Talento en la era de la IA

Gestión del cambio para la muy necesaria adopción de IA en las organizaciones

Por jpelayo · 23 de febrero de 2026

	Nombre	Qué cambia
0	Autocompletado Extremo	La IA sugiere la siguiente línea. Aceptas o rechazas. Solo una tecla tab más rápida. El humano sigue siendo quien escribe el software.
1	Becario de Código	Le das a la IA una tarea acotada: una función, un componente, un refactor. La IA ejecuta la tarea. El humano sigue siendo dueño de la arquitectura, el juicio y la integración.
2	Desarrollador Junior	La IA navega el código, maneja cambios en múltiples archivos, construye funcionalidades entre módulos. El humano sigue leyendo todo el código.
3	Desarrollador como Jefe	El humano deja de escribir y empieza a dirigir. Revisas a nivel de feature y de PR. La mayoría topa aquí porque soltar el código es psicológicamente difícil.
4	Desarrollador como Product Manager	Escribes una especificación, te vas, vuelves horas después y compruebas si los tests pasan. Dejas de leer código. Solo evalúas resultados.
5	La Fábrica Oscura	Entran especificaciones, sale software funcional. Ningún humano escribe o revisa código. La fábrica funciona con las luces apagadas. Aquí es hacia donde va la industria.

El core de una empresa son su talento y sus procesos. Una gran parte del peso de la gestión de procesos lo acarrean los sistemas desplegados y mantenidos por el departamento de IT, que en muchas ocasiones internalizan el desarrollo de software para mantener el sistema de negocio, adaptarlo o mejorarlo. Sus procesos departamentales han evolucionado desde hace 50 años con saltos más grandes o más pequeños, pero el que viene ahora es absurdamente enorme. La mayoría de las empresas no están dándose cuenta del orden de magnitud del cambio que viene con el desarrollo de código mediante IA. Y no es un cambio para mal, porque va a dar más protagonismo al core de la empresa (sus procesos), pero va a alterar la definición de talento. La empresa que haga hoy la apuesta correcta, mejorará sus perspectivas a largo plazo. Quien haga la apuesta perdedora o meta la cabeza bajo tierra, se va a hundir arrastrada por una parte de su organización que no va a ser competitiva.

El bucle se ha cerrado y va muy deprisa

El bucle autorreferencial se ha instalado tanto en Anthropic como en OpenAI. Codex 5.3 es el primer modelo de vanguardia que fue decisivo para crearse a sí mismo, y no es una metáfora. Se lanzó como producto directo del trabajo de código de su predecesor. OpenAI informó de una mejora de velocidad del 25% y un 93% menos de tokens desperdiciados, ganancias que el modelo encontró al analizar sus propias ineficiencias durante el proceso de entrenamiento.

Claude Code hace lo mismo. El 90% de su propio código fue escrito por Claude Code, un número que converge hacia el 100%. Boris Cherny, quien lo lidera, dice que dejó de escribir código hace meses. Su rol se desplazó a especificación, dirección y criterio. En Anthropic todos están definiendo arquitectura y las máquinas están escribiendo.

El bucle de retroalimentación se ha cerrado. La pregunta ya no es si la IA mejorará la IA, sino la aceleración de ese bucle; y qué significa para los cincuenta millones de personas que actualmente escriben software para ganarse el pan.

La curva en J

Sin embargo, aquí es donde estamos: la mayoría de las organizaciones no están viendo ganancias sino que se están volviendo más lentas. Los desarrolladores que intentan añadir IA a flujos de trabajo existentes se pasan el rato evaluando sugerencias, corrigiendo código casi-casi correcto, cambiando de contexto entre su modelo mental y el

del modelo, y depurando errores sutiles que parecen correctos-pero-no. El 46% de los desarrolladores en encuestas más amplias dicen no confiar plenamente en el código generado por IA.

Cuando añades una nueva herramienta sin rediseñar el flujo de trabajo a su alrededor, estás poniendo un motor nuevo en una transmisión vieja. La productividad cae antes de mejorar, a veces durante meses. La mayoría de las organizaciones están en ese punto ahora mismo, interpretando la desaceleración como evidencia de que la IA no funciona.

