



**La influencia de la Inteligencia Artificial en la codificación de aplicaciones en
la empresa SMART TMT.**

Juan Leonardi Ñustes Barrera

Gustavo Adolfo Vélez Cornejo

Universidad EAN

Facultad de Ingeniería

Especialización en Gerencia de Tecnología

Bogotá, Colombia

18/02/2026

**La influencia de la Inteligencia Artificial en la codificación de aplicaciones en
la empresa Smart TMT.**

Juan Leonardi Ñustes Barrera

Gustavo Adolfo Vélez Cornejo

Trabajo de grado presentado como requisito para optar al título de:

Especialista en Gerencia de Tecnología

Profesor (a):

Luisa Fernanda Carvajal Díaz

Unidad de estudio:

Seminario de Investigación

Universidad EAN

Facultad de Ingeniería

Especialización en Gerencia de Tecnología

Bogotá, Colombia

18/02/2026

Resumen

La presente investigación abordará la proyección de la eficiencia en términos de tiempo en el desarrollo de código fuente tradicional frente al desarrollo asistido por herramientas de inteligencia artificial en la empresa Smart TMT. Actualmente, las organizaciones dedicadas al desarrollo de software han ejecutado la programación de manera manual dependiendo principalmente del conocimiento y la experiencia de los desarrolladores, lo que implica una alta inversión de tiempo y recurso humano.

En el contexto actual de transformación digital, la implementación de herramientas de inteligencia artificial en la generación de código representa una alternativa orientada a optimizar la productividad y reducir tiempos de desarrollo y más aún, potencialmente a disminuir los costos operativos, es por esto que, entre los objetivos a alcanzar están estimar el tiempo empleado en el desarrollo de código fuente con el método tradicional versus apoyándose en inteligencia artificial, establecer los factores determinantes para el resultados obtenidos y determinar las habilidades que deben apropiar los desarrolladores para tomar ventaja de la inteligencia artificial.

Este cambio también redefine el rol del programador, quien pasa de ser un generador directo de código a desempeñar funciones de supervisión, validación y ajuste del código, especialmente ante posibles sesgos y errores derivados del uso de estas herramientas IA. Si bien la evolución de la inteligencia artificial ha sido exponencial, su implementación efectiva en el desarrollo de software requiere un enfoque integral que considere personas, procesos y metodologías, así como estrategias de capacitación, control de calidad y seguridad.

Palabras clave: Inteligencia artificial, Desarrollo de Software, Asistentes de IA, Sistemas, Aplicaciones, Programas.

Contenido

Resumen	6
Lista de Figuras	9
Planteamiento del Problema	11
Objetivos	12
<i>Objetivo general</i>	<i>12</i>
<i>Objetivos específicos</i>	<i>12</i>
Justificación	13
Marco Institucional	14
Marco Teórico	15
Variables y Enfoque Metodológico	21
<i>Variables</i>	<i>24</i>
<i>Preguntas de la encuesta</i>	<i>28</i>
Anexos.....	44
Referencias	455

Lista de Figuras

Figura 1. <i>Actividades estructurales del marco de trabajo del proceso</i>	16
Figura 2. <i>Incremento en la tasa de aceptación de código sugerido</i>	18
Figura 3. <i>Tasa de aceptación de código sugerido según la experiencia de los desarrolladores</i>	19
Figura 4. <i>Pronóstico del gasto en software empresarial</i>	20
Figura 5. <i>Número de desarrolladores en el mundo entre del año 2018 al 2024</i> ..	20.

Lista de Tablas

	Pág.
Tabla 1. <i>Estimación del tiempo de desarrollo con y sin IA por nivel de experiencia y nivel de complejidad de los diferentes.</i>	37
Tabla 2. <i>Resultado prueba Wilcoxon para los 30 pares de estimaciones</i>	38

Planteamiento del Problema

En la actualidad el mundo se encuentra en un proceso de transformación digital donde la tecnología ha tenido grandes avances en los últimos años, especialmente en el proceso de desarrollo de software. Actualmente, en muchas organizaciones el desarrollo de software es realizado de manera manual por los programadores, siendo un método que requiere conocimientos específicos, curva de aprendizaje y mucho tiempo de implementación, además del uso de metodologías robustas que resultan pesadas e ineficientes (Morales & Pardo, 2016).

Con el auge de la inteligencia artificial potenciada por las capacidades de cómputo actuales, estos procesos ahora pueden ser apoyados e incluso realizados en su totalidad por la inteligencia artificial, existen herramientas capaces de generar código, e incluso desarrollar funcionalidades completas de manera automatizada (Sarco & Toaquiza, 2022), que llegan a hacer pensar que, con el tiempo, la forma tradicional de desarrollar aplicaciones puede ser desplazada por los modelos inteligentes.

Si bien la aplicación de la inteligencia artificial, por una parte, representa una oportunidad para los desarrolladores y la sociedad en general, por la otra, presenta desafíos en materia de seguridad, privacidad y legales; toda vez que el código generado puede tener errores de seguridad, haber utilizado datos personales o incorporar partes de código de uso libre o con derecho de propiedad (Taulli, 2024).

Asimismo, la estandarización de códigos producidos por inteligencia artificial conserva un mismo patrón de desarrollo “sesgo” (Salazar & Cárdenas, 2024), que podría llegar a encontrar similitud en los proyectos de software desarrollados por la inteligencia artificial, por lo que se puede considerar que se perderá la creatividad e innovación en las aplicaciones, reduciendo la diversidad de enfoque en la solución de problemas complejos.

Ante esta nueva realidad, el rol de los miembros del equipo y en especial del desarrollador debe transformarse, pasando de ser un simple escritor de líneas de código a

convertirse en un supervisor y evaluador para ajustar el código, puesto que estas herramientas no son infalibles (Llerena, Viscaino, Culque, & Baño, 2023) . Adicionalmente, se dedicarán a actividades de mayor complejidad.

La revisión de la literatura presenta de manera general un impacto en el incremento de la productividad de los desarrolladores por la aplicación de la inteligencia artificial en el desarrollo de software (Dohmke, Iansiti, & Richards, 2023) , sin embargo, es importante determinar la productividad real versus la percibida, puesto que la productividad está afectada por la necesidad de supervisión por excesos de creatividad, alucinaciones, errores y ajustes en la arquitectura.

Por todo lo anterior, esta investigación pretende resolver la siguiente inquietud:
¿Cuál es la proyección en relación de eficiencia en tiempo en el desarrollo de código fuente tradicional versus el desarrollo de código fuente utilizando asistentes de inteligencia artificial (IA) en la empresa Smart TMT?.

Objetivos

Objetivo general

Determinar la proyección de la eficiencia del tiempo en el desarrollo de código fuente mediante el uso de asistentes de inteligencia artificial en comparación con el desarrollo tradicional en la empresa Smart TMT.

Objetivos específicos

- Estimar el tiempo empleado en el desarrollo de código fuente utilizando asistentes inteligencia artificial y el tiempo requerido en un desarrollo tradicional.
- Establecer los factores determinantes para el resultado obtenido con IA y con tradicional.
- Determinar las habilidades que deben adquirir los desarrolladores para tomar ventaja competitiva de la inteligencia artificial.

Justificación

En el contexto actual en el que las organizaciones están enfrentadas a cambios constantes y más rápidos por la competencia feroz, con nuevos modelos de negocio apalancados por la transformación digital, es primordial analizar el contexto para estar preparados y tomar las medidas, procurando estar a la vanguardia en materia tecnológica para suplir lo que está necesitando el mercado en un mundo cada vez más interconectado.

La inteligencia artificial hoy día está siendo de gran ayuda para aportar valor a los seres humanos en muchos ámbitos de la vida cotidiana, y su impacto es significativo en diferentes áreas (Salazar & Cárdenas, 2024) (Sarco & Toaquiza, 2022) , por lo cual, el estudio de la influencia de la inteligencia artificial en la codificación de aplicaciones resulta pertinente en esta coyuntura actual.

