Un Enfoque Multidimensional para la Agilidad: Integración de Equipos, Tecnología y Cultura en el Desarrollo de Software

Jesus Fernando Carvajal Anacona
Neiva, Colombia
Email: olpierjesusfernud@gmail.com

***Resumen*—Las metodologías ágiles han mejorado el desarrollo de software gracias a la adaptabilidad, colaboración y entrega continua de valor. En este artículo se analizan las distintas investigaciones sobre la aplicación de estas metodologías en la gestión de proyectos empresariales y tecnológicos Estas metodologías están representadas principalmente por Scrum y Kanban, emergen como marcos que le han dado un nuevo significado a la forma de planificar, ejecutar y evaluar proyectos, permiten ciclos de entrega más cortos y un enfoque orientado al cliente, estos enfoques han superado progresivamente a los modelos tradicionales como Waterfall, cuya rigidez ha evidenciado limitaciones en contextos donde la flexibilidad es indispensable.**

Dentro de las investigaciones analizadas sobresale el papel de Scrum como metodología predominante en proyectos empresariales y académicos. Su estructura basada en sprints, reuniones iterativas y roles definidos como el Product Owner, el Scrum Master y el Development Team ha permitido fortalecer la comunicación, mantener ciclos constantes de retroalimentación y garantizar un avance progresivo incluso en escenarios complejos. Sin embargo, la efectividad de Scrum no depende únicamente de su concepto, sino también del comportamiento, comunicación transversal, liderazgo y adaptabilidad cultural. Esto se deja claro en investigaciones sobre desarrollo global (GSD), donde los equipos distribuidos geográficamente enfrentan retos de zona horaria, barreras lingüísticas y diferencias organizacionales. En estos contextos, la metodología demuestra ser eficiente, siempre y cuando exista un liderazgo capaz de ordenar la colaboración remota y sostener la cohesión del equipo.

Del mismo modo, los estudios sobre diversidad de género revelan que la dinámica colaborativa también está profundamente influida por factores sociales. Los equipos con composición equilibrada entre hombres y mujeres presentan mejores resultados en diseño, calidad de trabajo y organización. Estos hallazgos muestran que la agilidad no se limita al uso de herramientas o prácticas técnicas, sino que implica una comprensión de la forma en cómo se relacionan las personas, la distribución de tareas y la percepción del rol individual dentro del grupo. Este componente social, muchas veces ignorado en la ingeniería de software tradicional, se convierte en un factor importante para determinar la calidad y la velocidad de entrega.

Otro punto a tomar en cuenta es el análisis de la aplicación diferenciada de metodologías ágiles según el ecosistema tecnológico. El contraste entre desarrollos para iOS y Android evidencia que, aunque ambos entornos pueden beneficiarse de Scrum y Kanban, presentan características propias que modifican tiempos, pruebas y niveles de exigencia. En iOS, la uniformidad del hardware acelera la validación, mientras que la diversidad de dispositivos Android hace más lento el proceso pero amplía los escenarios de prueba. Estos resultados resaltan que la agilidad no debe aplicarse como una fórmula absoluta, sino como un conjunto adaptable de principios capaces de ajustarse a las necesidades específicas de cada proyecto y plataforma.

La unión entre Agile, DevOps y tecnologías en la nube representa una de las transformaciones más relevantes del panorama actual. La integración entre estos enfoques permite automatizar despliegues, optimizar recursos y aumentar la frecuencia de entregas sin comprometer la calidad. No obstante, este modelo también enfrenta desafíos relacionados con la complejidad técnica, la proliferación de herramientas, las competencias requeridas en los equipos y la necesidad de garantizar seguridad en cada etapa del ciclo de vida del software. De ahí la relevancia de DevSecOps, que incorpora prácticas de ciberseguridad dentro de los sprints, automatiza pruebas de seguridad en pipelines CI/CD y redefine la forma en que se gestionan riesgos en sistemas ágiles. Esta integración es clave para lograr productos grandes capaces de resistir el aumento de amenazas, vulnerabilidades y ataques sofisticados.

Se observa una tendencia hacia modelos híbridos que combinan Agile con enfoques tradicionales o con marcos de mejora continua como Lean Six Sigma. Esta integración permite equilibrar agilidad con precisión analítica, adaptabilidad con eficiencia operativa y velocidad con control. Los estudios evidencian que estos enfoques híbridos reducen defectos, optimizan tiempos y refuerzan la toma de decisiones basada en datos. Sin embargo, requieren liderazgo sólido, capacitación constante y una adecuada cultura organizacional para evitar conflictos entre los principios de cada metodología.

Una contribución destacada es el análisis de la calidad de los requisitos, uno de los aspectos más críticos en la ingeniería de software. La comparación entre documentación convencional y documentación en pares revela que la segunda produce especificaciones más completas, consistentes y menos ambiguas. La colaboración directa evita interpretaciones equivocadas, identifica omisiones y hace que el documento evolucione de manera coherente con el proyecto. Esto demuestra que incluso en entornos ágiles, donde la documentación suele ser mínima, la calidad del SRS sigue siendo esencial para evitar sobrecostos, retrasos y fallos en el producto final.

Se explora la innovación mediante lógica difusa para

medir el nivel de agilidad real en los equipos. Este modelo permite traducir percepciones subjetivas en métricas objetivas, generando un sistema capaz de evaluar la madurez ágil con precisión y sin depender exclusivamente de indicadores parciales. La introducción de inteligencia artificial en la medición de prácticas ágiles constituye un avance significativo para organizaciones que buscan monitorear su evolución y detectar oportunidades de mejora.

Respecto al aseguramiento de calidad, los estudios sobre automatización de pruebas y optimización de pipelines CI/CD muestran cómo la velocidad del desarrollo moderno exige herramientas que reduzcan el esfuerzo manual, mejoren la estabilidad y eviten errores en producción. Automatizar pruebas E2E con Selenium, por ejemplo, ha demostrado ser fundamental para garantizar flujos críticos en sistemas de comercio electrónico. Al mismo tiempo, optimizar pipelines CI/CD mediante paralelización, análisis de métricas y entornos consistentes responde a la necesidad de entregas más confiables sin sacrificar la rapidez. Esto confirma que la automatización dejó de ser una ventaja competitiva para convertirse en una necesidad estratégica.

La investigación aborda el enfoque emergente conocido como Algorithm-driven Development (ADD), un modelo visual que convierte los requisitos en diagramas de flujo desde el inicio del proyecto y permite detectar errores antes de escribir el código. Este método ofrece una alternativa eficiente en proyectos grandes y complejos, donde TDD o BDD pueden fallar por restricciones de tiempo o volumen de trabajo. ADD fortalece la claridad lógica del sistema y reduce defectos de forma temprana, dejando ver una evolución en la forma de concebir la ingeniería de software.

Se destaca una tendencia transversal que recorre todos los estudios: el desarrollo sostenible del software. Esto implica no solo eficiencia técnica, sino prácticas que garanticen continuidad, seguridad integrada, documentación viva, equipos capacitados y procesos capaces de adaptarse a entornos cada vez más cambiantes. Las metodologías ágiles e híbridas adquieren un papel central en esta sostenibilidad, pues permiten iterar sin sacrificar calidad y anticiparse a las necesidades del cliente y del mercado.

Las investigaciones analizadas muestran que la agilidad ha dejado de ser un conjunto de prácticas para convertirse en una filosofía organizacional que permea tecnologías, personas, herramientas y comportamientos. Su éxito radica en la capacidad de integrar colaboración, automatización, liderazgo, seguridad y experimentación constante. Más que acelerar procesos, la agilidad cambia la manera en la que los equipos conciben el trabajo, enfrentan la incertidumbre y crean soluciones.

***Index Terms*—agile, devops, devsecops, lean six sigma, automatización, scrum, kanban, cultura organizacional, documentación colaborativa, ci/cd, seguridad continua**

I. Introducción

Desde la aparición del Manifiesto Ágil en 2001, las empresas han comprendido que el éxito de un proyecto no depende solo de las herramientas o modelos utilizados, sino de la capacidad de los equipos para responder al cambio, comunicarse efectivamente y mantener una entrega constante de valor[1].

Los diferentes estudios evidencian que marcos como Scrum y Kanban se han consolidado como referentes por su flexibilidad, su enfoque humano y su efectividad para organizar el trabajo en ciclos cortos y controlados. Estas prácticas han demostrado su valor tanto en proyectos pequeños como en entornos globales, donde la distancia y la diversidad cultural exigen empatía, liderazgo y comunicación continua. También se ha evidenciado que los factores de éxito en países en desarrollo, como Bangladesh, dependen de la planificación, el aprendizaje progresivo y la relación con los clientes a lo largo del ciclo de vida del proyecto[2], [3]..

