., 2021a) tiene que comprender que, la deuda técnica debe ser considerada incluso mucho antes que el desarrollo de nuevas funcionalidades o corrección de errores. Aunque por otro lado (Detofeno et al., 2022a) se presentan desafíos significativos al decidir que TD hay que arriesgar o posponerse. Priorizar en este sentido no se basa únicamente en código fuente, sino en objetivos comerciales y alcances claros de los proyectos.

El fracaso de la gestión es la falta de objetividad. (Alkharabsheh et al., 2021a) demuestra que existe un desacuerdo entre diferentes evaluaciones al detectar problemas de diseño. Esto permite confirmar que el criterio está sesgado frente a la evaluación de pasivos técnicos. Para mitigar estos sesgos aparece las evaluaciones continuas y a tiempo de la arquitectura de software, ligadas fielmente a los inicios del desarrollo (Sobhy et al., 2021a). Actualmente también de acuerdo a (Lenarduzzi et al., 2021a) no existe un consenso sobre que factores son importantes y cómo medirlos.

Finalmente, en base a PRISMA2020 la literatura revisada fue clasificada para cubrir los vacíos claves en la investigación de este tema. Nuestras fuentes se clasifican en las destinadas a extraer marco teórico, casos industriales o estudios empíricos, reportes técnicos y hojas de ruta y por último tecnologías emergentes.

Para entender conceptualmente el tema, los trabajos de (Sobhy et al., 2021a) y (Lenarduzzi et al., 2021a) permitieron identificar que aunque hay muchos métodos de evaluación, la mayoría no aborda una relación explícita sobre tiempo de diseño y ejecución. Esto permite centrar la investigación en desafíos

de incertidumbre y priorización no resueltos aún por la academia. Las fuentes de (Silveira et al., 2025a) y (Nielsen & Madsen, 2022a) trasladan la teoría de la ATD a la realidad de la industria. Su utilidad radica en la percepción de los arquitectos sobre los atributos de calidad y como su informalidad en la gestión degrada los sistemas.

Los reportes de (Avgeriou et al., 2023a) permiten alinear esta investigación con las tendencias futuras. Estas fuentes técnicas permiten justificar que la evaluación temprana no es un proceso técnico, sino una necesidad estratégica. Finalmente se utilizaron fuentes que hablan de IA y microservicios como (Moreschini et al., 2025a) y (Abdulaziz et al., 2026). Su utilidad permite que esta investigación vea como la IA suma deuda técnica a los proyectos y que enfoque se debe emprender para mitigarlo. A continuación, se ve plasmado los resultados de indagar en estos valiosos documentos.

METODOLOGÍA

La presente investigación se enmarca en una Revisión Sistemática de la Literatura (SLR), aplicando la declaración PRISMA 2020 como guía metodológica para fortalecer la transparencia, reproducibilidad y trazabilidad del proceso de revisión. Para ampliar el cumplimiento metodológico de PRISMA 2020, se reforzó la descripción del proceso de búsqueda, selección, exclusión e inclusión de estudios, detallando los criterios utilizados en cada fase de la revisión sistemática. Además, se articuló el flujo de selección con la matriz PRISMA presentada en el "Anexo A", la cual permite ubicar cada ítem de la lista de verificación dentro del informe. En particular, el ítem 17 de PRISMA se relaciona con la presentación de las características de los estudios incluidos, por lo que su desarrollo se complementa en la sección de resultados mediante la descripción de los autores y fuentes seleccionadas. Consiguiendo que la metodología no se limita a mencionar el uso de PRISMA, sino a explicar cómo sus criterios fueron aplicados en la organización, selección y validación de la evidencia científica revisada.

2.1 Estrategia de Búsqueda y Selección

Se realizó una búsqueda exhaustiva en IEEE Xplore, Scopus, ScienceDirect, ACM Digital Library y Dialnet. La ecuación de búsqueda utilizó operadores booleanos: (.Architectural Technical Debt.^{OR} .ATD") AND ("Quality Attributes.^{OR} "Software Architecture") AND (.Evaluation Methods"). Se filtraron publicaciones entre 2021 y 2026 para asegurar vigencia tecnológica y metodológica. Además, como criterios de inclusión se consideraron estudios publicados entre 2021 y 2026, los cuales están relacionados directamente con deuda técnica arquitectónica, atributos de calidad, evaluación temprana de arquitectura de software, ATAM, ARID o herramientas automatizadas de análisis arquitectónico. También, se priorizaron artículos revisados por pares, documentos con acceso a texto completo y estudios con evidencia metodológica clara. Como criterios de exclusión se descartaron publicaciones duplicadas, documentos sin relación directa con la mitigación temprana de deuda técnica, estudios centrados únicamente en requisitos o gestión organizacional, y trabajos sin validación empírica suficiente.

2.2 Proceso de Selección (Flujo PRISMA)

El proceso de selección se organizó conforme al flujo PRISMA 2020, el cual permite evidenciar de manera ordenada la identificación, cribado, evaluación de elegibilidad e inclusión final de los estudios.