Desde una perspectiva profesional y laboral, esta investigación es relevante porque las empresas de tecnología, las fábricas de software y los desarrolladores enfrentan un momento importante donde se empieza a considerar la adopción de herramientas basadas en inteligencia artificial, teniendo en cuenta que el uso de estas busca reducir tiempo, mejorar la productividad y optimizar costos. De la misma manera, se debe tener en cuenta los riesgos de dependencia, la calidad de los códigos generados y la vulnerabilidad de la información (Taulli, 2024) (Salazar & Cárdenas, 2024). Por consiguiente, se debe considerar un equilibrio entre los beneficios y riesgos, para que las organizaciones incorporen en el ciclo de vida de las aplicaciones el desarrollo asistido por inteligencia artificial en un marco de gobierno y buenas prácticas.

Así las cosas, es muy importante determinar como el desarrollo de software asistido por inteligencia artificial incide realmente en el cronograma de un proyecto y en la mejora en el time-to-market de los productos o servicios. La presente investigación se justifica, por tanto, en la necesidad de establecer una comparativa que proyecte la relación de eficiencia en tiempo

entre el método de desarrollo tradicional y el asistido por inteligencia artificial, ahondado a las habilidades que los desarrolladores deben fortalecer para sacar provecho de esta.

Marco Institucional

Smart TMT es una empresa privada colombiana fundada en el 2003, donde su casa matriz se encuentra ubicada en la ciudad de Cali (Valle del cauca) con presencia de personal en las regiones centro, occidente y eje cafetero del país, para atender los clientes de línea base y potenciales de las zonas de influencias mencionadas (SMART TMT, 2026). Esta organización se encuentra dentro del sector de tecnología y se especializa en el desarrollo de soluciones tecnológicas y consultoría a nivel nacional enfocada al fortalecimiento tributario, en la cual, sus clientes son alcaldías municipales y entidades estatales.

La empresa ofrece un conjunto de productos y servicios basados en un software de base instalada desarrollado mediante programación tradicional sobre la plataforma Oracle (Oracle developer). Esta solución opera bajo Web Based con base en tecnología JAVA J2EE, complementadas con herramientas de ofimáticas de Microsoft, Google y bajo licencias de libre licenciamiento (SMART TMT, 2026).

El portafolio tecnológico está compuesto por los siguientes módulos: Plataforma de Gestión, Portal de servicios web, Plataforma Valorización, Plataforma de reportes Analytics, Aplicación City, Interfaz Catastro, Interfaz Service y Reports (SMART TMT, 2026).

A través de estas soluciones, la compañía presta a sus clientes los siguientes servicios:

- a) Desarrollo a la medida, ajustado a cambios normativos legales o de entidad.
- b) soporte técnico para la atención de inconsistencias reportadas en plataforma, aplicaciones o interfaz.
- c) actualización de versiones de la plataforma y de base de datos.
- d) mantenimiento, corrección y optimización del código fuente de los productos ofrecidos.

Esto con el fin de obtener una gestión, operación, administración y control integral en la cadena de valor y recaudo de los

diferentes impuestos, contribuciones y tributos como: predial, industria y comercio, alumbrado público, bomberos, plusvalía y valorización, entre otros.

Lo anteriormente mencionado se encuentra respaldado por un equipo de desarrollo conformado por un (1) Arquitecto de Producto Máster, un (1) Líder de Desarrollo, un (1) desarrollador máster, nueve (9) desarrolladores senior y un (1) desarrollador junior. En relación con el nivel de conocimiento y experiencia en el uso de inteligencia artificial aplicada al desarrollo de software se identifica que cuentan con experiencia o conocimientos: el Arquitecto de Producto (1), el Líder de Desarrollo (1), un (1) desarrollador máster y un (1) desarrollador senior. Por otro lado, ocho (8) desarrolladores senior y un desarrollador junior (1) que no poseen experiencia o tienen un nivel muy bajo de conocimiento en el manejo de herramientas de inteligencia artificial para el desarrollo de software.

En este contexto y teniendo en cuenta su actividad especializada, la organización ha iniciado la implementación de inteligencia artificial en la etapa de desarrollo en la codificación del código de sus distintas plataformas generadas por el desarrollo tradicional, apoyado en el personal con conocimiento y experiencia con el fin de tener una mayor calidad, agilidad, seguridad, confiabilidad y rentabilidad en la generación de código para futuros desarrollos. Así mismo, busca optimizar el soporte y mantenimiento mediante uso continuo de esta tecnología, reduciendo la dependencia de ingenieros de desarrollo. Sin embargo, esta investigación se limita a determinar cuál es la proyección en términos de eficiencia de tiempo en el desarrollo de código fuente tradicional versus el desarrollo de código fuente utilizando asistentes de inteligencia artificial (IA) en la empresa Smart TMT.

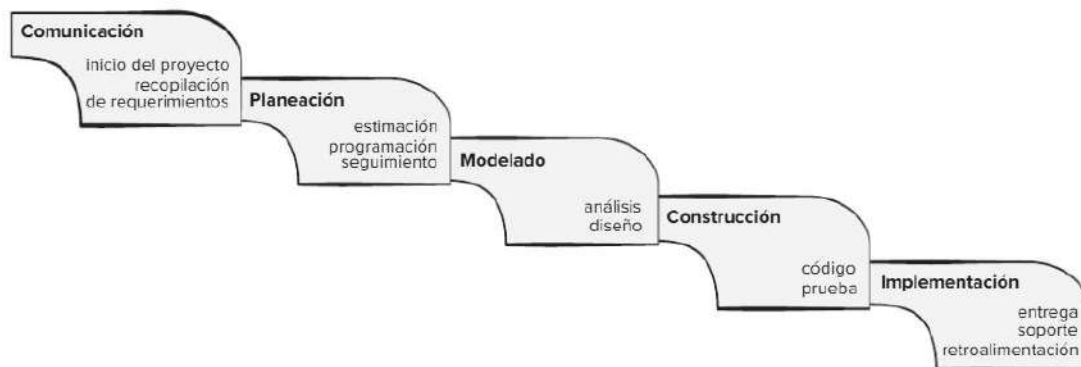
Marco Teórico

Con el propósito de entender como la inteligencia artificial incide en el proceso de construcción de software es relevante iniciar por entender que es un software, según (Pressman & Maxim, 2021), es un conjunto de tres elementos, instrucciones que al ejecutarse

realizan o desempeñan unas funcionalidades deseadas, estructuras de datos para gestionar la información de forma adecuada y finalmente la documentación que describe la forma de uso y operación del software.

De la misma manera, (Pressman & Maxim, 2021) en su libro definen las 5 actividades estructurales que deben realizarse en el marco de trabajo del proceso genérico para el ciclo de vida de la ingeniería del software: comunicación, planeación, modelado, construcción e implementación. La figura 1 describe el proceso en un modelo tradicional lineal de cascada.

Figura 1. Actividades estructurales del marco de trabajo del proceso



Nota. Tomado de “Ingeniería de Software: un enfoque práctico”, por Pressman, R. S., Maxim, B. R., 2021, p. 26.

“La IA se define como el desarrollo de algoritmos y sistemas capaces de realizar tareas que normalmente requerirían inteligencia humana, como el aprendizaje automático y la toma de decisiones” (Erazo, Ramos, Galarza, & Boné, 2023).

Si bien la inteligencia artificial es una tecnología antigua de la cual podemos ver casos como el test de Turing publicado en los años 50, cuyo propósito era demostrar la existencia de inteligencia en un dispositivo no biológico (Turing, 1950). Su vertiginoso desarrollo, crecimiento y auge esta apalancado en el avance de las capacidades de cómputo actuales, mejorando los

procesos de algoritmos complejos y el procesamiento de grandes volúmenes de información (Sarco & Toaquiza, 2022).

“Es innegable que en los últimos tiempos la evolución de la inteligencia artificial ha crecido a pasos agigantados y está destinada a redefinir las prácticas de desarrollo de software dado el potencial que ofrece para la construcción de diversas aplicaciones habilitadas por IA para ponerlas al servicio de muchas áreas importantes de desenvolvimiento rutinario de los seres humanos” (Salazar & Cárdenas, 2024).