Otros estudios han explorado dimensiones complementarias que amplían la comprensión de la agilidad. La diversidad de género, por ejemplo, ha mostrado influir en la dinámica colaborativa de los equipos, revelando que la equidad y la participación equilibrada mejoran la calidad del trabajo.[4].Investigaciones comparativas entre los ecosistemas de desarrollo para iOS y Android han puesto de manifiesto que la agilidad no es un método uniforme, sino una mentalidad que se adapta a las particularidades de cada entorno[5].

La integración de metodologías ágiles con otros enfoques ha abierto nuevas posibilidades. Su combinación con Lean Six Sigma une la flexibilidad del cambio con la disciplina del análisis basado en datos; la fusión con DevOps y Cloud Computing ha fortalecido la automatización, la entrega continua y la escalabilidad de los sistemas; y su aplicación dentro del marco del Project Management Institute (PMI) [6] ha demostrado que es posible equilibrar la velocidad con la estructura sin sacrificar la calidad[7], [8], [9].

La inteligencia artificial y la lógica difusa se han empleado para medir el grado de agilidad en los equipos, aportando métricas objetivas a procesos tradicionalmente subjetivos. Paralelamente, la automatización de pruebas y la optimización de flujos CI/CD [10], [11] (Integración y Despliegue Continuos) han reforzado la eficiencia del desarrollo, garantizando entregas más seguras y productos más estables. La inclusión de estrategias DevSecOps, por su parte, ha integrado la ciberseguridad desde las primeras etapas, haciendo de la protección del software un componente esencial del ciclo de vida del proyecto[12].

Las investigaciones coinciden en que el valor de la agilidad reside menos en los procedimientos y más en la mentalidad que la sustenta[13]. La colaboración, el aprendizaje constante y la apertura al cambio son los cimientos de un desarrollo sostenible, ético y orientado a las personas.

En sus primeras etapas, gran parte del trabajo seguía modelos lineales y poco flexibles que funcionaban bien para proyectos muy estables, pero que se volvían lentos y frágiles cuando el entorno cambiaba. Con el paso del tiempo, se entendió que los equipos no solo necesitan entregar productos, sino adaptarse, comunicarse, responder

a los cambios y trabajar de manera coordinada. De esta necesidad surgieron las metodologías ágiles, que con el tiempo se convirtieron en una manera de pensar y de organizar el trabajo, más allá de ser una simple lista de prácticas.[14]

A medida que la tecnología avanzó, Agile no permaneció sola, empezó a convivir con nuevos enfoques, herramientas y retos. Por ejemplo, el crecimiento del comercio electrónico obligó a acelerar pruebas automatizadas; el auge del cómputo en la nube exigió nuevas estrategias de integración y despliegue; la seguridad dejó de ser un elemento adicional y pasó a ser parte integral del proceso; DevOps y luego DevSecOps transformaron la manera de entregar software; y más recientemente, los modelos híbridos, la lógica difusa y la inteligencia artificial empezaron a apoyar la toma de decisiones. Poco a poco, el ecosistema del desarrollo moderno dejó de ser un conjunto aislado de prácticas y se convirtió en una red compleja donde la colaboración, la automatización, la calidad y la velocidad conviven con investigación, métricas, diversidad, liderazgo y cultura organizacional.

[15] y analizarlos como un sistema interconectado. Los estudios consultados muestran cómo las metodologías ágiles se integran con enfoques como DevOps, pruebas automatizadas, Lean Six Sigma, seguridad en el ciclo de vida, plataformas en la nube, documentación colaborativa, modelos híbridos y nuevas formas de medir la agilidad. Cada uno de estos componentes por separado aporta valor, pero su verdadera fuerza aparece cuando se conectan. Esto permite que los equipos respondan no solo a los cambios técnicos, sino también a los humanos, culturales y organizacionales.

Esta integración también revela algo importante: el desarrollo de software no es solo programación, es liderazgo, comunicación, análisis, retroalimentación, seguridad, pruebas, investigación, diseño, métricas y capacidad de aprender mientras se trabaja. Las metodologías ágiles han alcanzado su madurez no por seguir reglas estrictas, sino por permitir que cada organización las adapte a su realidad. Esto abre la puerta a prácticas combinadas, Agile con DevOps, Agile con nube, Agile con Lean Six Sigma, Agile con pruebas automatizadas y Agile con modelos híbridos. El resultado es una visión más completa del desarrollo moderno.

II. Marco teórico y trabajos relacionados

Las investigaciones revisadas ofrecen una visión amplia de cómo las metodologías ágiles se han consolidado, transformado y combinado con otros enfoques. Aunque cada artículo aborda un aspecto distinto, todos coinciden en que Agile no es solamente una técnica de trabajo: es una cultura organizacional basada en adaptabilidad, colaboración y mejora continua. Esta sección integra los principales aportes de los trabajos revisados, mostrando cómo se conectan entre sí y cómo dan forma al desarrollo actual.

Uno de los puntos claves del estado del arte es la consolidación histórica de Agile. López Menéndez de Jiménez describe cómo desde 2001 las organizaciones comenzaron a priorizar la adaptabilidad por encima de la planificación rígida y cómo metodologías como Scrum se volvieron populares gracias a su estructura de roles, ciclos cortos retroalimentación frecuente [1]. Este trabajo marca el punto de partida para entender por qué Agile empezó a desplazar enfoques tradicionales.

Otro eje relevante es el análisis de Agile en países en desarrollo, como lo muestran Shafir y colaboradores en Bangladesh. Señalan que, aunque Agile promete mejoras en tiempos y calidad, los entornos con limitaciones de experiencia, capacitación o recursos enfrentan retos especiales. Este tipo de estudios demuestra que la implementación exitosa depende del contexto y no solo de seguir buenas prácticas [3].

La diversidad dentro de los equipos también aparece como un factor importante. Stray et al. analizan cómo los grupos con mayor equilibrio de género tienden a tener mejores dinámicas de colaboración y distribución de tareas. No es un tema técnico, sino humano, pero afecta directamente al rendimiento y la calidad del trabajo [5]. Esto recuerda que las metodologías no operan solas, dependen de las personas que las aplican.

Varias investigaciones comparan o integran metodologías. Jain estudia diferencias en el desarrollo de iOS y Android cuando se aplica Scrum o Kanban, mostrando que cada entorno tiene particularidades que influyen en la elección de prácticas [4]. De manera similar, Moiseienko et al. presentan una aplicación Kanban y explican cómo cada herramienta o metodología debe ajustarse al ritmo del equipo [16].

La globalización del desarrollo tiene importancia a la vez. Reich y Reich analizan el uso de Scrum en equipos distribuidos y señalan desafíos como diferencias culturales, zonas horarias y barreras de comunicación. Destacan que la agilidad exige liderazgo empático, no solo reglas [2].

Algunas investigaciones profundizan en la integración de Agile con modelos modernos como DevOps y la nube. El Aouni y colegas muestran cómo Agile, Cloud y DevOps se complementan para ofrecer escalabilidad, automatización y velocidad [8]. Maidin et al. examinan tendencias futuras y explican por qué la seguridad debe tener el mismo nivel de importancia que la funcionalidad [17].

También aparecen enfoques híbridos que combinan agilidad con estructuras tradicionales, como Agile Híbrido o la integración de Agile con Lean Six Sigma, propuesta por Jibgah y colaboradores [7]. Este tipo de aproximaciones muestra que las organizaciones requerir flexibilidad sin perder control sobre la calidad y los tiempos.

La calidad del software y la documentación también reciben atención. Qamar et al. evalúan la documentación en pares y encuentran mejoras significativas en la claridad y consistencia de los requisitos [18]. Por su parte, Malla compara la velocidad y calidad entre metodologías ágiles

y tradicionales, mostrando que Agile reduce tiempos sin sacrificar estabilidad [14].

Solige analiza las pruebas automatizadas [10], mientras Cubillos presenta un caso práctico en un entorno e-commerce aplicando Scrum y Selenium IDE [19]. Ambos trabajos muestran cómo la automatización se vuelve un aliado para mantener calidad en procesos rápidos.

Varios artículos abordan medición y optimización. Aguayo et al. utilizan lógica difusa para medir agilidad de manera más precisa [6], mientras Shriram explora estrategias para mejorar el flujo CI/CD [11], mostrando que las métricas importan tanto como las prácticas de trabajo. Estos trabajos muestran que Agile no es un enfoque aislado. Se conecta con seguridad, pruebas, automatización, documentación, nube, cultura organizacional, diversidad, estadística, liderazgo y modelos híbridos.

Las distintas investigaciones analizadas coinciden en que la agilidad ha dejado de ser un método de desarrollo para consolidarse como una cultura organizacional basada en la adaptabilidad, la colaboración y el aprendizaje continuo. Desde los primeros estudios que explicaron los fundamentos del Manifiesto Ágil y su aplicación en entornos empresariales (López Menéndez de Jiménez, s. f.) [1], se evidenció que la clave del éxito no radica únicamente en las herramientas, sino en la capacidad de los equipos para adaptarse a los cambios y generar valor de manera constante.