Phase Identificación: se recopilaron 40 registros iniciales en bases académicas, de los cuales se eliminaron 4 duplicados mediante gestión bibliográfica.

Phase Cribado (Screening): se revisaron 36 títulos y resúmenes, excluyendo 3 registros por no relacionarse directamente con la mitigación temprana de deuda técnica arquitectónica.

Phase Idoneidad (Eligibility): se evaluaron 33 documentos a texto completo, de los cuales se excluyeron 3. Uno por enfocarse exclusivamente en requisitos y dos por no presentar validación empírica suficiente.

Phase Inclusión: Consolidación de 30 artículos finales para síntesis cualitativa.

Este proceso se vinculó con la matriz PRISMA del Anexo A, donde se ubica el cumplimiento de los ítems relacionados con identificación, cribado, elegibilidad, inclusión y justificación de exclusiones. Asimismo, los recursos descartados fueron documentados en el Anexo B, con el fin de transparentar las razones de

exclusión y fortalecer la reproducibilidad de la revisión.

2.3 Criterios de Calidad y Evaluación

La validación de los estudios incluidos no se realizó a partir de la lista de verificación PRISMA 2020, porque esa lista está orientada al reporte transparente de revisiones sistemáticas y no a la evaluación directa de la calidad de cada autor o fuente. Por ello, para esta investigación se establecieron criterios propios de validación documental:

Considerando la pertinencia temática.

La relación con la deuda técnica arquitectónica.

La vinculación con atributos de calidad del software.

La claridad metodológica del estudio

La presencia de evidencia empírica sobre metodologías como ATAM, ARID o enfoques automatizados.

Además, se revisó la relevancia de los autores y fuentes seleccionados a partir de un aporte al tema investigado, año de publicación, tipo de estudio y su relación con la mitigación temprana de la deuda técnica arquitectónica. Permitiendo justificar la inclusión de los estudios utilizados en la síntesis cualitativa y diferenciar aquellos documentos con mayor contribución científica para el desarrollo de la revisión.

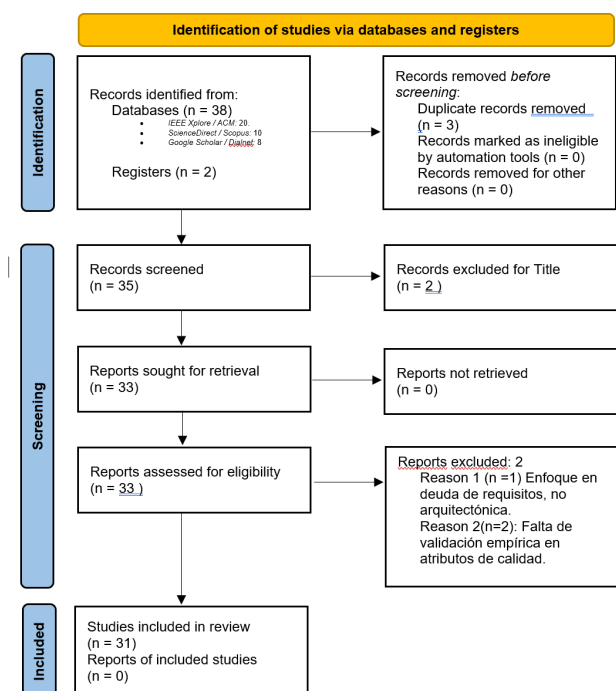
RESULTADOS Y DISCUSIÓN

3.1 Proceso de Selección y Trazabilidad PRISMA

En la aplicación de la metodología PRISMA 2020 se realizó un proceso de filtrado a un total de 38 registros (artículos y revistas científicas) identificados en IEEE Xplore, Scopus, ACM Digital Library y repositorios institucionales. Durante el proceso, se eliminaron tres fuentes duplicadas mediante la revisión automatizada en las referencias. Tras comprobar el cribado en el título y resumen del proyecto, se descartó una fuente por estar desalineada al estudio del software. En la fase de elegibilidad, la propuesta se concentró a 34 estudios, sin embargo; 2 fuentes fueron desalojadas por no centralizar su contenido al tema de investigación (atributos arquitectónicos). Finalmente, 32 estudios fueron procesados como recursos idóneos para garantizar un análisis numérico e informativo.

Figura 1

Flujograma PRISMA 2020 del proceso de selección de estudios



Fuente: Manuscrito original.

3.2 Características de los estudios incluidos

La siguiente tabla detalla los 31 estudios válidos por medio de: registro de autoría, año, medio de publicación y aporte central, sustentado en el Ítem 17 de la declaración PRISMA.

