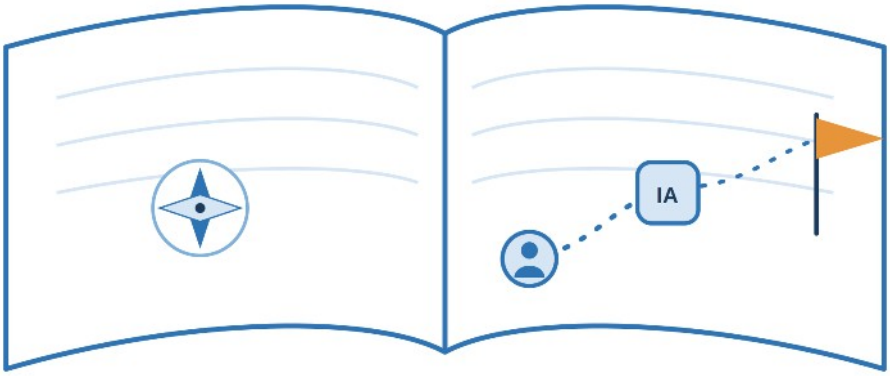
Scrum Manager
Uncovering better ways

Guía

Scrum en equipos con IA

Desarrollo de los temas del State of the art



Actualizado a junio de 2026

Scrum Manager® · Skill Arena

Sobre este documento

Este documento desarrolla en profundidad los temas del mapa de referencia (State of the art) para liderar equipos ágiles donde humanos y agentes de IA trabajan juntos, que está disponible en: Skill Arena.

Úsalo como material de consulta para mantener y contrastar tu práctica con el conocimiento profesional actual.

Es un documento formativo, no normativo. No pretende definir un «nuevo scrum» ni sustituir las prácticas que cada equipo haya adoptado. Su idea central: los principios ágiles son una brújula que no pierde su norte con la IA, pero las prácticas concretas necesitan adaptarse. Para cada tema se presentan opciones, evidencias y riesgos, respetando la autonomía de cada equipo.

Se complementa con la plataforma de entrenamiento y evaluación en Skill Arena. En el área Scrum en equipos con IA puedes realizar pruebas de entrenamiento para contrastar y mejorar tu nivel de conocimiento y si lo deseas también puedes obtener un diploma que acredita curricularmente la solvencia y vanguardia profesional en esta área.

Cada capítulo desarrolla uno de los temas del mapa, indica su estado de adopción y se cierra con las capacidades que un profesional debe ser capaz de demostrar, los errores frecuentes y referencias para profundizar.

Estado del conocimiento

El conocimiento sobre Scrum en equipos con IA evoluciona de forma extremadamente rápida: aunque los principios ágiles que lo sostienen permanecen estables, las prácticas concretas para integrar humanos y agentes se reinventan en cuestión de meses y solo una parte puede considerarse asentada. En los distintos apartados del documento, las etiquetas (ESTABLECIDO, EN CONSOLIDACIÓN, EMERGENTE) ayudan a identificar la madurez de cada concepto:

ESTABLECIDO consenso asentado; conocimiento que se da por necesario.

EN CONSOLIDACIÓN gana adopción con rapidez; aún no universal pero ya relevante.

EMERGENTE frontera reciente; alta relevancia y alta volatilidad.

Índice

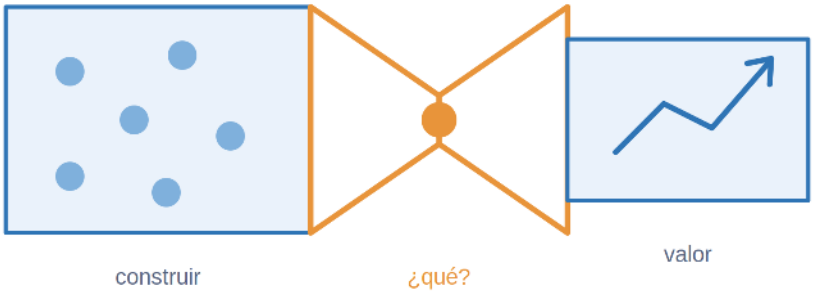
Capítulos del desarrollo, en el orden del State of the art.

BLOQUE A · EL CAMBIO DE PARADIGMA

Capítulo 1. El desplazamiento del cuello de botella a la gestión de producto

ESTABLECIDO

La capacidad de construir ha dejado de ser el factor limitante; decidir qué construir lo es.



El cuello de botella se desplaza de construir a decidir qué merece construirse.

Introducción

Durante décadas, la ingeniería fue el recurso escaso de los equipos de desarrollo. Las metodologías ágiles se optimizaron para maximizar el rendimiento del equipo que construía, porque construir era lo caro y lo lento. La IA generativa ha alterado esa premisa básica: cuando un equipo pequeño puede generar en horas lo que antes requería semanas, la construcción deja de ser el cuello de botella y la restricción se traslada a otra parte.

1.1 Dónde está ahora la restricción

El nuevo factor limitante es la gestión de producto: decidir qué merece construirse, validarlo con usuarios y mantener una visión estratégica coherente. La pregunta que define el trabajo cambia de «¿qué podemos construir?» —que antes acotaba la capacidad del equipo— a «¿qué deberíamos construir, y por qué?». Cuando la IA permite crear features más rápido de lo que la organización puede validar su deseabilidad, la calidad de la decisión sobre qué construir importa más que la velocidad de construcción.

1.2 Implicaciones para cada rol

El desplazamiento reconfigura el trabajo de todo el equipo: el product owner se vuelve más estratégico y menos táctico; el desarrollador desplaza su valor de la implementación a la orquestación y la revisión; el scrum master mueve su foco de facilitar eventos a facilitar la reflexión sobre qué y por qué se construye. Ninguno pierde relevancia; todos cambian de centro de gravedad.

El coche de carreras sin mapa. Un equipo que solo optimiza la velocidad de construcción es como un coche de carreras en un circuito sin mapa: puede ir muy rápido, pero eso no garantiza que llegue a la meta. El mapa —la gestión de producto— es ahora lo escaso.

Lo que debes saber hacer

Al dominar este tema, un profesional es capaz de:

- Explicar por qué la IA desplaza el cuello de botella de la construcción a la gestión de producto.
- Reformular la pregunta rectora del trabajo de «¿qué podemos construir?» a «¿qué deberíamos construir, y por qué?».