Investigaciones en contextos de países en desarrollo, como la de Shafir et al. (2025) [3], mostraron que los factores de éxito en la implementación de metodologías ágiles dependen tanto de la planificación y la comunicación como del compromiso con la mejora continua. Estas conclusiones se complementan con los hallazgos de Stray et al. (2025) [5], quienes demostraron que la diversidad de género dentro de los equipos ágiles influye positivamente en la calidad del trabajo y en la distribución de tareas, fortaleciendo las dinámicas colaborativas.

Por otra parte, Jain (2025) [4] exploró las diferencias en la aplicación de Scrum y Kanban en entornos de desarrollo para iOS y Android, resaltando que la agilidad no es una receta uniforme, sino una mentalidad que debe adaptarse a las características y necesidades de cada ecosistema. En una línea similar, Reich y Reich (2025) [2] destacaron que el uso de Scrum en proyectos globales exige no solo coordinación técnica, sino liderazgo, empatía y comunicación para mantener la cohesión entre equipos distribuidos.

Los estudios de Moiseienko, Moiseienko y Lubentsova (2025) [16] compararon la efectividad de Scrum y Kanban, concluyendo que la agilidad no depende de las herramientas, sino del aprendizaje y la mejora incremental del equipo. De manera complementaria, William (2025) [13] evaluó la eficacia de las metodologías de desarrollo en ecosistemas modernos, confirmando que la agilidad, frente a enfoques tradicionales como Waterfall, ha ganado predominio por su capacidad de adaptación y colaboración.

Desde una perspectiva gerencial, LS et al. (2025) [3] identificaron los principales desafíos en la implementación de metodologías ágiles, como la resistencia al cambio y la falta de experiencia, y propusieron estrategias basadas en liderazgo, capacitación y comunicación. En paralelo, El Aouni et al. (2025) [8] revisaron la integración de Agile, Cloud y DevOps, señalando que la sinergia entre estas prácticas permite optimizar los recursos y alcanzar procesos más escalables y eficientes.

Otros estudios profundizan en dimensiones más específicas. Khan y Ali (2025) [9] examinaron los retos de incorporar DevOps dentro del marco del Project Management Institute (PMI), resaltando la importancia de combinar la disciplina de la gestión tradicional con la velocidad y automatización propias de la agilidad. Malla (2025) [14] confirmó, a través de un estudio comparativo, que el uso de Agile mejora tanto la calidad del software como la velocidad de entrega, especialmente al integrarse con inteligencia artificial y estrategias Cloud-native.

En cuanto a la seguridad, Ok y Eniola (2025) [12] abordaron las mejores prácticas para la gestión de riesgos en entornos ágiles, subrayando que la ciberseguridad debe incorporarse desde las etapas iniciales mediante enfoques DevSecOps. De forma paralela, Maidin et al. (2025) [17] y Qamar et al. (2025) [18] destacaron la relevancia de integrar seguridad y documentación colaborativa para mejorar la sostenibilidad y la calidad de los requisitos en proyectos ágiles.

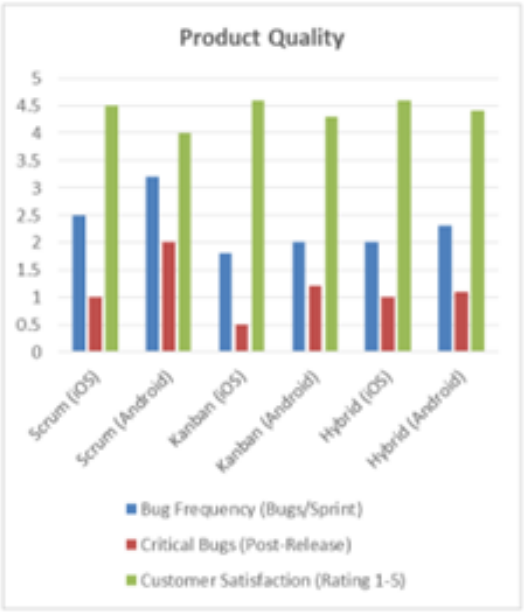
También se refleja en propuestas como la de Aguayo et al. (2025) [6], quienes emplearon lógica difusa para medir la agilidad mediante modelos de inteligencia artificial, y en la de Cubillos García (2025) [19], que aplicó Scrum y pruebas automatizadas E2E en entornos de comercio electrónico. Jawish et al. (2025) [15] propusieron el enfoque Algorithm-Driven Development (ADD), que prioriza la validación visual y estructurada del software desde su origen.

Por último, investigaciones como las de Solige (2025) [10] y Shriram (2025) [11] demuestran que la automatización de pruebas y la optimización de flujos CI/CD son pilares fundamentales para alcanzar un equilibrio entre velocidad, calidad y estabilidad.

III. Metodología de investigación aplicada

Se realizó una revisión sistemática de artículos científicos publicados entre los años 2015 y 2025, seleccionados por su relevancia en el ámbito de la ingeniería de software, la gestión de proyectos y la innovación tecnológica. Las fuentes fueron obtenidas de bases de datos académicas y revistas especializadas, priorizando investigaciones indexadas y de acceso verificable. Entre ellas se encuentran Procedia Computer Science, Information and Software Technology, International Journal of Technology, Management and Humanities, World Journal of Advanced Research and Reviews e IEEE Access, entre otras [4], [5], [8], [10], [14], [18].

En primer lugar, se realizó la búsqueda, selección y clasificación de los artículos según criterios temáticos: fundamentos del pensamiento ágil, estudios comparativos entre metodologías (Scrum, Kanban, Waterfall), integración con tecnologías emergentes (DevOps, Cloud, IA), gestión del riesgo, diversidad de equipos y seguridad del software [3], [7], [9], [12], [13]. En segundo lugar, se llevó a cabo una lectura crítica y analítica, enfocada en identificar coincidencias, diferencias y aportes relevantes entre los distintos enfoques. Finalmente, se elaboró una síntesis interpretativa que permitió articular los resultados de los estudios bajo una perspectiva integradora y contextualizada [15], [17].



Graph 2: Product Quality (Bug Frequency and Post-Release Issues)

Figura 1: Proceso de búsqueda, selección y clasificación de los artículos analizados.

La investigación se apoyó en un proceso de reflexión cruzado en el que se contrastaron los hallazgos teóricos con las implicaciones prácticas descritas en los estudios. Esto permitió construir una visión global sobre la evolución de las metodologías ágiles, su impacto en la calidad, la productividad y la cultura de trabajo en entornos de desarrollo de software [13], [14].

La metodología se enfoca en el estudio de las investigaciones como herramienta para comprender cómo las prácticas ágiles, más allá de ser marcos técnicos, representan una transformación cultural que une tecnología, liderazgo y colaboración en la creación de soluciones digitales sostenibles [1], [3], [5].

La lectura de los artículos deja claro que los temas principales son Scrum, Kanban, DevOps, nube, automatización, seguridad, híbridos ágiles, lógica difusa, colaboración, documentación, diversidad en los equipos y mejora continua [2], [6], [7], [8], [18]. La intención no fue clasificar los artículos de manera aislada, sino reconocer cómo cada uno aportaba una pieza al panorama general del desarrollo moderno.

Se tomó como eje común la evolución de Agile y su interacción con tecnologías y enfoques actuales. Según el análisis de las investigaciones estás se pueden conectar, por ejemplo: cómo DevOps se relaciona con automatización, cómo la documentación en pares influye en la calidad, cómo Scrum funciona en equipos distribuidos, o cómo Lean Six Sigma y Agile se complementan [2], [7], [10], [11], [18].

Después se organizaron las ideas siguiendo un hilo lógico, primero el origen de Agile y sus fundamentos; luego las adaptaciones según contexto, cultura y herramientas; después la integración con tecnologías como nube y CI/CD; y finalmente las propuestas emergentes como lógica difusa, seguridad continua e híbridos ágiles [1], [6], [8], [12], [17].

El objetivo principal es guiar la adopción de agilidad dentro de organizaciones que enfrentan los equipos, demandas cambiantes, riesgos de seguridad, múltiples plataformas tecnológicas, procesos distribuidos y necesidad de automatización. La metodología se construye como un marco práctico aplicable a proyectos de cualquier tamaño, combinando principios ágiles clásicos con prácticas contemporáneas, herramientas colaborativas y estrategias de integración tecnológica [9], [11], [12].

Las investigaciones dan a entender que los equipos con mayor capacidad de maniobra responden mejor a cambios repentinos. Esta flexibilidad abarca la comunicación interna, la reorganización de prioridades y la actualización de herramientas [3], [13].