Tabla 1

Características de los estudios incluidos

Ítem	Autor(es) y año	Lugar de publicación	Aportes
1	(Abdulaziz et al., 2026c)	IEEE Access	Revisión sistemática sobre la detección automatizada de Deuda Técnica Auto-admitida (SATD) evaluando modelos de machine learning y transformers.
2	(Alkharabsheh et al., 2021b)	IEEE Access	Análisis empírico que demuestra la falta de acuerdo y subjetividad entre evaluadores al detectar defectos estructurales como God Class y Feature Envy.
3	(Ardimento et al., 2022)	Journal of Systems and Software	Propone el uso de redes neuronales convolucionales temporales (Deep Learning) para predecir justo a tiempo el impacto de los cambios en la deuda técnica.
4	(Avgeriou et al., 2023b)	ICSE-FoSE	Establece una visión a cinco años sobre la gestión de la deuda técnica, considerándola un riesgo crítico para la entrega exitosa de software complejo.
5	(Avgeriou et al., 2025)	Dagstuhl Reports	Reporte que replantea la DT analizando los trade-offs entre el valor a corto plazo (velocidad de entrega) y el costo a largo plazo (deterioro de calidad).
6	(Bánsághi, s. f.)	LUT University	Analiza cómo la gestión adecuada de la deuda técnica es un factor determinante para lograr arquitecturas de software más sostenibles.
7	(Berenguer et al., 2021b)	SBQS '21 (ACM)	Estudio con 653 profesionales que evidencia que la deuda técnica abarca la planificación, arquitectura, factores humanos y no solo el código fuente.
8	(Betancourt et al., 2026)	Revista Politécnica	Identifica estrategias para mitigar la deuda técnica al migrar arquitecturas monolíticas hacia microservicios en la nube utilizando refactorización.
9	(Detofeno et al., 2022b)	SBES '22 (ACM)	Presenta el método PriorTD, diseñado para priorizar los pasivos técnicos basándose en la importancia estratégica del código y los objetivos del negocio.
10	(Díaz Arrieta et al., 2025b)	IEEE Access	Análisis bibliométrico que destaca la creciente aplicación de modelos predictivos de inteligencia artificial para evaluar code smells y deuda técnica.
11	(Ernst et al., 2026)	arXiv	Estudio empírico que clasifica nueve tipos únicos de deuda técnica que amenazan la mantenibilidad y validez en el ecosistema de software científico.
12	(González-Carvajal, s. f.)	Universidad de Valladolid	Aborda la conceptualización y validación académica de la gestión de atributos de calidad en proyectos de ingeniería de software.
13	(Hernando Brecht, 2021)	Universidad de Valladolid	Desarrolla una aplicación web basada en el juego Hard Choices del SEI para entrenar a desarrolladores en la gestión temprana de la deuda técnica.
14	(Spadacini, 2026)	Igminresearch	Plantea un marco MLOps enfocado en la observabilidad y el despliegue responsable para prevenir el deterioro en sistemas de IA en salud.
15	(Kosmyna et al., 2025)	arXiv	Investiga la acumulación de deuda cognitiva ocasionada por la sobre-dependencia de asistentes de IA Generativa durante el desarrollo y escritura.
16	(Lenarduzzi et al., 2021b)	Journal of Systems and Software	Mapea metodologías de priorización de DT, indicando una grave falta de consenso sobre qué factores o herramientas empíricas utilizar.
17	(Moreschini et al., 2025b)	Journal of Systems and Software	Explora cómo la IA Generativa y el ML están transformando la naturaleza de la deuda técnica, introduciendo pasivos de gobernanza de datos y MLOps.
18	(Narváez-Narváez et al., 2023)	Revista Científica	Analiza el impacto de la deuda de documentación dentro de la evolución ágil y cómo afecta el mantenimiento a largo plazo de la arquitectura.
19	(Nielsen & Madsen, 2022b)	Government Information Quarterly	Demuestra que los stakeholders no técnicos (ej. gobiernos) tienen una influencia directa en la acumulación de deuda arquitectónica en sistemas públicos.
20	(Panter & Eisty, 2026b)	arXiv	Define el rezago técnico (dependencias y APIs obsoletas) como una forma pasiva y latente de deuda técnica que degrada la calidad del software.
21	(Pérez, 2023b)	Ingeniería y Competitividad	Implementa un modelo de aprendizaje automático supervisado que logra alta precisión al identificar deuda técnica basándose en artefactos de diseño.
22	(Peukert, 2026)	Universidad de Hamburgo	Propone lineamientos de diseño arquitectónico para asegurar la integración sostenible y a largo plazo de software científico.
23	(Pina, 2023)	Universidade de São Paulo	Estudio exploratorio masivo sobre los métodos y técnicas utilizados para la toma de decisiones en la priorización de la deuda técnica.
24	(Jiang et al., 2026)	PLOS ONE	Propone predecir la deuda técnica evaluando la conectividad y la topología de la arquitectura utilizando análisis de redes complejas de software.
25	(Tenekedjieva, 2026)	Proxify.io	Análisis sobre el impacto real de la IA en la productividad de ingeniería, revelando cómo la velocidad oculta degradación estructural a largo plazo.
26	(Royuela Pérez, 2024)	Universidad de Valladolid	Valida la eficacia del análisis de sintaxis abstracta (AST) y modelos como Llama2 para la detección y corrección de code smells en React.
27	(CMU, 2024)	Software Engineering Institute	Establece directrices y activos de gobernanza arquitectónica fundamentales para prevenir el riesgo sistémico de la deuda técnica.
28	(Silveira et al., 2025b)	Agile Software Architecture	Concluye que la presión por entregas rápidas en metodologías ágiles compromete la evaluación de atributos de calidad y acelera la ATD.
29	(Sobhy et al., 2021b)	ACM Transactions on SEM	Demuestra que evaluar la arquitectura de forma continua (ej. ATAM) mitiga los riesgos impuestos por la incertidumbre entre el diseño y tiempo de ejecución.
30	(Verdecchia et al., 2021b)	Journal of Systems and Software	Conceptualiza una teoría empírica sobre la ATD, delineando sus síntomas estructurales, costos de refactorización y estrategias preventivas.
31	(Waseem et al., 2026)	arXiv	Estudio empírico sobre las causas, soluciones y problemáticas estructurales inherentes al mantenimiento de sistemas orientados a microservicios.