- Anticipar cómo el desplazamiento cambia el centro de gravedad de cada rol del equipo.
- Identificar dónde está el cuello de botella real de su propio equipo y si se ha desplazado.

Errores y antipatrones frecuentes

Seguir optimizando solo la velocidad de construcción. Es resolver un cuello de botella que ya no es el limitante; la mejora marginal de producir más rápido se desperdicia si no se valida.

Confundir más capacidad de construir con más valor. La capacidad de construir multiplicada no entrega valor por sí sola; necesita la guía de producto.

No reasignar el esfuerzo liberado. Si la IA libera capacidad de construcción y el equipo la dedica a construir aún más, agrava el problema en vez de resolverlo.

Para profundizar

· Scrum Guide Expansion Pack — AI and Scrum

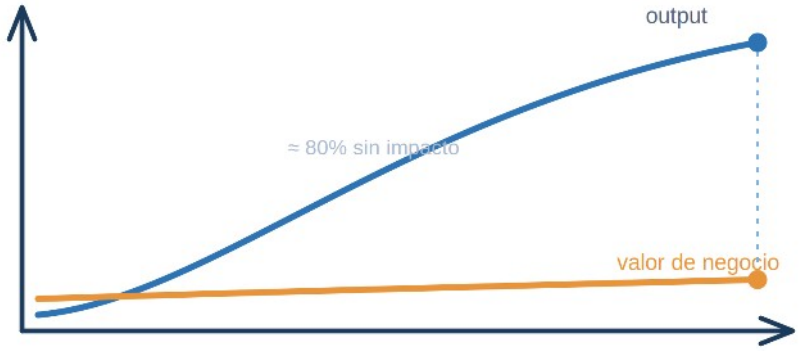
· Scrum Manager — Scrum en la era de la IA

BLOQUE A · EL CAMBIO DE PARADIGMA

Capítulo 2. La paradoja de la productividad

ESTABLECIDO

La mayoría de organizaciones que usan IA no observa impacto en sus resultados de negocio.



Más output no es más valor: la producción sube y el impacto de negocio apenas se mueve.

Introducción

Uno de los datos más reveladores del estado actual es la llamada «paradoja de la IA generativa»: aunque alrededor del 78 % de las empresas usaban IA en 2025 y la mayoría de los desarrolladores declaraban usar herramientas de codificación asistida, cerca del 80 % de las organizaciones reportaron no observar un impacto significativo en sus resultados de negocio. ¿Cómo es posible que tanta adopción genere tan poco impacto?

2.1 El diagnóstico

La brecha entre adopción y resultados se atribuye a un fenómeno concreto: los equipos producen funcionalidades a mayor velocidad, pero sin la guía de producto, la validación con usuarios ni la visión estratégica necesarias para que esa velocidad genere valor real. Producir más no es entregar más valor. La agilidad nunca fue velocidad; es la capacidad de responder eficazmente al cambio mientras se entrega valor.

2.2 Cómo superan la paradoja los equipos que lo logran

Los equipos que escapan de la paradoja comparten tres rasgos:

- **Mantienen la validación al ritmo de la producción.** Si pueden generar diez features por sprint, se aseguran de poder validar diez por sprint.
- **Invierten más en descubrimiento.** Como construir es más barato, la proporción de esfuerzo se desplaza hacia entender qué construir.
- **Miden resultados, no outputs.** No celebran cuántas features entregaron, sino qué impacto tuvieron en usuarios y negocio.

Lo que debes saber hacer

Al dominar este tema, un profesional es capaz de:

- Explicar la paradoja de la productividad y su causa principal.
- Reconocer las señales de que un equipo está atrapado en ella (backlog que crece sin validar, métricas de impacto planas).
- Aplicar los tres rasgos de los equipos que la superan a su propio contexto.

- Distinguir entre medir outputs y medir resultados.

Errores y antipatrones frecuentes

Celebrar el volumen de features entregadas. El número de features no dice nada sobre el valor; puede crecer mientras el impacto no cambia.

Reducir el discovery para aumentar el delivery. Es exactamente el movimiento que produce la paradoja: más construcción sin más comprensión.

Asumir que adoptar IA mejora los resultados por sí solo. La adopción sin transformación de la gestión es lo que explica el 80 % sin impacto.

Para profundizar

· Scrum Guide Expansion Pack — AI and Scrum

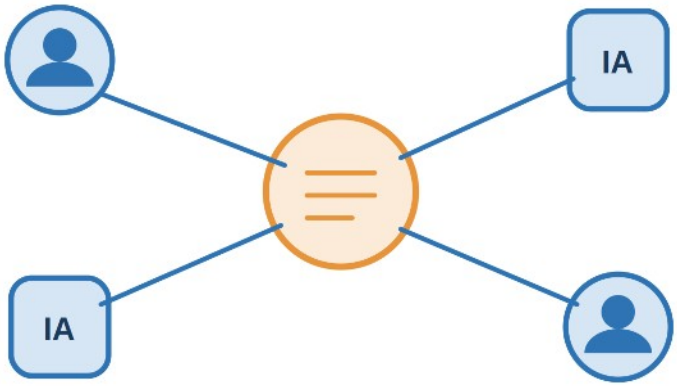
· Harvard / D3 — The Cybernetic Teammate

BLOQUE A · EL CAMBIO DE PARADIGMA

Capítulo 3. Equipos híbridos humano-IA

EN CONSOLIDACIÓN

El trabajo deja de ser una interacción exclusivamente entre humanos.



Humanos y agentes de IA coproducen: el equipo deja de ser solo humano.

Introducción

Los agentes de IA ya participan en la creación de código, la generación de tests, la revisión de arquitectura y la redacción de documentación. El trabajo del equipo deja de ser una interacción exclusivamente entre humanos, y las prácticas de gestión deben contemplar esta nueva realidad. La pregunta de fondo —¿es la IA un miembro del equipo, una herramienta o un colaborador externo? — no tiene respuesta definitiva, pero sí implicaciones prácticas inmediatas.