La dinámica del equipo influye directamente en la calidad del producto. Factores como diversidad de género, comunicación transversal, claridad en los roles y revisión conjunta de tareas aumentan la efectividad general [5].

La automatización, la nube, DevOps, las pruebas continuas, la documentación colaborativa, la lógica difusa y los tableros visuales se convierten en extensiones del trabajo humano, no en sustitutos. La tecnología amplifica el rendimiento del equipo cuando se integra con propósito [6], [8], [10], [11], [17].

El proceso inicia cuando el equipo adopta una preparación cultural que permite que los cambios fluyan sin resistencia. En lugar de presentar la transformación como un ajuste drástico, se introduce de manera gradual, mostrando cómo ciertas prácticas pueden resolver problemas cotidianos que antes drenaban tiempo y energía, la cultura del equipo adquiere relevancia porque influye directamente en la forma en que las personas responden ante bloqueos, revisiones y ajustes. A medida que se construye esta base cultural, el equipo define roles de manera clara, aunque sin convertirlos en barreras que limiten la interacción, cada rol funciona como una pieza que aporta una perspectiva distinta. La presencia de perfiles diversos permite enriquecer las discusiones técnicas y equilibrar la distribución de tareas, lo que disminuye la posibilidad de que ciertos integrantes

carguen con responsabilidades repetitivas o poco visibles. Se sugiere que el Product Owner participe activamente en la clarificación de requisitos y en la priorización del producto para evitar interpretaciones confusas, el Scrum Master se convierte en un facilitador que garantiza que las prácticas fluyan sin interrupciones innecesarias, mientras que el equipo técnico adopta una visión amplia del trabajo para no depender de especialistas aislados. La incorporación de un Security Champion agrega una capa de protección sin frenar el avance, pues permite que la seguridad se trate como parte del desarrollo y no como un control externo que aparece al final. Esta composición diversificada no pretende crear estructuras complejas; busca que cada integrante tenga claridad sobre su aporte y que la colaboración mantenga una dirección coherente [3], [5], [12].

Con los roles establecidos, se avanza hacia la construcción de un ciclo de trabajo que se sostenga por sí mismo, el ciclo se organiza en iteraciones que permiten retroalimentación frecuente, reducen la incertidumbre y mantienen el producto en movimiento constante, el tablero visual adquiere un papel esencial porque traduce las decisiones del equipo en una representación clara y compartida. Cada columna refleja el estado real del trabajo y permite que cualquier integrante detecte irregularidades sin necesidad de solicitudes formales, esta transparencia genera responsabilidad colectiva, reduce tiempos muertos y facilita el ajuste del trabajo cuando cambian prioridades o surge un riesgo técnico imprevisto [2], [16].

La automatización temprana forma parte del ciclo desde el primer día, por eso, plantea que el equipo desarrolle pruebas unitarias, análisis estático y validaciones integrales desde la primera iteración, este enfoque no solo mejora la calidad del código, sino que también acelera la detección de problemas. El pipeline de integración continua se convierte en una herramienta que acompaña cada decisión del equipo, su función no es únicamente ejecutar pruebas, sino ofrecer información constante sobre la estabilidad del producto, cuando un error aparece, se detecta en cuestión de segundos, lo que evita que el fallo avance a etapas donde sería más costoso corregirlo. Además, el equipo aprende a interpretar los resultados del pipeline como parte de su diálogo técnico, sin que se conviertan en simples números desconectados del proceso [10], [11].

La seguridad integrada desempeña un papel similar, no como un obstáculo externo. Esto implica revisar dependencias, analizar vulnerabilidades y evaluar riesgos desde la misma planeación del sprint, gracias a esta integración, la calidad del producto aumenta de manera orgánica, ya que los problemas de seguridad se abordan antes de que escalen. Las organizaciones que aplicaron este enfoque, según las investigaciones revisadas, lograron reducir fallos críticos sin necesidad de controles adicionales que suelen frenar el avance del desarrollo. La clave está en que la seguridad se convierte en una práctica continua, accesible para todo el equipo, y no en un control impuesto [9], [12].

La arquitectura en la nube facilita que el desarrollo mantenga consistencia, la nube permite replicar entornos sin depender de configuraciones manuales y agiliza pruebas que requieren condiciones específicas. Esta capacidad de replicación mejora la fiabilidad de las pruebas y reduce discrepancias entre ambientes, lo que se traduce en menor cantidad de errores difíciles de reproducir, los despliegues se vuelven más estables y el equipo puede experimentar con mayor libertad sin comprometer el entorno principal. Este apoyo tecnológico se integra con DevOps, que agrega prácticas de despliegue continuo y mantenimiento de entornos estables [8], [17].

La documentación ocupa un lugar fundamental en esta metodología porque las investigaciones muestran que los proyectos con documentación inconsistente tienden a acumular errores, la propuesta complementaria sugiere que la documentación se construya mediante trabajo en pares, esta estrategia aumenta la claridad, evita contradicciones y mantiene la información sincronizada con el trabajo real del equipo. La creación de diagramas y representaciones visuales facilita la comprensión de la lógica del sistema y permite que las decisiones técnicas se tomen con una visión más amplia del producto, esta documentación no se trata como un requisito formal, sino como una guía que permite que todos avancen con seguridad [15], [18].

IV. Implementación del software

La implementación de los resultados obtenidos en esta investigación se plantea desde una perspectiva aplicada, orientada a mover los principios de las metodologías ágiles a contextos organizacionales reales. Con base en los hallazgos de los estudios analizados, se propone un modelo de adopción progresiva que combine la flexibilidad de Agile con la estructura necesaria para garantizar resultados sostenibles y de calidad. La aplicación práctica parte de la construcción de equipos que integren diversos roles (desarrolladores, líderes de proyecto, diseñadores, testers y representantes del cliente) bajo un esquema colaborativo inspirado en Scrum. Este modelo fomenta reuniones breves y periódicas (daily meetings), revisión de avances al final de cada ciclo (sprint review) y reflexión colectiva para mejorar procesos (retrospective). Esta dinámica permite ajustar el trabajo de acuerdo con los cambios en los requisitos o el entorno, garantizando una entrega constante de valor al cliente o usuario final [1], [5].

A su vez se plantea la incorporación de tableros visuales tipo Kanban como apoyo a la gestión de tareas y control de flujo de trabajo, siguiendo las recomendaciones de Moiseienko, Moiseienko y Lubentsova [16]. Esto facilita la transparencia, la priorización de actividades y la detección temprana de cuellos de botella, reforzando la autonomía de los equipos.

Para proyectos de mayor complejidad o distribuidos geográficamente, se sugiere integrar prácticas de automatización y despliegue continuo (CI/CD), tomando como referencia los aportes de Shriram (2025) y Solige (2025)

[10], [11]. Estas herramientas permiten optimizar los tiempos de entrega, reducir errores y mantener una calidad constante en cada versión del software. En esta misma línea, se recomienda la adopción de estrategias DevSecOps, en las que la seguridad se integre desde las primeras etapas del desarrollo asegurando una visión preventiva frente a las vulnerabilidades digitales [9], [12].

La implementación debe considerar un enfoque de mejora continua inspirado en Lean Six Sigma, como lo sugieren Jibgah et al. [7]. Esto implica medir la eficiencia de los procesos, eliminar actividades innecesarias y promover la toma de decisiones basada en datos. En contextos organizacionales complejos, esta integración contribuye a equilibrar la rapidez de la agilidad con la precisión del control de calidad.

Se recomienda la creación de una cultura organizacional ágil, apoyada en la capacitación constante, la comunicación transparente y el liderazgo participativo. El éxito de la implementación no depende solo de adoptar marcos de trabajo, sino de transformar la mentalidad de los equipos hacia una colaboración genuina [3], [13].

De esta manera, la aplicación de las metodologías ágiles, combinadas con prácticas modernas como DevOps y Lean Six Sigma, constituye una estrategia viable para elevar la eficiencia, la calidad y la capacidad de adaptación de los proyectos de software en entornos competitivos y cambiantes [7], [8], [9].

La integración de todas las prácticas abordadas en las investigaciones se pueden imaginar como la construcción de un ecosistema de desarrollo moderno. No se trata de aplicar Scrum por un lado, DevOps por otro y automatización en un tercer espacio. La implementación real sucede cuando estos elementos se conectan de forma orgánica dentro del ciclo de trabajo [6], [8], [10].

El punto de partida es Agile. Los equipos trabajan en iteraciones cortas, con revisiones frecuentes y una comunicación constante. Dentro de este marco se aplican roles claros como Product Owner, Scrum Master y desarrolladores. Este enfoque sirve como base para coordinar el trabajo [1], [5].

Luego aparecen las herramientas. Los tableros Kanban dan visibilidad al flujo, permiten identificar bloqueos y mejorar el ritmo general [16]. La integración de DevOps aporta automatización, despliegues frecuentes y entornos más estables [8], [11]. Las plataformas en la nube permiten escalabilidad, replicación rápida y acceso remoto [17].