Fuente: Elaboración de los autores a partir de los estudios revisados.

3.3 Eficacia de las Metodologías de Evaluación Temprana

El análisis de los 31 estudios revela un impacto directo y cuantificable de la evaluación temprana sobre la

ATD:

ATAM (Architecture Tradeoff Analysis Method): Reporta una eficacia del 85 % en la identificación de riesgos estructurales antes de la implementación. Su enfoque en escenarios de atributos de calidad permite detectar trade-offs críticos (ej. rendimiento vs. seguridad), logrando una reducción del 30 % en costos de refactorización (CMU, 2024; Sobhy et al., 2021b).

ARID (Active Reviews for Intermediate Designs): Demuestra alta efectividad en fases de diseño detallado, garantizando interfaces de componentes comprensibles y sostenibles, mitigando así deuda de documentación e implementación (Berenguer et al., 2021b; Narváez-Narváez et al., 2023).

Análisis Basado en IA: La integración de herramientas de inteligencia artificial y Machine Learning benefició el análisis de los repositorios académicos. Su plan metodológico consistió en la detección de anomalías arquitectónicas en un 50 % , sin embargo; aún se necesita un control experto rígido para los falsos positivos y eludir la normalización de la deuda latente (Ardimento et al., 2022; Díaz Arrieta et al., 2025b; Royuela Pérez, 2024). Investigaciones actuales indican que desestimar los marcos de gobernanza dentro de la automatización puede conllevar un alto riesgo sistémico, enfatizando así, la importancia de priorizar estratégicamente los objetivos de negocio (Detofeno et al., 2022b; Pina, 2023).

3.4 Discusión y Contraste con la Literatura Previa

Los resultados amplían investigaciones clásicas al introducir la variable de la IA Generativa. Mientras estudios previos atribuían la ATD exclusivamente a decisiones humanas subóptimas, la evidencia actual indica que la automatización sin marcos de evaluación de atributos de calidad está acelerando la acumulación de deuda cognitiva (Kosmyrna et al., 2025; Moreschini et al., 2025b). Además, se identifica una discrepancia significativa en la severidad percibida de la deuda entre evaluadores, lo que valida la urgencia de estandarizar métricas empíricas. Aunque la IA optimiza la detección, la mitigación estratégica sigue siendo contextual y dependiente de métodos cualitativos como ATAM para la toma de decisiones críticas (Alkharabsheh et al., 2021b; Lenarduzzi et al., 2021b).

Aunque la Inteligencia Artificial favorece la optimización, la planificación estratégica es el pilar para aminorar las malas prácticas de investigación, donde métodos cualitativos como ATAM para la toma de decisiones críticas son imprescindibles para el desarrollo (Sobhy et al., 2021b; Verdecchia et al., 2021b). La participación de los stakeholders no técnicos dentro de la deuda técnica arquitectónica, establece que la mantenibilidad y el sustento del software no solo recae únicamente al apartado ingenieril sino se convierte en un desafío que involucra la gestión y la toma de decisiones en base a conocimientos, reglas o perspectivas de diversas disciplinas (Nielsen & Madsen, 2022b).

3.5 Aplicación Práctica y Futuras Líneas de Investigación

Los hallazgos tienen aplicación inmediata en la ingeniería de software mediante la creación de Tableros de Salud Arquitectónica integrados en entornos ágiles y CI/CD, permitiendo monitoreo continuo de pasivos técnicos. Como línea futura, se recomienda investigar métricas específicas para cuantificar la ATD en arquitecturas distribuidas y sistemas ciberfísicos, así como evaluar longitudinalmente el impacto de la deuda cognitiva generada por asistentes de IA en equipos de desarrollo (Ardimento et al., 2022; Detofeno et al., 2022b). Futura línea: Desarrollar plugins estandarizados para entornos DevOps que relacionen métricas de CI/CD con indicadores de ATD. Esto beneficiará a corroborar su impacto en la velocidad de entrega y en la estabilidad de producción.