3.1 La democratización de la capacidad técnica

Un perfil de negocio puede hoy generar prototipos funcionales sin intervención directa de ingeniería. El product owner visualiza ideas sin esperar a que el equipo las implemente; el diseñador genera código básico; el stakeholder crea mockups funcionales. Las barreras entre roles se difuminan y las dependencias tradicionales se reducen. Pero emerge un riesgo: más personas pueden crear más cosas, lo que amplifica la necesidad de validación y gobernanza.

3.2 Decisiones que cada equipo debe tomar

Independientemente de la postura conceptual sobre qué es la IA, todo equipo híbrido debe definir explícitamente cuatro cosas: qué puede y qué no puede hacer la IA en este equipo, quién es responsable de revisar lo que produce, cómo se integra en el flujo de trabajo diario, y qué límites se le ponen. Estas decisiones son parte de la autoorganización del equipo, no de la herramienta.

La responsabilidad recae siempre en un humano. Sea cual sea el papel que el equipo asigne a la IA, cuando un agente comete un error la rendición de cuentas no puede diluirse: debe haber claridad sobre qué persona responde de lo que la IA produce.

Lo que debes saber hacer

Al dominar este tema, un profesional es capaz de:

- Describir cómo la democratización técnica difumina las barreras entre roles y qué riesgo introduce.

- Enumerar las cuatro decisiones explícitas que todo equipo híbrido debe tomar sobre la IA.
- Justificar por qué la responsabilidad del resultado recae siempre en un humano.
- Reconocer el equipo híbrido como objeto de la autoorganización, no como configuración de la herramienta.

Errores y antipatrones frecuentes

Dejar indefinido el papel de la IA. Sin acuerdo explícito sobre qué hace y quién la revisa, aparecen tareas que nadie supervisa: zonas de riesgo.

Diluir la responsabilidad en «lo hizo la IA». La rendición de cuentas debe recaer en una persona identificable.

Tratar la democratización como ausencia de gobernanza. Que más personas puedan construir aumenta, no reduce, la necesidad de validación.

Para profundizar

· Scrum.org — recursos

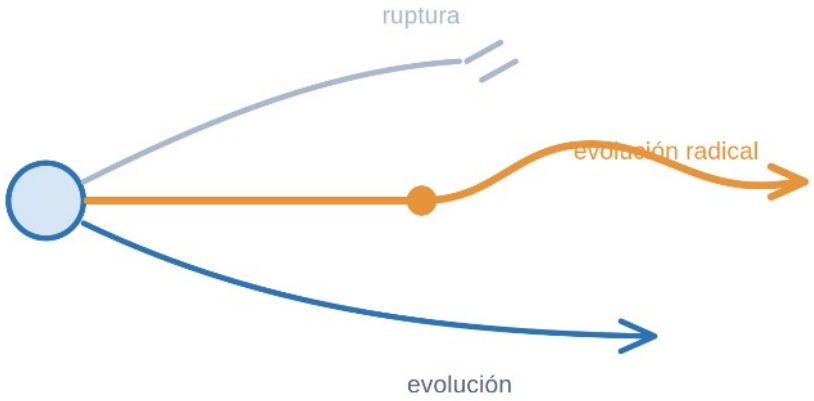
· Scrum Guide Expansion Pack

BLOQUE A · EL CAMBIO DE PARADIGMA

Capítulo 4. El debate ruptura vs. evolución, y la «evolución radical»

EN CONSOLIDACIÓN

La comunidad se divide entre quienes ven el fin de scrum y quienes lo ven como multiplicador de la agilidad.



Ruptura, evolución o evolución radical: principios estables, prácticas que se transforman.

Introducción

Ante el impacto de la IA, la comunidad ágil se ha dividido en dos posturas. Comprenderlas antes de decidir el propio camino evita seguir modas en cualquier dirección. Este capítulo expone ambas y la posición intermedia que adopta el enfoque de Scrum Manager.

4.1 Las dos posturas

La ruptura. Sostiene que las prácticas estándar de scrum han llegado al fin de su efectividad: los ciclos de dos semanas son lentos para la IA, los equipos se reducen, los roles evolucionan o desaparecen, y la historia de usuario no es el formato adecuado para la IA. Algunos llegan a decir que «scrum ha muerto» y abogan por vibe coding, SDD o flujo continuo sin sprints.

La evolución. Argumenta que la IA es un multiplicador de fuerza para la agilidad, no su reemplazo: los principios (empirismo, adaptación, colaboración) son más críticos que nunca, la velocidad de la IA hace más necesaria la reflexión, y el problema no es scrum sino cómo se ha implementado.

4.2 La tercera vía: evolución radical

La posición de Scrum Manager es intermedia: los principios ágiles mantienen su vigencia, pero las prácticas concretas necesitan una transformación profunda —no ajustes cosméticos—, y ese cambio debe hacerlo cada equipo según su contexto, no esperar a que alguien lo prescriba. Coincide con la ruptura en la magnitud del cambio y con la evolución en el valor de los fundamentos. En junio de 2025, el Scrum Guide Expansion Pack declaró formalmente que scrum es «mutable», validando lo que la comunidad ya venía practicando.

El riesgo de cada postura. La ruptura corre el riesgo de tirar al bebé con el agua del baño: descartar mecanismos valiosos como la retrospectiva en nombre de la velocidad. La evolución corre el riesgo de subestimar el cambio: aplicar parches a prácticas que necesitan repensarse. La evolución radical busca evitar ambos.

Lo que debes saber hacer

Al dominar este tema, un profesional es capaz de:

- Exponer los argumentos principales de las posturas de ruptura y de evolución.
- Definir la «evolución radical» y en qué coincide con cada una de las otras dos.
- Identificar el riesgo característico de cada postura.
- Tomar una posición informada sobre qué conservar y qué transformar en su propio equipo.

Errores y antipatrones frecuentes

Adoptar la ruptura por entusiasmo y eliminar mecanismos valiosos. Descartar la retrospectiva o la reflexión en nombre de la velocidad es el error más costoso.

Adoptar la evolución como excusa para no cambiar nada. Aplicar parches cosméticos subestima la magnitud real del cambio.

Esperar a que «alguien» prescriba la adaptación. Scrum es conocimiento abierto; la adaptación es trabajo del propio equipo.

