



AI-Driven Development:

the new era of code





La nueva era del código: reescribiendo las reglas del desarrollo de software

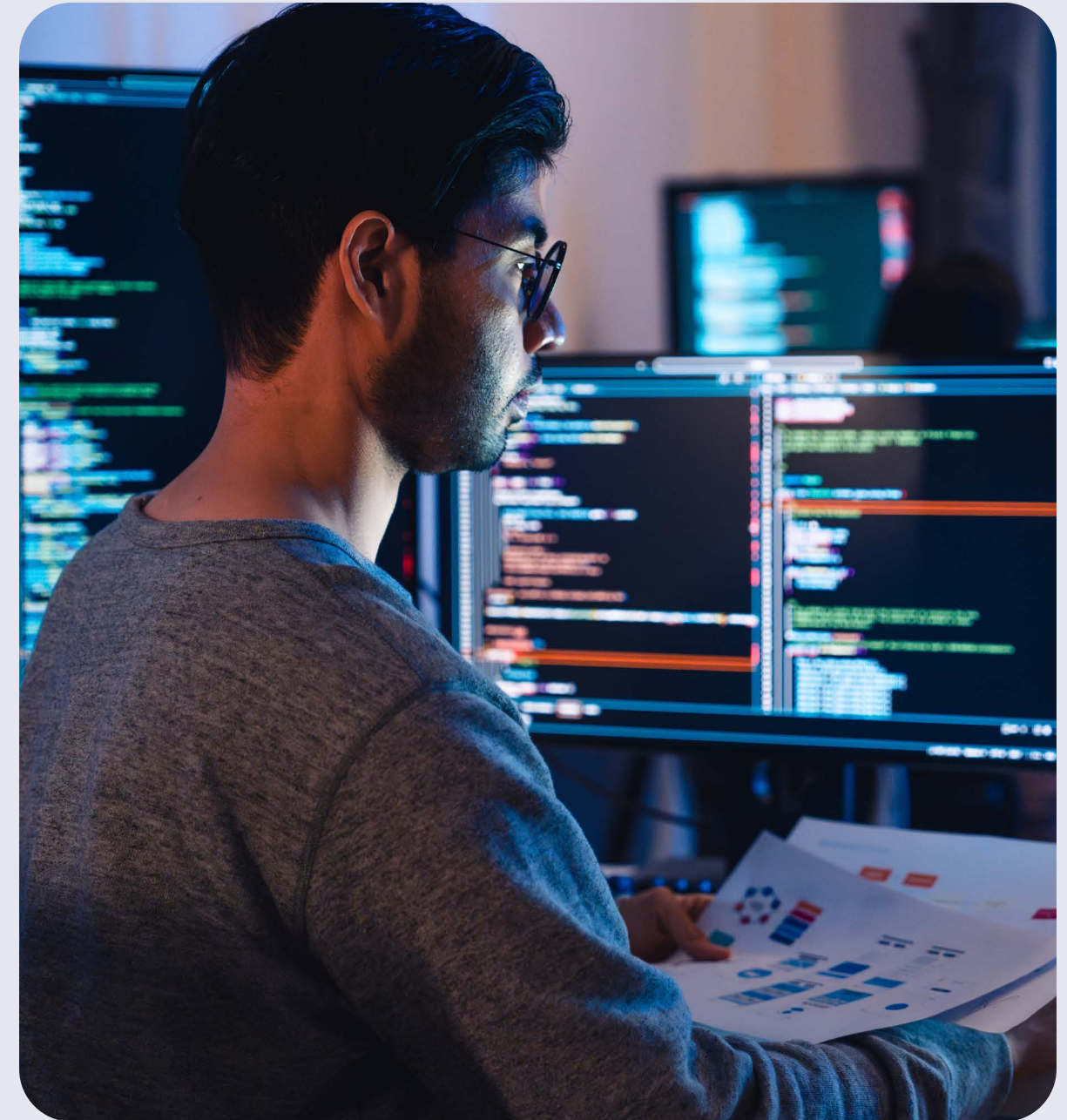
La nueva era del software replantea los fundamentos del desarrollo

Anteriormente, **el desarrollo de software era en gran medida una tarea manual**. Los desarrolladores emplearon un enfoque práctico y metódico al escribir código, basándose en su experiencia personal, la colaboración y herramientas que proporcionaban una automatización limitada. Con el tiempo, la industria implementó una mayor automatización, como canales de integración y despliegue continuo (CI/CD), pruebas automatizadas y prácticas DevOps.

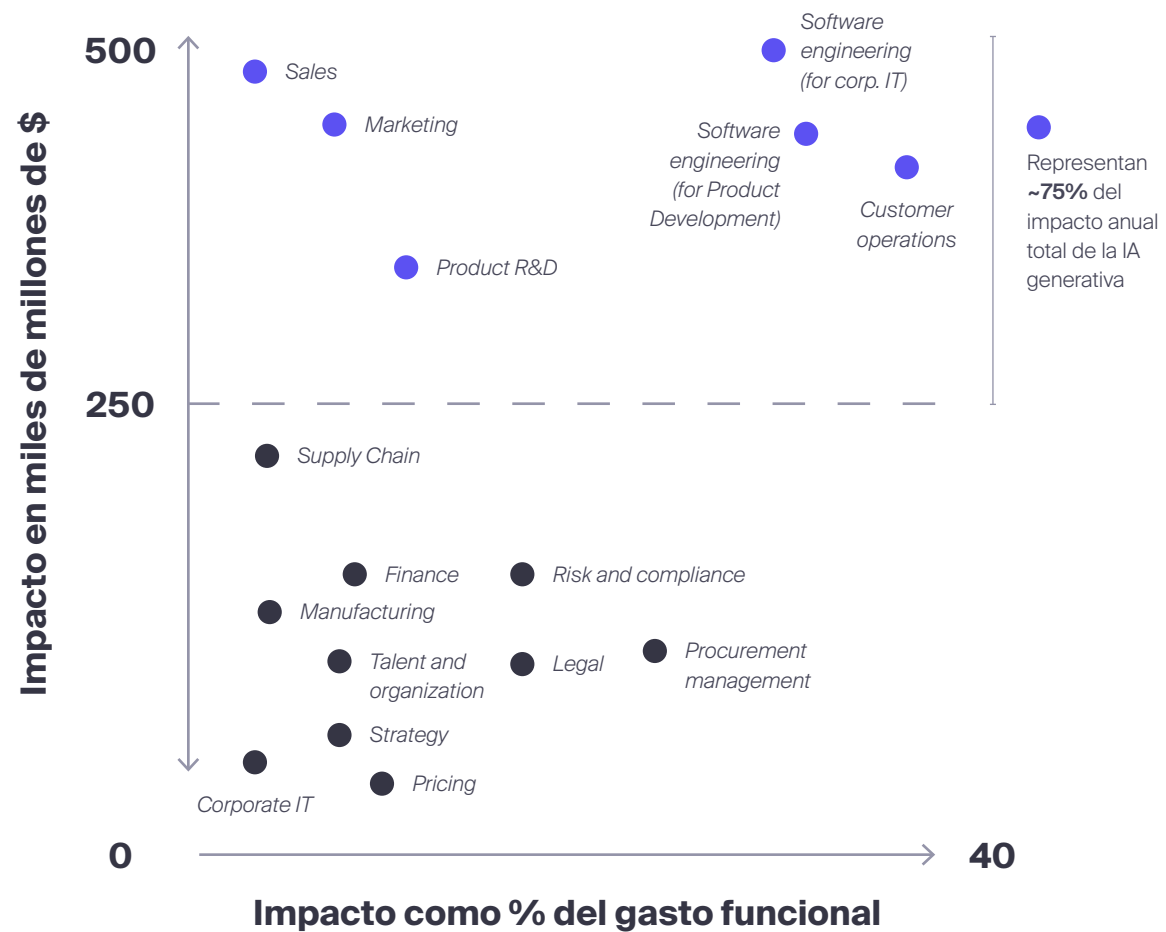
Sin embargo, este modelo centrado en la ejecución humana está dando paso a una nueva etapa, marcada por la irrupción de la inteligencia artificial generativa. A diferencia de los avances anteriores, **estas tecnologías amplifican las capacidades del desarrollador**, traduciendo requerimientos expresados en lenguaje natural a código funcional, generando funciones completas y diseñando casos de prueba con mínima intervención.

Este salto tecnológico **representa un verdadero punto de inflexión**. La nueva era del código está siendo moldeada por un cambio de paradigma profundo, en el que la GenAI y otras tecnologías emergentes redefinen cómo se concibe, implementa y evoluciona el software.

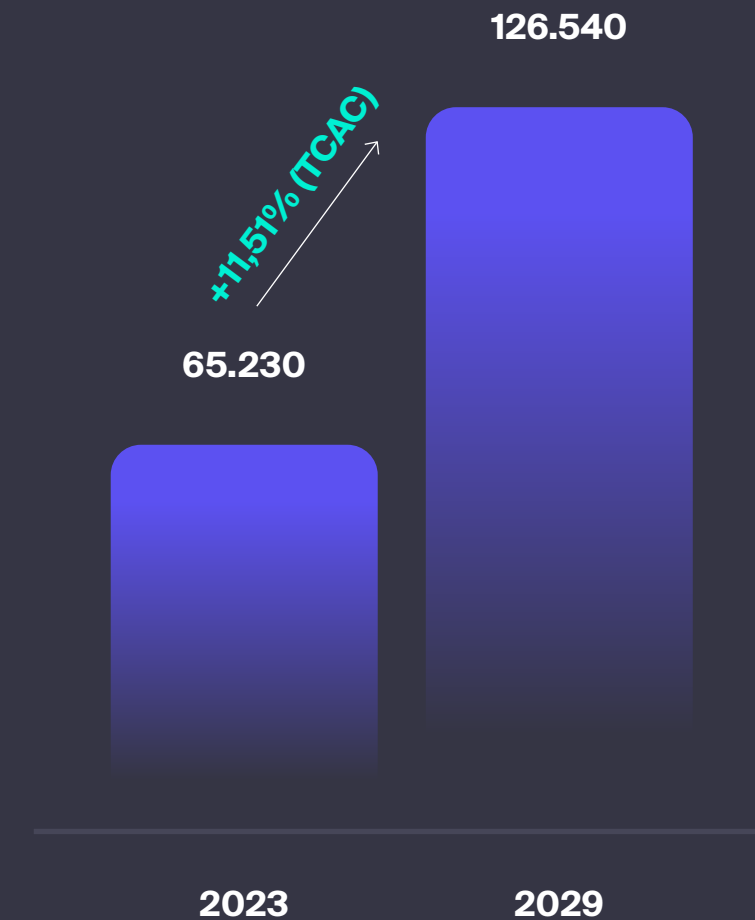
Se estima que la GenAI podría aportar entre 2,6 y 4,4 billones de \$ anuales a la economía mundial, y una parte significativa de este impacto provendrá de la ingeniería de software, junto con áreas como operaciones con clientes, marketing y ventas, e I+D. De hecho, **el mercado global de la ingeniería de software se valoró en 65.230 millones de \$ en 2023 y se espera que casi se duplique hasta alcanzar los 126.540 millones de \$ para 2029**, con una tasa de crecimiento anual compuesta (TCAC) del 11,51%. Esta rápida expansión ilustra hasta qué punto las herramientas y los procesos basados en IA están revolucionando la industria.



La aplicación de la GenAI en un número limitado de funciones podría concentrar la mayor parte del impacto potencial de esta tecnología en los distintos casos de uso corporativos



Tamaño del mercado global de la ingeniería de software
(en millones de \$)



Las nuevas reglas impulsadas por automatización e IA transforman radicalmente el desarrollo y el paradigma del software

Nuevas reglas y prácticas guían el desarrollo con inteligencia artificial

La adopción de la IA en la ingeniería de software no solo mejora la productividad, sino que **está modificando los cimientos de cómo se conciben, crean, prueban y despliegan las aplicaciones.** El rol del desarrollador, la forma en que se gestionan los proyectos y las prácticas para asegurar la calidad están siendo redefinidas bajo esta nueva lógica.

En el núcleo de esta transformación **se encuentra la capacidad de la IA para generar código de forma autónoma.** Estas herramientas actúan como asistentes de codificación en tiempo real, analizando el contexto y sugiriendo bloques completos de código. Este avance no reemplaza al desarrollador humano, sino que lo libera de tareas repetitivas y de bajo valor añadido, permitiéndole concentrarse en problemas de mayor nivel, como el diseño de algoritmos complejos, la optimización del rendimiento o la resolución de errores críticos.

Otro fenómeno son las nuevas plataformas de democratización del código, ahora potenciadas por IA. Estas **permiten que perfiles no técnicos puedan participar directamente en el desarrollo de aplicaciones,** describiendo funcionalidades en lenguaje natural que la plataforma traduce automáticamente en software funcional.

En el proceso DevOps, la IA mejora la estabilidad operativa mediante supervisión proactiva y despliegues más seguros. Todo ello exige, sin embargo, una gobernanza responsable, capaz de garantizar transparencia, evitar sesgos y mantener el control humano sobre los procesos automatizados.



En paralelo, el área de *testing*, uno de los cuellos de botella clásicos en el ciclo de desarrollo, **también ha sido transformada por herramientas que implementan inteligencia artificial**. Estos sistemas generan y ejecutan casos de prueba adaptativos, detectando anomalías en el comportamiento del software con una precisión y rapidez superiores a las técnicas manuales. La automatización de pruebas unitarias y de interfaz (UI) contribuye a mantener altos estándares de calidad y libera recursos que antes eran consumidos por tareas tediosas.

La influencia de la IA no se limita al desarrollo, sino que se extiende a la operación y entrega continua del software. Actualmente, la industria explora activamente la **posibilidad de un desarrollo de software completamente automatizado**. Esta visión contempla que la IA pueda encargarse de todo el ciclo: desde la recolección de requisitos hasta la implementación y mantenimiento (*Automated Machine Learning*).



La velocidad y precisión
al generar código no es opción,
sino la clave para revolucionar
el desarrollo moderno

La clave en ingeniería moderna: agilidad y precisión

El entorno evoluciona constantemente y se torna cada vez más competitivo, rápido y eficaz. Sin embargo, **los equipos de ingeniería aún dedican una enorme cantidad de tiempo a escribir código, ejecutar pruebas y corregir errores.** A medida que las aplicaciones escalan, este proceso se vuelve más complejo, costoso y difícil de mantener.

La falta de automatización puede generar ciclos de desarrollo más largos, incremento de la deuda técnica y mayores costes operativos. De hecho, se estima que los **desarrolladores reescriben, en promedio, el 26% del código**, lo que para una empresa mediana puede suponer más de 4,7 millones de dólares en costes anuales. En contraste, se ha demostrado que en proyectos con *Proof of Concept* (PoC) apoyadas con IA han logrado hasta un 40% de ahorro en costes y una aceleración del 40-60% en la toma de decisiones técnicas.



Por ende, su uso se convierte en una prioridad estratégica:



Aceleración de la entrega de software

Automatizando desde la generación de código hasta las pruebas y despliegues continuos se reduce significativamente el tiempo necesario para lanzar nuevas funcionalidades, permitiendo ciclos de entrega más cortos y una mayor capacidad de respuesta.



Reducción de costes de desarrollo y operaciones

Con el apoyo de herramientas automatizadas se reduce la carga de trabajo de los equipos, disminuyen los errores costosos y se optimiza el uso de la infraestructura. Traduciéndose en ahorros operativos significativos, especialmente en proyectos de gran escala.



Mejora la calidad del software desde fases tempranas

Identificando vulnerabilidades y patrones anómalos antes de que el software entre en producción (DevSecOps). Esto incrementa la robustez del producto final, minimiza la necesidad de correcciones poslanzamiento y reduce el riesgo de incidencias críticas.