El impacto que tiene la inteligencia artificial en el desarrollo de software en las empresas va más allá de adquirir las plataformas; para tomar ventaja deben realizar un adecuado manejo del cambio que tenga en cuenta personas, procesos y metodologías. Se debe realizar un fortalecimiento de las habilidades de los desarrolladores en temas como ingeniería de prompts, lineamientos de calidad, seguridad y gestión de riesgos en el código generado (Taulli, 2024), transformando al desarrollador generando valor en actividades más creativas y estrategias y de supervisor para revisar y evaluar del código.

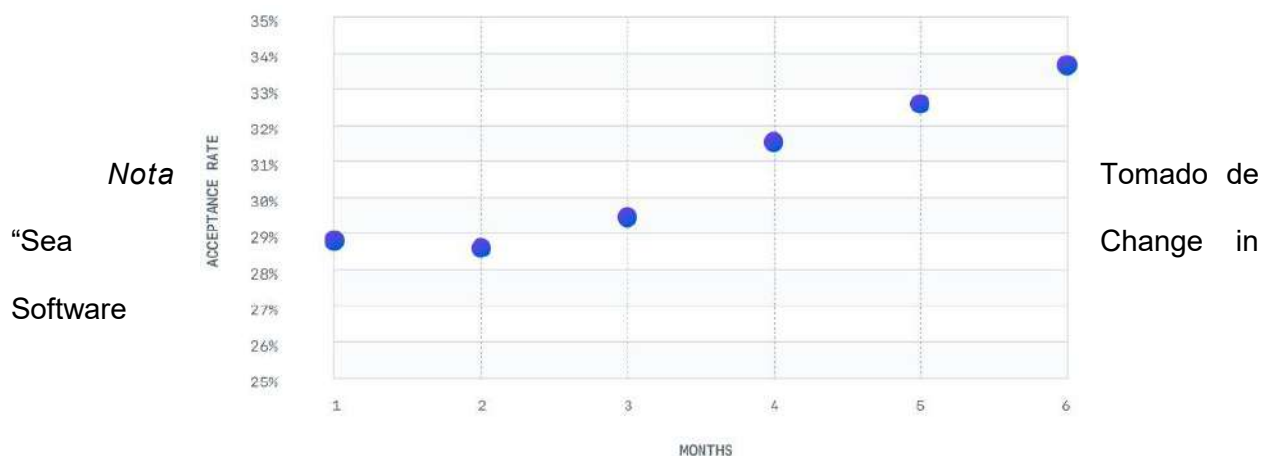
Además de los cambios operativos, técnicos, la aplicación de inteligencia artificial en la codificación de aplicaciones genera un impacto significativo en las estructuras organizacionales ya que puede modificar los perfiles de contratación y esto podría reducir la introducción en el mercado laboral para nuevos profesionales enfocados al desarrollo de software sin competencias o habilidades a la codificación de código con asistentes de inteligencia artificial, generando una modificación en la dinámica del empleo en empresas de tecnología, desarrollo de software y en general.

Una investigación realizada por la firma de análisis Keystone.AI (Dohmke, Iansiti, & Richards, 2023), arroja resultados interesantes en la incidencia que tiene el desarrollo de software asistido por inteligencia artificial en campos como la economía y la productividad de los desarrolladores. Reportan que al asistente de IA Github Copilot ha sido activado por más de un millón de desarrolladores y adoptado por más de 20.000 organizaciones. Partieron de una

investigación previa del año 2022, un experimento controlado en el que segmentaron los desarrolladores aleatoriamente en dos grupos a los cuales se les cronometró el tiempo que tomaron para implementar un servidor HTTP en el lenguaje JavaScript, un grupo desarrollaba con el asistente y el otro de la manera tradicional. El experimento mostró que los desarrolladores que utilizaron la IA implementaron el servidor 58.8% más rápido. (Dohmke, Iansiti, & Richards, 2023).

Un año después de que el asistente GitHub Copilot fue liberado de manera general al mercado, los investigadores iniciaron a analizar el impacto del asistente en los desarrolladores durante un periodo de tiempo y usan la tasa de aceptación de líneas de código propuestas por el asistente como métrica de productividad. Analizando las estadísticas de la plataforma, encontraron que los usuarios aceptaban el 30% de las sugerencias de código, lo que representa una evidente mejora en la productividad. Adicionalmente, evidenciaron que la tasa de aceptación aumentó a lo largo del tiempo pasando de cerca del 29% en el primer mes a casi 34% en el sexto, como se ilustra en la Figura 2, se observa una tendencia al alza y oportunidad de crecimiento. (Dohmke, Iansiti, & Richards, 2023).

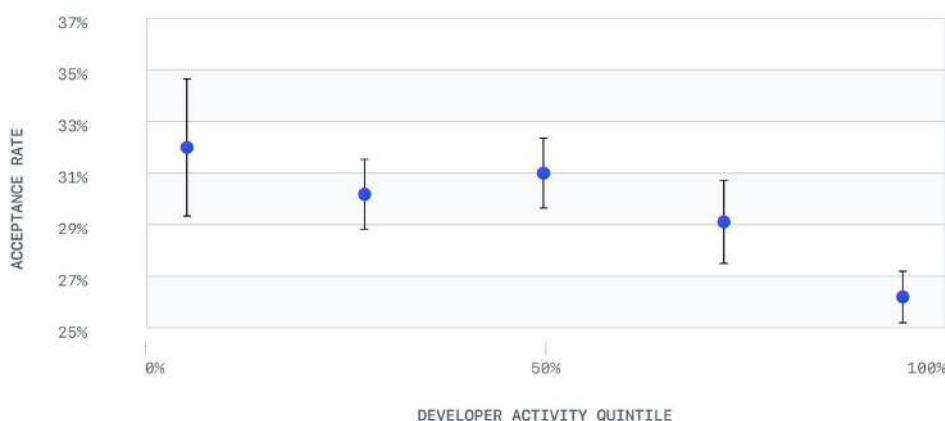
Figura 2. *Incremento en la tasa de aceptación de código sugerido*



Development: Economic and Productivity Analysis of the AI-Powered Developer Lifecycle”, por Dohmke, T., Iansiti, M., Richards, G., 2023, p. 4.

Además, observaron que los desarrolladores con menos experiencia se apalancaban más en la IA que los de mayor bagaje. En la Figura 3 se ilustra a los desarrolladores agrupados en quintiles según su experiencia por el número de actividades realizadas en el repositorio git. En el quintil de más a la izquierda, menos experimentados, se puede ver que la tasa de aceptación está cercana al 32%, mientras que en el quintil más a la derecha, más experimentados, la tasa de aceptación de código cae a cerca del 26%. (Dohmke, Iansiti, & Richards, 2023).

Figura 3. Tasa de aceptación de código sugerido según la experiencia de los desarrolladores



Nota. Tomado de “Sea Change in Software Development: Economic and Productivity Analysis of the AI-Powered Developer Lifecycle”, por Dohmke, T., Iansiti, M., Richards, G., 2023, p. 6.

En adición a lo anterior, estos mismos autores (Dohmke, Iansiti, & Richards, 2023) citando otras fuentes indican que debido a la dinámica de digitalización en la que se encuentran las organizaciones a nivel mundial, existe una creciente demanda en software que no puede ser suplida de la mejor manera debido a una escasez de desarrolladores. El estudio de Statista de la Figura 4, muestra el crecimiento sostenido en la inversión que han hecho las

organizaciones en software empresarial iniciando el año 2009 con \$225 billones y con un pronóstico para el año 2025 de \$1250 billones. (Kanovich, 2025)