Para profundizar

· Scrum Guide Expansion Pack

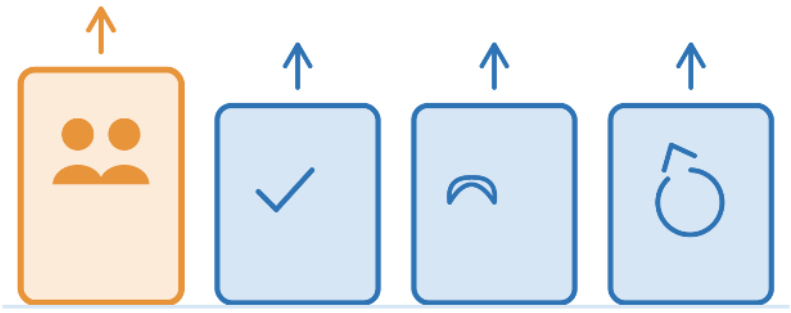
· Age of Product — A Critical Reality Check

BLOQUE B · LOS PRINCIPIOS ÁGILES COMO BRÚJULA

Capítulo 5. Los valores del Manifiesto Ágil se amplifican con la IA

ESTABLECIDO

Los cuatro valores no solo sobreviven a la IA: se vuelven más críticos.



Los cuatro valores del Manifiesto no solo sobreviven: la IA los vuelve más críticos.

Introducción

Antes de analizar qué debe cambiar, conviene anclar qué no debe cambiar. Los cuatro valores del Manifiesto Ágil, redactados en 2001, no solo siguen vigentes veinticinco años después: la IA los hace más críticos, no menos. Este capítulo recorre cómo cada valor se amplifica.

5.1 Los cuatro valores bajo la IA

Individuos e interacciones sobre procesos y herramientas. Cuando las herramientas incluyen agentes que generan código, la conversación humana sigue siendo irremplazable. La IA libera tiempo mental; la pregunta es a qué se dedica. Si se dedica a más interacción de calidad, el valor se amplifica; si a generar más features sin conversación, se pierde.

Software funcionando sobre documentación exhaustiva. Generar prototipos funcionales en horas refuerza el software como mecanismo de validación. Emerge una paradoja: las especificaciones detalladas se vuelven esenciales para guiar a la IA, pero son documentación ejecutable —lenguaje común humano-IA—, no burocracia de cajón.

Colaboración con el cliente sobre negociación contractual. Si construir lo equivocado es barato y rápido, el riesgo aumenta, y la única forma de mitigarlo es más colaboración con el cliente. La IA amplifica tanto los aciertos como los errores en la comprensión del problema.

Respuesta al cambio sobre seguimiento de un plan. La IA hace operativamente viable detectar el cambio en tiempo real, lo que antes requería ciclos trimestrales. Pero el sensor es inútil si el equipo no tiene el hábito de responder, y responder requiere los momentos de reflexión que dan las retrospectivas.

Lo que debes saber hacer

Al dominar este tema, un profesional es capaz de:

- Explicar cómo se amplifica cada uno de los cuatro valores del Manifiesto con la IA.
- Distinguir la documentación ejecutable (spec) de la burocracia documental.
- Argumentar por qué la colaboración con el cliente gana importancia cuando construir es barato.
- Relacionar la respuesta al cambio con la necesidad de momentos de reflexión.

Errores y antipatrones frecuentes

Usar el tiempo liberado por la IA para producir más. Desaprovecha la oportunidad de amplificar la interacción y la validación, que es donde está el valor.

Confundir las specs con documentación burocrática. La spec para IA es lenguaje ejecutable, no papeleo de archivo.

Tener el sensor del cambio sin el hábito de responder. Detectar el cambio en tiempo real no sirve si el equipo no reflexiona ni adapta.

Para profundizar

· Manifiesto Ágil

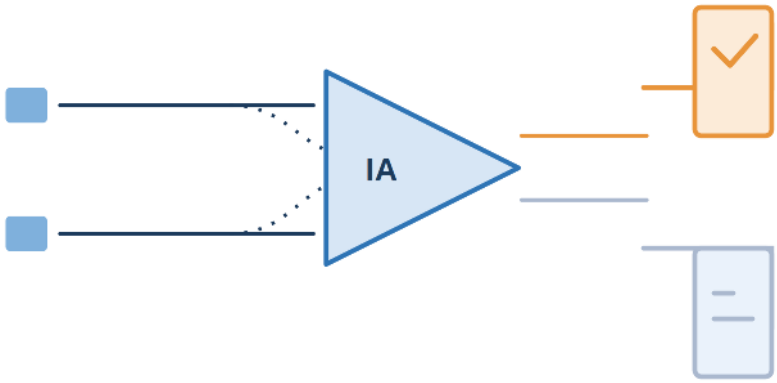
· Scrum Manager BoK — El manifiesto ágil

BLOQUE B · LOS PRINCIPIOS ÁGILES COMO BRÚJULA

Capítulo 6. La IA como amplificador, no como sustituto

ESTABLECIDO

La IA no es neutral: multiplica lo que el equipo ya trae, fortalezas y carencias.



La IA amplifica lo que el equipo ya trae: potencia las buenas prácticas y agrava las carencias.

Introducción

Una idea transversal emerge de toda la investigación reciente: la IA amplifica lo que el equipo ya trae. No es neutral; es un multiplicador de tendencias existentes. Entender esto reencuadra dónde está la responsabilidad del resultado: en el equipo y sus prácticas, no en la herramienta.

6.1 El multiplicador en ambas direcciones

A un equipo con buenas prácticas de producto, la IA las potencia: valida más rápido, explora más opciones, detecta problemas antes. A un equipo con carencias, la IA las expone y agrava: produce más features sin validar, acumula más deuda técnica, genera más ruido sin claridad estratégica. La IA en manos de un buen equipo es un motor más potente para un conductor experto; en manos de un equipo con problemas, es un motor más potente para quien no domina el vehículo.

6.2 La separación del juicio

La mayor amenaza no es que la IA reemplace a los profesionales ágiles, sino que revele que ciertas organizaciones nunca necesitaron verdaderos profesionales ágiles, sino simplemente a alguien que gestionara una herramienta. La IA separa a quienes aportan juicio, experiencia y capacidad de manejar la complejidad humana de quienes solo realizaban tareas mecánicas. El valor diferencial está en lo que la IA no puede hacer: juicio, empatía, visión, liderazgo.