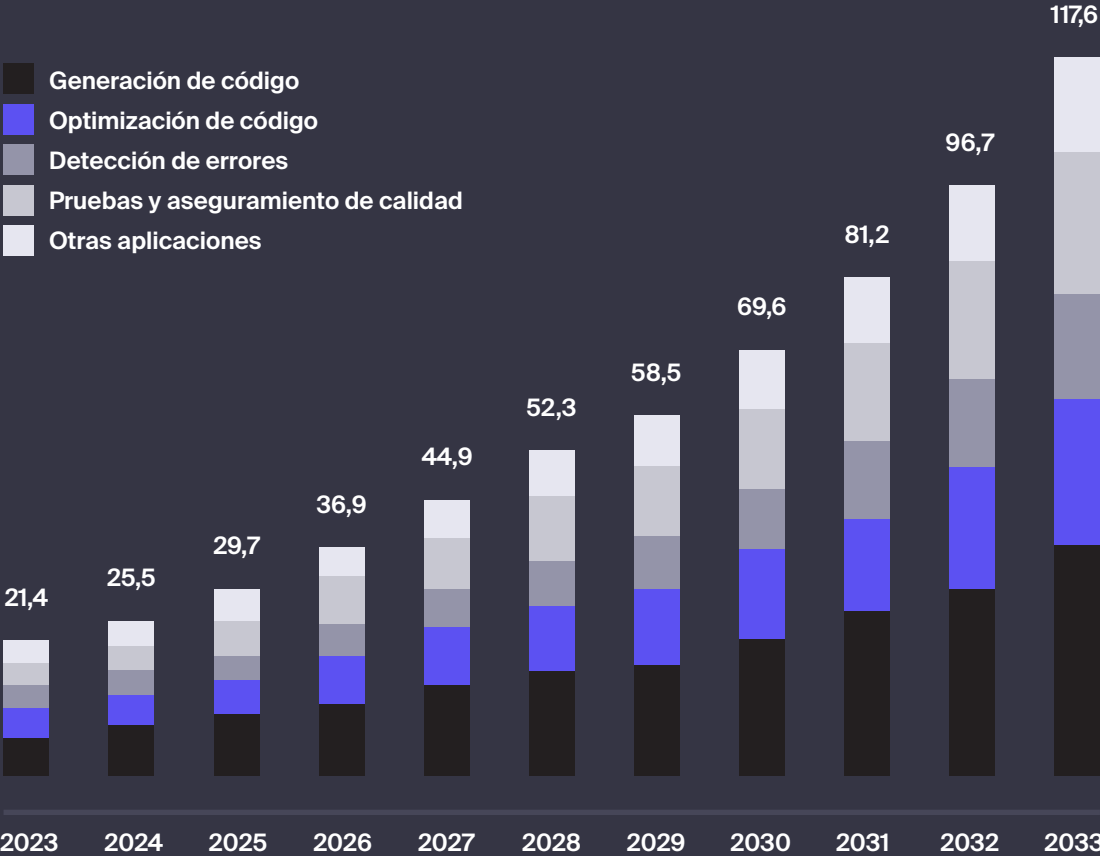


Optimización de la asignación de recursos humanos

Los desarrolladores pueden centrarse en aspectos estratégicos y de mayor relevancia, como la mejora de la experiencia de usuario. Elevando la productividad del equipo y permitiendo un mejor aprovechamiento del talento.

Fuente: Market Research

Tamaño del mercado de IA generativa en el desarrollo de software (en millones de \$)



El mercado crecerá a una tasa compuesta anual (TCAC) de: 19,1%

El tamaño de mercado previsto para 2033 es de: 117,6 millones de \$

No adaptarse implica perder competitividad, talento clave y acumular deuda técnica que compromete la innovación y sostenibilidad futura

El coste de no adaptarse: menos eficiencia, fuga de talento y deuda técnica

La automatización de procesos internos, la personalización a escala o la creación de agentes cognitivos están redefiniendo el mercado, y **las empresas que no adapten sus capacidades de desarrollo a esta nueva lógica corren el riesgo de quedar fuera del juego.**

El problema no es únicamente tecnológico sino, también financiero, operativo y estratégico. Mientras los **líderes del mercado construyen plataformas inteligentes y optimizan sus procesos con IA** generativa, quienes permanecen anclados en modelos tradicionales se enfrentan a una creciente brecha en eficiencia, costes y escalabilidad. Y esa brecha no deja de ampliarse.

Y el coste del talento no queda atrás, se calcula que **incorporar un equipo de IA hoy implica**

salarios altamente competitivos. Ingenieros de ML en EE.UU. pueden llegar a alcanzar los 180.000 \$ anuales, mientras que los *Data Scientists Senior* pueden alcanzar los 200.000 \$ en grandes corporaciones. Las empresas que no integran estos perfiles en su arquitectura operativa quedarán dependientes de terceros o verán limitada su capacidad de innovación.

Las organizaciones que no actúen ahora estarán pagando en productividad, en relevancia, en talento y en eficiencia un precio cada vez más difícil de justificar ante sus *stakeholders*.



El precio real de no adaptarse

Costes ocultos por falta de automatización



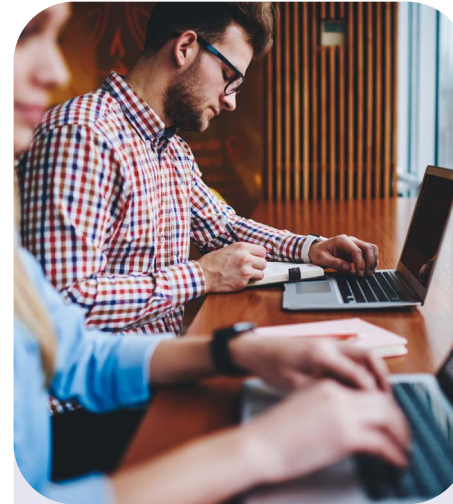
Cada flujo no optimizado con IA es tiempo humano desaprovechado. Por ejemplo, un solo modelo de IA que automatiza un flujo repetitivo puede ahorrar miles de horas al año. **No aplicarlo es asumir ese coste operativo de forma recurrente.**

Costes crecientes en mantenimiento y evolución



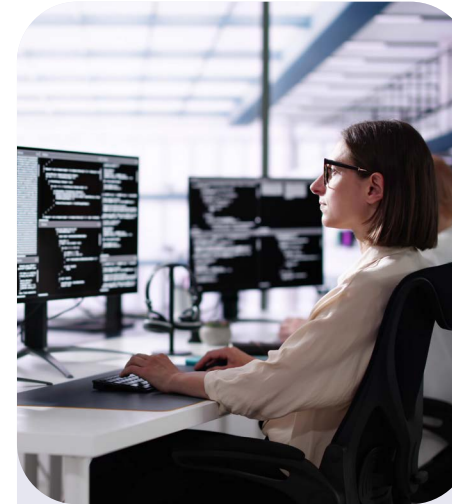
Mientras el **software tradicional requiere un 15-25% anual en mantenimiento**, los sistemas de IA, correctamente diseñados, pueden redistribuir ese esfuerzo mediante aprendizaje continuo y optimización automática. Sin IA, el mantenimiento crece en complejidad, coste y dependencia humana.

Desajuste presupuestario



Mantener una estrategia tradicional implica malgastar hasta un 30-50% del presupuesto en tareas que la IA ya permite automatizar (como preparación de datos, pruebas o refactorización). En cambio, adoptar IA reduce drásticamente los costes de desarrollo incremental y acelera el retorno sobre la inversión.

Desfase en talento y productividad



Los desarrolladores están migrando hacia entornos que integran IA como parte del ciclo de desarrollo. **No ofrecer estas herramientas impacta directamente en la atracción, retención y rendimiento** del equipo técnico.

Dependencia de infraestructuras rígidas y poco escalables



Entrenar modelos de IA **puede requerir entre 10.000 y 100.000 \$ mensuales en recursos de cómputo según la escala**. Pero no invertir en esa infraestructura significa quedarse sin capacidad para evolucionar productos inteligentes o responder al mercado en tiempo real.

La ingeniería aumentada redefine el desarrollo al integrar IA confiable, mejorando decisiones, velocidad, calidad y adaptabilidad en escala

Fundamentos de cambio: IA e ingeniería aumentada

El concepto de **Augmented Engineering** representa una transformación estructural en la manera en que concebimos, diseñamos y desplegamos soluciones tecnológicas. Integrando inteligencia artificial, *Machine Learning*, realidad aumentada (RA) y herramientas **generativas** en el núcleo de la práctica, potenciando la capacidad humana para enfrentar la complejidad técnica y organizativa de la era digital.

En el ámbito del desarrollo de software, esta evolución es especialmente significativa. La irrupción de la IA generativa ha desencadenado una aceleración en los ciclos de desarrollo, al **reducir el tiempo necesario para tareas** que históricamente han sido intensivas en esfuerzo humano, desde la redacción de código hasta la depuración, las pruebas o el despliegue. Sin embargo, esta transformación no puede limitarse a la mera adopción de asistentes de codificación. La verdadera promesa de la

ingeniería aumentada radica en un enfoque integral donde la inteligencia artificial actúa como colaboradora cognitiva en todas las etapas del ciclo de vida del software.

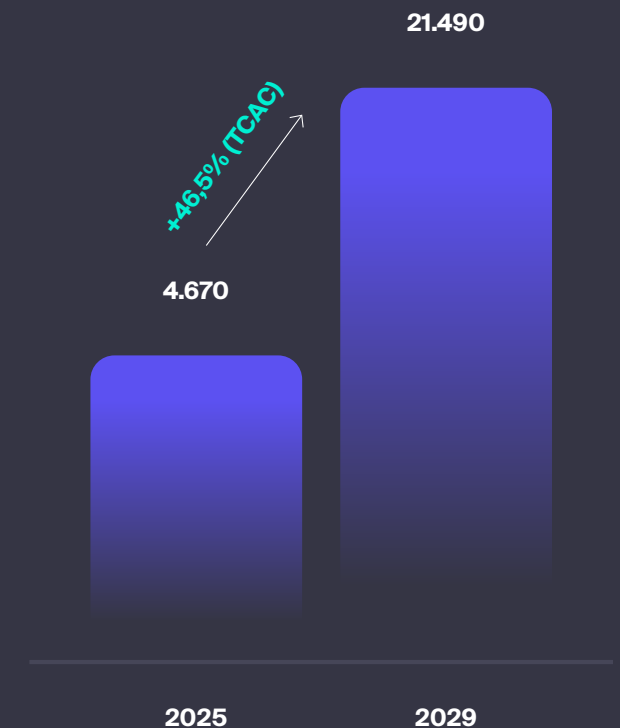
Esto se basa en la simbiosis entre el conocimiento experto del desarrollador y la capacidad analítica, generativa y predictiva de los sistemas de IA. En lugar de sustituir al profesional, la IA amplifica su impacto, analizando grandes volúmenes de código, anticipando errores potenciales, sugiriendo mejoras estructurales y acelerando las pruebas automatizadas. **Es una inteligencia que, bien integrada, no interrumpe el flujo creativo, sino que lo potencia.**

Según un estudio, **para 2028 el 75% de los desarrolladores empresariales utilizará asistentes de codificación basados en IA**, frente a menos del 10% en 2023. Más aún, se estima que el 80% de las actividades

del ciclo de vida del software involucrarán generación de código mediante GenAI en 2025, incrementando la productividad hasta en un 75% en determinados casos de uso.

Sin embargo, el camino hacia la ingeniería aumentada no está exento de desafíos. A diferencia de otros entornos tecnológicos más exploratorios o tolerantes al error, el desarrollo de software requiere niveles de precisión, validación estructural y cumplimiento normativo que los modelos generativos generalistas no están preparados para asumir por sí solos. **Esto implica una arquitectura tecnológica y organizativa capaz de soportar esa colaboración humano-máquina**, con sistemas de orquestación inteligentes, plataformas de automatización, entornos *Cloud* escalables, bases de datos vectoriales, herramientas de seguridad autónoma, modelos específicos de dominio y mecanismos continuos de retroalimentación.

Tamaño del mercado de ingeniería de software aumentada con IA (en millones de \$)



La ingeniería aumentada no solo automatiza tareas, sino también amplifica la capacidad humana para decisiones y adaptación continua en desarrollo

Ingeniería de software aumentada en el ciclo de vida del desarrollo

1

Ideación, visión y análisis potenciados:

modelos como GPT, combinados con RAG, permiten automatizar el estudio de tendencias de mercado, identificar espacios de oportunidad y sintetizar grandes volúmenes de información sobre clientes, competencia y tecnologías emergentes. Además, puede actuar como copiloto creativo en sesiones de *brainstorming*, sugiriendo funcionalidades, diferenciadores y flujos de usuario basados en casos similares, datos históricos y predicciones de comportamiento.

2

Evaluación de viabilidad:

analiza en paralelo la factibilidad técnica y la rentabilidad del proyecto. Utilizando bases de datos de proyectos anteriores, la IA estima recursos, tiempos y riesgos asociados. Asimismo, puede construir simulaciones de ROI, modelar escenarios alternativos y prever limitaciones tecnológicas.

3

Captura y priorización de requisitos:

se beneficia de capacidades como el procesamiento de lenguaje natural, que permiten extraer, estructurar y clasificar requerimientos desde múltiples fuentes. A su vez, la IA puede mantener actualizada la priorización de requisitos en función del contexto del *sprint*, restricciones técnicas o metas de negocio. La trazabilidad se fortalece con modelos que identifican dependencias y conflictos entre requisitos, reduciendo ambigüedades y omisiones.

4

Diseño asistido y arquitectura evolutiva:

actúa como guía que propone arquitecturas, patrones de diseño y decisiones técnicas basadas en el contexto funcional, las restricciones no funcionales y la experiencia acumulada. Puede incluso generar diagramas UML, validar estructuras y simular el impacto de modificaciones antes de su implementación. El diseño iterativo se enriquece con recomendaciones explicadas, y su revisión se agiliza mediante modelos capaces de detectar inconsistencias, omisiones o desviaciones del estándar.

5

Implementación, codificación aumentada y contextualizada:

los asistentes de codificación transforman el desarrollo diario. La IA puede sugerir funciones completas, refactorizar bloques obsoletos, generar código contextualizado y ofrecer explicaciones. Además, se adapta al estilo y preferencias del desarrollador, acelerando el *onboarding* en nuevos lenguajes o *frameworks*.

6

Pruebas automatizadas y prevención de fallos:

genera casos de prueba unitarios y funcionales, prioriza las pruebas según el riesgo y detecta puntos ciegos en la cobertura. También identifica defectos potenciales y vulnerabilidades a partir del análisis del código, aprendizaje de *bugs* históricos o análisis estático/dinámico. Con esta automatización inteligente, se reduce drásticamente el tiempo de *testing* sin comprometer la calidad.

7

Despliegue continuo y gestión de releases:

la IA mejora el despliegue mediante la generación automatizada de configuraciones específicas por entorno (IaC), la optimización de estrategias de *releases* y el control de versiones. Puede sugerir horarios óptimos, predecir fallos y coordinar la reversión automática ante anomalías. Además, facilita la documentación y trazabilidad de cada cambio, manteniendo la gobernanza incluso en entornos de alta velocidad.

8

Operaciones inteligentes y resiliencia continua:

en producción, permite un monitoreo proactivo y una respuesta adaptativa. Mediante el análisis en tiempo real de *logs*, trazas y métricas, la IA detecta desviaciones, predice incidentes y sugiere ajustes automáticos en el escalado de recursos. La gestión de incidentes se automatiza parcialmente con ChatOps, mientras que el rendimiento se optimiza mediante el reconocimiento de patrones y cuellos de botella. La IA también actúa como primer nivel de soporte, proponiendo soluciones basadas en incidentes anteriores.

9

Mantenimiento evolutivo y mejora continua:

detecta código muerto, deuda técnica y oportunidades de mejora. A través de los análisis semánticos, propone refactorizaciones, migraciones tecnológicas o reestructuración de componentes. Además, puede responder a consultas contextuales sobre el código, generar documentación viva y anticipar efectos colaterales de cambios.

10

Fin de vida: migración y retirada responsable:

finalmente, la IA también contribuye al retiro de sistemas. Puede generar planes de migración de datos entre plataformas, redactar *scripts* de apagado progresivo y generar contenidos formativos personalizados para usuarios que migran a nuevos sistemas. Incluso puede ayudar a gestionar la resistencia al cambio, identificando patrones en el *feedback* y proponiendo mensajes, ajustes o materiales de ayuda para facilitar la transición.

La ingeniería aumentada exige IA con visión sistémica, adaptabilidad organizacional y alineación entre personas, procesos y tecnología

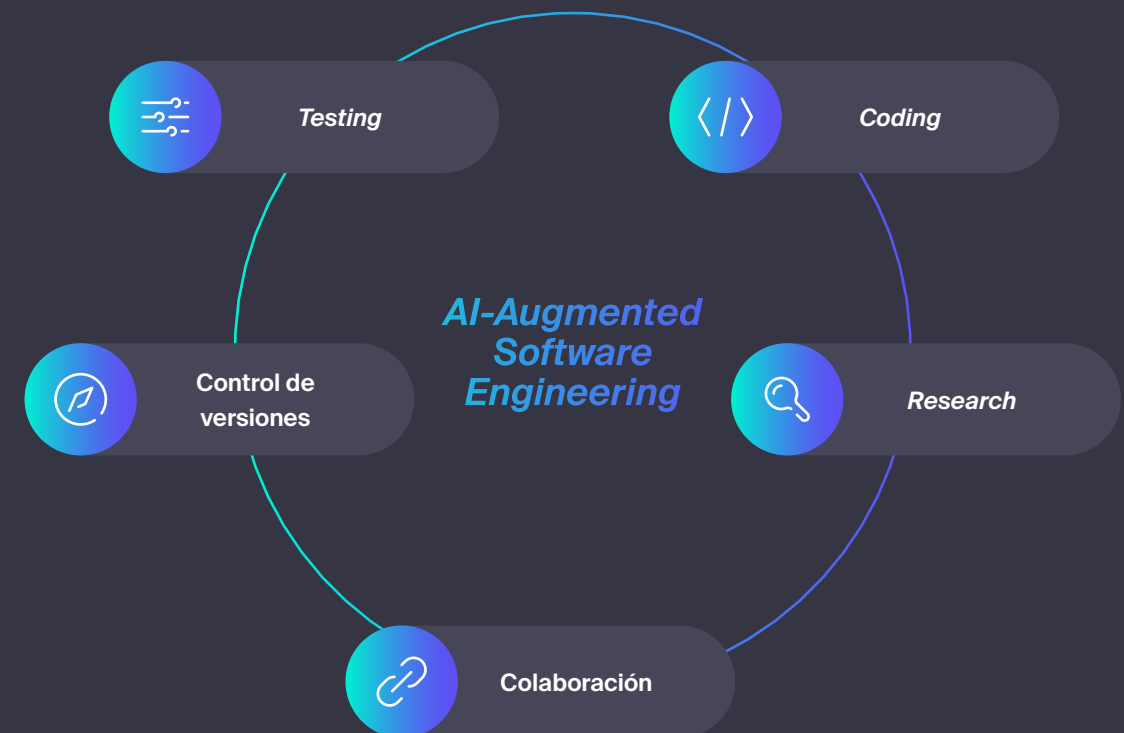
AI-Augmented Software Engineering (AIASE)

No obstante, en entornos de ingeniería crítica, como financiera, de salud o de administración pública, los **desafíos son mayores**.