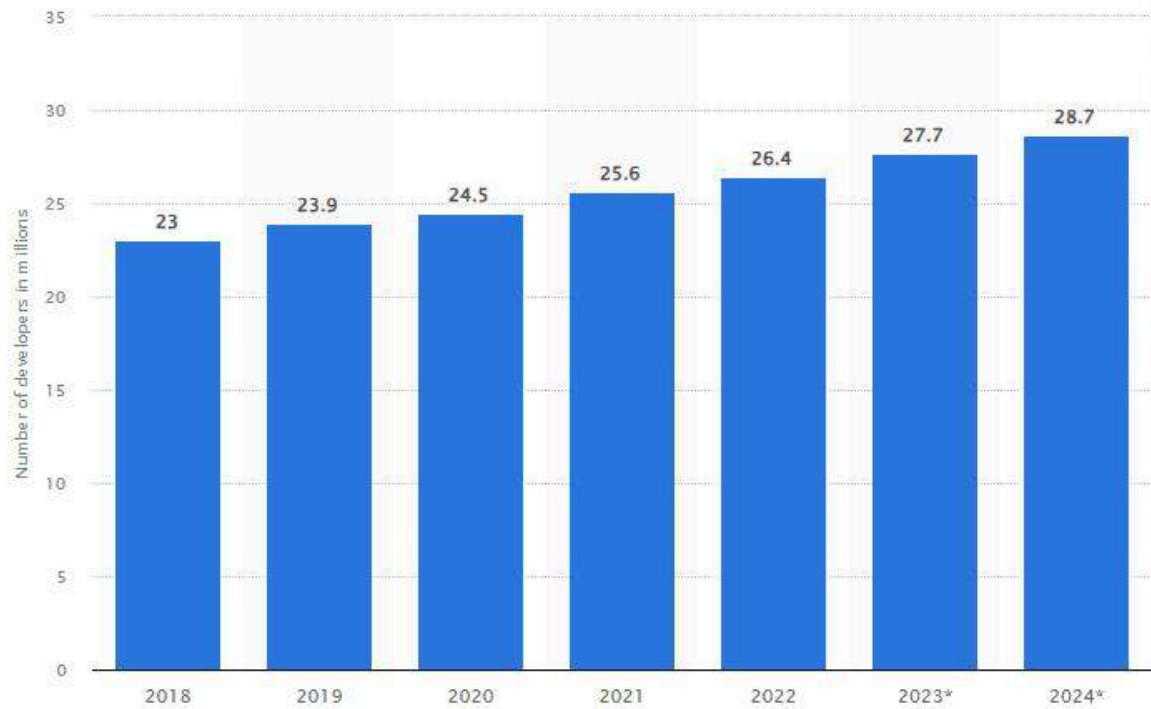
Figura 4. *Pronóstico del gasto en software empresarial*



Nota. Tomado de “Software Enterprise: The Changing Ecosystem”, por Kanovich, R., 2023, p. 2.

Se estima que en año 2024 en el mundo existían cerca de 29 millones de desarrolladores (Statista Research Department, 2025) y en un estudio predicen que para el 2030 la escasez de talento en el sector de tecnología podría ser de 4.3 millones de trabajadores (Korn Ferry, 2018), afectando el avance de la revolución digital. Lo importante de la incorporación de la IA al desarrollo de software es que puede ayudar a cerrar esa brecha de falta de talento al aumentar la productividad de los desarrolladores.

Figura 5. *Número de desarrolladores en el mundo entre del año 2018 al 2024*



Nota. Tomado de “Statista: Number of software developers worldwide in 2018 to 2024”, por Statista Research Department.

Variables y Enfoque Metodológico

El estudio a desarrollar es transversal y estudio de caso debido a que la recolección de datos se hará en un momento único aplicando el instrumento de encuesta y entrevista semiestructurada, ambas, se realizarán a través de formularios de Google Forms, respectivamente. Por lo anterior, el enfoque investigativo va ser mixto, donde el cuantitativo se va a encargar de proyectar la relación de eficiencia en tiempo proyectada requerido en el desarrollo de código fuente apoyado con la inteligencia artificial en comparación con el desarrollo tradicional y el cualitativo se encargará de determinar los factores determinantes para los resultados obtenidos y las habilidades que deben apropiarse y fortalecer los desarrolladores para tomar ventaja competitiva de la inteligencia artificial.

El alcance del estudio incluirá, por una parte, el tipo de análisis descriptivo, porque busca especificar las habilidades que deben poseer los desarrolladores para tomar ventaja de

la IA, y por la otra, correlacional, porque pretende analizar cómo las variables independientes (complejidad del desarrollo y nivel de experiencia) afectan a la variable dependiente (eficiencia en tiempo proyectada).

El trabajo investigativo se llevará a cabo en las siguientes tres fases:

Fase 1: Recolección de información y determinar la línea base: Se trata de buscar en la documentación de la empresa tres casos de uso ya implementados de diferente complejidad y el tiempo aprobado en su momento por un experto sin IA y con IA. Se requieren dos semanas.

Fase 2A: Diseño y aplicación del Instrumento (Entrevista): Se entrega una entrevista semiestructurada al arquitecto del producto, líder de desarrollo y a un (1) desarrollador con experiencia en el uso de IA. El instrumento incluye preguntas abiertas y descripción técnica de los tres casos analizados, permitiendo explicar su razonamiento técnico y expresar sus percepciones sobre el uso de la IA en relación con: los factores determinantes de la eficiencia en tiempos, las habilidades que generan ventajas competitivas y barreras para su adopción. Se requieren dos semanas. Una diseñando la entrevista y una recopilando los datos.

Fase 2B: Diseño y aplicación del Instrumento (Encuesta/Estimación): Se entrega a los 11 desarrolladores la descripción técnica de los tres casos. Se les pide responder las preguntas de la encuesta entre las cuales se solicita estimar:

Tiempo de desarrollo sin IA: Basado en su conocimiento técnico puro.

Tiempo de desarrollo con IA: Basado en el uso de asistentes para aquellos que ya los conocen o proyectan su uso. Se requieren cuatro semanas, una diseñando la encuesta y tres recopilando los datos.

Fase 3: Análisis Cuantitativo: Para el análisis inicial se aplicará la fórmula de la variable dependiente Eficiencia en Tiempo Proyectada, tomado los valores de la estimación realizada por los desarrolladores para cada caso de uso utilizando IA y del método tradicional. Adicionalmente se realizará la comparación contra el histórico, esto permitirá validar la productividad real versus la proyectada, por complejidad del desarrollo y por experiencia del

desarrollador. Con el propósito de fortalecer el resultado obtenido se incluirá rigor científico a través de una validación estadística que lo fundamente.

Inicialmente el análisis se apoyará en la prueba de Wilcoxon, sobre los pares de estimaciones que realicen desarrolladores que actualmente utilizan asistentes de IA, en los tres niveles de complejidad de caso de uso evaluados, para determinar si el uso de asistentes de IA produce una reducción significativa en los tiempos de desarrollo estimados. Por otra parte, se examinará si la complejidad del desarrollo o la experiencia del desarrollador son factores que inciden en la Eficiencia en Tiempo Proyectada (ETP). Para esto, se empleará la Prueba de Kruskal-Wallis, con el fin de identificar si existen diferencias significativas en la ETP según el nivel de complejidad del requerimiento (Baja, Media y Alta), permitiendo validar estadísticamente si la complejidad actúa como un factor que reduce la ventaja de la IA en tareas de mayor envergadura técnica. Y se utilizará la misma prueba para comparar la ETP entre los tres niveles de experiencia profesional (Junior, Senior y Master). El objetivo es corroborar si los desarrolladores con menor trayectoria presentan una ganancia de eficiencia superior frente a los perfiles con mayor experiencia.

Para realizar la validación estadística de los resultados, se utilizará el complemento de Excel Real Statistics, el cual es gratis y de uso libre e incluye entre otras, las pruebas Wilcoxon y Kruskal-Wallis.

Complementariamente, se realizará un análisis comparativo entre la **Eficiencia en Tiempo Proyectada** y las competencias específicas de los desarrolladores, desde la explicación técnica y el uso de la IA en relación con los factores determinantes, habilidades y barrera de adopción. Esto permitirá identificar si habilidades críticas, como la competencia en ingeniería de prompts y la capacidad de detección de alucinaciones, actúan como factores determinantes en la magnitud de los resultados obtenidos con el uso de IA.