Al mismo tiempo, la seguridad se integra desde el inicio a través de prácticas de DevSecOps, modelos de amenazas, revisiones de código y automatización de análisis estático [9], [12]. La documentación en pares ayuda a mantener claridad en los requisitos, especialmente cuando los proyectos cambian con rapidez [18]. Y métodos como Lean Six Sigma introducen medición y reducción de defectos en momentos clave del ciclo [7].

La implementación también requiere entender el factor humano. La colaboración, diversidad de pensamiento y distribución justa de tareas influyen directamente en los resultados. La cultura ágil solo funciona cuando los miembros del equipo se sienten escuchados, valorados y capaces de aportar [3], [5].

Los enfoques modernos como la lógica difusa ayudan a medir la agilidad de manera más realista, usando datos donde las percepciones suelen ser ambiguas [6]. Todo esto forma un sistema en el que cada elemento refuerza a los demás.

La puesta en marcha del marco integrado comienza cuando el equipo adopta prácticas que transforman su dinámica diaria y fortalecen la calidad del producto desde los primeros ciclos de trabajo. La implementación se desarrolla como un proceso vivo donde cada integrante participa en la construcción del flujo, la observación de sus efectos y la mejora continua [13], [17].

No se trata simplemente de organizar tareas en un tablero, sino de crear un ritmo que permita observar la evolución del producto sin distancia entre la intención y el resultado. En esta etapa los integrantes descubren cómo se comporta el flujo bajo presión real, cómo responden cuando aparece un bloqueo inesperado, qué tan rápido detectan inconsistencias en la definición de una historia, cuánto tardan en identificar una falla en el pipeline y qué ocurre cuando se ajusta la prioridad de una funcionalidad en pleno desarrollo [10], [11].

A medida que el equipo avanza, el tablero visual se convierte en el eje operativo. No funciona como un simple registro de tareas, sino como una representación del estado mental del proyecto [16]. Cada columna revela la salud del flujo, pues concentra información sobre el avance, las dependencias, las tareas que requieren verificación, las pruebas automatizadas que deben ejecutarse o los análisis de seguridad que todavía no se completan. Cuando un elemento se estanca, el bloqueo se vuelve visible para todos y la conversación surge de forma natural. La transparencia que ofrece el tablero reduce la necesidad de reuniones prolongadas y evita confusiones sobre quién debe actuar ante un retraso o un fallo en el código.

Mientras el equipo asimila esta dinámica, la automatización se introduce de manera gradual. Las primeras pruebas unitarias dan paso a evaluaciones más amplias que incluyen análisis estático, validaciones sobre rendimiento y pruebas end-to-end [10], [11]. A diferencia de otros enfoques donde la automatización se deja para etapas finales, aquí se integra desde el comienzo, lo que permite que cada cambio en el código produzca información inmediata. El equipo aprende a interpretar los resultados del pipeline como parte de su trabajo cotidiano.

A la vez, la seguridad se incorpora al mismo ritmo que la automatización. La presencia de un Security Champion dentro del equipo facilita que las decisiones relevantes se tomen durante el desarrollo y no tras la entrega [9], [12]. La revisión del código, el análisis de dependencias, la identificación de vulnerabilidades y la preparación de modelos de amenazas se vuelven actividades naturales.

En la medida en que el proyecto crece, los entornos en la nube permiten replicar escenarios reales sin depender de configuraciones locales [8], [17]. La implementación utiliza infraestructuras que facilitan despliegues frecuentes, pruebas de carga y verificación de cambios sin afectar a los usuarios finales.

La interacción entre roles tiene un papel fundamental en esta implementación [3], [5]. El Product Owner mantiene un contacto constante con el equipo, aclara dudas, ajusta historias cuando surgen nuevos requisitos y ayuda a organizar la visión del producto. El Scrum Master se enfoca en eliminar barreras, garantizar que los integrantes tengan las herramientas necesarias y acompañar al equipo cuando surgen conflictos o malentendidos. Los desarrolladores rotan tareas para evitar que ciertas responsabilidades recaigan siempre en la misma persona.

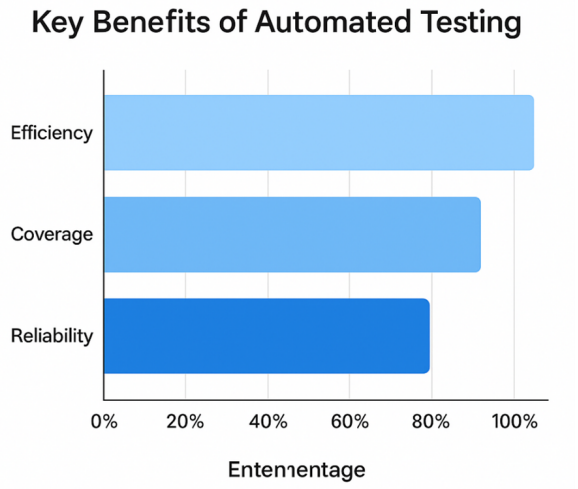


Figura 2: Interacción entre roles del equipo ágil durante la implementación del marco integrado.

La documentación se integra de manera constante mediante revisiones colaborativas [15], [18]. Dos miembros del equipo se unen para redactar, corregir y pulir los documentos esenciales del proyecto, lo que evita inconsistencias y malentendidos.

Con el paso del tiempo, la implementación genera suficiente información para evaluar el avance del grupo. La medición no se limita a cuantificar tareas terminadas, sino que se enfoca en la forma en que el equipo construye y refina su flujo [6], [17]. Las métricas recopilan tiempos de ciclo, estabilidad del pipeline, defectos detectados en etapas tempranas, claridad de la documentación, efectividad de los despliegues y nivel de automatización alcanzado.

La implementación demuestra su valor cuando el equipo enfrenta cambios drásticos en prioridades, alcance o requisitos [2], [13]. La estructura integrada permite reorganizar el trabajo sin comprometer la calidad del producto. Las decisiones se apoyan en información confiable.

Con el paso de varios sprints, la organización observa una transformación en el equipo. La comunicación fluye sin obstáculos, los problemas se abordan de manera directa y las responsabilidades se reparten con equidad. La calidad del producto mejora gracias a la automatización, la seguridad integrada y la claridad en la documentación [10], [11], [12]. Los despliegues se vuelven más frecuentes y estables, lo que incrementa la confianza del Product Owner y reduce la ansiedad que suele acompañar las entregas finales.

V. Evaluación y resultados

El análisis de las investigaciones revisadas permite comprender que la agilidad no es únicamente una metodología para gestionar proyectos, sino un marco de pensamiento que transforma la cultura organizacional y la manera en que las personas perciben el trabajo. La discusión central gira en torno a la capacidad de las metodologías ágiles para integrar tres dimensiones que antes se trataban por separado: la eficiencia técnica, la colaboración humana y la adaptabilidad al cambio [13], [14].

Scrum y Kanban continúan siendo los pilares de la práctica ágil debido a su simplicidad y efectividad en la organización del trabajo. Sin embargo, su éxito depende menos de su estructura formal que del compromiso y la comunicación entre los miembros. Esta observación se refuerza en los estudios de Reich y Reich (2025), quienes demostraron que la distancia física o la diversidad cultural no constituyen un obstáculo cuando existe liderazgo empático y cohesión en los equipos [2], [16].

Los hallazgos de Shafir et al. (2025) y Stray et al. (2025) amplían la comprensión del enfoque ágil al mostrar que su éxito no se limita a contextos tecnológicamente avanzados. La experiencia de países en desarrollo y los análisis sobre diversidad de género confirman que la adaptabilidad y la inclusión son factores determinantes para el rendimiento y la innovación. Estas perspectivas demuestran que la agilidad es también una herramienta de democratización dentro del entorno empresarial, capaz de equilibrar las diferencias y potenciar las capacidades individuales [3], [5].

Otro punto clave es la evolución del pensamiento ágil hacia una integración con tecnologías emergentes, la fusión con DevOps, Cloud y Lean Six Sigma representa una tendencia hacia modelos híbridos más completos, donde la automatización y el análisis de datos fortalecen la calidad y la sostenibilidad de los procesos. Estos avances técnicos se acompañan de una transformación en la mentalidad gerencial. En las investigaciones de LS et al. (2025) se observa que el liderazgo ágil no se basa en el control, sino en la guía, la escucha y la confianza. Así, el papel del gestor tiene un nuevo significado, ya no es quien dicta el rumbo del proyecto, sino quien facilita el entorno para que el equipo pueda autogestionarse y aprender de manera continua [3], [7], [8], [9], [17].

De igual forma, los trabajos centrados en seguridad y en la optimización de flujos CI/CD destacan que la velocidad y la calidad no deben ser metas opuestas, el verdadero valor de la agilidad se encuentra en el equilibrio entre ambos factores, donde la automatización, la prevención de errores y la mejora constante garantizan productos más confiables y sostenibles [10], [11], [12].