Para que la IA genere valor real en estos entornos, es necesario dotarla de capacidades extendidas. En primer lugar, los sistemas deben ser capaces de procesar insumos complejos, como esquemas técnicos, simulaciones, flujos de negocio o requisitos legales, **para generar software que funcione y respete las restricciones estructurales, normativas u operativas del dominio**. A esto se suma la validación automática de conformidad, incorporando desde el diseño reglas sectoriales, estándares regulatorios y requerimientos de auditoría, un enfoque de *compliance by design*.

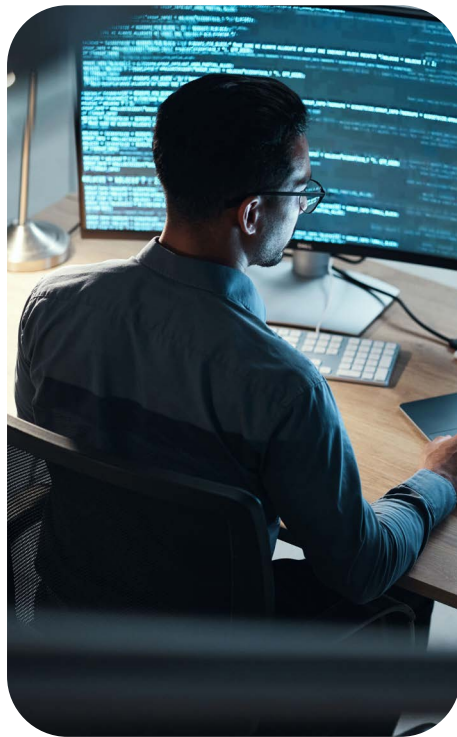
A su vez, es igual de esencial asegurar la trazabilidad de cada decisión algorítmica, desde la fuente de los datos hasta el impacto de cualquier cambio en producción. Esto es crítico en marcos regulatorios como el EU AI Act o normativas específicas de sectores como el financiero (Basel III, EBA *guidelines*) o el sanitario (HIPAA, MDR). Y aunque la automatización crece, en estos entornos **sigue siendo obligatorio el control humano significativo** (*human-in-the-loop*), donde los sistemas pueden asistir, pero las decisiones críticas siguen pasando por un experto cualificado.

AIASE ofrece apoyo constante en diversas áreas del desarrollo de software



Su impacto puede observarse
claramente al desglosarlo
en cinco áreas clave:

Coding



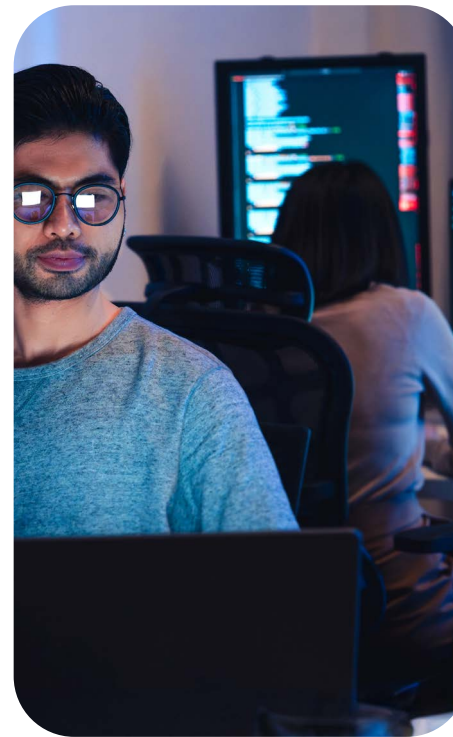
la IA permite la generación asistida de código y la revisión del mismo, basado en NLP y contexto del proyecto.

Research



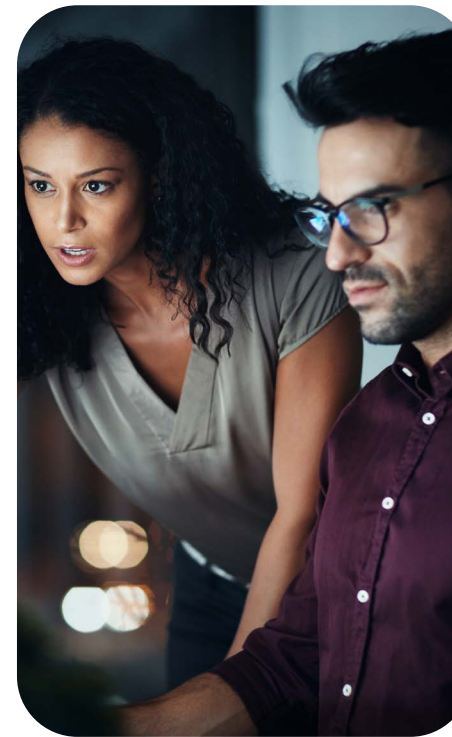
los IDE aumentados por IA actúan como *hubs* de conocimiento, facilitando información técnica y documentación sin interrupciones.

Controles de versiones



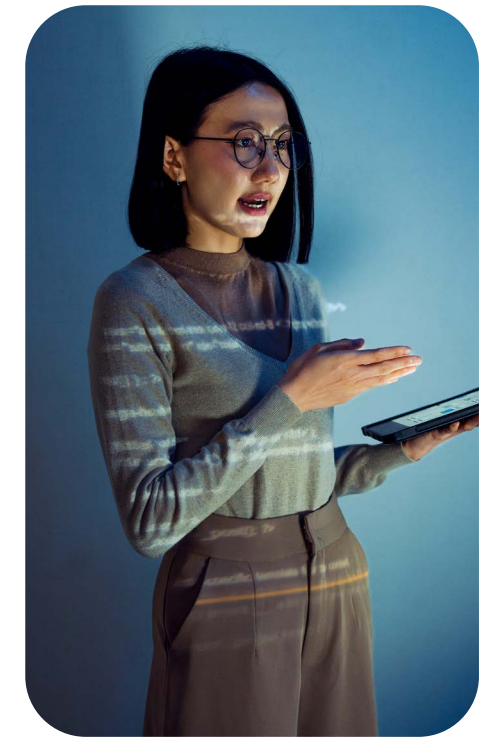
a través del análisis continuo, permiten detectar regresiones, anomalías, cuellos de botella o deuda técnica en tiempo real.

Colaboración



estas soluciones notifican, documentan y comparten contexto de manera automática, manteniendo la coherencia del equipo.

Testing



la IA es capaz de generar, priorizar y mantener pruebas inteligentes y adaptativas, incluyendo casos de autorreparación ante cambios.

Hay desafíos éticos, técnicos y culturales que requieren estrategia clara para gobernanza, escalabilidad y adopción responsable

Desafíos estratégicos del desarrollo de software potenciado por IA

Inversión inicial elevada:

las soluciones de IA pueden implicar costes significativos de licenciamiento, infraestructura y talento especializado. Esto supone una barrera de entrada especialmente crítica para organizaciones pequeñas o con entornos *legacy*.



Riesgos de privacidad y seguridad de los datos:

los modelos de IA requieren grandes volúmenes de datos, que muchas veces incluyen información confidencial, lo que genera preocupaciones en torno al cumplimiento normativo (ej. GDPR, CCPA) y la protección de datos sensibles.



Escasez de talento especializado:

la adopción efectiva de IA exige conocimientos en *Machine Learning*, ingeniería de datos, MLOps y herramientas específicas, perfiles que son escasos y altamente demandados



Integración con sistemas legacy:

los sistemas heredados pueden presentar barreras técnicas para la integración de soluciones de IA modernas, lo que ralentiza la adopción y genera fricciones operativas.



Explicabilidad y confianza:

las decisiones generadas por modelos de IA a menudo no son interpretables por los equipos técnicos o *stakeholders*, lo que dificulta la validación, el cumplimiento y la adopción cultural.



Riesgos éticos y sesgo algorítmico:

la IA puede amplificar sesgos presentes en los datos o modelos mal entrenados, generando resultados injustos o discriminatorios. Además, la automatización excesiva puede generar tensiones laborales y resistencia al cambio.



Pérdida del factor humano:

el uso intensivo de IA puede derivar en dependencia tecnológica, pérdida de pensamiento creativo o erosión de habilidades blandas como la empatía, el juicio contextual o el diseño centrado en el usuario.



Recomendaciones

- Soluciones específicas de bajo coste de entrada
- Herramientas basadas en suscripción a la nube
- Modelos *open source* optimizados localmente
- Herramientas con encriptación *end-to-end*, control de acceso y auditoría
- Gobernanza sólida y monitoreos
- Arquitecturas híbridas que mantengan datos de manera local
- *Upskilling* con formaciones, Colaborar con consultoras expertas
- Establecer comunidades internas de práctica en IA
- Estrategias de integración progresiva (*phased approach*)
- Soluciones que operen en capas superficiales sin modificar el *core legacy*
- Diseñar las API que conecten sistemas *legacy* con capacidades inteligentes.
- Soluciones de IA que incorporen mecanismos XAI
- Trazabilidad y visualizaciones del razonamiento
- Documentación de decisiones del sistema para todos los usuarios
- Marcos éticos claros que rijan su uso
- Auditorías periódicas para evaluar sesgos y revisar su impacto social
- Diseñar IA como tecnología complementaria
- Herramientas de colaboración humano-máquina
- Promover modelos colaborativos
- Evaluar el impacto en el rendimiento integral del equipo

Entre estos desafíos, la latencia también emerge como una barrera crítica para sistemas impulsados por IA en producción

Latencia, el nuevo “downtime” de la IA

En el contexto de las aplicaciones generativas, la latencia ha emergido como un factor tan disruptivo como históricamente lo ha sido la caída de servicios: es el nuevo “downtime”. La premisa que subyace es clara, **si una respuesta generada por IA no llega a tiempo, no llega con valor**. Con la popularización de interfaces conversacionales y agentes autónomos, se exige que el tiempo de respuesta se asemeje al de una conversación humana. Este desafío es aún mayor cuando se trata de modelos de razonamiento, que pueden ofrecer respuestas de mayor calidad si se les permite un tiempo adicional para “pensar”.

En el desarrollo de software, la latencia **es el tiempo que transcurre entre que un desarrollador solicita una acción y la recepción de la respuesta por parte del sistema de IA**. Este retardo, medido en segundos, puede parecer trivial en otros contextos, pero en un flujo creativo representa una interrupción real. El efecto de la latencia es directo, una IA que responde con lentitud pierde valor como asistente ya que desincentiva su uso, altera el flujo de trabajo y genera una experiencia percibida como torpe o poco fiable,

este fenómeno se amplifica en contextos colaborativos. Las demoras afectan la eficiencia en revisiones de código, la calidad de las iteraciones ágiles y la moral de los equipos.

Además, introduce un nuevo tipo de deuda técnica. A medida que los equipos confían más en la IA para tareas rutinarias o críticas, la falta de inmediatez deteriora la experiencia general, frena la adopción a largo plazo y puede llegar a comprometer la eficacia operativa. En términos cuantificables, **se estima que en escenarios óptimos una IA con baja latencia puede mejorar la productividad por encima del 20%**, pero si esa misma herramienta responde con retardos constantes, esos beneficios se erosionan hasta volverse contraproducentes.

A medida que las organizaciones integran sistemas de IA generativa en sus pipelines de desarrollo, la latencia se convierte en un nuevo KPI. Ya no basta con que el modelo sea preciso o versátil; debe ser también rápido, estable y consistente. Las empresas líderes están comenzando a rediseñar sus arquitecturas para enfrentar este desafío, recurriendo a estrategias como:

Edge inference

donde los modelos se ejecutan parcialmente en dispositivos locales para reducir el ida y vuelta con la nube.

Modelos ligeros adaptados

como LLaMA 3, Mistral o versiones optimizadas para inferencia rápida.

Cacheo contextual

que reduce la necesidad de procesar desde cero en cada interacción.

Arquitecturas RAG optimizadas

que combinan recuperación de contexto con generación localizada.

Una organización que logra reducir al mínimo la latencia de sus herramientas aumentadas se posiciona estratégicamente para liderar en velocidad, calidad y adaptabilidad. Por el contrario, aquellas que toleren entornos lentos, inconexos o frustrantes corren el riesgo de quedar atrás en la carrera de la transformación digital.

La ingeniería de software aumentada
impulsa un nuevo paradigma con código
inteligente, adaptable, rápido y humano

Visión de futuro, el nuevo ADN del desarrollo de software

En la década actual y las futuras, un amplio catálogo de tecnologías emergentes alcanzará una adopción generalizada y transformará radicalmente la forma en que las organizaciones diseñan, desarrollan y operan software. Esto permitirá **redirigir el talento hacia áreas de alto impacto como iniciativas estratégicas, resolución de incertidumbre técnica o cocreación con el negocio** y, al mismo

tiempo, mejorará la resiliencia operativa. Los entornos de prueba se enriquecen con rutas de código no obvias, capaces de detectar fallos antes de que escalen, y ofrecer soluciones automatizadas. Desencadenando en menor deuda técnica, mayor robustez y una entrega continua alineada con los objetivos de negocio.





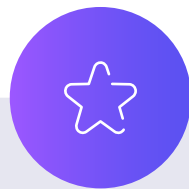
De codificación manual a colaboración humano-IA

Los desarrolladores ya no trabajan solos, gracias a los asistentes de codificación se **reduce la carga cognitiva y acelera la entrega.**



Bases de código adaptativas

Los repositorios dejan de ser estáticos para aprender y evolucionar. Modelos entrenados sobre el histórico de tests y despliegues pueden **anticipar errores, sugerir refactorizaciones o aplicar parches de seguridad.**



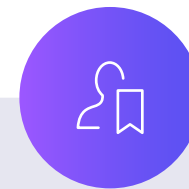
Democratización del desarrollo

El AIASE impulsa un acceso inclusivo al desarrollo de software. Plataformas LCNC con capacidades generativas permiten que **usuarios sin formación técnica contribuyan a la creación de productos digitales.**



Nuevos roles y habilidades

Los desarrolladores asumen un rol de diseñadores de sistemas inteligentes. La competencia clave **será saber colaborar con sistemas autónomos, validar outputs, entender su lógica, y tomar decisiones éticas y responsables.**



Más calidad, menos errores

Diversas herramientas están reduciendo en miles los errores en producción **mediante pruebas y refactorización automatizada** gracias a la ingeniería aumentada de software.



87%

de los desarrolladores indicó que GitHub Copilot les permitió **conservar la energía mental durante tareas repetitivas.**

30%

de las **sugerencias de código ofrecidas por GitHub Copilot son aceptadas** en promedio por los usuarios.

75%

de las grandes corporaciones utilizará al menos cuatro herramientas LCNC tanto para **aplicaciones de IT como para iniciativas de citizen development.**

67%

de las **organizaciones de IA maduras están creando nuevos roles relacionados con esta**, el 87% ya cuentan con equipos de IA.

87%

menos defectos en producción y 70% **menos tiempo en revisiones** manuales debido a herramientas de revisión de código con IA.



Tendencias clave en *AI-Driven Development*

La nueva economía del software se construye con código asistido, LCNC, VibeOps, agentes autónomos y automatización inteligente

5.000 **M de \$**

El gasto global en TI en 2024

850.000 **M de \$**

Tamaño del mercado de desarrollo de software para 2028

Tendencias clave hacia el futuro del desarrollo de software

A medida que las organizaciones enfrentan entornos económicos cada vez más restrictivos, el desarrollo de software no solo persiste, sino que consigue reinventarse. Las inversiones en transformación digital siguen creciendo, se estima que el **gasto global en TI alcanzó los 5.000 millones de \$ en 2024**, con una fuerte orientación hacia plataformas ágiles, inteligencia artificial y herramientas que aumentan la productividad.

Además, se espera que el mercado de desarrollo de software supere los 850.000 millones de \$ en 2028. Esto evidencia que el **software sigue siendo un motor para la transformación digital**. Este contexto está impulsando una nueva era del código, marcada por aceleración, colaboración humano-IA y sostenibilidad.