Variables

Independientes:

- **Complejidad por Horas de Desarrollo:** Define la magnitud del esfuerzo requerido para desarrollar una especificación funcional, puede tener tres valores en el dominio:
 - **Baja:** Actividades de diseño o construcción de alcance reducido, con pocos elementos, reglas básicas, ajustes menores, bajo impacto técnico y mínima dependencia de componentes externos. Corresponde normalmente a requerimientos pequeños o intermedios. Entre 14 a 114 Horas, según el tipo de actividad.
 - **Media:** Actividades de diseño o construcción con alcance intermedio, que implican varios elementos funcionales, validaciones, cálculos, cambios estructurados o ajustes sobre componentes existentes. Corresponde normalmente a requerimientos medianos o grandes. Entre 115 a 386 Horas, según el tipo de actividad.
 - **Alta:** Actividades de diseño o construcción de amplio alcance, con alta cantidad de elementos, reglas complejas, integraciones, personalización avanzada, componentes nuevos o esfuerzos no acotados. Corresponde normalmente a requerimientos muy grandes. Desde 387 a 1.021 Horas, según la actividad.

Consideramos que para la investigación debemos tratarla como ordinal categórica porque permite agrupar los requerimientos por complejidad independiente del número exacto de horas que se requieren para el desarrollo y como la complejidad del desarrollo incide en la variable dependiente **Eficiencia en Tiempo Proyectada**.

- **Nivel de Experiencia de los Desarrolladores:** Define la capacidad técnica que tiene el desarrollador por el tiempo que se ha dedicado a desarrollar, nivel de conocimientos y trayectoria profesional. El dominio tiene tres valores posibles:
 - Master: Profesional con trayectoria altamente consolidada, conocimiento técnico profundo y visión integral del desarrollo de software. Se distingue por su capacidad para liderar definiciones técnicas, orientar decisiones de arquitectura, resolver problemáticas de alta complejidad y contribuir estratégicamente al fortalecimiento técnico del equipo y de las soluciones desarrolladas.
 - Senior: Profesional con experiencia consolidada, amplio dominio técnico y capacidad para desempeñarse con autonomía en el análisis, diseño, desarrollo e implementación de soluciones. Cuenta con criterio para resolver problemas de mediana y alta complejidad, y su trayectoria evidencia participación sostenida y relevante en proyectos de desarrollo.
 - Junior: Profesional con experiencia inicial en desarrollo de software, con conocimientos fundamentales del stack tecnológico y capacidad para ejecutar tareas previamente definidas. Requiere acompañamiento en la toma de decisiones técnicas de mayor complejidad y su trayectoria se caracteriza por aportes principalmente operativos y de alcance puntual.

Consideramos que para la investigación debemos tratarla como ordinal categórica porque agrupa a los desarrolladores de acuerdo con su trayectoria y nivel de competencia tanto técnica como de habilidades blandas, y permite

determinar cómo esta incide en la variable dependiente **Eficiencia en Tiempo**

Proyectada por cada uno de los grupos.

Moderadora

- **Método de Desarrollo:** Desarrollo tradicional y/o asistido con IA como variable moderadora ya que modifica la relación entre las variables dependientes en cuanto a la complejidad de desarrollo y nivel de experiencia que influye en el impacto para la variable dependiente **Eficiencia en Tiempo Proyectada**. El dominio tiene dos valores:
 - Desarrollo Asistido por IA
 - Desarrollo Tradicional

Dependientes:

- **Eficiencia en Tiempo Proyectada:** Variable numérica que define el porcentaje de ganancia en tiempo proyectado el realizar desarrollo de aplicaciones con asistentes de IA versus el desarrollo tradicional. Es importante resaltar que para el estudio se trabajará basado en una estimación en horas indicada por los desarrolladores para cada caso de uso de diferente nivel de complejidad.

La fórmula que se usará para determinar la eficiencia en tiempo proyectada será:

$$ETP = \left(\frac{\sum T_{tradicional} - \sum T_{IA}}{\sum T_{tradicional}} \right) * 100$$

Donde

ETP = Eficiencia en Tiempo Proyectada.

$\sum T_{tradicional}$ = La sumatoria de los tiempos estimados por los desarrolladores para los tres casos de uso de forma tradicional.

$\sum TIA$ = La sumatoria de los tiempos estimados por los desarrolladores para los tres casos de uso utilizando asistentes de inteligencias artificial.

Caracterizadoras:

Se tiene un grupo de variables caracterizadoras con el fin de identificar posibles factores determinantes y habilidades que deben tener y/o adquirir los desarrolladores en el contexto de la **Eficiencia en Tiempo Proyectada** con las herramientas IA y no como estas hacen variar la **Eficiencia en Tiempo Proyectada**. A continuación, definimos las siguientes:

Competencia en ingeniería de prompts: Nivel que posee el desarrollador para estructurar instrucciones claras y coherentes a la herramienta de IA para que estén alineadas a los requerimientos a desarrollar. Se identificará y se relacionará según los resultados obtenidos de la entrevista con los expertos en manejo de herramientas IA de la empresa.

Competencia para identificar errores lógicos o alucinaciones: Nivel de competencia del desarrollador para reconocer alucinaciones, errores sintácticos, semánticos o lógicos del código generado por la herramienta IA en los requerimientos a desarrollar. Se identificará y se relacionará según los resultados obtenidos de la entrevista con los expertos en manejo de herramientas IA de la empresa.

Verificación y aseguramiento de calidad del código sugerida por IA: Acción que el desarrollador realiza para revisar, validar y ajustar las recomendaciones generadas por la IA que garantiza los estándares técnicos, de seguridad y calidad de los requerimientos a desarrollar. Se identificará y se relacionará según los resultados obtenidos de la entrevista con los expertos en manejo de herramientas IA de la empresa.

Preparación para integrar IA en el entorno de desarrollo: Percepción de preparación técnica que tiene el desarrollador para incorporar IA en el entorno de desarrollo en los requerimientos a desarrollar. Se identificará y se relacionará según los resultados obtenidos de la entrevista con los expertos en manejo de herramientas IA de la empresa.

Capacidad de integración arquitectónica: Capacidad que tiene el desarrollador para integrar el código generado por la IA dentro de la arquitectura del sistema de información sin afectar la cohesión de los requerimientos a desarrollados y a desarrollar. Se identificará y se relacionará según los resultados obtenidos de la entrevista con los expertos en manejo de herramientas IA de la empresa.

Técnicas e instrumentos de recolección de datos

Encuesta

Para la recolección de los datos que se usaran para el análisis cuantitativo se utilizará una encuesta, en la que los desarrolladores responderán preguntas de respuesta cerradas y abiertas para el caso de capturar las estimaciones del esfuerzo para desarrollar los diferentes casos de uso sin y con apoyo o asistentes de IA.

Preguntas

Se recolectarán las siguientes variables para caracterizar a la población:

1. Especifique su nivel de experiencia en desarrollo
 - a. Master
 - b. Senior
 - c. Junior
2. Por favor seleccione su rango de edad
 - a. 18–24

- b. 25–34
- c. 35–44
- d. 45–54
- e. 55 o más

3. ¿Cuál es su nivel de formación académica?

- a. Técnico / Tecnólogo
- b. Profesional
- c. Especialización
- d. Maestría
- e. Doctorado

A continuación, se realizan preguntas específicas relacionadas con el estudio de investigación.

4. ¿Utiliza actualmente asistentes de IA (GitHub Copilot, ChatGPT, etc.) para tareas de codificación? Si responde “No” pasa a la pregunta 9

- a. Si
- b. No

5. ¿Cuál?

- a. Cursor
- b. Devin
- c. JetBrains
- d. Claudecode
- e. Windsurf
- f. GitHub Copilot
- g. Otra

Basándose en la descripción funcional de cada caso de uso, por favor indique el tiempo estimado del esfuerzo en horas de ingeniero para desarrollarlos con asistentes de inteligencia artificial.

6. Caso de uso 1, ¿Cuánto tiempo en horas?

7. Caso de uso 2, ¿Cuánto tiempo en horas?

8. Caso de uso 3, ¿Cuánto tiempo en horas?

Basándose en la descripción funcional de cada caso de uso, por favor indique el tiempo estimado del esfuerzo en horas de ingeniero para desarrollarlos sin asistentes de inteligencia artificial.

9. Caso de uso 1, ¿Cuánto tiempo en horas?

10. Caso de uso 2, ¿Cuánto tiempo en horas?

11. Caso de uso 3, ¿Cuánto tiempo en horas?

Entrevista

Para la recolección de los datos que se usaran para el análisis cualitativo se utilizará la herramienta de la entrevista, en la que los desarrolladores responderán preguntas de respuesta abierta para obtener sus percepciones.