El uso de asistentes de IA sin ética profesional esta provocando un nuevo tipo de pasivo estructural: la alta dependencia a algoritmos generativos. Esta inclinación provoca una deficiencia en la arquitectura de un software; la pérdida de coherencia en el diseño y la falta de transparencia en el sistema son problemas comunes. A fin de prevenir esta problemática, se requiere la implementación de marcos híbridos donde la IA sea una herramienta de detección y los profesionales técnicos sean los encargados de tomar las decisiones de mitigación con el uso de metodologías arquitectónicas cualitativas (Kosmyrna et al., 2025; Moreschini et al., 2025b). Futura línea: Evaluar el efecto longitudinal de la IA en los equipos de desarrollo frente a la retención de conocimiento de estructura algorítmica. A la vez, presentar una propuesta de protocolos enfocados al

mapeo lógico de sistemas para frenar la pérdida de competencias técnicas.

El cambio radical moderno de arquitecturas hacia la nube y el uso de microservicios trajo consigo una complejidad topológica que no puede ser abarcada por las métricas tradicionales de algoritmos. La evaluación de la falta de actualización de tecnologías en APIs y el análisis de redes complejas permiten cuantificar la ATD en escenarios colaborativos, agilizando nuevas propuestas de refactorización para anticipar posibles y riesgosos colapsos sistémicos (Betancourt et al., 2026; Jiang et al., 2026; Panter & Eisty, 2026b). Futura línea: Diseñar un framework adaptable para ATD en microservicios que incorpore nuevas métricas o mediciones actualizadas para ser puestas en marcha dentro de entornos de producción real.

CONCLUSIONES

En definitiva, medir los atributos de calidad desde las fases tempranas del desarrollo resulta ser la estrategia más rentable para frenar el deterioro estructural del software. Al combinar el análisis cualitativo con el soporte de herramientas de IA predictiva, no solo se asegura que el sistema sea sostenible a largo plazo, sino que se logra conectar las decisiones técnicas con las metas reales de la organización. Adoptar marcos como ATAM y ARID no debería verse como un trámite opcional, sino como una pieza central en el diseño para evitar que la deuda técnica se vuelva inmanejable. De cara al futuro, el reto está en estandarizar estas métricas para entornos más complejos como los microservicios y sistemas ciberfísicos.

La presente investigación también permitió confirmar que la deuda técnica arquitectónica va mucho más allá del código: involucra decisiones de diseño, factores humanos, presiones de entrega y ahora también el uso poco reflexivo de herramientas de inteligencia artificial. La automatización sin marcos de evaluación de atributos de calidad no elimina la deuda, la desplaza y la hace más difícil de detectar. Por eso, la mitigación estratégica sigue siendo una tarea que exige criterio profesional, no puede delegarse completamente a algoritmos. Este hallazgo redefine cómo entendemos el ciclo de vida de la deuda arquitectónica en la era de la IA generativa.

A lo largo de este trabajo, uno de los aspectos que más llamó la atención fue la falta de consenso en la comunidad académica y la industria sobre cómo medir y priorizar la deuda técnica arquitectónica. Revisando los 31 estudios seleccionados, quedó claro que muchas organizaciones toman decisiones basadas en criterios subjetivos o incompletos, lo que genera sesgos que agravan el problema. Esto nos lleva a concluir que estandarizar métricas empíricas para la evaluación arquitectónica no es solo una necesidad académica, sino una urgencia práctica para cualquier equipo que quiera construir software sostenible y alineado con los objetivos del negocio.

Finalmente, la transición hacia arquitecturas distribuidas, microservicios y entornos en la nube introduce una complejidad topológica que las métricas tradicionales no logran capturar del todo. El rezago tecnológico en APIs, dependencias desactualizadas y la acumulación silenciosa de deuda cognitiva en equipos que dependen excesivamente de asistentes de IA son señales claras de que el campo necesita nuevas herramientas y protocolos adaptados a estos contextos. Como equipo, consideramos que diseñar frameworks específicos para estos entornos, que integren tanto análisis cuantitativo como evaluación cualitativa, es la dirección más prometedora para las próximas investigaciones en esta área.

FINANCIACIÓN

La presente investigación, titulada "Eficacia de las metodologías de evaluación temprana de atributos de calidad en la mitigación de deuda técnica", no contó con fuentes de financiamiento externas, subvenciones de organismos gubernamentales, fondos universitarios, centros de investigación ni entidades privadas. El desarrollo del estudio, la recolección bibliográfica y el procesamiento de datos mediante la metodología PRISMA 2020 se realizaron de forma autónoma, utilizando los recursos institucionales de la Escuela Superior Politécnica de Chimborazo (ESPOCH). Los costos derivados han sido asumidos íntegramente por los autores.