Este panorama exige repensar cómo se construye software, con qué herramientas, por quiénes y con qué impacto. En este escenario emergen **una serie de tendencias clave que están redefiniendo los pilares del desarrollo**: desde la incorporación masiva de la inteligencia artificial hasta la democratización del código, pasando por la sostenibilidad digital, los nuevos roles del desarrollador y la evolución hacia plataformas como modelo operativo.

Lo que antes era un proceso secuencial, artesanal y altamente especializado, hoy se convierte en una práctica colaborativa, asistida por IA, más abierta, veloz y alineada con objetivos de negocio. Comprender estas tendencias no es solo una cuestión de innovación, es **una hoja de ruta para asegurar la competitividad futura**.

Tendencias clave



1. Low-Code / No-Code



2. Asistentes de código avanzados



3. Automatización de la arquitectura



4. Agentes autónomos de mantenimiento



5. Ingeniería inversa y refactorización



6. Vibe Coding y VibeOps

1 **Low-Code / No-Code:** el otro motor silencioso del cambio

El desarrollo de software ya no está reservado exclusivamente para perfiles con años de experiencia en codificación. Las organizaciones que antes requerían grandes equipos técnicos para construir incluso aplicaciones sencillas ahora están **transformando sus procesos mediante plataformas Low-Code y No-Code (LCNC)**. Estas herramientas permiten crear aplicaciones a través de interfaces visuales y componentes predefinidos, minimizando la necesidad de escribir código y aumentando la velocidad, accesibilidad y escalabilidad del desarrollo.

La incorporación de inteligencia artificial en estas plataformas potencia aún más sus capacidades, simplemente con describir una funcionalidad en lenguaje natural el sistema puede traducirlo en una aplicación operativa. No solo agilizando la entrega de soluciones sino, también, **ampliando aún más la base de usuarios capaces de participar en el desarrollo**, desde analistas de negocio hasta *Product Managers*, que pueden ahora crear o adaptar herramientas sin depender constantemente de un equipo de ingeniería.

Tanto las pymes como grandes corporaciones se pueden beneficiar de las ventajas competitivas ya que, **un 48 % de los líderes**

de TI y ejecutivos C-level afirman haber acelerado el desarrollo de aplicaciones gracias al Low-Code, mientras que un 45 % destaca su impacto directo en la reducción de costes. No sorprende, entonces, que el mercado global de plataformas *Low-Code* estuviera valorado en 24.830 millones de \$ en 2023 y proyecte un crecimiento anual del 22,5 % hasta 2030.



Democratización del desarrollo: cuando todos pueden crear software

Este movimiento se basa en **una filosofía digital orientada a la democratización del desarrollo de software**, eliminando la barrera del conocimiento técnico profundo. Incluyendo tanto ventajas como nuevos desafíos:

Ventajas

Facilidad de uso y aplicación

Visualización inmediata de resultados

Reducción significativa
de costes de desarrollo

Menor carga de mantenimiento
y mayor productividad

Desafíos

Restricciones de personalización

Dependencia de entornos proveedores

Perdida de buenas prácticas de diseño
y programación por falta de experiencia

Soluciones poco escalables o de baja
calidad sin una adecuada gobernanza

80%

de las empresas
afirma que el *Low-Code*
mejora la productividad

79%

afirma que el *Low-Code*
reduce los costes
operativos

73%

afirma que el
Low-Code mejora
el *time-to-market*

2 Asistentes de código avanzados

El **mercado de las herramientas de desarrollo asistido por IA alcanzó los 12.560 millones de \$ en 2024** y continuará creciendo anualmente un **24,5% hasta 2030**.

Los asistentes de código con IA representan una nueva capa de inteligencia dentro del entorno de desarrollo. **Su lógica está construida sobre los modelos de lenguaje a gran escala**, como GPT o Codex, entrenados sobre millones de líneas de código y patrones sintácticos provenientes de múltiples lenguajes, *frameworks* y repositorios públicos. Esta base les permite no solo generar código coherente, sino comprender el contexto funcional de lo que se está programando, proponer soluciones completas y detectar problemas antes de que estos lleguen a ejecutarse.

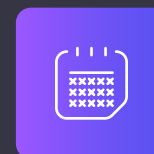
De esta manera, a IA **actúa como un acelerador cognitivo que libera tiempo y esfuerzo** mental para que la persona se concentre en las tareas de mayor complejidad conceptual. Al reducir la fricción en la escritura de código repetitivo, la creación de pruebas unitarias o la refactorización estructural, los asistentes de IA permiten que el talento técnico se oriente más hacia el diseño, la estrategia y la arquitectura.

La clave de su funcionamiento reside en **su capacidad para interactuar** fluidamente con los entornos de desarrollo integrados más utilizados, como Visual Studio Code, IntelliJ IDEA o plataformas en la nube como GitHub. Esto significa que el desarrollador no tiene que aprender un nuevo sistema, sino que incorpora la IA directamente en su flujo de trabajo habitual, convirtiéndola en una extensión natural de su proceso creativo.

Más del 30% del nuevo código en empresas como Google o Microsoft ya se genera mediante IA, con proyecciones que apuntan a superar el 90% hacia el final de la década.

La verdadera potencia de estos asistentes radica en su capacidad para entender el contexto amplio en el que se encuentran. Es decir, no solo operan sobre la línea de código que se está escribiendo, sino que aprenden de la estructura del proyecto, respetan convenciones internas y se integran con otras herramientas del ecosistema, lo que garantiza consistencia y productividad a lo largo del ciclo de vida del software.

Adopción y frecuencia de uso del asistente de codificación de IA por desarrolladores



49%

los usa a diario



39%

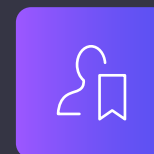
los usa
semanalmente



58%

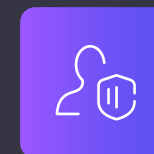
comenzó a usarlos
en los últimos 6 meses

Principales preocupaciones de los desarrolladores sobre el uso de la IA en la ingeniería de software



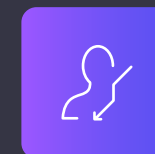
55%

se preocupa por la
calidad del código
generado por IA



48%

tiene inquietudes
sobre seguridad
y privacidad



46%

teme una menor
demanda de
desarrolladores

3 Automatización de la arquitectura

La incorporación de inteligencia artificial en la arquitectura implica una reconfiguración del proceso decisional, donde modelos de aprendizaje automático, analítica predictiva y automatización permiten que los **sistemas se diseñen con mayor eficiencia, se escalen dinámicamente, se mantengan y documenten en tiempo real** y anticipen los fallos, cuellos de botella y riesgos de seguridad antes de desplegarse.

En paralelo, **la automatización de la documentación arquitectónica representa otro salto de valor**. A través del NLP, los sistemas son capaces de extraer información técnica directamente de las especificaciones, del código o de los artefactos generados en el entorno de desarrollo, y transformarla en documentación estructurada, legible y dinámica. Esto facilita enormemente el traspaso de conocimiento, la incorporación de nuevos perfiles al equipo y la auditoría técnica en contextos regulados.

Además del diseño y la documentación, la IA también ha comenzado a modificar las dinámicas de despliegue y mantenimiento, especialmente mediante la automatización del aprovisionamiento de infraestructura, la

integración de pipelines CI/CD inteligentes y el monitoreo continuo del rendimiento del sistema. Dando lugar a un nuevo concepto de arquitectura, la arquitectura autocurativa (*self-healing*), capaz de detectar anomalías, diagnosticar fallos y aplicar medidas correctivas sin necesidad de intervención humana directa. Del mismo modo, los **sistemas autooptimizados (*self-optimising*) ajustan automáticamente recursos, balancean cargas y escalan servicios según patrones de uso, logrando así una arquitectura más eficiente y sostenible en costes**.

En términos de gestión del cambio, esto implica que las organizaciones deben comenzar a integrar la IA no como una herramienta externa, sino como una capacidad interna del sistema, incorporada desde el diseño inicial. No obstante, **este nuevo enfoque conlleva también responsabilidades**. La introducción de IA en la arquitectura requiere revisar las prácticas de gobernanza, incorporar mecanismos de validación continua de los modelos, asegurar la transparencia de las recomendaciones y mantener una supervisión humana efectiva que garantice la alineación con los objetivos del negocio, la seguridad del sistema y los principios éticos.

Automatización del modelado

Herramientas como Structurizr o ArchiMate con capacidades AI permiten:



Generar planos de arquitectura a partir de requisitos.



Dibujar mapas de dependencias.



Proponer patrones arquitectónicos según casos de éxito previos.



Reducir el esfuerzo manual y mejora la estandarización.

Reconocimiento de patrones

IA analiza arquitecturas exitosas anteriores para:



Detectar patrones replicables.



Sugerir estructuras de microservicios, caché, colas de mensajes, entre otras.



Optimizar rendimiento y escalabilidad desde el diseño inicial.

4 Agentes autónomos de mantenimiento

A diferencia de los asistentes tradicionales y de la mano de la automatización y detección de patrones, los **asistentes autónomos son capaces de descomponer objetivos, planificar tareas, escribir y probar código y aprender del resultado sin necesidad de intervención humana.**

Uno de los avances más significativos es el paso del **mantenimiento correctivo tradicional hacia un modelo predictivo y preventivo**, gracias a modelos que analizan patrones históricos de fallos, uso del sistema y rendimiento. Estas herramientas pueden anticipar incidentes antes de que ocurran, permitiendo acciones que reducen los tiempos de inactividad, optimizan el uso de recursos y evitan interrupciones en servicios .

En paralelo, la capacidad de analizar grandes volúmenes de *logs*, métricas y telemetría en tiempo real permite a los sistemas **identificar anomalías, clasificar automáticamente errores y proponer correcciones específicas.** Estos procesos se están convirtiendo en dominios nativos de agentes IA capaces de operar sin intervención directa, aunque siempre bajo supervisión humana en contextos sensibles.

El caso de SapFix, la herramienta desarrollada por Meta, representa un hito en la automatización del *debugging*. Este sistema genera parches para errores detectados, los valida mediante pruebas automatizadas y los somete a revisión humana antes de su despliegue. Lo relevante aquí no es solo la automatización del parcheo, sino la **integración fluida entre IA y humanos en un ciclo de mejora continua, trazable y validado.** Este tipo de colaboración híbrida sienta las bases de una ingeniería de confiabilidad aumentada, donde los humanos supervisan y afinan, pero no intervienen directamente en cada incidente.

Más allá del *debugging*, su auge plantea un nuevo nivel de automatización en la gestión de sistemas. Estos agentes pueden **aplicar parches, reiniciar servicios, escalar recursos o ejecutar tareas de mantenimiento** basadas en condiciones contextuales. En el ámbito de la ciberseguridad, pueden incluso actuar como centinelas activos, bloqueando amenazas en tiempo real y asegurando el cumplimiento normativo sin requerir intervención inmediata del equipo de TI.

En el ámbito del soporte, la IA está optimizando el ciclo completo de atención a incidencias. Los sistemas de NLP permiten **clasificar tickets, priorizarlos según criticidad y generar respuestas automáticas fundamentadas en bases de conocimiento.** Esto libera a los equipos de soporte de tareas y mejora los tiempos de resolución, al tiempo que proporciona a los usuarios finales una experiencia más ágil y personalizada.

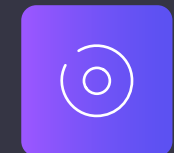
Además, plataformas avanzadas permiten hoy integrar estos flujos con asistentes conversacionales y agentes LLM capaces de **interactuar con los usuarios, entender descripciones informales de problemas, diagnosticar causas probables y ofrecer soluciones automáticas o guiadas.** El resultado es una capa de inteligencia distribuida que transforma el soporte en un proceso conversacional, omnipresente y adaptativo.

Plataformas como **Retool con Agents** o **Amazon con Kiro**, ya muestran que los agentes pueden completar de forma autónoma más del 30% de las tareas complejas sin ningún tipo de supervisión del equipo desarrollador.



75%

identifica las pruebas impulsadas por IA como clave en su estrategia para 2025



16%

de los encuestados ha adoptado la IA en pruebas, lo que muestra que la adopción real aún está rezagada

5 Ingeniería inversa y refactorización automatizada

Modernizar el software heredado es uno de los **retos más urgentes**. Miles de líneas de código siguen en operación en lenguajes ya obsoletos.

El principal catalizador de este cambio es la **capacidad de los LLM para comprender y generar código en múltiples lenguajes**, así como para inferir estructuras, patrones y funcionalidades desde bases de código sin documentación. Estas capacidades habilitan nuevas formas de abordar la modernización: desde el análisis automatizado de funciones hasta la traducción semánticamente precisa a lenguajes modernos, pasando por la detección de vulnerabilidades y la reconstrucción de conocimiento técnico perdido. El impacto organizacional de estas herramientas no es menor. En lugar de embarcarse en migraciones “*big bang*” costosas y arriesgadas, **las empresas pueden adoptar estrategias de modernización progresiva apoyadas en inteligencia artificial**. Casos como el de bancos japoneses que han convertido millones de líneas de COBOL a Java en menos de un año o el gobierno estatal americano que han migrado sistemas de desempleo de PL/I a Python ilustran que la promesa es real y ya está en marcha.

A nivel técnico, la IA está transformando la ingeniería inversa en al menos tres dimensiones. Primero, a través del

entendimiento automatizado de código, con modelos capaces de identificar propósitos de funciones, dependencias entre módulos y estructuras de control complejas. Segundo, mediante la traducción inteligente de código, donde el foco ya no es una simple conversión sintáctica, sino la preservación del contexto y la lógica de negocio. Y tercero, al reforzar la seguridad de estos sistemas, detectando vulnerabilidades históricas y proponiendo remediaciones conforme a estándares modernos de ciberseguridad. Esta automatización también está llegando al análisis de *malware* y código hostil. Modelos como DeGPT, diseñados para mejorar la legibilidad y simplificación de código descompilado, **permiten entender con rapidez el comportamiento de software malicioso, mapearlo contra frameworks como MITRE ATT&CK y generar reportes automáticos** con recomendaciones de mitigación. En este campo, la IA no solo ahorra tiempo; representa una diferencia crítica en la respuesta frente a amenazas avanzadas.

En EE.UU., más del 80% del gasto público en TI sigue destinado a mantenimiento de sistemas antiguos, lo que subraya la urgencia de soluciones automatizadas. Herramientas como las desarrolladas por IBM ya **han demostrado mantener más del 90% de la cobertura funcional con intervención mínima**.



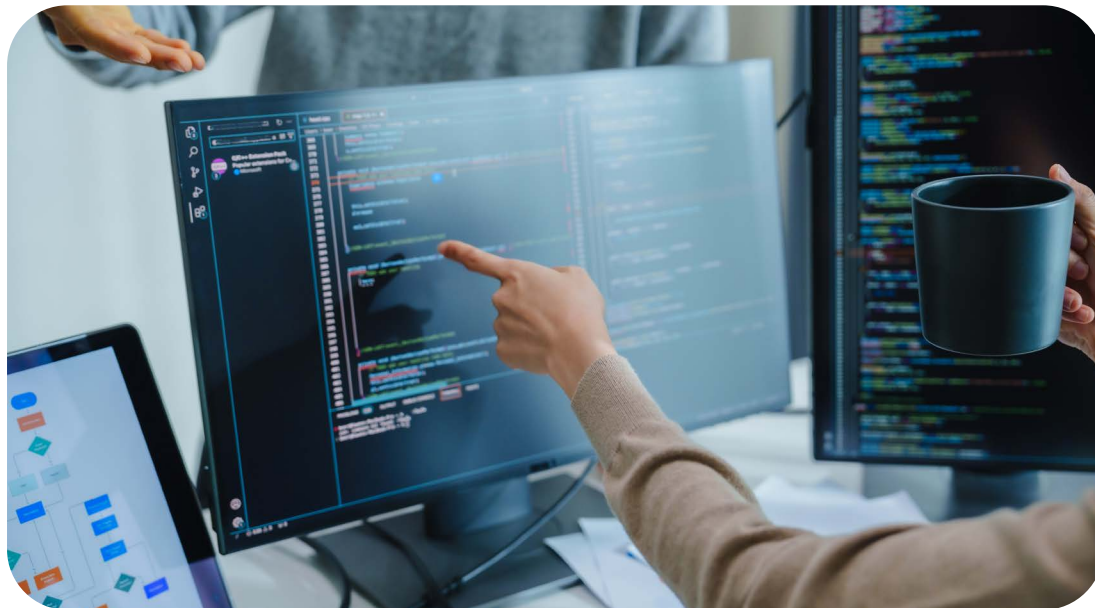
6 Vibe Coding & VibeOps: programación por vibra

A comienzos de 2025, se introdujo un concepto que rápidamente captó la atención de la comunidad tecnológica: el **Vibe Coding**. Esta **tendencia redefine la programación como una actividad guiada por la intención del desarrollador**, quien expresa en lenguaje natural lo que necesita, mientras modelos de lenguaje LLM traducen esa intención en código ejecutable.

Poco después emergió su evolución natural, el **VibeOps**. Un enfoque operacional impulsado

por IA que busca maximizar la productividad del desarrollador eliminando fricciones operativas y permitiéndole centrarse en la creación de valor.