1. Factores determinantes de la eficiencia (Relación complejidad–tiempo)

a. Basado en los tres casos de uso (baja, media y alta complejidad), ¿qué elementos técnicos específicos determinan que la IA sea un aliado real o un obstáculo en el desarrollo? y ¿Por qué cree que en tareas de alta complejidad el ahorro de tiempo suele ser menor?

b. Si usted estima un tiempo de supervisión para las sugerencias de la IA. ¿Cómo decide cuándo confiar plenamente en el código generado y qué criterios utiliza para detectar posibles alucinaciones o errores lógicos?

c. ¿Cómo influye la arquitectura actual empleada en su empresa en la velocidad con la que puede integrar código generado por IA? ¿Considera que las herramientas de IA se alinean bien con los estándares de calidad de la empresa?

2. Habilidades para la ventaja competitiva

a. Más allá de dar instrucciones básicas, ¿cómo ha perfeccionado su técnica de ingeniería de prompts (uso de contexto, delimitadores y ejemplos) para obtener resultados de mayor calidad?

b. Con el uso de asistentes, el trabajo manual disminuye. ¿Qué habilidades de diseño modular y pensamiento estratégico considera que son ahora más importantes que el simple conocimiento de la sintaxis del lenguaje?

c. ¿Qué procesos sigue para asegurar que el código sugerido por la IA no introduzca vulnerabilidades de seguridad (CWE) o problemas de propiedad intelectual en los repositorios de la empresa?

3. Factores determinantes y barreras de adopción

a. Si la curva de aprendizaje se percibe como empinada, ¿qué tipo de formación específica o acompañamiento técnico cree que facilitaría que el equipo adoptara la IA con mayor confianza?

b. En cuanto al riesgo de estandarización del código (sesgo). ¿De qué manera utiliza la IA para potenciar su creatividad en lugar de simplemente repetir patrones preexistentes?

c. ¿En qué medida la preocupación por la privacidad de los datos de la empresa limita su uso diario de herramientas basadas en la nube como ChatGPT o Claude?

Validación del instrumento

Encuesta

Según la validación del instrumento aplicando la V de Aiken sobre la encuesta, se elimina la pregunta donde se solicitaba los nombres y apellidos del desarrollador, el dato no es

relevante para la investigación y puede generar que no se responda transparentemente e invalidar o sesgar las respuestas debido a la pérdida de anonimato. Por otra parte, como con el nombre se podía identificar o asociar el nivel de experiencia del desarrollador, se hace necesario incluir esa pregunta para saber qué nivel de experiencia tiene el desarrollador que la está contestando.

Entrevista

Según la validación del instrumento aplicando la V de Aiken sobre la entrevista, no se hizo ninguna modificación sobre las preguntas del instrumento porque todas fueron entendidas por los expertos.

Resultados

1) Cálculo de la Eficiencia en Tiempo Proyectada (ETP).

Para realizar el análisis de la ETP, se ha utilizado la información de los 11 desarrolladores que respondieron la encuesta y la estimación real del juicio de experto. La ETP se calcula mediante la fórmula:

$$ETP = \left(\frac{\sum T_{tradicional} - \sum TIA}{\sum T_{tradicional}} \right) * 100$$

Donde

ETP = Eficiencia en Tiempo Proyectada.

$\sum T_{tradicional}$ = La sumatoria de los tiempos estimados por los desarrolladores para los tres casos de uso de forma tradicional.

$\sum TIA$ = La sumatoria de los tiempos estimados por los desarrolladores para los tres casos de uso utilizando asistentes de inteligencias artificial.

Eficiencia en Tiempo Proyectada General

Considerando a los 10 profesionales que utilizan asistentes de IA (9 de la encuesta y el experto), se analizó un total de 30 casos de uso evaluados bajo ambos métodos, obteniéndose

un ETP General de 26.76%. En términos globales, la organización proyecta una reducción de más de una cuarta parte del tiempo total de desarrollo al integrar herramientas como Claudecode y ChatGPT.

Eficiencia por Nivel de Complejidad

El análisis revela que la IA es más disruptiva en tareas de menor complejidad, aunque el ahorro en horas absolutas es mayor en tareas complejas:

ETP Baja Complejidad: 39.60%. Es el área con mayor optimización porcentual, donde desarrolladores como el número 7 alcanzan reducciones de hasta el 70.7%.

ETP Media Complejidad: 29.01%. Representa un ahorro estable y significativo en la carga de trabajo intermedia.

ETP Alta Complejidad: 23.67%. Aunque el porcentaje es menor, la reducción de tiempo es crítica debido al alto volumen de horas que exigen estos requerimientos.

Eficiencia por Nivel de Experiencia

Se observa que la ganancia de eficiencia se distribuye de manera competitiva en todos los rangos, con una ligera ventaja en los niveles de mayor jerarquía:

ETP Nivel Master: 27.47%. Incluye al experto y al desarrollador 1, mostrando que la IA potencia la capacidad arquitectónica de los perfiles más experimentados.

ETP Nivel Senior: 27.31%. Los 7 seniors analizados muestran una eficiencia casi idéntica a la de los Masters, confirmando que la IA es una herramienta estándar para este segmento.

ETP Nivel Junior: 22.19%. Aunque es el nivel con menor ganancia ETP, representa una aceleración significativa para perfiles en etapa de formación. Este resultado va un poco en contravía de los estudios internacionales en los que los desarrolladores de menor experiencia presentaban tasas de aceptación de código superiores al de los más experimentados, cercanas al 32% versus el 26%,

Eficiencia del Juicio de Experto

La estimación del experto (Master con uso de Claudecode con rol de arquitecto en la organización) sirve como la "medida de oro" o referencia de alta competencia para el análisis:

ETP Experto (Baja): 45.83%.

ETP Experto (Media): 23.49%.

ETP Experto (Alta): 17.39%.

ETP Total Experto: 21.54%.

El juicio de experto es más conservador en la alta complejidad comparado con el promedio de los desarrolladores, lo que sugiere una visión más realista de los desafíos técnicos de mayor envergadura que la IA no puede automatizar completamente.

El análisis de la Eficiencia en Tiempo Proyectada (ETP) demuestra que podrá tener un impacto positivo y consistente de la Inteligencia Artificial en los procesos de Smart TMT. Con una eficiencia general del 26.76%** , la tecnología se posiciona como un catalizador crítico de productividad. Es relevante que la eficiencia alcanza su pico en la baja complejidad (40.14%), actuando como una herramienta de automatización para tareas rutinarias. Asimismo, el hecho de que tanto los niveles Master (27.47%) como Senior (27.31%) presenten resultados casi idénticos indica que la IA puede ser adoptada con éxito, permitiendo una optimización transversal de la fuerza laboral independientemente de la competencia técnica. La tabla 1, ilustra los datos recolectados en la encuesta:

Tabla 1

Estimación del tiempo de desarrollo con y sin IA por nivel de experiencia y nivel de complejidad de los diferentes.

Desarrollador	Nivel de Experiencia	Nivel de Complejidad	Estimación con IA (Horas)	Estimación sin IA (Horas)
Experto	Master	Baja	26	48
Experto	Master	Media	254	332
Experto	Master	Alta	361	437
1	Master	Baja	24	40

1	Master	Media	175	260
1	Master	Alta	240	372
2	Senior	Baja	49	63
2	Senior	Media	220	340
2	Senior	Alta	432	510
3	Senior	Baja	24	32
3	Senior	Media	210	280
3	Senior	Alta	262	390
4	Junior	Baja	41	60
4	Junior	Media	311	365
4	Junior	Alta	360	490
5	Senior	Baja	24	46
5	Senior	Media	194	326
5	Senior	Alta	280	443
6	Senior	Baja	44	52
6	Senior	Media	330	358
6	Senior	Alta	430	447
7	Senior	Baja	12	41
7	Senior	Media	183	320
7	Senior	Alta	259	428
9	Senior	Baja	21	56
9	Senior	Media	203	337
9	Senior	Alta	390	408
11	Senior	Baja	34	57
11	Senior	Media	223	326
11	Senior	Alta	353	486

2) Validación de los resultados con la prueba Wilcoxon Pareada

La prueba de Wilcoxon se utilizó para comparar la estimación con IA versus la estimación sin IA, en los 10 profesionales (9 desarrolladores y 1 experto) que utilizan asistentes como Claudecode, con 30 pares de datos analizados, el estadístico arrojó una significancia extrema ($p < 0.0001$). En el 100% de los casos evaluados, el uso de IA resultó en una reducción de horas de desarrollo, esto confirma que la IA no es solo un apoyo marginal, sino un catalizador de eficiencia sistemático. No se registró ningún escenario donde la IA aumentara el tiempo de ejecución, validando la adopción de herramientas como Claudecode como una estrategia de optimización real para la empresa. La Tabla 2, ilustra el resultado de la prueba.