Copilot es la ilustración más clara. Veinte millones de usuarios, 42% de cuota de mercado, y estudios de laboratorio que muestran un 55% más de velocidad en completar código en tareas aisladas. En producción... la historia es más complicada: pull requests más grandes, mayores costes de revisión, más vulnerabilidades de seguridad. Las organizaciones que ven ganancias del 25 al 30% son las que rediseñaron todo su proceso de desarrollo, cómo escriben specs, revisan código, estructuran el CI/CD.

El verdadero cuello de botella

Cada ceremonia en una organización de software existe por una limitación humana. Las standups sincronizan a personas que comparten un código. La planificación de sprint gestiona la memoria de trabajo humana. La revisión de código atrapa errores humanos. El QA existe porque los constructores no pueden evaluar su propio trabajo objetivamente. Cuando el humano ya no escribe el código, nada de esto desaparece con elegancia. Se convierte en fricción.

El cambio estructural es más difícil de ver que el cambio tecnológico. El valor del manager se desplaza de coordinar al equipo a definir la especificación con suficiente claridad para que los agentes construyan la funcionalidad. Escribir una especificación lo suficientemente precisa para un agente de IA es una habilidad genuinamente distinta. Las máquinas construyen lo-que-has-descrito. Si lo que has descrito es ambiguo, aparece relleno en los huecos. El cuello de botella ha pasado de la velocidad de implementación a la calidad de la especificación.

Para otra ocasión queda hablar sobre la capacidad media del personal técnico de articular verbalmente mensajes complejos, y de su expresión escrita, gramatical y ortográficamente correcta. Por ahora solo esto: lean más.

La técnica de escenarios

Los tests tradicionales viven dentro del código. El agente de IA puede leerlos, lo que significa que puede optimizarse para superarlos en lugar de construir código fuente adecuado a la función objetivo: es el clásico problema de estudiar para el examen. Notas perfectas, comprensión superficial.

Hay que usar escenarios. Los escenarios viven fuera del código, almacenados por separado para que el agente nunca los vea. Funcionan como un holdout set, el mismo concepto que en machine learning se usa para prevenir el sobreajuste. El agente construye el software; los escenarios evalúan si realmente funciona. El agente nunca ve los criterios de evaluación. No puede manipular el sistema.

A un volumen de trabajo adecuado, los agentes de IA operan a una escala donde el coste de cómputo se vuelve significativo, y aun así suele ser más barato que los humanos a los que reemplaza: si no has gastado 1000 dólares en computación por ingeniero al día, tu organización tiene margen de mejora.

Los procesos

La mayoría del software empresarial son sistemas existentes vegetantes a lo largo de los años, ejecutándose en producción, sosteniendo ingresos. Monolitos recrecidos durante quince años de añadir funcionalidades. Gestión de configuración que vive en la cabeza de tres personas que recuerdan por qué tal variable de entorno tiene ese numerito.

No puedes aplicar la "fábrica oscura" a un sistema legacy. La especificación no existe. Los tests, si los hay, cubren el 30% del código; el otro 70% funciona sobre tradición oral. El sistema es la especificación y la única descripción completa de lo que hace el software.

El camino empieza por documentar. Hacer ingeniería inversa del conocimiento implícito en un sistema en ejecución requiere experiencia de dominio, honestidad sin filtros,

pensamiento sistémico: exactamente las capacidades humanas que importan más en la era de la "fábrica oscura", no menos.

La migración tiene este aspecto: primero, usa la IA en nivel 2 o 3 para acelerar el trabajo existente. Segundo, usa la IA para documentar lo que tu sistema realmente hace. Tercero, rediseña el pipeline de validación y despliegue. Cuarto, comienza a trasladar el desarrollo nuevo a agentes autónomos. Ese camino lleva tiempo. Quien te diga lo contrario te está vendiendo algo.