6.3 Qué amplifica y qué no

La IA amplifica la velocidad de ejecución, la capacidad de exploración, el acceso a información y la productividad individual. Pero no amplifica automáticamente la claridad sobre qué construir, la conexión con las necesidades del usuario, la cohesión del equipo ni la capacidad de reflexionar y aprender. Estas últimas siguen dependiendo de las personas y sus prácticas.

Lo que debes saber hacer

Al dominar este tema, un profesional es capaz de:

- Explicar por qué la IA es un amplificador no neutral de las tendencias del equipo.
- Distinguir qué amplifica la IA automáticamente y qué no.

- Argumentar la «separación del juicio» y su implicación para el valor profesional.
- Diagnosticar qué fortalezas y qué carencias está amplificando la IA en su propio equipo.

Errores y antipatrones frecuentes

Esperar que la IA arregle problemas de proceso. Los agrava: un equipo con mala validación produce más sin validar.

Buscar soluciones técnicas a carencias humanas. La respuesta a una debilidad expuesta por la IA es mejor práctica, no más IA.

Definir el valor propio por tareas que la IA puede hacer. El valor diferencial está en el juicio, la empatía y la visión, no en la operativa automatizable.

Para profundizar

· Harvard / D3 — The Cybernetic Teammate

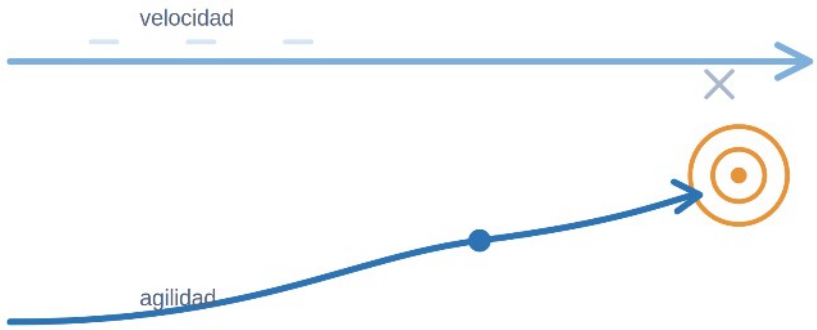
· Scrum Manager — Scrum en la era de la IA

BLOQUE B · LOS PRINCIPIOS ÁGILES COMO BRÚJULA

Capítulo 7. Velocidad no es agilidad

ESTABLECIDO

La agilidad es responder al cambio mientras se entrega valor, no la velocidad de producción.



La velocidad recta no acierta el objetivo móvil; la agilidad corrige el rumbo y lo alcanza.

Introducción

Un error frecuente y peligroso es asumir que ir más rápido equivale a ser más ágil. La confusión existía antes de la IA, pero la IA la amplifica dramáticamente. Distinguir velocidad de agilidad es uno de los criterios que más protegen a un equipo del «feature factory» acelerado.

7.1 La distinción

La agilidad es la capacidad de responder eficazmente al cambio mientras se entrega valor. Un equipo puede ser muy rápido y muy poco ágil: entrega features a gran velocidad, pero sin saber si son las correctas, sin capacidad de cambiar de rumbo cuando la evidencia lo requiere, sin momentos de reflexión. Y un equipo puede ser relativamente lento y muy ágil: entrega menos, pero validado, responde rápido al mercado y mejora continuamente su forma de trabajar.

7.2 La velocidad que importa y la ilusoria

No toda velocidad es igual. La que importa es la de validación (cuánto tardamos en saber si una idea funciona), la de aprendizaje (en incorporar feedback) y la de adaptación (en cambiar de rumbo). La que puede ser ilusoria es la de generación de código (si no se usa o está mal), la de cierre de historias (si no entregan valor) y la de features entregadas (si no se adoptan). Los datos del sector son elocuentes: con IA, la duplicación de código aumentó, el refactoring cayó y la confianza de los desarrolladores en el código generado descendió.

Las preguntas que protegen. ¿Entregamos más valor o solo más código? ¿Podemos cambiar de rumbo cuando la evidencia lo pide? ¿Aprendemos a la velocidad que producimos? ¿La calidad se mantiene o se degrada? Si las respuestas no son claramente positivas, la velocidad puede estar siendo contraproducente.

Lo que debes saber hacer

Al dominar este tema, un profesional es capaz de:

- Definir agilidad y distinguirla de velocidad con ejemplos.
- Clasificar tipos de velocidad en los que importan y los que pueden ser ilusorios.
- Usar las preguntas de protección para diagnosticar si la velocidad es productiva.

- Interpretar los datos de degradación de calidad como síntoma de velocidad sin agilidad.

Errores y antipatrones frecuentes

Medir el éxito por velocidad de producción. Es la métrica que más fácil se infla y menos dice sobre el valor entregado.

Sacrificar la capacidad de cambiar de rumbo por ir rápido. Un equipo comprometido con su plan a alta velocidad es rápido pero no ágil.

Ignorar la degradación de calidad mientras sube la velocidad. La deuda técnica acumulada en silencio acaba frenando todo.

Para profundizar

· Scrum Manager BoK — El manifiesto ágil

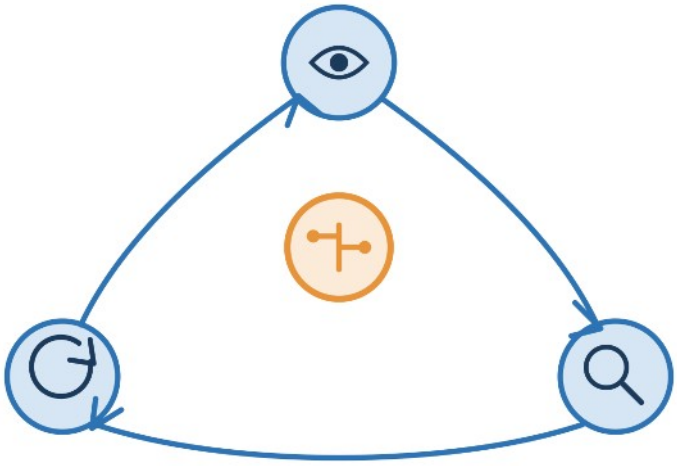
· Scrum Guide Expansion Pack — AI and Scrum

BLOQUE B · LOS PRINCIPIOS ÁGILES COMO BRÚJULA

Capítulo 8. Empirismo, reflexión y autoorganización

ESTABLECIDO

El espíritu de scrum que la IA debe reforzar, no erosionar.