Ambas tendencias **comparten la visión común de transformar la experiencia de desarrollo en un proceso fluido, multimodal y asistido** por inteligencia artificial, donde el desarrollador se convierte en arquitecto de intenciones más que en ejecutor de tareas mecánicas.



Vibe Coding: de la intención al código ejecutable

Este enfoque prioriza la creación rápida sobre la optimización prematura o "*code first, refine later*". Esta tendencia se alinea perfectamente con los marcos ágiles, facilitando el prototipado rápido, ciclos iterativos y validación temprana de conceptos. Así, la programación se convierte en una experiencia más dialógica e intuitiva, donde el *developer* mantiene el control conceptual y la IA actúa como copiloto técnico.

La ventaja está en acelerar el paso de una idea a un MVP, facilitando la exploración técnica con bajo coste, mayor adaptabilidad y posibilidad de pivotar más rápido, minimizando costes hundidos. La meta es un entorno de desarrollo donde agentes de IA actúan como asistentes omnipresentes, es decir, sugieren en tiempo real, automatizan procesos tediosos y hasta generan la estructura base de aplicaciones completas. Herramientas ya disponibles apuntan en esa dirección, por ejemplo, entornos como Replit o el IDE Cursor con integración de modelos generativos permiten este flujo conversacional con la máquina al construir código.

Un desarrollador *vibe* puede decir por voz lo que necesita, ver el código generado al instante, ejecutarlo, y si algo falla simplemente describir el error a la IA para que lo corrija.

Actualmente su acción es limitada, pero, hacia 2030 con LLM mucho más potentes y especializados, **podría volverse una modalidad estándar de desarrollo para prototipos e incluso para proyectos serios en ciertas circunstancias.**

Esto requiere nuevas habilidades para los desarrolladores, incluida una comunicación clara, un juicio agudo y un pensamiento general. El Informe de empleos de LinkedIn de 2025 señala un **aumento del 300% en la demanda de puestos como "AI Prompt Engineer"**, lo que muestra cómo la codificación de vibraciones está reescribiendo las descripciones de los puestos de trabajo.

VibeOps integra operaciones inteligentes en el flujo de desarrollo, eliminando fricciones y potenciando la creatividad sin interrupciones

VibeOps: operaciones invisibles, productividad visible

VibeOps nace como una extensión natural del concepto de *vibe coding*, introducido por Andrej Karpathy, donde **el programador guía, refina y supervisa código generado por IA sin abandonar su entorno creativo**. El salto que propone VibeOps es aún más disruptivo: eliminar las fricciones operativas (infraestructura, despliegue, monitoreo, mantenimiento) delegándolas a sistemas inteligentes que responden a instrucciones en lenguaje natural, directamente desde el editor de código.

Mientras que DevOps exige a los equipos conocimientos específicos en herramientas como YAML, Terraform o scripts de monitoreo, **VibeOps plantea una experiencia radicalmente distinta**.

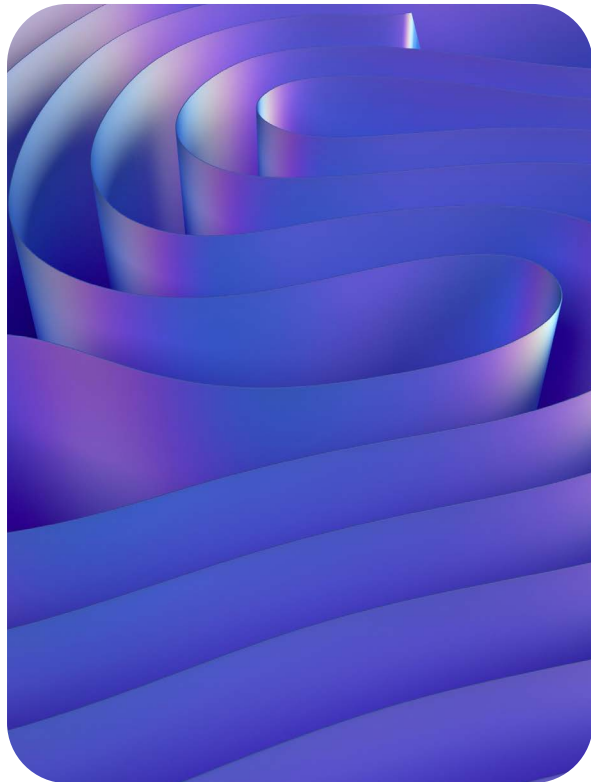
Este enfoque transforma la relación del desarrollador con el ciclo de vida operativo del software. Ya no se trata de colaborar con los equipos de operaciones, sino de que **las**

operaciones se integren dentro del flujo de desarrollo, sin generar interrupciones ni necesidad de cambio de contexto.



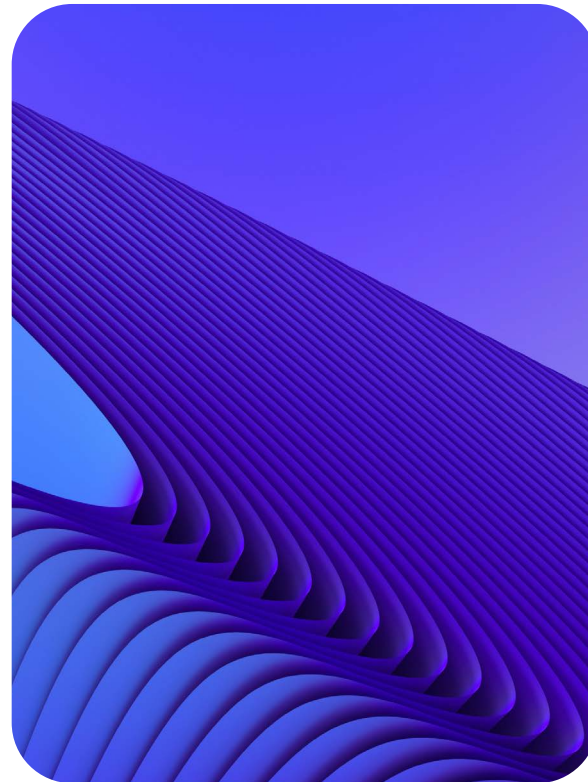
DevOps tradicional	VibeOps impulsado por IA
Infraestructura como código	Infraestructura conversacional guiada por IA
Configuración manual	Autoconfiguración basada en contexto
Pipelines CI/CD complejos	Despliegue instantáneo desde el editor
Scripts de monitoreo	Alertas, logs, rollback automáticos por defecto
Soporte especializado	Respuesta asistida por LLM en lenguaje natural

Esta filosofía se fundamenta en cuatro pilares:



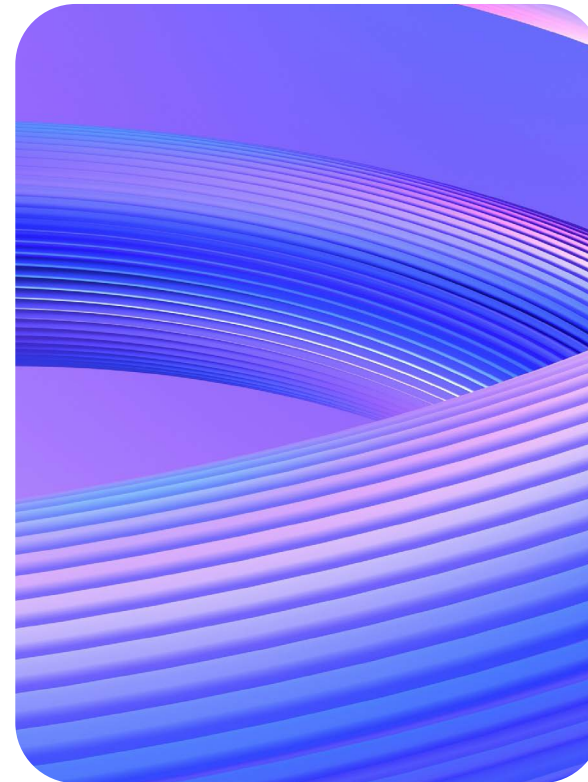
Definición conversacional de infraestructura

Los desarrolladores describen lo que necesitan en lenguaje natural; la IA diseña, configura e implementa directamente desde el entorno de desarrollo.



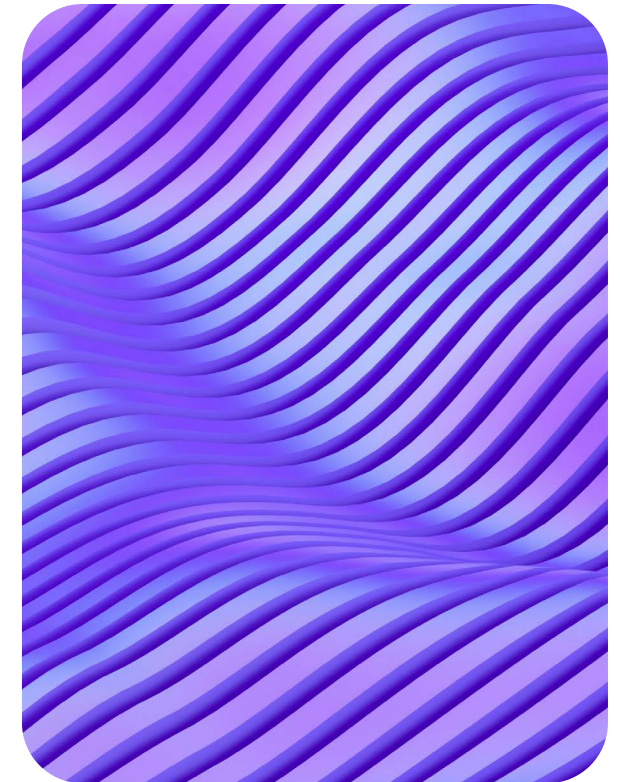
Respuesta asistida ante incidentes

En caso de errores, la IA analiza *logs*, diagnostica problemas y propone soluciones sin que el desarrollador abandone su tarea principal.



Optimización del entorno de desarrollo

La IA aprende del comportamiento del usuario para anticiparse a cuellos de botella, ajustar recursos y personalizar la experiencia sin intervención directa.



Continuidad entre plataformas

Permite mover código entre entornos locales, *staging* o producción sin cambiar de mentalidad ni configuración, unificando la experiencia de desarrollo.

VibeOps redefine el éxito en ingeniería enfocándose en métricas que mejoran la experiencia, eficiencia y creatividad del desarrollador

Implementar VibeOps implica repensar qué se mide, cómo se mide y para qué se hace. Ya no se trata de capturar únicamente datos de rendimiento del sistema, sino de crear visibilidad sobre la fricción, los bloqueos y los aceleradores que afectan al flujo creativo del desarrollador.

A diferencia de los enfoques tradicionales centrados en métricas de rendimiento del sistema (*uptime*, errores, *throughput*), **VibeOps propone un nuevo conjunto de indicadores donde el foco es la experiencia del desarrollador:**



Reducción del tiempo de cambio de contexto (*context switching*)



Frecuencia de despliegue



Tiempo medio para crear nuevos entornos (*time to environment*)



Duración del estado de flujo creativo



Tiempo de ciclo de idea a producción

Estas métricas son la brújula que guía la implementación de VibeOps, alineando las operaciones con el objetivo último: **acelerar la entrega de valor mediante una experiencia de desarrollo sin fricción.**

La tendencia no reemplaza los principios fundamentales de DevOps (colaboración, automatización, entrega continua), pero los expande con una capa crítica, es decir, el diseño intencional de la experiencia del desarrollador como fuente primaria de eficiencia y creatividad.



Impacto de la IA en la consultoría de software: el futuro de las empresas de ingeniería

Generative AI revoluciona el desarrollo de software, redefiniendo el papel del desarrollador como coordinador y evaluador de código

La transformación de las empresas de consultoría en la era del código inteligente

El impacto que experimentan las consultoras impulsadas por la IA Gen y la automatización abarca cambios tanto estratégicos como operativos. La entrega de proyectos puntuales está siendo sustituida por los **modelos de socios estratégicos, activos y continuos, capaces de anticiparse y adaptarse al ritmo de la innovación**. A nivel de organización interna, se está cambiando drásticamente la forma en que se concibe, desarrolla, prueba y mantiene el software, caracterizada por una **automatización avanzada**, un incremento notable en la productividad y la consolidación de nuevos marcos colaborativos.

Este hecho ha generado la **aparición de nuevas trayectorias profesionales**. El rol del desarrollador está transitando de ser un escritor de código al de un revisor y validador de resultados generados por IA. A medida que los LLM ganan precisión y fiabilidad, este perfil evoluciona hacia un orquestador del desarrollo, encargado de supervisar, ajustar y garantizar la calidad de los *outputs* automáticos.

De la mano de ello, **se abren debates clave sobre el futuro del trabajo, la ética en la integración de IA y el papel de la consultoría** en un ecosistema tecnológico dominado por plataformas inteligentes y soluciones autónomas.

Además, la naturaleza poco predecible de esta tecnología conlleva el **surgimiento de nuevas funciones especializadas**. Ya no basta con dominar lenguajes de programación tradicionales, ahora se espera que estos profesionales sepan seleccionar, gestionar e integrar múltiples modelos de IA dentro de sistemas existentes.

En paralelo, el aprendizaje continuo y la actualización constante de **habilidades se vuelven imperativos**. Las organizaciones deben impulsar programas de formación adaptados, que incluyan tanto capacidades técnicas como habilidades blandas, como la comunicación efectiva y la gestión de proyectos.

Los clientes de consultoría quieren servicios impulsados por IA

89%



espera que los servicios de consultoría incorporen IA para la mejora de productividad y calidad.

86%



afirma estar buscando activamente servicios que incorporen IA y activos tecnológicos.



Finalmente, más allá de potenciar la productividad individual, existe la oportunidad de transformar el desarrollo colaborativo de software. Al integrarse en marcos como Agile o DevOps, optimiza flujos, anticipa fallos y refuerza la coordinación. La automatización de pruebas, despliegues y planificación alivia la carga operativa, pero **exigirá reforzar la trazabilidad, validación y control ético para garantizar la confianza**.

Generative AI impulsa a consultoras de software a reinventar procesos, adoptar nuevas habilidades y ofrecer servicios basados en resultados

La organización interna no es lo único que se está transformando por la inteligencia artificial. Los **modelos de negocios actuales deben sufrir cambios estructurales y migrar** hacia esquemas de mayor valor añadido, donde el foco se ubique en el impacto estratégico y operativo que sus servicios generen al cliente.

A día de hoy, se están integrando de forma activa herramientas avanzadas como generadores automáticos de código (GitHub Copilot, Codeium, entre otros) en los flujos de trabajo, lo que permite acelerar el *delivery* sin comprometer la calidad. Pero el cambio va más allá, se está buscando posicionarse como proveedores de capacidades, ofreciendo servicios

especializados como la personalización de modelos de IA según el contexto de negocio del cliente, auditorías de calidad sobre código generado automáticamente o **consultoría para la integración de inteligencia artificial en productos digitales existentes**.

Esta transformación viene acompañada de un **mayor énfasis en la creación de activos reutilizables de propiedad intelectual (IP)**, como *frameworks* especializados, bibliotecas de conocimiento, modelos entrenados para industrias específicas o soluciones preconfiguradas listas para adaptarse, permitiendo elevar el valor entregado y reducir la dependencia de desarrollos desde cero.

En adición, emergen nuevos esquemas de monetización más alineados con los resultados del cliente. Se anticipa que, gracias a la capacidad de la IA para conectar datos fragmentados y hacer visible el valor generado de extremo a extremo, los modelos de *pricing* tenderán a basarse en resultados (*outcome-based pricing*), donde el cliente **pagará por los KPI alcanzados o beneficios obtenidos**. Este movimiento se enmarca también en una transición desde el modelo tradicional de Software como Servicio (SaaS) hacia un enfoque de Inteligencia Artificial como Servicio (**AlaaS**), **donde agentes inteligentes proporcionan soporte continuo, aprenden con el uso y evolucionan sin necesidad de actualizaciones manuales**.

En conjunto, estos **cambios exigen reimaginar no solo cómo se desarrolla el software, sino cómo se captura y entrega el valor**. De esta manera, al transformarse en socios estratégicos, será necesario implicarse en cuatro frentes claves: el soporte a la toma de decisiones con IA, la automatización de tareas internas para escalas con mayor eficiencia, convertirse en líderes de la adopción responsable de tecnologías mediante la ética y *compliance* y conformar equipos multidisciplinares que integren capacidades de negocio, datos y tecnologías.

Impacto de la nueva era del código con IA e ingeniería aumentada en diversas áreas de la organización

Organización, equipo
y nuevos roles

Nuevas prácticas
y metodología

Herramientas
y arquitectura

Oferta de
soluciones

Productividad
y rendimiento

La colaboración inteligente impulsa nuevos roles estratégicos y técnicos, elevando productividad, calidad e innovación

Colaboración humano y máquina que redefine el rol del desarrollador

La aplicación e implementación de la inteligencia artificial está reformulando de forma integral la experiencia de los desarrolladores de software. Desde cómo se escribe, hasta cómo se concibe, construye, valida y entrega. Esta transformación implica un replanteamiento del flujo de trabajo, los roles en equipos y la colaboración entre las distintas áreas. A raíz de ello, el **desarrollador se convierte en un orquestador de capacidades**, combinando los recursos de la IA con el conocimiento humano para construir soluciones más rápidamente y con mayor coherencia técnica.