El talento

La ingeniería de software siempre ha sido un modelo de aprendizaje vestido de ropa empresarial. Los juniors aprenden haciendo: funcionalidades simples, bugs pequeños, inmersión en el código. Los seniors mentorizan, revisan, atrapan errores. En cinco a siete años un junior se convierte en senior a través de experiencia acumulada. La IA rompe ese modelo por abajo. Si la IA maneja las funcionalidades simples y los bugfixes, ¿dónde aprenden los juniors? Si la IA revisa código más rápido que un senior, ¿dónde ocurre la mentorización?

Y sin embargo necesitamos más ingenieros de elite, no menos. El nivel mínimo requerido está subiendo hacia exactamente las habilidades que siempre han sido más difíciles de ganarse. El junior de 2026 necesita el pensamiento sistémico que se esperaba de un ingeniero de nivel medio en 2020, no porque el trabajo se haya vuelto más difícil, sino porque el trabajo de nivel básico se automatizó.

Lo que importa ahora es: pensamiento sistémico, intuición de cliente, capacidad de tener todo un producto/proyecto/programa en la cabeza, y capacidad de escribir una especificación lo suficientemente precisa para que un agente autónomo la implemente correctamente. Esas habilidades siempre han separado a los grandes ingenieros de los mediocres. La diferencia es que "adecuado" ya no es una posición viable a ningún nivel, porque "adecuado" es lo que hacen los modelos de IA.

La demanda

Cada vez que el coste de computación ha caído, de los mainframes a los PCs, de los PCs a la nube, de la nube al serverless, el total de software producido no se ha mantenido plano: ha explotado. Han surgido nuevas categorías que antes eran económicamente imposibles: SaaS, móvil, streaming, analítica en tiempo real.

Estamos bajando el coste de producción de software en un orden de magnitud. Un hospital regional, un fabricante de tamaño medio, una empresa familiar de logística no pueden permitirse software a medida a los costes laborales actuales. Un sistema de inventario personalizado puede costar 500.000 dólares y tardar más de un año. Esa necesidad no cubierta es ahora abordable.

Aunque no es ningún consuelo para quienes sufren la pérdida de empleo, sepan que la demanda de software nunca se ha saturado. La restricción siempre se traslada al siguiente cuello de botella. Aquí, ese cuello de botella es el criterio: saber qué construir y para quién. La "fábrica oscura" no reemplaza a las personas más difíciles de reemplazar. Las amplifica.

Lo que viene

La "fábrica oscura" es real. Un pequeño número de equipos produce software sin que ningún humano escriba o revise código, entregando resultados listos para producción que mejoran con cada nueva generación de modelos. El bucle de retroalimentación está cerrado y esos equipos se mueven cada vez más rápido con cada iteración.

La mayoría de las empresas están atascadas en el nivel 2, volviéndose mediblemente más lentas con herramientas que creen que las están haciendo más rápidas. La vanguardia está más adelante de lo que casi nadie quiere admitir. El centro está más atrás de lo que los equipos de vanguardia reconocen. La distancia entre ellos no es una brecha tecnológica. Es una brecha de personas, de cultura, de organización, de voluntad de cambiar.

Las empresas que sobreviven no son las que compran la mejor herramienta de código. Son las que hacen el trabajo duro, lento y poco glamuroso de documentar lo que hacen sus sistemas, reconstruir sus organizaciones en torno al criterio en lugar de la coordinación, y son suficientemente humildes y honestas consigo mismas para saber que esta transición no ocurrirá tan rápido como quieren, porque las personas cambian despacio.

La "fábrica oscura" no necesita más ingenieros. Necesita mejores. Mejor significa personas capaces de pensar con claridad sobre lo que debería existir, describirlo con suficiente precisión para que las máquinas lo construyan, y evaluar si lo que se ha decidido construir realmente sirve a humanos reales. Eso siempre ha sido la parte difícil de la ingeniería de software. Solo hemos, durante años, dejado que la complejidad de la implementación ocultara cuánta gente era realmente buena en ello. Las máquinas han

retirado esa cobertura. Estamos a punto de descubrir la meritocracia real, el valor del auténtico talento.

Crédito y agradecimientos: Nathan Jones