Transparencia, inspección y adaptación: el ciclo empírico que la IA debe reforzar.

Introducción

Los principios que sustentan scrum son invariantes que la IA no debería erosionar, sino reforzar. Pero ese refuerzo no es automático: requiere disciplina deliberada, justo cuando la velocidad invita a abandonarla. Este capítulo desarrolla los cuatro pilares del espíritu que se mantiene.

8.1 Empirismo

Transparencia, inspección y adaptación se potencian con la IA: dashboards más inteligentes, análisis del progreso, ciclos de feedback más rápidos. Pero la capacidad solo genera valor si se ejerce con disciplina. El riesgo es que la velocidad de la IA supere la capacidad del equipo para reflexionar sobre lo que construye. La IA genera datos; el equipo debe convertirlos en aprendizaje. El empirismo no es automático: requiere momentos deliberados de inspección y adaptación.

8.2 Reflexión y colaboración humana

La mejora continua requiere reflexión, y la reflexión requiere tiempo. La IA libera tiempo, pero la tentación es usarlo para producir más, no para reflexionar más. Los equipos que prosperan protegen deliberadamente el tiempo de reflexión. Y la colaboración humana —construir confianza, tomar decisiones difíciles, resolver conflictos con empatía— no puede delegarse: los eventos no son solo intercambios de información, son rituales que construyen equipo.

8.3 Autoorganización

Scrum confía en que los equipos deciden cómo hacer su trabajo. La IA lo hace más importante, no menos: con más herramientas y opciones, las decisiones sobre cómo organizarse son más complejas. La autoorganización en la era de la IA incluye decidir qué tareas se delegan a la IA, qué nivel de revisión se aplica, cómo se integra en el flujo y qué límites se ponen. Un equipo que espera que le digan exactamente cómo usar la IA está renunciando a la autoorganización.

Lo que debes saber hacer

Al dominar este tema, un profesional es capaz de:

- Explicar por qué el empirismo se vuelve más importante, no menos, con la IA.
- Justificar la protección deliberada del tiempo de reflexión frente a la presión de producir.
- Argumentar por qué la colaboración humana no es delegable a la IA.
- Enumerar las decisiones de autoorganización que un equipo debe tomar sobre la IA.

Errores y antipatrones frecuentes

Asumir que la IA hace el empirismo automático. Genera datos, pero convertirlos en aprendizaje exige inspección y adaptación deliberadas.

Usar el tiempo liberado para producir, no para reflexionar. Es la renuncia silenciosa al empirismo.

Esperar que la herramienta dicte cómo organizarse. Decidir cómo usar la IA es responsabilidad del equipo, no de la IA.

Para profundizar

· Scrum Guide Expansion Pack

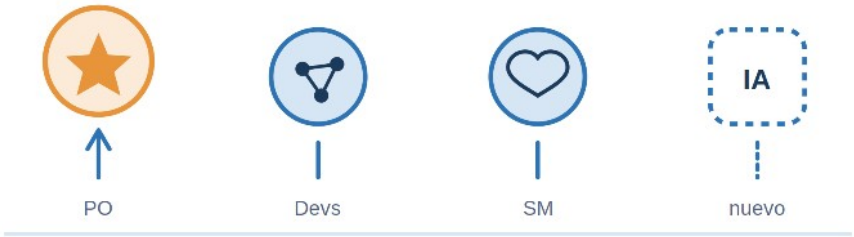
· Manifiesto Ágil — principios

BLOQUE C · ROLES EN TRANSFORMACIÓN

Capítulo 9. Product owner → gestor estratégico del valor

EN CONSOLIDACIÓN

Al volverse la gestión de producto el cuello de botella, el rol gana relevancia, no la pierde.



Ningún rol desaparece: todos desplazan su foco con la IA, y el PO gana relevancia estratégica.

Introducción

Con la IA capaz de construir features más rápido de lo que la organización puede validar su deseabilidad, el product owner se convierte en el factor limitante de la entrega de valor. Paradójicamente, esto hace su rol más importante, no menos —siempre que evolucione de lo táctico a lo estratégico.

9.1 Del backlog a la estrategia

Un product owner que se limita a escribir historias y priorizar un backlog se queda corto, porque esa tarea la asiste la IA. Se necesita un perfil que gestione activamente qué merece construirse, diseñe experimentos de validación continua, decida dónde la IA aporta valor real frente a dónde genera ruido, y mantenga la visión estratégica cuando la velocidad invita a perderse en los detalles.

9.2 Competencias emergentes

El rol incorpora capacidades nuevas:

- **Data literacy:** interpretar datos, diseñar métricas de éxito y decidir con evidencia, crítico cuando la IA genera más outputs que medir.
- **Estrategia de producto con IA:** saber qué puede y qué no puede hacer la IA, y anticipar cómo cambiará el mercado.
- **Gestión de la paradoja producción-validación:** diseñar flujos donde la capacidad de validar escale con la de producir.
- **Comunicación técnica sobre IA:** sin ser técnico, entender lo suficiente para decidir qué delegar y qué riesgos asumir.

El riesgo de obsolescencia. Es bajo si el rol evoluciona hacia la visión estratégica; alto si se limita a escribir historias de usuario, tarea que la IA asiste significativamente.

Lo que debes saber hacer

Al dominar este tema, un profesional es capaz de:

- Explicar por qué el product owner se convierte en el cuello de botella estratégico.
- Enumerar las competencias emergentes del rol y por qué cada una gana relevancia.
- Diseñar un flujo donde la validación escale con la producción.

- Distinguir el product owner táctico (en riesgo) del estratégico (revalorizado).

Errores y antipatrones frecuentes

Limitarse a escribir y priorizar historias. Es la parte del rol que la IA más asiste; quedarse ahí es el camino a la obsolescencia.

No diseñar la validación al ritmo de la producción. Deja crecer un backlog de features construidas y no validadas.

Delegar la visión estratégica. Es justo lo que la IA no puede aportar y lo que define el valor del rol.