Se entiende que en los resultados discutidos las metodologías ágiles han madurado desde un enfoque centrado en el software hacia un modelo de pensamiento que puede aplicarse a diversos sectores, su eficacia depende de la capacidad de las organizaciones para asumir el cambio como una oportunidad y no como una amenaza. La flexibilidad, la comunicación y el aprendizaje colectivo son los motores que sostienen esta creencia [13], [14].

Como conclusión general, puede decirse que las metodologías ágiles no solo han mejorado la calidad y rapidez del desarrollo de software, sino que han dado una nueva definición a la naturaleza del trabajo en equipo y la gestión del conocimiento, su integración con metodologías como Lean Six Sigma, DevOps o PMI muestra que la agilidad puede convivir con estructuras tradicionales sin perder su esencia adaptativa. La incorporación de la inteligencia artificial y la automatización anticipa una nueva etapa en la que la agilidad será también medible, predecible y más cercana al usuario final [6], [7], [8], [9], [15].

La aplicación del marco integrado produjo efectos visibles que no sustituyen los resultados previos de la organización, sino que los amplían desde una perspectiva práctica y tecnológica. A medida que el equipo puso en marcha los ciclos iterativos, la automatización temprana y la seguridad incorporada desde el inicio, comenzaron a aparecer cambios que no dependieron de ajustes formales, sino del modo en que las personas interactuaron con el flujo de trabajo. La observación del comportamiento diario del equipo mostró mejoras que surgieron casi de manera espontánea, impulsadas por la transparencia del tablero, el uso constante de pipelines estables y la claridad que trajeron las revisiones colaborativas [10], [11], [12], [17], [18].

Uno de los cambios más evidentes fue la forma en que el equipo abordó los errores, la presencia de pruebas automáticas permitió detectar fallos en cuestión de segundos, lo que redujo el tiempo que antes se invertía en buscar causas ocultas o reproducir escenarios no tan claros, la rapidez en la detección cambió la percepción del error, dejó de verse como una amenaza para el avance del sprint y pasó a convertirse en un recordatorio de que el flujo funcionaba como debía. La reducción de incertidumbre también disminuyó la presión durante los cierres de ciclo, ya que el equipo sabía que la mayor parte de los problemas técnicos se revelaban antes de llegar a la fase de revisión [10], [11].

La integración de seguridad dentro del sprint tuvo un efecto similar, la revisión constante del código, el análisis de dependencias y la validación de amenazas evitaron

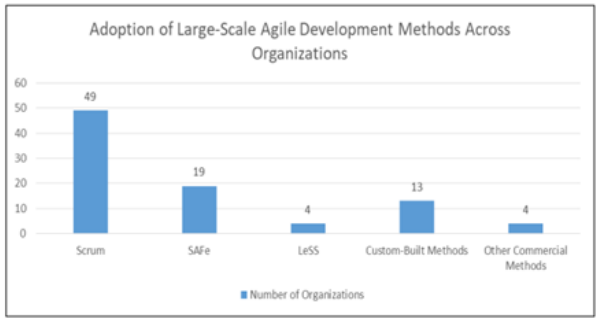


Figura 3: Resultados del proceso de automatización temprana y detección continua de errores mediante CI/CD.

que los problemas se acumularan, el equipo empezó a anticiparse a escenarios que antes no consideraban, lo que contribuyó a que la calidad del producto aumentara sin necesidad de incorporar controles adicionales, la seguridad dejó de ser una carga para convertirse en una parte natural del proceso y esa transición redujo significativamente la aparición de vulnerabilidades [9], [12], [17].

El trabajo en equipo se fortaleció cuando el intercambio de tareas, la rotación de responsabilidades y las decisiones técnicas revisadas en pares mejoraron la comprensión colectiva del sistema. La dinámica diaria cambió y las conversaciones se volvieron más fluidas, los bloqueos se resolvían sin fricción y las discusiones técnicas adquirieron un enfoque más práctico. Los integrantes se apoyaron mutuamente para superar dificultades técnicas y evitar acumulación de trabajo en manos de un solo perfil, esta distribución equilibrada no solo aumentó la eficiencia, sino que generó un ambiente de confianza donde cada persona podía intervenir sin temor a cometer errores [3], [5].

Los tableros visuales contribuyeron a un cambio en la percepción del avance, la visibilidad constante del flujo permitió que los integrantes anticiparan retrasos antes de que afectaran al sprint completo, cuando un elemento permanecía inmóvil más tiempo del esperado, la reacción del equipo surgía sin necesidad de que alguien solicitara intervenir. Esta capacidad de detectar bloqueos temprano redujo interrupciones, mejoró la continuidad del trabajo y mantuvo un ritmo estable a lo largo de los ciclos. El proyecto dejó de depender de reportes tardíos o reuniones extensas para identificar cuellos de botella [2], [16].

El uso de entornos en la nube generó otro conjunto de resultados. La eliminación de diferencias entre ambientes disminuyó fallos derivados de configuraciones inconsistentes. Las pruebas de despliegue se realizaron con mayor confianza y los cambios pudieron ejecutarse sin generar impactos inesperados. La replicación rápida de entornos facilitó experimentos controlados, lo que ayudó a afinar decisiones importantes sin arriesgar la estabilidad del sistema principal. Las tareas que antes requerían coordinación entre múltiples áreas se resolvieron con mayor rapidez, ya que el equipo tenía acceso directo a las herramientas

necesarias [8], [17].

El trabajo colaborativo sobre la documentación contribuyó a consolidar una base de conocimiento clara, los documentos reflejaron mejor las decisiones reales del proyecto y permitieron que nuevos participantes comprendieran la estructura del sistema con rapidez. Las revisiones en pares evitaron contradicciones y ofrecieron un estilo más uniforme, lo que eliminó interpretaciones ambiguas durante el desarrollo. Esta mejora en la claridad documental influyó directamente en la calidad de las funcionalidades, pues el equipo redujo la cantidad de ajustes derivados de malentendidos [15], [18].

El monitoreo constante del pipeline, el análisis del tiempo de ciclo y la revisión del flujo permitieron detectar patrones que antes pasaban desapercibidos. El equipo aprendió a tomar decisiones basadas en datos y no en percepciones fragmentadas. Esta nueva forma de interpretar los avances favoreció ajustes más precisos, pues los integrantes podían identificar con claridad qué parte del proceso requería refuerzo y cómo debía abordarse esa necesidad. El análisis mediante lógica difusa ofreció una visión más matizada de la madurez del equipo, mostrando no solo el progreso, sino también los matices entre fortalezas y áreas de mejora [6], [10], [11].

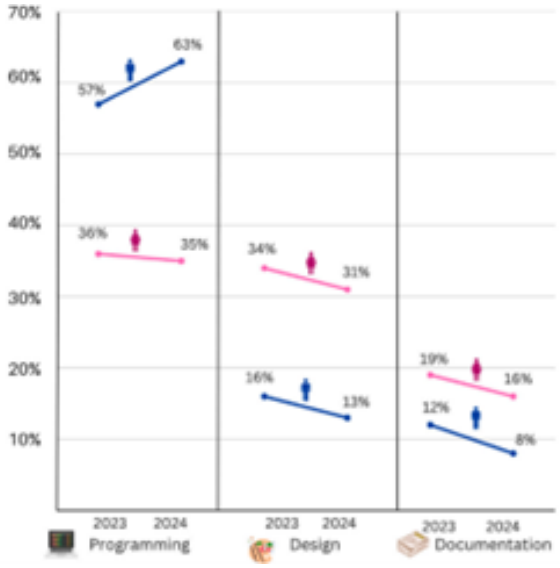


Figura 4: Evaluación del rendimiento del equipo mediante métricas de flujo y análisis basado en lógica difusa.

La capacidad para responder a cambios repentinos ganó solidez. Cuando surgieron variaciones en prioridades o el Product Owner ajustó la dirección del producto, el equipo pudo reorganizar el flujo sin generar desorden. La presencia de automatización, documentación clara y entornos replicables permitió implementar modificaciones sin comprometer la estabilidad del sprint. La organización observó entregas estables incluso en periodos de alta demanda, lo que incrementó la confianza en el equipo y facilitó la toma de decisiones estratégicas [2], [3], [13].

La experiencia acumulada a lo largo de varios ciclos también generó una mejora sostenida en el ritmo de entrega. Aunque el objetivo no era aumentar velocidad sin control, la reducción de errores, la estabilidad del pipeline, la claridad en los roles y la seguridad integrada implicaron que el equipo entregara funcionalidades con mayor regularidad. El tiempo invertido en corrección disminuyó de manera natural, lo que permitió dedicar más esfuerzo a tareas de valor real. La predictibilidad del sprint creció, y esa estabilidad benefició tanto al equipo como a la dirección [10], [11], [12], [14].