Tabla 2*Resultado prueba Wilcoxon para los 30 pares de estimaciones*

	Estimación con IA (Horas)	Estimación sin IA (Horas)
median	215	329
count	30	
# unequal	30	
T+	0	
T-	465	
T	0	
	one tail	two tail
mean	232.5	
std dev	48.61326979	ties
z-score	4.772359502	yates
effect r	0.871309651	
p-norm	0.000000910	0.000001821
p-exact	0.000000001	0.000000002
p-simul	N/A	N/A

3) Validación de los resultados con la prueba Kruskal-Wallis

La prueba se utilizó para examinar si la categorización de los casos de uso (**Baja, Media y Alta**) es un factor que permite determinar si las diferencias de tiempo entre los tres niveles de dificultad son estadísticamente significativas en los dos escenarios, tanto en el método tradicional como con IA. El resultado en tradicional o sin IA fue: $H=31.13$, $p<0.0001$ y con IA: $H=23.69$, $p<0.0001$. En ambos casos la significancia de la prueba fue contundente y demuestra que la complejidad técnica es la variable independiente más robusta de la investigación. Incluso cuando la IA reduce el tiempo total (validado por Wilcoxon), la jerarquía de esfuerzo se mantiene intacta. Un caso de alta complejidad sigue requiriendo significativamente más tiempo que uno de baja, lo que demuestra que la IA optimiza el proceso, pero no elimina la brecha impuesta por la complejidad técnica.

Por otra parte, la misma prueba se utilizó para estudiar si el nivel de experiencia (**Junior, Senior y Master**) del desarrollador influye en la velocidad de entrega de la empresa o si la IA beneficia más a un nivel de experiencia que a otro. El resultado en tradicional o sin IA fue: $p=0.5741$ y con IA: $p=0.7371$. Lo cual determina para ambos casos que no es estadísticamente significativo por ser $p>0.05$, demostrando que el nivel de experiencia por si solo no genera variaciones críticas en el tiempo estimado para estos casos de uso y que la IA no aumenta la productividad en mayor medida en un nivel de experiencia particular.

4) Factores determinantes

De acuerdo con las encuestas realizadas al equipo de desarrollo y la entrevista a los expertos que lideran el equipo que tienen conocimiento en el manejo de la inteligencia artificial, se identifican diferencias significativas entre el desarrollo tradicional y el desarrollo asistido por IA que no son exclusivamente de la herramienta, si no de una serie de factores estructurales, técnicos y humanos.

Por lo anterior se establecen que los factores determinantes para el resultado obtenido con IA y con tradicional, son:

1. **Contexto y Claridad de las Reglas de Negocio:** Es el factor más crítico ya que la IA se convierte en un aliado real en tareas repetitivas o que no requieren de un contexto masivo del producto ya construido, no obstante, se transforma en un obstáculo cuando carece de entrenamiento adecuado o del contexto requerido para ejecutar actividades.
2. **Arquitectura del Software:** La estructura del sistema preexistente determina la velocidad de integración del código generado. Al no estar estandarizados, divididos en partes pequeñas e independientes, con servicios no desacoplados y capas que no se encuentran bien definidas, la IA no va a ser eficiente ya que puede generar componentes no aislados con alto riesgo. Además, en sistemas con lógica

- distribuida en triggers, procedimientos almacenados o componentes altamente dependientes, la integración de IA es más lenta y requiere mayor análisis humano, disminuyendo el ahorro de tiempo, como sucede en la empresa.
3. **Complejidad y el Análisis:** Paradoja en las tareas de alta complejidad ya que ahorro de tiempo en requerimientos complejos es proporcionalmente menor debido a que el esfuerzo principal no es escribir el código, sino comprender el negocio, analizar dependencias y validar escenarios. Mientras que en requerimientos simples la IA reduce significativamente el tiempo, por lo que en las complejas requiere de intervención humana para evitar errores no previstos, por ello evidencia que nivela los tiempos respecto al desarrollo tradicional.
 4. **Evolución de las Habilidades.** El desarrollo con IA desplaza el enfoque de las habilidades técnicas tradicionales, ya que hay que enfocarse en la Ingeniería de Prompts porque el éxito con IA depende de la capacidad de estructurar prompts con contexto completo, roles definidos, restricciones y ejemplos. Igualmente, el conocimiento profundo de la sintaxis pierde relevancia frente a la capacidad de diseñar arquitecturas modulares y comprender procesos de negocio, por lo que el desarrollador moderno debe ser un auditor de código más que un simple escritor de líneas.
 5. **Seguridad y Supervisión:** La inteligencia artificial puede generar errores o vulnerabilidades en el desarrollo, por lo que requiere revisiones, pruebas y herramientas de análisis para garantizar la seguridad del código. La confianza en los resultados no es automática y depende de validaciones técnicas que aseguren el cumplimiento de los requisitos.

5) Habilidades determinantes que deben adquirir los desarrolladores para obtener ventaja competitiva de IA.

Basado en los resultados obtenidos a través de las entrevistas con los expertos en manejo de IA, se determina las siguientes habilidades:

1. **Ingeniería de Prompts Avanzada:** Dominar el uso de **contexto** detallado, delimitadores, ejemplos concretos y la definición explícita de roles (como arquitecto o analista) para obtener resultados de alta calidad.
2. **Diseño de Arquitectura Modular:** Desarrollar la capacidad de diseñar e implementar sistemas con bajo acoplamiento y alta cohesión, ya que las arquitecturas modulares permiten que la IA genere componentes aislados de forma más eficiente y con menor riesgo.
3. **Análisis de Reglas de Negocio y Alcance:** Definir con precisión los objetivos, casos de uso y procesos de negocio, ya que la IA se vuelve un obstáculo si carece de reglas claras o de un alcance bien acotado.
4. **Auditoría y Validación de Código (Detección de Alucinaciones):** Capacitarse y cursar como un supervisor crítico, capaz de identificar errores lógicos y "alucinaciones" (asumir escenarios no especificados), verificando que el código cumpla exactamente con los requerimientos.
5. **Seguridad y Desarrollo Seguro:** Adquirir habilidades en auditoría técnica y explorar herramientas para detectar vulnerabilidades como inyección SQL o exposición de credenciales que la IA podría introducir inadvertidamente.
6. **Automatización de Pruebas:** Dominar la automatización de pruebas unitarias, de integración y funcionales, ya que es el mecanismo principal para garantizar que el código generado sea correcto y no afecte procesos de misión crítica.

7. **Pensamiento Estratégico y Gestión de Riesgos:** Desarrollar la identificación de dependencias, riesgos de escalabilidad y el análisis de impacto de los cambios en el sistema global, más que en la escritura manual de líneas de código.
8. **Gestión de Contexto y Dependencias:** Desarrollar la habilidad de saber de cómo y cuándo suministrar a la IA el dominio del negocio, la arquitectura existente y las restricciones del sistema, ya que va a determinar si la herramienta será un aliado real o un generador de reprocesos.
9. **Gobernanza de Datos y Privacidad:** Capacitarse en técnicas de anonimización de información y seguir políticas de privacidad para utilizar herramientas en la nube sin vulnerar los derechos de propiedad intelectual o la confidencialidad de la empresa.
10. **Pensamiento Innovador y Curiosidad Intelectual:** Utilizar la IA no solo para repetir patrones, sino como una herramienta de lluvia de ideas y exploración de alternativas arquitectónicas, retando las soluciones tradicionales para potenciar la creatividad.