Para profundizar

· Scrum Guide Expansion Pack — AI and Scrum

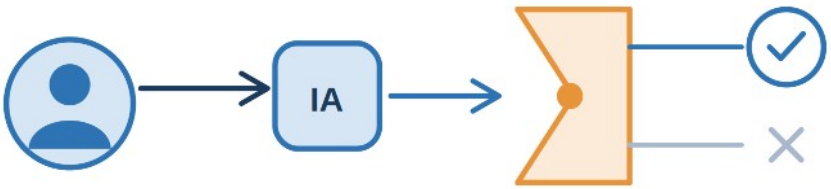
· Scrum Manager — Scrum en la era de la IA

BLOQUE C · ROLES EN TRANSFORMACIÓN

Capítulo 10. Desarrolladores → product builders

EN CONSOLIDACIÓN

El valor se desplaza de la implementación a la orquestación y la revisión.



El valor pasa de escribir a orquestar y revisar: dirigir la IA y juzgar lo que genera.

Introducción

El equipo de desarrollo se transforma en lo que diversos autores llaman «product builders»: perfiles que combinan herramientas no-code, generación de código asistida por IA y comprensión profunda del producto. El valor se desplaza de escribir código a orquestar y revisar.

10.1 El desplazamiento del valor

Antes, el valor estaba en saber escribir código eficiente, conocer el lenguaje, dominar el framework. Ahora está en saber qué pedir a la IA, cómo evaluar lo que genera, cuándo rechazarlo y cómo integrarlo en una arquitectura coherente. Esto no hace irrelevantes las competencias técnicas: al contrario, se necesita más conocimiento técnico para evaluar código generado que para escribirlo desde cero. Un desarrollador junior puede aceptar código que un senior identificaría como problemático.

10.2 Nuevas competencias y el dato contraintuitivo

Las competencias esenciales son prompt engineering, spec writing, revisión de código generado y context engineering. Y un dato revelador: desarrolladores senior experimentaron una disminución del 19 % en su rendimiento en tareas complejas al usar IA, por el tiempo dedicado a revisar y probar el output. No significa que no deban usarla: significa que la IA no es un atajo mágico, requiere inversión de tiempo y juicio experto. Los seniors aportan ese juicio; los juniors pueden no tenerlo aún.

10.3 Tamaño del equipo

La recomendación clásica de 3-9 personas está en revisión: equipos de 2-3 con IA pueden lograr lo que antes requería equipos mayores. Esto trae menos necesidad de coordinación y más autonomía por persona, pero también riesgos: menos diversidad de perspectivas, mayor dependencia de individuos clave, menos resiliencia y riesgo de silos de conocimiento.

Lo que debes saber hacer

Al dominar este tema, un profesional es capaz de:

- Explicar el desplazamiento del valor de la implementación a la orquestación.
- Argumentar por qué revisar código generado exige más experiencia, no menos.
- Enumerar las nuevas competencias esenciales del rol.
- Sopesar las ventajas y los riesgos de los equipos más pequeños con IA.

Errores y antipatrones frecuentes

Asumir que la IA hace prescindible el conocimiento técnico. Evaluar código generado requiere más juicio experto que escribirlo.

Esperar de la IA un atajo sin coste. El tiempo de revisión es real; los seniors pueden incluso ralentizarse en tareas complejas.

Reducir el equipo sin gestionar sus riesgos. Equipos minúsculos ganan agilidad pero pierden resiliencia y diversidad si no se compensa.

Para profundizar

· Scrum Manager — SDD (State of the art)

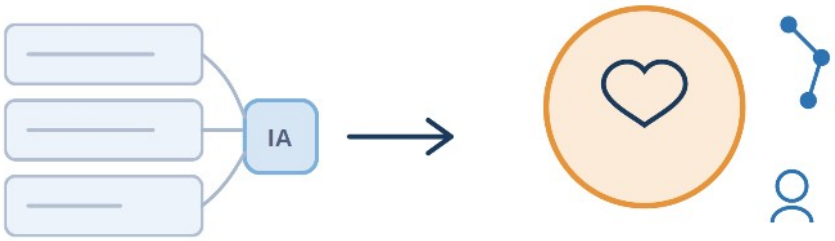
· Anthropic — Building effective agents

BLOQUE C · ROLES EN TRANSFORMACIÓN

Capítulo 11. Scrum master: transformación o extinción

EN CONSOLIDACIÓN

El rol con mayor riesgo de obsolescencia si se define solo como facilitador de eventos.



La IA absorbe lo operativo; el scrum master se desplaza a lo que la IA no replica.

Introducción

Este es el rol con mayor riesgo de obsolescencia si su valor se define exclusivamente como «facilitador de eventos». Pero también es el que más puede ganar si evoluciona hacia competencias que la IA no puede replicar.

11.1 El riesgo real

La IA puede hacer muchas tareas operativas del scrum master tradicional: sugerir técnicas de facilitación, generar agendas, tomar notas y resúmenes, analizar métricas de flujo, detectar patrones en retrospectivas y proponer acciones basadas en datos. Si el valor del rol se limita a esto, la IA lo hace prescindible.

11.2 Lo que la IA no puede hacer

Las competencias que la IA no replica son precisamente las que definen el valor de un scrum master evolucionado: inteligencia emocional (leer el estado del equipo, detectar conflictos latentes, crear seguridad psicológica), pensamiento sistémico (ver conexiones entre problemas), liderazgo (inspirar, dar feedback difícil con empatía, sostener al equipo en crisis), toma de decisiones éticas y coaching organizacional. La IA puede analizar el sentimiento de un texto; no puede sentir la tensión en una sala.

11.3 La evolución del rol

El scrum master evolucionado facilita sistemas de trabajo híbridos humano-IA, establece la gobernanza del uso de IA, hace coaching sobre las nuevas competencias, lidera la reflexión sobre cómo la IA cambia el trabajo y protege los momentos de reflexión contra la presión de la velocidad. En equipos pequeños y maduros, sus responsabilidades pueden interiorizarse sin un rol dedicado, y el scrum master evoluciona hacia el servicio a múltiples equipos o la transformación organizacional.

Lo que debes saber hacer

Al dominar este tema, un profesional es capaz de:

- Identificar qué tareas operativas del rol asume la IA.
- Enumerar las competencias que la IA no puede replicar y que definen el valor del rol.
- Describir las funciones del scrum master evolucionado en equipos híbridos.
- Reconocer cuándo el rol puede interiorizarse en el equipo y hacia dónde evoluciona entonces.