VI. Discusión

La integración de metodologías ágiles con prácticas modernas como DevOps, automatización, modelos híbridos, seguridad continua, documentación colaborativa, análisis de datos y plataformas en la nube muestra que el desarrollo de software ya no puede entenderse como un proceso lineal ni como un conjunto de técnicas desconectadas. Lo que emerge de los estudios analizados es un panorama en el que la adaptabilidad se vuelve una condición necesaria para sobrevivir en un entorno tecnológico cambiante. Sin embargo, esa adaptabilidad no significa improvisación: implica entender qué prácticas combinan mejor entre sí y cómo se relacionan con las necesidades de los equipos, las organizaciones y los usuarios [7], [8], [10].

Uno de los aspectos más interesantes es que Agile, en casi todos los trabajos revisados, aparece como una especie de “columna vertebral”, pero nunca como una solución total. Agile organiza el trabajo, facilita la colaboración y reduce la rigidez típica de los enfoques tradicionales; aun así, no resuelve por sí solo problemas de seguridad, ni automatiza procesos, ni garantiza calidad en entornos de alta complejidad técnica. Por eso tantos artículos presentan combinaciones, Agile y DevOps, Agile y Cloud, Agile y Lean Six Sigma, Agile y documentación en pares, Agile y seguridad continua. Cada combinación responde a un vacío específico [7], [8], [12], [18].

Un buen ejemplo de esto es el papel que juega la automatización. En metodologías tradicionales, las pruebas se realizaban al final del proceso, lo cual generaba acumulación de errores, sobrecostos y retrasos. En contraste, la automatización (integrada desde el inicio gracias a DevOps) permite ejecutar pruebas en cada cambio del código. Estudios como los de Solige o el caso del comercio electrónico analizado por Cubillos muestran que la automatización reduce riesgos y acelera la entrega [10], [19]. Esto refuerza la idea de que Agile necesita herramientas que permitan sostener la velocidad sin sacrificar calidad. La automatización no reemplaza al equipo, pero sí amplifica su capacidad para detectar problemas y mejorar el producto antes de que lleguen al usuario final.

Otro aspecto central es la seguridad. A lo largo de varios artículos aparece una preocupación común, la rapidez del desarrollo ágil puede dejar espacios donde las vulnerabilidades crecen. Por eso modelos como DevSecOps buscan

integrar la seguridad en cada etapa. Los estudios muestran que incluir revisiones de código, análisis estáticos, threat modeling y pruebas automatizadas de seguridad evita que los sprints sacrifiquen protección a cambio de velocidad [9], [12], [17]. Las métricas de seguridad también permiten evaluar la postura de riesgo del proyecto de forma continua. Esto demuestra que la seguridad no puede verse como un "anexo", sino como un componente fundamental del ciclo de vida del software moderno.

En este mismo sentido aparece el rol de la documentación. Durante muchos años, Agile fue interpretado como un rechazo a la documentación, lo cual llevó a problemas de claridad y ambigüedad en los requisitos. El estudio sobre documentación en pares revela que documentar no solo es compatible con Agile, sino necesario para mantener coherencia en proyectos que cambian con velocidad. El trabajo colaborativo mejora la claridad de los requisitos y evita que se pierda información crítica [18]. Esto muestra que Agile ha ido evolucionando: ya no se entiende como "menos documentación", sino como "la documentación necesaria, hecha de la manera correcta".

Los enfoques híbridos también aparecen con fuerza. Muchas organizaciones han comprobado que las metodologías ágiles puras no siempre se ajustan a sus procesos internos, especialmente en sectores con regulación estricta. Modelos como Agile Híbrido o la integración con Lean Six Sigma ofrecen un balance entre flexibilidad y estructura. Lean Six Sigma aporta disciplina, medición, análisis y reducción de defectos; Agile aporta dinamismo, ciclos cortos y orientación al cliente. Cuando se combinan, se crea un entorno más equilibrado, donde el equipo se mueve rápido pero con control [3], [7]. Esto demuestra que la industria ya no busca ser completamente ágil o completamente tradicional; busca ser adaptable según las exigencias del proyecto.

La diversidad en los equipos también ocupa un lugar relevante en la discusión. Aunque pueda parecer un tema social más que técnico, los estudios muestran que la composición del equipo afecta directamente la calidad del trabajo. Los grupos con mayor presencia de mujeres o con distribución equilibrada de género tienden a tener mejores dinámicas, más comunicación y distribución de tareas más justa. La colaboración mejora cuando se reduce la homogeneidad [5]. Esto recuerda que la calidad del software no depende solo de herramientas o metodologías, sino de las relaciones humanas dentro del equipo.

La globalización del desarrollo introduce otros desafíos. La distancia, las diferencias culturales y las zonas horarias complican la comunicación y la coordinación. Scrum, aplicado en equipos distribuidos, funciona solo si existe un liderazgo capaz de mantener cohesión. No basta con hacer reuniones diarias; hace falta empatía, claridad en la comunicación y organización del tiempo. Los estudios muestran que Agile en entornos globales requiere más madurez que en equipos locales [2]. La agilidad no puede ser solo rápida, sino también consciente del contexto humano.

Otro punto de discusión es la importancia de medir la agilidad. Aunque Agile promueve ciclos cortos y adaptativos, muchas organizaciones no saben qué tan ágiles son realmente. Los modelos basados en lógica difusa aportan una manera de medir factores subjetivos sin perder precisión. Esto abre nuevas vías para que los equipos evalúen su desempeño más allá de las percepciones personales. Medir lo que antes era intangible es un paso importante para profesionalizar aún más las prácticas ágiles [6].

La discusión general muestra un patrón común en todos los estudios: la combinación de prácticas es más poderosa que cualquier metodología individual. Agile no compite con DevOps ni con Lean Six Sigma ni con la automatización; se complementan. El desarrollo de software moderno se mueve hacia ecosistemas donde las herramientas, la cultura organizacional, la tecnología, la seguridad y las personas trabajan juntas. Las organizaciones que entienden esto tienen más posibilidades de crear productos sostenibles, seguros y de alta calidad. Las que se aferran a un único enfoque suelen quedarse cortas ante la complejidad real [7], [8], [10], [11], [12].

La discusión revela que el futuro del desarrollo de software no está en escoger la metodología "correcta", sino en saber combinar enfoques, adaptarse a los equipos y cultivar una cultura donde el aprendizaje continuo sea la base del trabajo. La tecnología evoluciona rápido, pero las necesidades humanas dentro del desarrollo también deben evolucionar a la misma velocidad [13], [14].

VII. Conclusiones y trabajo futuro

Las conclusiones derivadas de este estudio muestran que el desarrollo moderno de software está experimentando una transformación impulsada por la integración de múltiples prácticas, metodologías y tecnologías. No se trata únicamente de que Agile se haya vuelto popular o que DevOps aporte velocidad; se trata de que la industria ha entendido que ningún enfoque por sí solo resuelve los desafíos actuales. Lo que realmente marca la diferencia es la capacidad de combinar prácticas y ajustar cada una al contexto particular del proyecto y de la organización.

Se puede concluir que Agile ha logrado consolidarse porque ofrece un marco flexible que se adapta a la naturaleza cambiante del desarrollo contemporáneo. Sin embargo, su éxito no está en aplicar ceremonias como sprints o retrospectivas, sino en transformar la cultura del equipo. Agile funciona cuando fomenta conversaciones claras, revisiones constantes y apertura al cambio. Pero los estudios también demuestran que Agile, por sí solo, tiene límites: necesita apoyo de otras prácticas para afrontar retos como seguridad, automatización, documentación y escalabilidad.

Uno de los aportes más significativos de la integración de prácticas modernas es la automatización. Las pruebas automatizadas permiten mantener calidad incluso cuando los equipos trabajan con ciclos cortos y entregas frecuentes. Esto demuestra que la agilidad no significa sacrificar

calidad, sino apoyarse en herramientas que permiten mantenerla bajo presión. La automatización también reduce la carga de trabajo repetitiva, libera tiempo para la innovación y hace que los equipos puedan enfocarse en problemas que realmente requieren criterio humano.

La seguridad, por otro lado, ha pasado de ser un complemento a ser un elemento central del desarrollo. En un mundo donde los ataques informáticos son más frecuentes y sofisticados, no es viable entregar software rápido sin asegurarlo. La integración de DevSecOps demuestra que la seguridad no debe esperar al final; debe acompañar cada etapa del proceso. Las organizaciones que adoptan seguridad temprana reducen costos, evitan vulnerabilidades críticas y construyen productos más confiables. Las metodologías ágiles modernas ya no intentan "agregar seguridad después"; buscan integrarla desde la planificación.