La IA **también está asumiendo un papel activo en la definición de historias de usuario, criterios de aceptación y requisitos funcionales**, permitiendo a los equipos de desarrollo transmitir esta información con mayor claridad a los analistas de negocio y *stakeholders*.

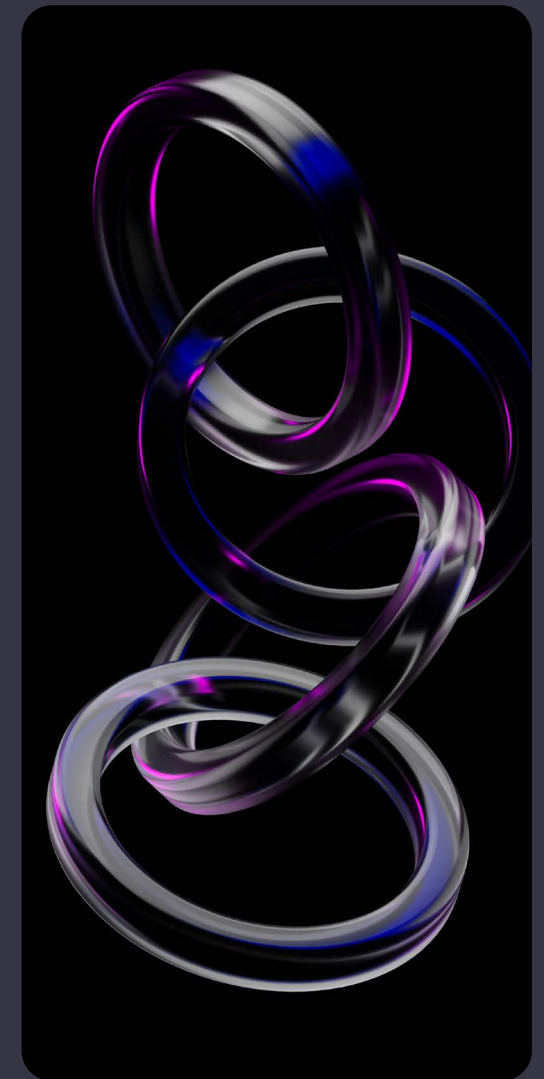
En términos de diseño de interfaces, la **IA puede establecer estructuras base de páginas, flujos de navegación y componentes funcionales**. A partir de estas estructuras, los diseñadores intervienen para definir las capas interactivas y de experiencia de usuario más complejas, asegurando una navegación intuitiva y centrada en el usuario final.

Además, está abriendo la puerta a una entrega continua real. Equipos ágiles pueden utilizarla para generar grandes volúmenes de código, redactar *pull requests* (PR) y mantener un flujo constante de estas. **Esto acelera el *time-to-market* y convierte el ciclo de desarrollo en un proceso mucho más fluido, reduciendo tiempos de espera y dependencia entre tareas.**

La aparición de la colaboración como punto de inflexión

Este impacto generado por la IA, su democratización y profesionalización se extiende más allá del desarrollador puro. **El nuevo modelo colaborativo abre también la puerta a trayectorias profesionales emergentes**, roles como el *Prompt Engineer*, el *AI Trainer*, el *Data Steward* o el *Platform Engineer*, entre muchas otras. Muchos de estos perfiles están siendo formalizados por consultoras IT y otras empresas especializadas que lideran el despliegue de soluciones avanzadas y necesitan asegurar la correcta integración entre capacidades humanas e inteligencia artificial.

Es por ello por lo que **se valora cada vez más la adaptabilidad, la capacidad de aprendizaje continuo y las competencias interdisciplinarias**. Las empresas que apuestan por equipos híbridos tenderán a innovar más rápido, reducir errores y atraer talento de nueva generación interesado en trabajar con tecnologías punteras.



La nueva era del código redefine equipos y consultoría, pasando de la ejecución a la estrategia, automatización e innovación



Prompt Engineer

Especialista en diseñar instrucciones para modelos generativos, ajustando sus respuestas a contextos específicos. Acelera la creación de prototipos y garantiza resultados útiles y controlados.



AI Software Architect

Orquesta el ecosistema necesario para llevar la IA a producción, alineando ciencia de datos, negocio y regulación. Es clave para escalar proyectos con garantías estructurales.



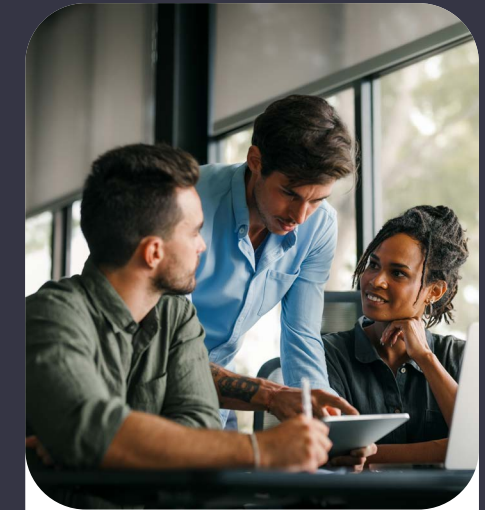
MLOps Engineer

Automatiza y gestiona el ciclo de vida de la IA, asegurando su rendimiento, trazabilidad y cumplimiento. Es el puente entre ciencia de datos, ingeniería y operaciones.



AI Ethicist y Risk Analyst

Evalúan impactos sociales, sesgos y riesgos regulatorios de la IA, definiendo marcos éticos y de gobernanza para un desarrollo responsable y conforme a normativas.



Data Steward y AI Trainer

El primero asegura la calidad y gobernanza de los datos; el segundo entrena y ajusta modelos para garantizar que sus respuestas sean relevantes, éticas y adaptadas al contexto real.

Del refuerzo de plantilla al refuerzo de capacidades y nuevas presiones

El modelo de *staff augmentation*, basado en incrementar el número de desarrolladores o analistas externos para cubrir picos de demanda o compensar carencias de *skills*, está siendo sustituido por un enfoque centrado en capability augmentation. En lugar de sumar personas, **las organizaciones buscan multiplicar su capacidad efectiva mediante el uso estratégico de, por ejemplo, herramientas de IA, automatización de procesos, copilotos inteligentes, frameworks especializados y plataformas de desarrollo asistido.** De esta manera, los equipos de desarrollo se vuelven más pequeños, pero más especializados, capaces de generar mayor valor.

Este cambio hacia la *augmentación* de capacidades tiene efectos directos sobre los perfiles de talento que demandan las organizaciones. Tareas que tradicionalmente recaían en roles junior como

escribir funciones básicas, hacer pruebas unitarias, investigar documentación o analizar datos, están siendo asumidas por sistemas inteligentes, generando de esta manera una presión estructural sobre los puestos de entrada. A partir de ahora, **se espera que incluso los perfiles junior comprendan cómo colaborar con IA y cómo generar valor desde el primer momento.**

Simultáneamente, **los nuevos perfiles híbridos cobran mayor protagonismo,** los profesionales capaces de interpretar salidas generadas por IA, afinar *prompts* para obtener mejores resultados, tomar decisiones basadas en datos y aportar una visión crítica sobre lo que la IA propone darán lugar a equipos más especializados, donde la capa operativa se automatizará y la capa consultiva y estratégica se fortalecerá.



Con LCNC, nacen desarrolladores ciudadanos, abriendo el desarrollo de software a perfiles no técnicos

La nueva figura del *citizen developer*

El auge de estas plataformas LCNC ha impulsado la aparición del desarrollador ciudadano. Estos son **profesionales sin perfiles técnicos que, gracias a estas herramientas, crean aplicaciones funcionales para resolver las necesidades de negocio con mayor autonomía y rapidez.**

Las empresas están invirtiendo en estos nuevos profesionales en busca de unos beneficios que ya son tangibles. Los *citizen developers* han reducido hasta en un 70 % los tiempos de entrega de aplicaciones y en un 50 % los costes de desarrollo. Además, se **calcula que el 30 % de las aplicaciones de automatización con IA generativa son creadas por *citizen developers* en 2025.**

Por lo que es esperable que, junto a los desarrolladores profesionales, conformen un ecosistema ágil donde se una el conocimiento de negocio con la robustez técnica. Todo ello, englobado bajo una buena gobernanza que guíe a los desarrolladores ciudadanos y aporte un soporte a la innovación mientras minimiza los posibles riesgos de su implementación.

Se distinguen por un conjunto de características que los habilitan para crear aplicaciones sin ser expertos en programación:

- **No requieren conocimientos técnicos avanzados:** se apoyan en plataformas de bajo o nulo código con interfaces gráficas intuitivas para construir aplicaciones, en lugar de escribir código complejo.
- **Orientación a la resolución de problemas del negocio:** son profesionales que provienen de otras áreas funcionales y detectan necesidades concretas en sus procesos diarios.
- **Conocimiento profundo del negocio:** a diferencia de un desarrollador tradicional, el *citizen developer* entiende de primera mano los procesos y objetivos de su departamento, permitiéndole alinear perfectamente la solución tecnológica con las necesidades reales de la operación.
- **Colaboración multidisciplinaria:** el *citizen developer* suele trabajar de forma colaborativa, sirviendo de puente entre el área de negocio y el área técnica.

Modelo de *workflow* de *citizen developer*



Surge el enfoque de menos líneas y más **impacto** impulsando desarrollo rápido y colaborativo con resultados tangibles

Relevancia estratégica del *citizen development*

Aceleración de la Transformación Digital

Al habilitar que más personas contribuyan en la creación de soluciones, **el desarrollo ciudadano democratiza la innovación dentro de la empresa.**

Reduce costes y tiempos

Las plataformas LCNC minimizan la necesidad de desarrollos complejos, lo cual **abarata costes de TI y acelera los tiempos** de entrega de aplicaciones.

Mejora la Productividad

Estos empleados tienden a estar más **comprometidos y motivados**, ya que participan directamente en la construcción de herramientas que facilitan su trabajo.

Alivia la carga del departamento de IT

Al delegar en las áreas de negocio la creación de aplicaciones sencillas o la automatización de tareas puntuales.

Una nueva visión con **menos líneas y más impacto**

De este movimiento, **se acrecienta la filosofía de menos líneas, más impacto.** Buscando lograr mayor resultado de negocio escribiendo la menor cantidad de código posible, algo que las plataformas LCNC ya hacen realidad, con las herramientas modernas un usuario puede crear mediante componentes predefinidos y flujos visuales, reduciendo drásticamente el volumen de código escrito a mano.

En términos prácticos, significa que una persona puede construir una aplicación útil con unos pocos clics y configuraciones, en un tiempo significativamente reducido que con un lenguaje tradicional. Esta eficiencia se puede traducir en soluciones desplegadas en días o semanas, en lugar de meses, y adaptadas exactamente a la necesidad identificada. Es más, este paradigma no solo acelera el desarrollo, sino que **también puede reducir errores, ya que menos código escrito en muchas ocasiones equivale a menos posibilidades de bugs, y facilita el mantenimiento.** Las actualizaciones se hacen ajustando parámetros o reemplazando componentes visuales, en vez de tener que refactorizar amplias secciones de código fuente.

Gracias a las plataformas *low-code/no-code*, las organizaciones pueden conseguir **grandes resultados en digitalización y automatización**

con un esfuerzo de programación muy reducido, lo que empodera a todos los empleados para contribuir a la innovación.



Nuevas prácticas con IA y automatización optimizan desarrollo y entrega, impulsando eficiencia, calidad y rápida respuesta a cambios

IA y automatización redefinen prácticas de desarrollo y entrega de software

Más allá de su papel como herramienta puntual, la inteligencia artificial **se está consolidando como una capa operativa que atraviesa cada eslabón del desarrollo de software**. Desde la escritura inicial del código hasta su puesta en producción y evolución continua, actúa como un motor que optimiza flujos, anticipa fallos y adapta las soluciones al contexto cambiante.

Su valor no reside únicamente en acelerar tareas individuales, sino en habilitar un ecosistema más cohesivo e inteligente, donde el *testing* se ajusta automáticamente al comportamiento del usuario, la gestión de proyectos se vuelve predictiva, y la experiencia del cliente se personaliza en tiempo real.

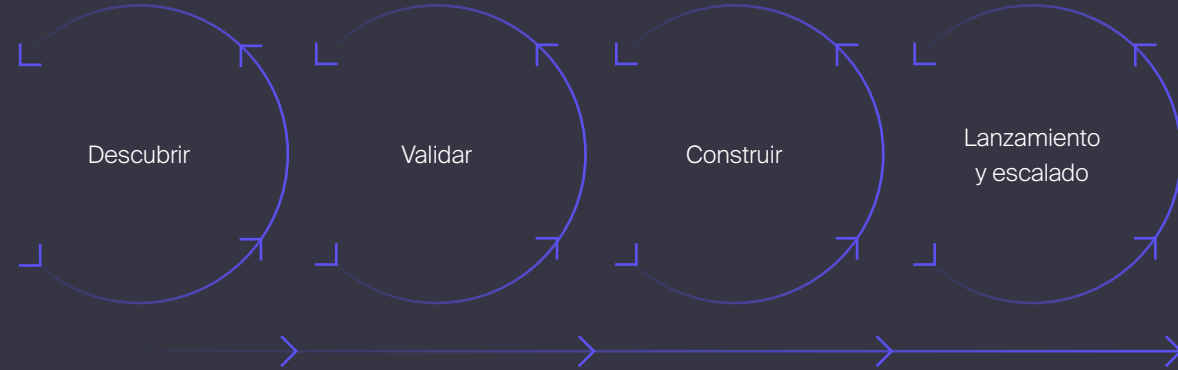
En este sentido, **se convierte en una infraestructura que hace posible una nueva forma de construir, desplegar y mantener software** en entornos cada vez más complejos y dinámicos.

De hecho, su integración está acelerando de forma decisiva la entrega de software. Al poder automatizar una gran cantidad de tareas, su uso reduce los tiempos de entrega, mejora la calidad del software y permite detectar errores antes de que se conviertan en problemas críticos, **convirtiendo la totalidad del proceso en un desarrollo ágil proactivo, revisable y eficiente**, clave para sostener ciclos de desarrollo continuos y en constante cambio.



Reimaginando el ciclo de vida del desarrollo de software de fragmentado a acelerado con IA

Ciclo de vida actual del desarrollo de productos de software



Datos dispersos y aislados entre investigación, uso y éxito



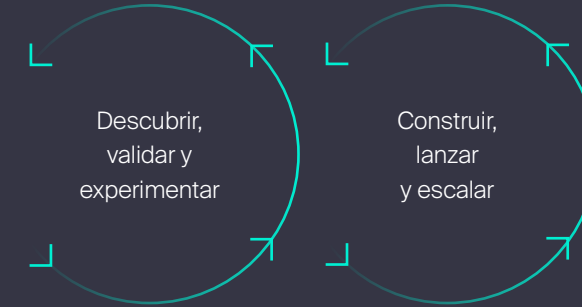
Propiedad fragmentada entre producto, marketing, ingeniería, éxito



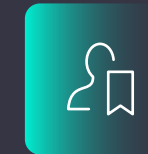
Tiempos largos, retrasos entre fases

Ciclo de vida del desarrollo de productos habilitados por IA

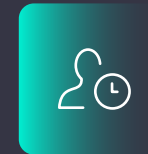
Datos integrados de clientes que permiten obtener *insights* en todos los horizontes



Datos integrados en tiempo real que generan *insights*



Propiedad alineada, gerentes de producto como "mini CEO"



Ciclo comprimido radicalmente, llegada más rápida al mercado

Integrar seguridad desde el inicio con *Shift Left Testing* reduce riesgos, costos y fortalece la confianza en el software

Shift left testing en seguridad: integrar controles desde el inicio

Como resultado de la evolución de la ingeniería de software impulsada por IA y automatización, los procesos de *testing* y seguridad han experimentado una transformación significativa. En particular, las organizaciones están adoptando el enfoque "*shift-left*" en sus prácticas de DevSecOps, **trasladando las pruebas y validaciones de seguridad a las etapas más tempranas del ciclo de vida del desarrollo**. En lugar de dejar estas tareas para el final, ahora se integran desde el diseño y la codificación inicial.