Conclusiones

1. La integración de asistentes de Inteligencia Artificial en el ciclo de desarrollo de software genera una optimización disruptiva y sistemática de la productividad. Se evidencia una Eficiencia en Tiempo Proyectada (ETP) general del 26.76%. La Prueba de Wilcoxon ratifica que esta mejora no es aleatoria ni subjetiva, ya que en el 100% de los casos analizados el uso de IA redujo los tiempos de desarrollo frente al método tradicional.
2. La complejidad técnica del requerimiento es la variable más influyente en la carga horaria, independientemente de la tecnología utilizada (Tradicional o con IA), y demuestra que la clasificación de casos de uso es una metodología de planeación robusta.

3. La Inteligencia Artificial actúa como un estándar nivelador de productividad dentro de la organización, permitiendo que perfiles con menor trayectoria alcancen ritmos de entrega competitivos, siempre que el requerimiento esté bien delimitado por su nivel de complejidad.
4. El factor determinante para que la IA supere al desarrollo tradicional no es solo la velocidad de generación de texto, sino la capacidad del equipo para proveer contexto, mantener una arquitectura limpia y ejercer una supervisión técnica rigurosa basada en estándares de calidad y seguridad.
5. El ahorro de tiempo que ofrece la IA es significativo en tareas simples, pero en la alta complejidad, la ventaja competitiva reside en quien domina el análisis, la supervisión técnica y la integración estratégica del código.
6. para optimizar el desarrollo de software en Smart TMT, es más efectivo centrar la planificación en la complejidad técnica del requerimiento que en el perfil del desarrollador, siempre que se garantice el uso de asistentes de IA como catalizadores de la productividad individual

Recomendaciones

Como recomendación derivada del presente trabajo investigativo, se sugiere lo siguiente:

1. Incorporar un área formal estructurada dentro del proceso de construcción y gestión de requerimientos, cuya responsabilidad principal sea capacitar, promover, fortalecer y supervisar la adopción de herramientas de IA en las actividades de desarrollo de software, a partir de las siguientes consideraciones:
 - a. Capacitación en Ingeniería de Prompts Avanzada: Formar al equipo en el uso de estructuras de prompts que incluyan contexto completo, definición de roles específicos (arquitecto o analista funcional), objetivos, restricciones y el uso de delimitadores para reducir ambigüedades.

- b. Priorizar la claridad en las Reglas de Negocio: Antes de iniciar la utilización de IA formalmente, es fundamental que los requerimientos tengan definiciones, reglas y alcances extremadamente claros, ya que de esto depende que la eficiencia aumente y se eviten reprocesos.
- c. Implementar un Marco de Validación Obligatorio: No confiar a ciegas en el código generado; es importante establecer protocolos que incluyan pruebas unitarias, de integración y funcionales como mecanismo estándar para garantizar que el resultado cumpla con lo solicitado de los requerimientos estimados.
- d. Integrar Auditoría de Seguridad (DevSecOps): Adoptar herramientas de inspección como Sonar y aplicar estándares de desarrollo seguro (OWASP) para detectar automáticamente vulnerabilidades o patrones inseguros que la IA pueda introducir.
- e. Establecer Políticas de Privacidad y Gobernanza de Datos: Para mitigar el riesgo en el uso de herramientas en la nube, es necesario capacitar al personal en técnicas de anonimización de información y el uso de datos simulados, asegurando que no se comparta información sensible o código propio crítico.
- f. Fomentar el Acompañamiento Técnico Práctico: La formación no debe ser solo teórica sino sesiones guiadas y mentorías aplicadas a proyectos reales de la empresa con el fin que el equipo adopte la IA con mayor confianza y aprenda a manejar casos de alta complejidad.
- g. Alineación con Estándares Internos: Hay que fortalecer los modelos de IA mediante la creación de manuales de estilo (guía compuesta por un conjunto de criterios preceptuados) y bibliotecas de componentes propios de la empresa, asegurando que el código generado se alinee con la nomenclatura implementada con patrones de diseño específicos.
- h. Segmentación de Tareas Complejas: En tareas de alta complejidad el ahorro de tiempo es menor debido al esfuerzo de análisis, se recomienda utilizar la IA para la

desagregación o segmentación de problemas grandes en partes pequeñas con alcances acotados.

- i. Supervisión Humana en Procesos Críticos: Establecer niveles de revisión diferenciados según la criticidad de los componentes que afecten procesos transaccionales o de misión crítica deben tener una supervisión humana mucho más estricta que las tareas simples o repetitivas.

Con ello, la implementación de esta unidad permitiría establecer lineamientos, metodologías y estándares para el uso adecuado de tecnologías basadas en IA, favoreciendo una integración estructurada dentro de los procesos organizacionales.

2. El código generado mediante herramientas de inteligencia artificial no debe ser considerado como un producto final autónomo, sino como un recurso de apoyo sujeto a procesos de revisión, validación y seguimiento por parte del desarrollador responsable. Esta supervisión permitirá garantizar la calidad del producto desarrollado, reducir posibles riesgos asociados a errores técnicos, vulnerabilidades de seguridad o inconsistencias funcionales, y asegurar el cumplimiento de los requerimientos y objetivos establecidos en los diferentes proyectos de la empresa. De esta manera, la combinación entre las capacidades de la inteligencia artificial y el criterio técnico humano puede contribuir al fortalecimiento de los procesos de desarrollo y a la optimización de los resultados obtenidos.

Anexos

Anexo A. Matriz Horas Construcción de software de desarrollo - Ficha Técnica.

Anexo B. V de Aiken.

Anexo C. Encuesta Desarrolladores Estimación Casos de Uso

Anexo D. Entrevista sobre el uso de Inteligencia Artificial en el desarrollo de software

Anexo E. Descripción tres casos de uso

Anexo F. Estimación aprobada juicio de experto

Anexo G. Análisis resultados

Referencias

- Dohmke, T., Iansiti, M., & Richards, G. (2023). *Sea Change in Software Development: Economic and Productivity Analysis of the AI-Powered Developer Lifecycle*. Keystone. Obtenido de <https://github.blog/news-insights/research/the-economic-impact-of-the-ai-powered-developer-lifecycle-and-lessons-from-github-copilot/>
- Erazo, A., Ramos, F., Galarza, P., & Boné, M. (2023). La inteligencia artificial aplicada a la optimización de programas informáticos. *Journal of Economic and Social Science Research*, 3.
- Kanovich, R. (2025). *Software Enterprise: The Changing Ecosystem*. Artisan Partners.
- Korn Ferry. (2018). *Future of Work : The Global Talent Crunch*. Korn Ferry. Korn Ferry. Obtenido de Korn Ferry: <https://www.kornferry.com/>
- Llerena, L., Viscaino, F., Culque, W., & Baño, F. (2023). Estudio neutrosófico sobre la afectación de la inteligencia artificial en el desarrollo de software. *Neutrosophic Computing and Machine Learning*, 29.
- Morales, J. J., & Pardo, C. J. (2016). Revisión sistemática de la integración de modelos de desarrollo de software dirigido por modelos y metodologías ágiles. *Informador Técnico*.
- Pressman, R., & Maxim, B. (2021). *Ingeniería de Software: Un enfoque práctico* (Novena ed.). MC Graw Hill.
- Salazar, P., & Cárdenas, J. (2024). La evolución de la inteligencia artificial y su impacto en el desarrollo de software. *Polo del Conocimiento*.
- Sarco, W., & Toaquiza, D. (2022). APLICACIÓN DE LA INTELIGENCIA ARTIFICIAL EN EL DESARROLLO DE SOFTWARE. *Universidad Técnica de Cotopaxi*.
- SMART TMT. (2026). *Smart TMT*. Obtenido de <https://smarttmt.com/>

Statista Research Department. (2025). *Number of software developers worldwide in 2018 to 2024*. Statista Research Department;. Obtenido de

<https://www.statista.com/statistics/627312/worldwide-developer-population/>

Taulli, T. (2024). *AI-Assisted Programming: Better Planning, Coding, Testing, and Deployment*.

O'Reilly Media, Inc.

Turing, A. (1950). Computing machinery and intelligence. *Mind*.