La documentación en pares representa otra conclusión importante. Muchos equipos interpretaron mal el mensaje de Agile y creyeron que la documentación era innecesaria, lo cual generó confusiones, inconsistencias y retrabajo. La evidencia muestra que documentar es parte fundamental de construir productos claros y sostenibles. La colaboración en la documentación permite capturar mejor los requisitos, evita contradicciones y facilita el mantenimiento a largo plazo. Es un recordatorio de que Agile no elimina la documentación; elimina la documentación innecesaria y la reemplaza por documentación útil.

Otra conclusión relevante es que la diversidad en los equipos impacta directamente la calidad del software. Los estudios muestran que equipos con presencia equilibrada de mujeres y hombres distribuyen mejor las tareas, colaboran con mayor fluidez y abordan los problemas desde perspectivas más amplias. Esto demuestra que la composición del equipo no es un detalle administrativo, sino un factor que influye en el diseño, la calidad, la solución de problemas y el rendimiento general.

La globalización del desarrollo también aporta conclusiones, pues los equipos distribuidos enfrentan retos que no se resuelven con reuniones diarias o herramientas digitales. Se requiere liderazgo empático, claridad en los roles y comunicación constante. Scrum y otras metodologías ágiles pueden funcionar en entornos globales, pero solo si el equipo entiende que la distancia exige mayor cuidado en la coordinación. Las organizaciones que ignoran este factor suelen experimentar malentendidos, retrasos y pérdida de cohesión.

Por otro lado, los modelos híbridos confirman que las metodologías estrictas (ya sean totalmente ágiles o totalmente tradicionales) no siempre se ajustan a la realidad. Las organizaciones necesitan flexibilidad y estructura al mismo tiempo. Agile Híbrido y la integración con Lean Six Sigma permiten obtener lo mejor de cada enfoque, la velocidad y adaptabilidad de Agile, y la precisión y análisis estadístico de Lean Six Sigma. Esta combinación permite mejorar la calidad del producto sin frenar la entrega continua.

La incorporación de la nube y las tuberías CI/CD demuestran que la infraestructura también se ha convertido en un factor fundamental en la agilidad. La nube permite escalar, replicar ambientes y trabajar de forma distribuida. CI/CD, cuando está bien optimizado, permite entregas constantes y confiables. Esto muestra que la agilidad no es solo cultural, sino también técnica. Sin una buena base tecnológica, Agile se queda corto.

Las herramientas de medición, como la lógica difusa, aportan otra conclusión interesante: la industria ya no se conforma con evaluar la agilidad de manera superficial. Se están desarrollando métodos para medir la adaptabilidad, colaboración y rendimiento de los equipos con más precisión. Esto convierte la agilidad en un proceso medible, no solo en una filosofía.

De forma general, puede concluirse que el desarrollo de software moderno es un sistema complejo compuesto por personas, procesos y tecnología. La agilidad no consiste en moverse rápido, sino en moverse con sentido. Las organizaciones que adoptan métodos de forma rígida suelen fracasar; las que entienden la esencia y construyen un ecosistema coherente tienen mejores resultados. La verdadera agilidad surge cuando la cultura, la comunicación, la automatización, la seguridad, la documentación, la diversidad y las métricas trabajan juntas.

El análisis integrado de todos los estudios demuestra que el futuro del desarrollo de software se dirige hacia una mezcla cada vez más grande de prácticas. No habrá una única metodología dominante. Habrá equipos que combinen Agile con DevOps, otros con prácticas de seguridad, otros con automatización intensiva, otros con Lean Six Sigma o con modelos híbridos. Lo importante será la capacidad de adaptar esas combinaciones a las necesidades de cada proyecto.

En resumen, el desarrollo moderno exige equipos capaces de aprender, experimentar, comunicar, automatizar, evaluar y asegurar. La combinación de metodologías y tecnologías estudiadas aquí revela un camino claro, el futuro de la ingeniería de software depende de la integración, no de la separación; de la colaboración, no de la rigidez; de la evolución constante, no de la repetición. Las organizaciones que abracen esta visión estarán mejor preparadas para crear software seguro, estable, eficiente y sostenible en el tiempo.

Apéndice

- **Datos**: fuente, versión, licencias, anonimización.
- **Código**: repositorio, commit hash, instrucciones de ejecución.
- **Entorno**: SO, versión de compiladores, dependencias, semillas.
- **Procedimiento**: pasos exactos para replicar resultados.
- **Resultados**: tablas/figuras generadas automáticamente en `build/`.

Referencias

[1] R. E. López Menéndez de Jiménez. «Unidad 1. Actividad 2.2.» Universidad Nacional Autónoma de México, Facultad de Contaduría y Administración. dirección: http://fcaenlinea.unam.mx/anexos/1728/Unidad_1/u1_act2_2.pdf

[2] A. Reich y N. Reich, «Scrum in Global Software Development: Challenges, Risks, and Mitigation Strategies for Effective Project Management,» *Journal of Policy Options*, vol. 8, n.º 1, págs. 43-50, 2025.

[3] G. LS et al., «Challenges and Solutions in Agile Software Development: A Managerial Perspective on Implementation Practices,» *International Journal of Advanced Computer Science & Applications*, vol. 16, n.º 3, 2025.

[4] A. Jain, «A Comparative Assessment of Agile Methodologies in iOS and Android Development,» *International Journal of Technology, Management and Humanities*, vol. 11, n.º 02, págs. 1-7, 2025.

[5] V. Stray, I. Stabell, G. E. Sæter, A. Barbala e Y. Lindsjørn, «Teamwork in agile software development: A mixed-method study of gender diversity and collaboration dynamics,» *Information and Software Technology*, pág. 107 840, 2025.

[6] S. O. R. Aguayo, P. D. Reyes, J. R. H. Morales y R. A. Díaz, «Lógica difusa y el manifiesto ágil: Innovación en la medición de agilidad en el desarrollo de software,» 2025.

[7] D. Jibgah, T. Ananya, B. Elly y S. Grace, «The Integration of Agile Methodologies with Lean Six Sigma in Software Engineering Projects,» 2025.

[8] F. El Aouni, K. Moumane, A. Idri, M. Najib y S. U. Jan, «A systematic literature review on Agile, Cloud, and DevOps integration: Challenges, benefits,» *Information and Software Technology*, vol. 177, pág. 107 569, 2025.

[9] M. Z. Khan y M. L. Ali, «CHALLENGES & OPPORTUNITIES OF ADOPTING DEVOPS IN THE PMI CONTEXT,» *Contemporary Journal of Social Science Review*, vol. 3, n.º 1, págs. 1357-1365, 2025.

[10] S. Solige, «Automated testing in modern software development: navigating challenges and opportunities,» *World Journal of Advanced Research and Reviews*, vol. 26, n.º 1, págs. 955-961, 2025, ISSN: 2581-9615.

[11] K. M. P. Shriram, «Engineering efficiency through CI/CD pipeline optimization,» *International Journal of Science and Research Archive*, vol. 14, n.º 1, págs. 908-916, 2025, ISSN: 2582-8185.

[12] E. Ok y J. Eniola, «Best Practices for Cybersecurity Risk Management in Agile Software Development,» 2025.

[13] J. R. William, «Evaluating the Effectiveness of Software Development Methodologies in Modern Technology Ecosystems,» *ISCSITR-INTERNATIONAL JOURNAL OF COMPUTER SCIENCE AND ENGINEERING*, vol. 1, n.º 1, págs. 8-12, 2025.

[14] P. Malla, «Analyzing the impact of agile methodologies on software quality and delivery speed: A comparative study,» *World Journal of Advanced Research and Reviews*, vol. 25, n.º 01, págs. 1207-1216, 2025.

[15] P. Jawish, P. Evrard, A. Lemerle, A. Genin, L. Dergham y S. Lanfranchi, «Algorithm-Driven Development: A Proactive Approach to Improving Software Quality and Reducing Defects,» *SSRN*, Available at SSRN 5242955.

[16] N. Moiseienko, M. Moiseienko y D. Lubentsova, «A web-based Kanban application for enhancing agile project management practices,» en *CEUR Workshop Proceedings*, 2025, págs. 131-138.

[17] S. S. Maidin, N. Yahya, M. A. bin Fauri Fauzi y N. S. A. A. Bakar, «Current and Future Trends for Sustainable Software Development: Software Security in Agile and Hybrid Agile through Bibliometric Analysis,» *Journal of Applied Data Sciences*, vol. 6, n.º 1, págs. 311-324, 2025.

[18] N. Qamar, N. Sabahat y A. Mosavi, «Evaluating the Impact of Pair Documentation on Requirements Quality in Agile Software Development,» *IEEE Access*, 2025.

[19] S. V. Cubillos García, «Automatización de pruebas para el sector Ecommerce de Corona. co: Pruebas Automatizadas en un Entorno B2C,» Práctica empresarial, Tesis de mtría., 2025.