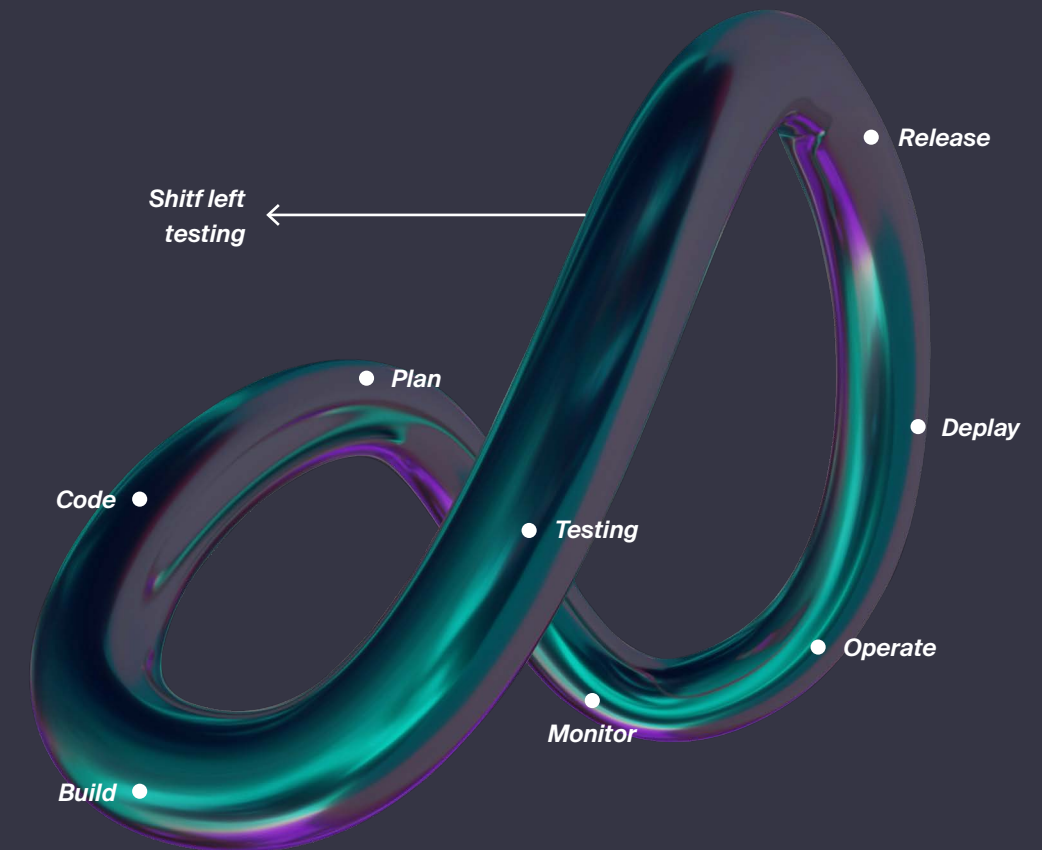
Este cambio estratégico se traduce en productos más seguros, ya que **permite detectar y corregir vulnerabilidades desde el inicio**, reduciendo significativamente los riesgos y fortaleciendo la integridad del software. Además, implica un ahorro de costes considerable, ya que abordar los problemas de seguridad en fases avanzadas suele requerir reescrituras costosas y expone a las organizaciones a sanciones derivadas de brechas de seguridad. Al mismo tiempo, eleva el nivel técnico del equipo al involucrar a los desarrolladores en etapas tempranas.

Otro impacto es el impulso a la **colaboración transversal**, "*shift left*" fomenta la **cooperación directa entre desarrolladores y especialistas en seguridad**, facilitando una comunicación más eficiente, un intercambio de conocimiento más fluido y una comprensión compartida del proceso.

Grandes compañías, como Google o Microsoft, ya emplean herramientas impulsadas por IA para automatizar el *testing* de sus productos, generando automáticamente casos de prueba, priorizando ejecuciones y anticipando defectos. También se han desarrollado **herramientas que utilizan algoritmos de *Machine Learning* para realizar pruebas de *fuzzing***, identificando vulnerabilidades con mayor precisión y eficiencia que los métodos tradicionales.

Finalmente, algunas plataformas de *streaming* ya utilizan **sistemas de IA para monitorizar en tiempo real su infraestructura de *streaming***, detectando anomalías, anticipando interrupciones y garantizando así una experiencia de usuario estable y confiable.

Shift left testing para una alta calidad con velocidad de entrega



Herramientas como Copilot y GPT lideran la revolución del código asistido, transformando productividad y calidad del software

Herramientas y tecnologías habilitadoras en ingeniería de software

El mercado de IA generativa en desarrollo de software, valorado en 21,4 millones de \$ en 2023, se espera que alcance los 117,6 millones en 2033, con una tasa de crecimiento anual del 19,1 %. La irrupción de herramientas de generación de código asistida por IA como GitHub Copilot, GPT, Tabnine, CodeWhisperer o Replit han sido una de las innovaciones más disruptivas en este sector, **las cuales permiten traducir instrucciones en lenguaje natural a código funcional.**

La mayoría de las soluciones, combinando procesamiento de lenguaje natural con aprendizaje profundo, no solo generan código bajo demanda, también **escanean activamente el entorno de desarrollo, detectan errores y proponen mejoras, aprendiendo sobre el contexto del equipo** a medida que se utilizan. De esta manera, se reducen los ciclos de desarrollo, se mejora la calidad del código, se aceleran las entregas y se amplía la capacidad creativa del equipo.

Herramientas

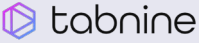
 ChatGPT

 GitHub Copilot


Amazon CodeWhisperer

 bolt

 replit

 tabnine

Fuente: AI News Dive



Mayor velocidad de desarrollo:
permitiendo a los equipos centrarse en funciones nuevas.



Automatización de tareas monótonas:
liberando tiempo para resolver retos complejos.



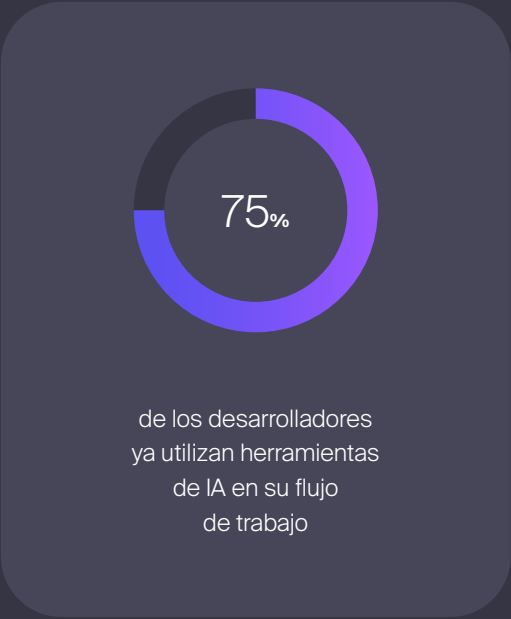
Código más limpio y legible:
con menos errores y mayor mantenibilidad.



Aprendizaje acelerado para desarrolladores junior, gracias a explicaciones y sugerencias en tiempo real.

Estas herramientas han demostrado **acelerar la finalización de tareas hasta en un 56 %**. Los desarrolladores que las utilizan reportan mayores niveles de satisfacción: **entre el 60 % y el 75 % afirman sentirse menos frustrados, más enfocados y más satisfechos con su labor diaria.** El impacto es especialmente notable en organizaciones maduras y entre desarrolladores senior, quienes saben cómo aprovechar mejor la IA como extensión de sus capacidades.

El beneficio no se limita al rendimiento inmediato, **un 73% de los desarrolladores que usan GitHub Copilot afirman que les ayuda a mantenerse en estado de flujo,** mientras que **el 87 % señala que les permite conservar esfuerzo mental** durante tareas repetitivas.



Cada fase del ciclo de software se beneficia de asistencia aumentada, reduciendo errores y acelerando la entrega con precisión

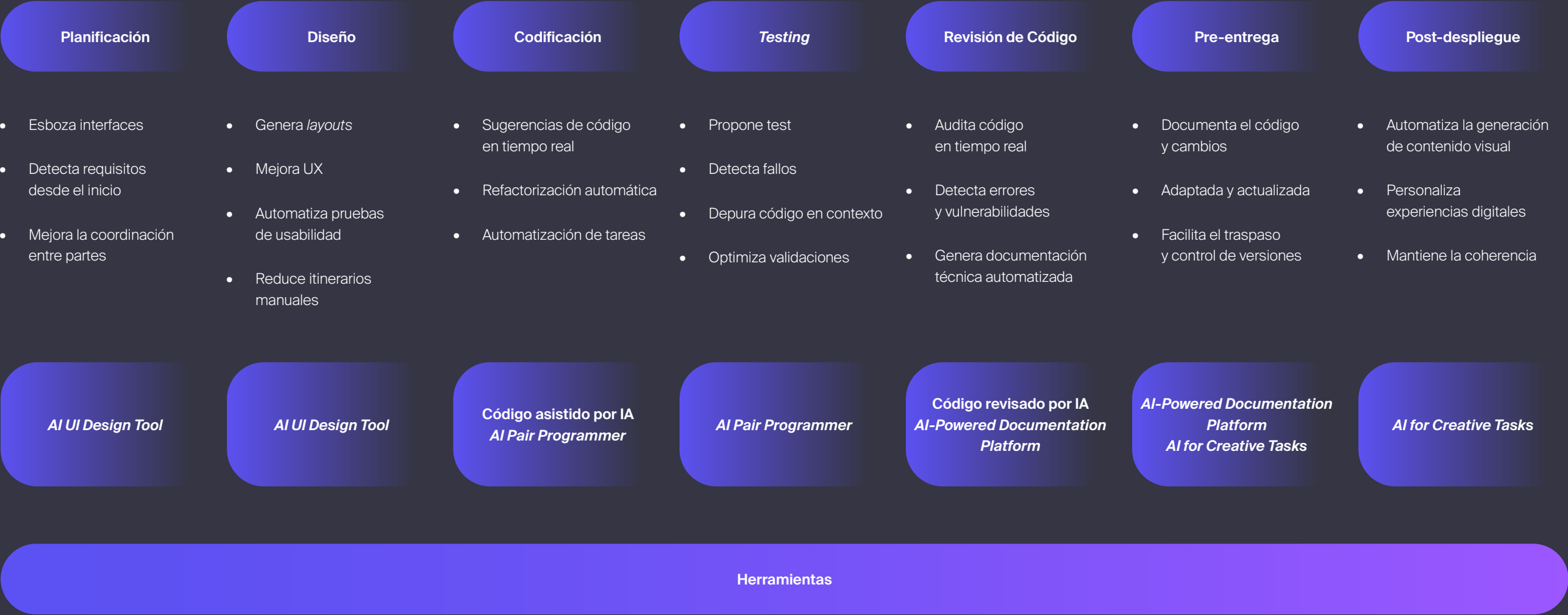
Herramientas aplicadas en todo el ciclo de vida del desarrollo de software

La transformación de la ingeniería de software en la era de la inteligencia artificial es el resultado de un ecosistema de tecnologías habilitadoras, herramientas inteligentes y arquitecturas flexibles que actúan como catalizadores de una nueva forma de concebir,

construir y mantener soluciones digitales. Este entorno tecnológico está facilitando el auge del desarrollo asistido por IA y la expansión de plataformas LCNC, **haciendo posible una ingeniería de software más rápida, inclusiva y centrada en la personalización.**



Uso de herramientas en cada etapa del ciclo de vida del desarrollo de software



Una arquitectura modular e inteligente convierte cada herramienta en un aliado inteligente y cada flujo en una oportunidad de optimización

Arquitectura de soporte en la ingeniería de software aumentada: una infraestructura inteligente y modular

La arquitectura de soporte detrás de las tecnologías habilitadoras y herramientas de IA en el ciclo de vida del desarrollo de software y la ingeniería aumentada es un **entramado complejo y modular que integra plataformas, servicios, protocolos y capas de automatización.**

Esta arquitectura permite que las soluciones de IA sean escalables, seguras, colaborativas y adaptables a diferentes etapas y necesidades del desarrollo. Es más, su diseño busca facilitar ese trabajo conjunto entre humanos y máquinas, desde la ideación hasta el mantenimiento continuo.



Características de la Arquitectura de Soporte



Modularidad y reutilización:

esta arquitectura se apoya en componentes desacoplados que pueden adaptarse a las necesidades de cada equipo, producto o cliente. Esto permite mantener la agilidad incluso en entornos complejos y facilita la evolución constante.



Escalabilidad y flexibilidad:

diseñada para crecer con el proyecto, se adapta a equipos grandes o reducidos, proyectos sencillos o multinivel, entornos *Cloud*, *on-prem* o híbridos.



Inteligencia embebida:

la IA no opera como una capa externa, actúa como una parte integral del flujo con copilotos que asisten en el IDE, motores de recomendación, sistemas que ajustan automáticamente pipelines y modelos que aprenden del comportamiento de los equipos.



Integración fluida y sin fricciones:

diversas plataformas permiten conectar, automatizar y optimizar sin interrupciones. La interoperabilidad entre sistemas es una prioridad técnica que define la experiencia del desarrollador y la solidez operativa.

Componentes típicos del Stack AIASE

Módulos de IA/ML	generación de código, refactorización, detección de errores, predicción de fallos.
Orquestadores de flujos	conectan desarrollo, <i>testing</i> , despliegue y monitoreo.
Smart IDE	asistentes de código, <i>prompts</i> contextuales, métricas de productividad.
Plataformas de observabilidad	monitorización continua de DevEx y rendimiento del software.
Controladores de IaC	infraestructura definida por código con ayuda de lenguaje natural.

Su impacto estratégico y operativo

Una arquitectura de soporte bien diseñada no es visible para el usuario final, pero determina radicalmente el éxito de los equipos de desarrollo. **Su impacto se manifiesta en los ciclos de desarrollo más cortos y entregas más ágiles, la reducción de errores manuales y mayor confiabilidad, el aumento en la satisfacción del desarrollador (DevEx) y en la creciente innovación del sector.**

La oferta IT se transforma con IA, posicionando a consultoras como agentes clave de innovación, eficiencia y acompañamiento continuo

Evolución de las soluciones de consultoría IT en la nueva era de la ingeniería de software aumentada

A medida que estas tecnologías se vuelven más accesibles y más complejas, se hace evidente que un verdadero reto, **elegir bien el stack correcto, gobernar los datos, adaptar la arquitectura, formar al talento interno y alinear todo ello con los objetivos del negocio.** En este punto de unión entre la visión y la ejecución, entre arquitectura y cultura, es donde el papel de las consultoras se vuelve determinante.

Desde el propio diseño e implementación de arquitecturas modernas (*Cloud*, híbrido, microservicios, DevSecOps, entre otras) hasta la integración de soluciones de IA, automatización o plataformas LCNC. La intervención de las consultoras de IT **facilita la maximización del ROI** de estas tecnologías, acelerando los ciclos de entrega y mitigando los riesgos operativos y regulatorios.

Particularmente en entornos donde se implementan herramientas de AIASE, **el acompañamiento experto permite resolver desafíos estructurales** que van desde la gestión del ciclo de vida de los modelos (versionado, reentrenamiento, actualización), hasta la selección de tecnologías de código abierto o cerrado en función del caso de uso y del marco regulatorio. Además de todo ello, las estas empresas juegan un rol clave en garantizar la trazabilidad, la interoperabilidad entre sistemas legacy y modernos y en establecer entornos de pruebas seguros antes de escalar soluciones a producción.

A partir de ahora, **su propuesta de valor ya no gira en torno al desarrollo a medida o al outsourcing puntual, sino que se estructura como una oferta integral**, modular y continuada capaz de responder a los múltiples desafíos técnicos, culturales y estratégicos de las organizaciones que compiten en la economía digital.

Núcleos de intervención estratégica en procesos de transformación digital



Cuando la IA libera el talento,
la efectividad deja de ser una meta
y se vuelve el estándar

Productividad potenciada por IA que impulsa resultados impactantes

Tareas como escribir nuevo código, refactorizar o documentar pueden completarse hasta dos veces más rápido.

Concretamente, la documentación de código y la creación de nuevo código se acortan en torno a un 50%, la refactorización de código se ejecuta en dos tercios del tiempo habitual.

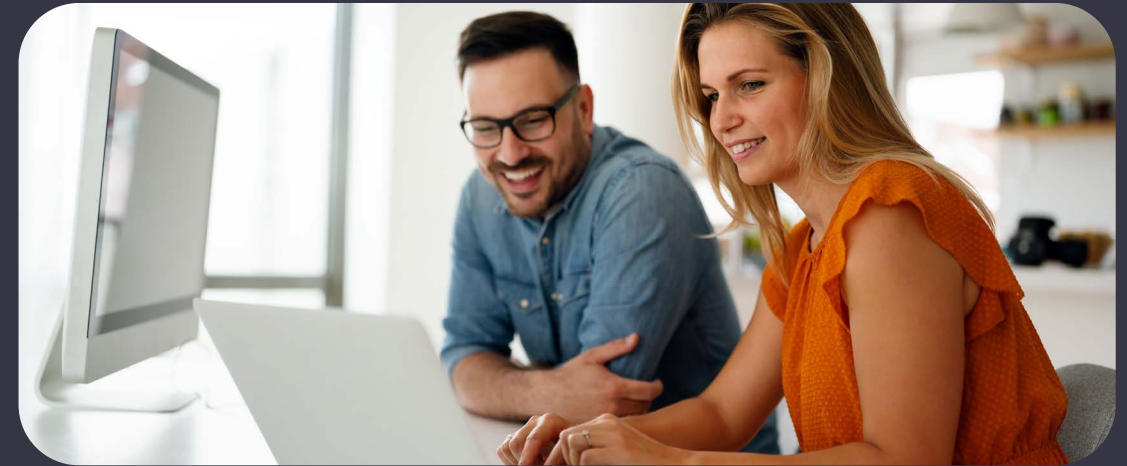
Cabe destacar que este impacto disminuye en tareas complejas o cuando se trata de perfiles junior. En estos casos, el ahorro de tiempo puede ser menor e incluso se observó que algunos tardaron un 7% a 10% más con herramientas de IA, debido a su falta de dominio en entornos técnicos específicos.