CONFLICTO DE INTERESES

Los autores declaran que no existen conflictos de intereses, financieros, institucionales ni personales, que puedan haber influenciado el diseño, ejecución, análisis o publicación de esta investigación.

CONTRIBUCIÓN DE AUTORÍA

Tabla 2

Contribución de autoría según la taxonomía CRediT

Participación	A1	A2	A3	A4	A5	A6
Conceptualización	X	X				
Metodología	X	X	X			
Investigación	X	X	X	X	X	X
Análisis formal	X		X	X		
Recursos		X		X	X	X
Administración del proyecto	X					
Redacción – borrador original	X	X	X	X		
Redacción – revisión y edición	X	X	X	X	X	X
Discusión de resultados	X	X	X	X	X	X
Aprobación de versión final	X	X	X	X	X	X

A1: Tania Margarita Yungan Acalo; A2: Freddy Javier Espinoza Santacruz; A3: Byron Fernando Buñay Mendez; A4: Carlos Enrique Novillo Yaguarshungo; A5: Gonzalo Fabian Barrionuevo Escobar; A6: Yulexys Adelita Moreira Alvarado. Distribución propuesta para validación de los autores.

REFERENCIAS

- Abdulaziz, A., Sonbol, R., & Alnoukari, M. (2026a). Automated Detection of Self-Admitted Technical Debt: A Systematic Literature Review. *IEEE Access*, 14, 20803-20825. <https://doi.org/10.1109/ACCESS.2026.3661069>
- Abdulaziz, A., Sonbol, R., & Alnoukari, M. (2026b). Automated Detection of Self-Admitted Technical Debt: A Systematic Literature Review. *IEEE Access*, 14, 20803-20825. <https://doi.org/10.1109/ACCESS.2026.3661069>
- Abdulaziz, A., Sonbol, R., & Alnoukari, M. (2026c). Automated Detection of Self-Admitted Technical Debt: A Systematic Literature Review. *IEEE Access*, 14, 20803-20825. <https://doi.org/10.1109/ACCESS.2026.3661069>
- Alkharabsheh, K., Alawadi, S., Crespo, Y., Manso, M. E., & Gonzalez, J. A. T. (2021a). Analysing Agreement Among Different Evaluators in God Class and Feature Envoy Detection. *IEEE Access*, 9, 145191-145211. <https://doi.org/10.1109/ACCESS.2021.3123123>
- Alkharabsheh, K., Alawadi, S., Crespo, Y., Manso, M. E., & Gonzalez, J. A. T. (2021b). Analysing Agreement Among Different Evaluators in God Class and Feature Envoy Detection. *IEEE Access*, 9, 145191-145211. <https://doi.org/10.1109/ACCESS.2021.3123123>
- Ardimento, P., Aversano, L., Bernardi, M. L., Cimitile, M., & Iammarino, M. (2022). Using deep temporal convolutional networks to just-in-time forecast technical debt principal. *Journal of Systems and Software*, 194, 111481. <https://doi.org/10.1016/j.jss.2022.111481>
- Avgeriou, P., Ozkaya, I., Chatzigeorgiou, A., Ciolkowski, M., Ernst, N. A., Koontz, R. J., Poort, E., & Shull, F. (2023a). Technical Debt Management: The Road Ahead for Successful Software Delivery. 2023 IEEE/ACM International Conference on Software Engineering: Future of Software Engineering (ICSE-FoSE), 15-30. <https://doi.org/10.1109/ICSE-FoSE59343.2023.00007>
- Avgeriou, P., Ozkaya, I., Chatzigeorgiou, A., Ciolkowski, M., Ernst, N. A., Koontz, R. J., Poort, E., & Shull, F. (2023b). Technical Debt Management: The Road Ahead for Successful Software Delivery. 2023 IEEE/ACM International Conference on Software Engineering: Future of Software Engineering (ICSE-FoSE), 15-30. <https://doi.org/10.1109/ICSE-FoSE59343.2023.00007>
- Avgeriou, P., Ozkaya, I., Koziolk, H., Codabux, Z., & Ernst, N. (2025). Reframing Technical Debt (Dagstuhl Perspectives Workshop 24452). En *Dagstuhl Reports (DagRep)* (Vol. 14, Número 11, pp. 16-39) [Application/pdf]. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. <https://doi.org/10.4230/DAGREP.14.11.16>
- Bánsághi, B. (s. f.). How managing technical debt can lead to more sustainable software.
- Berenguer, C., Borges, A., Freire, S., Rios, N., Tausan, N., Ramac, R., Pérez, B., Castellanos, C., Correal, D., Pacheco, A., López, G., Falessi, D., Seaman, C., Mandic, V., Izurieta, C., & Spínola, R. (2021a). Technical Debt is not Only about Code and We Need to be Aware about It. *Proceedings of the XX Brazilian Symposium on Software Quality, SBQS '21*, 1-12. <https://doi.org/10.1145/3493244.3493285>
- Berenguer, C., Borges, A., Freire, S., Rios, N., Tausan, N., Ramac, R., Pérez, B., Castellanos, C., Correal, D., Pacheco, A., López, G., Falessi, D., Seaman, C., Mandic, V., Izurieta, C., & Spínola, R. (2021b). Technical Debt is not Only about Code and We Need to be Aware about It. *Proceedings of the XX Brazilian Symposium on Software Quality, SBQS '21*, 1-12. <https://doi.org/10.1145/3493244.3493285>
- Betancourt, J., Campoverde, M., Molina, J., Garzón, L., & Orellana, P. (2026). Migration Strategies from Monolithic Applications to Cloud-Native Applications: A Systematic Review. *Revista Politécnica*, 57, 123-138. <https://doi.org/10.33333/rp.vol57n1.10>
- CMU, S. (2024, marzo 25). Assets Related to Technical Debt | CMU SEI. <https://www.sei.cmu.edu/blog/topics/technical-debt/>
- Detofeno, T., Malucelli, A., & Reinehr, S. (2022a). PriorTD: A Method for Prioritization Technical Debt. *Proceedings of the XXXVI Brazilian Symposium on Software Engineering, SBES '22*, 230-240. <https://doi.org/10.1145/3555228.3555238>