Por lo contrario, los desarrolladores experimentados que utilizan herramientas apoyadas en IA en tareas complejas son hasta un 30% más propensos a completar

su trabajo dentro del plazo, gracias al soporte que ofrece para explorar nuevos lenguajes, frameworks o estructuras con mayor autonomía.

30-45%

el impacto directo de la IA en la productividad de la ingeniería de software, expresado como porcentaje del gasto anual actual en esta función.



La efectividad en la entrega de resultados

La efectividad en la entrega de soluciones de software **depende de múltiples factores críticos: precisión del código, calidad de producto y funcional, cumplimiento de plazos y capacidad de adaptación continua.**

Anteriormente, los equipos dedicaban un volumen considerable de esfuerzo a tareas repetitivas como la escritura de código base, la detección manual de errores, la gestión operativa y la coordinación de despliegues. Esto no solo ralentizaba los ciclos de desarrollo, sino que también exponía a los proyectos a fallos humanos, brechas de calidad y retrasos en la entrega.

Hoy, la incorporación de IA permite que los equipos sean mucho más efectivos al automatizar estas tareas, reconfigurando por completo este panorama. Gracias a la automatización, el refuerzo de los estándares de calidad y la optimización de procesos de extremo a extremo, los desarrolladores e ingenieros pueden operar con mayor agilidad y fiabilidad. **Esto libera talento para enfocarse en tareas de mayor valor añadido** (como la innovación, el diseño y la toma de decisiones), logrando un impacto más alto con menor esfuerzo operativo.

La clave para entregas efectivas es combinar IA con pruebas rigurosas, seguridad y adaptación del equipo

Entre los avances más destacados en términos de efectividad se encuentran:



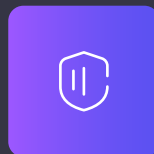
Detección automática de errores en tiempo real



Generación de pruebas automáticas que mitigan fallos antes de llegar a producción



Optimización del rendimiento del código para maximizar su eficiencia



Refuerzo de la seguridad mediante la identificación proactiva de vulnerabilidades

Aun a pesar del progreso, **la integración plena de la IA en el desarrollo de software enfrenta desafíos relevantes** ya que, por ejemplo, persisten obstáculos como la acumulación de deuda técnica por código generado sin mantenimiento, errores contextuales de la IA, o su limitada capacidad para comprender las particularidades de cada entorno organizativo. A esto, se suman preocupaciones sobre privacidad, seguridad de los datos y cumplimiento normativo, que siguen siendo barreras importantes.

Por último, la compatibilidad de las herramientas de IA con los ecosistemas tecnológicos existentes no siempre es inmediata, lo que complica su adopción fluida. Por otro lado, **la confianza en la calidad del código generado por IA ha aumentado sustancialmente**, pero aún persisten riesgos relacionados con la fiabilidad que pueden mitigarse con protocolos de prueba y mecanismos de validación para asegurar que el código generado cumpla con los estándares técnicos y de negocio. Solo así será posible escalar sus beneficios sin comprometer la integridad del producto final.

En conclusión, **las organizaciones que apuestan por adoptar la ingeniería aumentada enfrentan desafíos** que van mucho más allá del despliegue tecnológico. Requerirán de acompañamiento experto para seleccionar el *stack* adecuado, adaptar la arquitectura a contextos regulados o complejos, gobernar el ciclo de vida de los modelos de IA y garantizar su integración sin fricciones con sistemas legacy.

Es por ello que las consultoras están migrando hacia un enfoque de capacidades, soporte continuo, estratégico y orientado a resultados.

La productividad se ha visto significativamente potenciada gracias a copilotos de código, generación automatizada de documentación, herramientas de refactorización asistida o *testing* inteligente, sin embargo, **este impacto positivo dependerá de un uso experto y bien contextualizado**, pues en entornos no maduros o en manos inexpertas, la IA puede ralentizar el proceso o generar deuda técnica. La efectividad en la entrega, por su parte, también se ha visto redefinida. Debido a la liberación de carga de trabajo de los equipos, se puede apreciar una mejora en la calidad de los productos y un incremento en la motivación y la retención del talento.

Ahora bien, **la confianza en los beneficios de la AIASE debe ir acompañada de estructuras de validación, protocolos de calidad y visión estratégica**. Solo las organizaciones que asuman esta transición como un proceso de madurez organizativa podrán escalar sus capacidades sin comprometer la integridad técnica ni el valor del producto final.

La adopción masiva de IA implica riesgos críticos como dependencia tecnológica, sesgos algorítmicos y un elevado impacto ambiental



KPIS y *engineering velocity* en la era del desarrollo asistido por IA

Esta transformación del rol de las consultoras IT, la productividad y la efectividad de resultados **tiene una consecuencia directa en cómo se mide el rendimiento en los equipos de desarrollo.** Tradicionalmente, los equipos han confiado en indicadores como la velocidad de entrega, el tiempo de ciclo o la cobertura de pruebas. Sin embargo, en entornos asistidos por IA, estos KPI evolucionan para reflejar nuevas dimensiones en la calidad, eficiencia e impacto.

En primer lugar, **aparecen métricas específicas del uso de IA**, como la precisión del modelo (*model accuracy*), la latencia de inferencia o el *throughput* en entornos de producción, que resultan clave cuando se despliegan asistentes de codificación, generación de pruebas o documentación automatizada. Al mismo tiempo, herramientas de análisis inteligente del código permiten medir la complejidad, identificar patrones problemáticos y evaluar automáticamente la mantenibilidad, elevando el listón de calidad desde etapas tempranas.

También se amplían los indicadores hacia la experiencia del desarrollador, como la reducción de la carga cognitiva, la mejora del foco o el estado de flujo sostenido, que son aspectos críticos para maximizar el rendimiento sostenible del equipo.

Por su parte, el concepto de ***engineering velocity* se reconfigurará a más allá del número de tareas completadas**, lo que importará ahora será cuánto valor real se entrega y en qué medida la IA potenciará la autonomía, colaboración y toma de decisiones del equipo. La automatización acelera la ejecución, pero el verdadero beneficio está en cómo humanos y agentes inteligentes se coordinan.

Ahora, se debe medir tanto la velocidad como el equilibrio entre rapidez, calidad, bienestar del equipo y valor aportado. **Adaptar los sistemas de métricas para medir la contribución de la IA será esencial para quienes buscan mantenerse competitivos** sin comprometer su sostenibilidad técnica y humana en el desarrollo de software. Aquí, las consultoras desempeñan la función clave de ayudar a las organizaciones a incorporar herramientas de IA y rediseñar sus arquitecturas, sus flujos de trabajo y, fundamentalmente, sus sistemas de métricas.

Porque sin un sistema de medición que nos permita interpretar correctamente ese impacto, las organizaciones corren el riesgo de diluirse entre la complejidad o la inercia operativa. **Medir bien es, hoy más que nunca, una ventaja competitiva.** Y hacerlo con inteligencia, no solo artificial, sino organizativa, es lo que marcará la diferencia entre adoptar IA y realmente transformarse con ella.



¿Cómo las
empresas de
consultoría pueden
liderar la nueva
era del código?

Las consultoras se reinventan, combinando *Low-Code*, *expertise* técnico y colaboración continua para liderar la transformación digital sostenible

¿Cómo se adaptan las consultoras a esta tendencia sin quedar obsoletas?

Lejos de quedar desplazadas, las consultoras están evolucionando para capitalizar el potencial de la IA y posicionarse como formadores y socios estratégicos en la implementación de estas plataformas. Si antes su propuesta de valor se centraba en proveer de equipo técnico y profesionales para desarrollar software a medida, ahora **deben asumir un rol más maduro, guiando, integrando y escalando el desarrollo digital de sus clientes.**

El nuevo paradigma abre nuevas oportunidades para quienes sepan adaptarse con visión estratégica. El reto ya no está en competir por la escritura de código, sino en **liderar la orquestación del desarrollo ágil, seguro y escalable dentro de cada organización.**

Las consultoras que comprendan esta nueva lógica podrán evolucionar en este nuevo paradigma que crece en el sector. Aquellas que insistan en competir desde el "*body shopping*" y las horas/hombre, por el contrario, corren el riesgo de ser superadas por un ecosistema más ágil, descentralizado y empoderado.

Esta transformación implica cinco grandes líneas de adaptación:

Facilitadores de gobierno, formación y escalabilidad

Actuar como aliados estratégicos y formadores, creando Centros de Excelencia, estandarizando metodologías y estableciendo buenas prácticas para evitar el *shadow IT*.

Especialistas en la integración avanzada y la última milla técnica

Posicionarse como los responsables de asegurar la calidad, seguridad y rendimiento, completando las capacidades de los *citizen developers*.

Socios de evolución continua

Pasar de una entrega puntual de software a ciclos de mejora continua, estableciendo relaciones de colaboración a largo plazo que permitan acompañar al cliente en su evolución.

Expertos y creadores de propiedad intelectual reutilizable

Convertir su *expertise* sectorial en módulos o plantillas reutilizables, posicionándose como creadores de propiedad intelectual, no solo como proveedores de código a demanda.

Agentes de cambio y mentores organizacionales

Liderar la transformación cultural, redefiniendo flujos, colaboración e innovación, reduciendo riesgos y acelerando la adopción consciente de estas tecnologías.

Un nuevo modelo de colaboración, con las consultoras como impulsores del desarrollo inteligente y sostenible

Ejemplo del recorrido del acompañamiento continuo y visión compartida

1 Kickoff

La consultora no espera requisitos detallados, sino que cocrea la visión junto al cliente.

- Identifica oportunidades para integrar la IA.
- Alinea los objetivos de negocio con la estrategia tecnológica
- Define indicadores de innovación, productividad y eficiencia.

2 Diseño y Arquitectura

Se diseña una arquitectura modular, escalable e inteligente.

- Permite integrar IA, automatización y observabilidad de forma nativa.
- Soporta iteración rápida y adaptación continua.
- Aplica principios de "AI by design" como autocuración y reconocimiento de patrones.

3 Desarrollo y testing

Se incorporan asistentes de codificación, pruebas automatizadas y validación inteligente.

- Acelera los ciclos de desarrollo y reduce tareas repetitivas.
- Mejora la calidad del código generado por IA mediante revisión y evaluación.
- Alinea lógica técnica con necesidades del negocio mediante prompts efectivos.

4 Deploy/ Observabilidad

Las implementaciones se automatizan y supervisan en tiempo real.

- Se configuran sistemas de autoescalado, monitoreo proactivo y alerta inteligente.
- Asegura visibilidad total del rendimiento y experiencia del desarrollador.
- Permite experimentar en producción con bajo riesgo.

5 Evolución/ feedback

La consultora permanece tras el lanzamiento para optimizar y evolucionar continuamente.

- Ajusta herramientas y flujos según el uso real.
- Optimiza la experiencia del desarrollador y reduce interrupciones.
- Integra nuevas tecnologías o funcionalidades según necesidad.

6 Cultura e innovación

La consultora impulsa una cultura de ingeniería aumentada e innovación continua.

- Promueve prácticas centradas en IA, automatización y agilidad.
- Capacita a los equipos en nuevas herramientas y flujos.
- Ayuda a construir capacidades internas sostenibles.

7 Crecimiento empresarial

La consultora actúa como guía estratégica de crecimiento y eficiencia tecnológica.

- Aconseja sobre escalabilidad, eficiencia energética y automatización.
- Introduce conceptos como GreenOps, VibeOps y DevEx inteligente.
- Ayuda al cliente a posicionarse como líder innovador en su industria.

La adopción masiva de IA implica riesgos críticos como dependencia tecnológica, sesgos algorítmicos y un elevado impacto ambiental

Esta adopción masiva conlleva un cierto número de riesgos que se deben tener en consideración y gestionarse de manera proactiva:



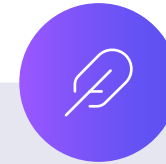
Dependencia tecnológica y erosión de habilidades

A medida que los desarrolladores se apoyan más en herramientas de IA, existe el peligro de la sobre dependencia. Esto puede **generar vulnerabilidad operativa ante fallos o interrupciones en las plataformas proveedoras**. Además, podría erosionar habilidades en los equipos técnicos, especialmente en perfiles más junior, convirtiendo a los equipos en "profesionales menos experimentados".



Sesgos de algorítmicos y alucinaciones

Estos sistemas aprenden de datos históricos que pueden contener sesgos, ya sean de calidad de código, estilo o incluso éticos o discriminatorios, lo que implica el **riesgo de que reproduzcan errores o generen resultados inseguros**. Adicionalmente, las "alucinaciones" de la IA suponen un gran peligro en el desarrollo de software. De hecho, se ha documentado que muchas sugerencias ocasionalmente son irrelevantes o incorrectas.



Sostenibilidad y huella de carbono

Los modelos de código más eficientes consumen aproximadamente 50 toneladas de CO₂ pese a entrenarse en infraestructuras energéticamente optimizadas. Por lo que, ante la creciente presión regulatoria y social, las organizaciones deberán adoptar enfoques más sostenibles, como la reutilización de modelos ya entrenados, el uso de energía renovable o el diseño de algoritmos más eficientes.

Liderar la transformación requiere *partners* con visión, capacidad técnica y criterio estratégico, no proveedores centrados en ejecución

Take action now: liderar la nueva era del código

La transformación que atraviesa el desarrollo de software no se limita únicamente a nuevas herramientas, **implica un gran cambio estructural que redefina las bases del delivery tecnológico:** desde la configuración de los equipos hasta los mecanismos de gobernanza y los modelos de relación con el negocio. Es por ello por lo que no basta con adaptarse, las organizaciones que aspiren a liderar deberán anticiparse al cambio de manera rápida y reconfigurar profundamente la forma en la que crean. La ventaja competitiva no vendrá del acceso a la tecnología, sino de la capacidad para convertirla en un motor de cambio sostenible y diferencial.

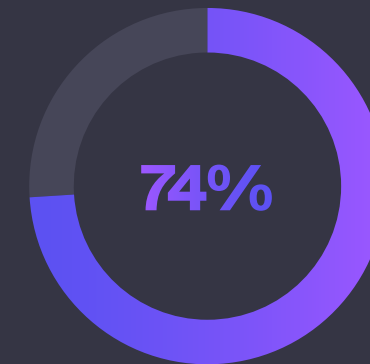
Es por ello por lo que, **contar con un partner estratégico y tecnológico es una condición y requisito para poder siquiera competir.**

El modelo de colaboración que emerge no es transaccional, sino transformacional. El papel deja de ser el de ejecutoras de proyectos para convertirse en habilitadoras de futuro, combinando asesoría tecnológica, integración fluida de nuevas herramientas y visión estratégica, además de traducir la complejidad en oportunidades, integrar capacidades que

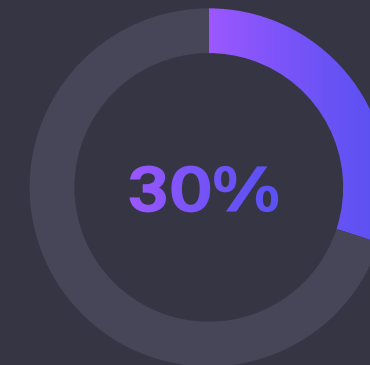
el cliente no puede escalar internamente, y acompañar la transformación con un enfoque integral, ágil y sostenible.

Sus aportaciones son varias: desde la asistencia con capacidades avanzadas, talento especializado, metodologías inteligentes y acceso continuo hasta el conocimiento de tendencias globales o futuras y la ayuda con los marcos regulatorios y éticos. Este acompañamiento **permite optimizar procesos, acelerar ciclos de innovación, reducir costos operativos, fortalecer la resiliencia digital y escalar iniciativas sin sobrecargar las capacidades internas.**

Contar con un socio estratégico significa tener una guía constante en un entorno donde el cambio es acelerado y la presión competitiva no deja margen para errores. Las consultoras no solo ayudan a implementar tecnologías como la IA, también actúan como catalizadoras de innovación, anticipándose a las tendencias del mercado y **ayudando a las empresas a construir modelos de negocio preparados para el futuro que llega.**



de las empresas tiene dificultades para encontrar talento IT, por ello el desarrollo de talento es una prioridad dentro de las alianzas estratégicas



más eficiencia operativa para las compañías que integran IA a través de alianzas estratégicas

La nueva era del código se lidera con visión compartida, colaboración constante y una estrategia basada en cocrear valor

Las compañías que apuestan por este tipo de alianzas no solo ejecutan mejor; piensan y deciden con mayor certeza y se adaptan más rápido. Los beneficios de contar con un socio estratégico son múltiples:



En definitiva, la nueva era del código no solo cambia la manera de crear y desarrollar código, **es una oportunidad estratégica para que las organizaciones eleven su potencial apoyándose en socios o *partners* tecnológicos capaces de liderar, guiar y ejecutar los desafíos de la transformación digital.**



softtek.com