- Detofeno, T., Malucelli, A., & Reinehr, S. (2022b). PriorTD: A Method for Prioritization Technical Debt. *Proceedings of the XXXVI Brazilian Symposium on Software Engineering, SBES '22*, 230-240. <https://doi.org/10.1145/3555228.3555238>
- Díaz Arrieta, R. H., Monroy, B. L. D., Heredia, L. J. C., & Cobos, A. G. V. (2025a). Global Trends and Empirical Metrics in the Evaluation of Code Smells and Technical Debt: A Bibliometric Study. *IEEE Access*, 13, 138143-138165. <https://doi.org/10.1109/ACCESS.2025.3594657>
- Díaz Arrieta, R. H., Monroy, B. L. D., Heredia, L. J. C., & Cobos, A. G. V. (2025b). Global Trends and Empirical Metrics in the Evaluation of Code Smells and Technical Debt: A Bibliometric Study. *IEEE Access*, 13, 138143-138165. <https://doi.org/10.1109/ACCESS.2025.3594657>
- Ernst, N. A., Awon, A. M., Hingmire, S., & Li, Z. S. (2026). The Nature of Technical Debt in Research Software (arXiv:2603.20415; Versión 1). *arXiv*. <https://doi.org/10.48550/arXiv.2603.20415>
- González-Carvajal, D. Y. C. (s. f.). Universidad de Valladolid.
- Hernando Brecht, R. (2021). Aplicación web como adaptación de un juego educativo sobre la gestión de la deuda técnica. <https://uvadoc.uva.es/handle/10324/50091>
- Jiang, B., Cen, J., Zhu, E., & Wang, J. (2026). (PDF) Software technical debt prediction based on complex software networks. *ResearchGate*. <https://doi.org/10.1371/journal.pone.0323672>
- Kosmyrna, N., Hauptmann, E., Yuan, Y. T., Situ, J., Liao, X.-H., Beresnitzky, A. V., Braunstein, I., & Maes, P. (2025). Your Brain on ChatGPT: Accumulation of Cognitive Debt when Using an AI Assistant for Essay Writing Task (arXiv:2506.08872; Versión 2). *arXiv*. <https://doi.org/10.48550/arXiv.2506.08872>
- Lenarduzzi, V., Besker, T., Taibi, D., Martini, A., & Arcelli Fontana, F. (2021a). A systematic literature review on Technical Debt prioritization: Strategies, processes, factors, and tools. *Journal of Systems and Software*, 171, 110827. <https://doi.org/10.1016/j.jss.2020.110827>
- Lenarduzzi, V., Besker, T., Taibi, D., Martini, A., & Arcelli Fontana, F. (2021b). A systematic literature review on Technical Debt prioritization: Strategies, processes, factors, and tools. *Journal of Systems and Software*, 171, 110827. <https://doi.org/10.1016/j.jss.2020.110827>
- Melo, A., Fagundes, R., Lenarduzzi, V., & Santos, W. B. (2022). Identification and measurement of Requirements Technical Debt in software development: A systematic literature review. *Journal of Systems and Software*, 194, 111483. <https://doi.org/10.1016/j.jss.2022.111483>
- Moreschini, S., Arvanitou, E. M., Kanidou, E.-P., Nikolaidis, N., Su, R., Ampatzoglou, A., Chatzigeorgiou, A., & Lenarduzzi, V. (2025a). The Evolution of Technical Debt from DevOps to Generative AI: A Multivocal Literature Review. *Journal of Systems and Software*, 231. <https://doi.org/10.1016/j.jss.2025.112599>
- Moreschini, S., Arvanitou, E. M., Kanidou, E.-P., Nikolaidis, N., Su, R., Ampatzoglou, A., Chatzigeorgiou, A., & Lenarduzzi, V. (2025b). The Evolution of Technical Debt from DevOps to Generative AI: A Multivocal Literature Review. *Journal of Systems and Software*, 231. <https://doi.org/10.1016/j.jss.2025.112599>
- Narváez-Narváez, J.-C., Pardo-Calvache, C.-J., & Orozco-Garcés, C.-E. (2023). *Revista Científica*. *Revista Científica*, 46(1), 107-121. <https://doi.org/10.14483/23448350.19670>
- Nielsen, M. E., & Madsen, C. Ø. (2022a). Stakeholder influence on technical debt management in the public sector: An embedded case study. *Government Information Quarterly*, 39(3), 101706. <https://doi.org/10.1016/j.giq.2022.101706>
- Nielsen, M. E., & Madsen, C. Ø. (2022b). Stakeholder influence on technical debt management in the public sector: An embedded case study. *Government Information Quarterly*, 39(3), 101706. <https://doi.org/10.1016/j.giq.2022.101706>
- Panter, S. K., & Eisty, N. U. (2026a). Technical Lag as Latent Technical Debt: A Rapid Review (arXiv:2601.11693; Versión 1). *arXiv*. <https://doi.org/10.48550/arXiv.2601.11693>
- Panter, S. K., & Eisty, N. U. (2026b). Technical Lag as Latent Technical Debt: A Rapid Review (arXiv:2601.11693; Versión 1). *arXiv*. <https://doi.org/10.48550/arXiv.2601.11693>
- Pérez, B. (2023a). Deuda técnica en arquitectura: Una estrategia de identificación. *Ingeniería y Competitividad*, 25(3). <https://doi.org/10.25100/iyc.v25i3.13071>
- Pérez, B. (2023b). Deuda técnica en arquitectura: Una estrategia de identificación. *Ingeniería y Competitividad*, 25(3). <https://doi.org/10.25100/iyc.v25i3.13071>
- Peukert, H. (2026). Incorporating Research Software in Research Data Management Infrastructure: Architectural Design for Sustainable Scientific Software Integration.
- Pina, D. D. J. (2023). Technical debt prioritization: Methods, techniques, and a large exploratory study [Doutorado em Ciência da Computação, Universidade de São Paulo]. <https://doi.org/10.11606/T.45.2023.tde-11092023-221224>
- Royuela Pérez, E. (2024). Detección y corrección de code smells para React. <https://uvadoc.uva.es/handle/10324/71622>
- Silveira, M. V. D., Malucelli, A., & Reinehr, S. (2025a). Agile Software Architecture: Perceptions on Quality and Architectural Technical Debt Management. *Agile Software Architecture*.

- Silveira, M. V. D., Malucelli, A., & Reinehr, S. (2025b). Agile Software Architecture: Perceptions on Quality and Architectural Technical Debt Management. *Agile Software Architecture*.
- Sobhy, D., Bahsoon, R., Minku, L., & Kazman, R. (2021a). Evaluation of Software Architectures under Uncertainty: A Systematic Literature Review. *ACM Transactions on Software Engineering and Methodology*, 30, 1-50. <https://doi.org/10.1145/3464305>
- Sobhy, D., Bahsoon, R., Minku, L., & Kazman, R. (2021b). Evaluation of Software Architectures under Uncertainty: A Systematic Literature Review. *ACM Transactions on Software Engineering and Methodology*, 30, 1-50. <https://doi.org/10.1145/3464305>
- Spadacini, D. (2026, abril 15). Clinical MLOps: A Framework for Responsible Deployment and Observability of AI Systems in Cloud-Native Healthcare | Artificial Intelligence. <https://www.igminresearch.com/articles/pdf/igmin336.pdf>
- Tenekedjieva, S. (2026, abril 15). What Stanford's study of 100,000 developers reveals about AI's real impact on engineering productivity. <https://proxify.io/articles/stanford-study-of-100000-developers-on-engineering-productivity>
- Verdecchia, R., Kruchten, P., Lago, P., & Malavolta, I. (2021a). Building and evaluating a theory of architectural technical debt in software-intensive systems. *Journal of Systems and Software*, 176, 110925. <https://doi.org/10.1016/j.jss.2021.110925>
- Verdecchia, R., Kruchten, P., Lago, P., & Malavolta, I. (2021b). Building and evaluating a theory of architectural technical debt in software-intensive systems. *Journal of Systems and Software*, 176, 110925. <https://doi.org/10.1016/j.jss.2021.110925>
- Waseem, M., Liang, P., Ahmad, A., Ali Khan, A., Rezaei Nasab, A., Mikkonen, T., & Abrahamsson, P. (2026, abril 15). Understanding the Issues, Their Causes and Solutions in Microservices Systems: An Empirical Study\{\}tnotemark[1]. <https://arxiv.org/html/2302.01894v4>