Errores y antipatrones frecuentes

Definir el rol solo como facilitar eventos. Es exactamente lo que la IA hace prescindible.

Competir con la IA en tareas operativas. El valor está en lo que la IA no puede hacer, no en hacer más rápido lo que sí puede.

Abandonar la protección de los momentos de reflexión. Es una de las funciones más valiosas del rol en la era de la IA.

Para profundizar

· Scrum.org — El Scrum Master y la IA

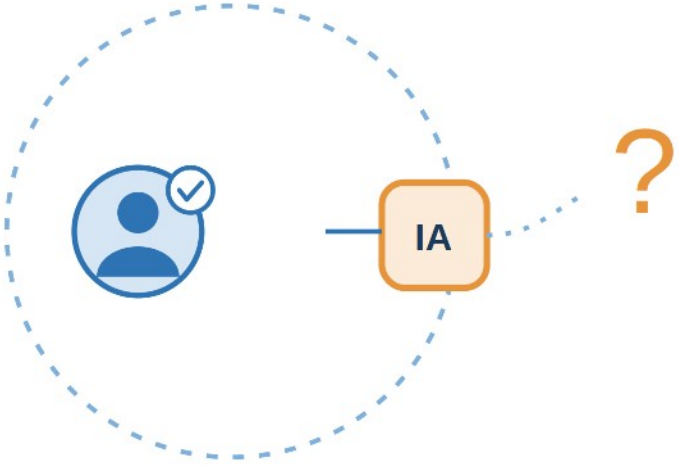
· Scrum Manager — Scrum en la era de la IA

BLOQUE C · ROLES EN TRANSFORMACIÓN

Capítulo 12. La IA como rol: ¿comprometida o implicada?

EMERGENTE

¿Es la IA un miembro del equipo, una herramienta o un colaborador externo?



¿Miembro, herramienta o colaborador externo? La pregunta sigue abierta, con un humano siempre responsable.

Introducción

Una pregunta emergente afecta a todos los roles: dónde sitúa el equipo a la IA en su dinámica de trabajo. En scrum se distingue entre «comprometidos» (quienes intervienen directamente en construir el producto, responsables del resultado) e «implicados» (otras partes interesadas que aportan pero no responden). ¿Dónde encaja la IA?

12.1 Las dos posturas

La IA como comprometido potencial. El Scrum Guide Expansion Pack (2025) sugiere que la IA puede ser un «Product Developer» más, con la condición de que al menos un humano mantenga la responsabilidad. Renombra «Developers» a «Product Developers» y explicita que pueden ser humanos o automatizados. Las versiones recientes incorporan además la IA como actor y la figura de «Supporters».

La IA como herramienta, no comprometido. Otras voces consideran que tratar a la IA como miembro es una analogía imperfecta: la IA no tiene compromiso (no se siente responsable), ni motivación (no se esfuerza más ante un deadline), ni participa significativamente en retrospectivas, ni desarrolla relaciones de confianza. Desde esta perspectiva, es una herramienta potentísima al servicio de los comprometidos, pero no un comprometido.

12.2 Implicaciones prácticas, más allá del debate

Independientemente de la postura filosófica, cada equipo debe resolver cuestiones concretas: quién rinde cuentas cuando un agente comete un error (siempre un humano); cómo se hace «onboarding» a la IA proporcionándole contexto (estándares, arquitectura, historial); y cómo se contempla la IA en los eventos —en la planning se decide qué se le delega, en la daily se revisa su output, en la retrospectiva se evalúa cómo funciona la colaboración con ella.

El debate conceptual no debe bloquear la decisión práctica. Cada equipo puede definir qué puede y qué no puede hacer la IA, quién la revisa, cómo se integra y qué límites tiene, sin necesidad de

cerrar antes la cuestión filosófica de si es o no un miembro del equipo.

Lo que debes saber hacer

Al dominar este tema, un profesional es capaz de:

- Explicar la distinción entre comprometidos e implicados y por qué la IA es difícil de ubicar.
- Exponer la postura del Scrum Guide Expansion Pack y la postura alternativa.
- Enumerar las decisiones prácticas que cada equipo debe tomar con independencia del debate conceptual.
- Diseñar un «onboarding» de contexto para un agente de IA en su equipo.

Errores y antipatrones frecuentes

Bloquear las decisiones prácticas esperando resolver el debate conceptual. Lo importante es definir qué hace la IA y quién responde, no zanjar si es «miembro».

Tratar a la IA como comprometido sin un humano responsable. La rendición de cuentas debe recaer siempre en una persona.

Omitir el onboarding de contexto. Sin contexto (estándares, arquitectura, historial), el output de la IA es irrelevante o erróneo.

Para profundizar

· Scrum Guide Expansion Pack

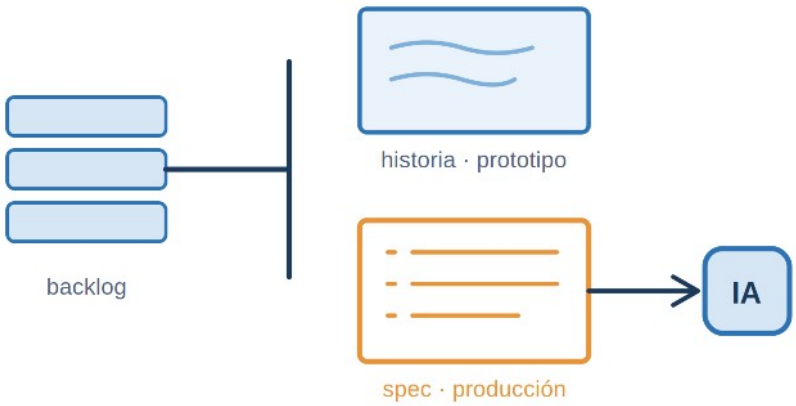
· Age of Product — A Critical Reality Check

BLOQUE D · ARTEFACTOS Y EVENTOS ADAPTADOS

Capítulo 13. De historias de usuario a especificaciones

EN CONSOLIDACIÓN

Las historias funcionan para humanos pero son ambiguas para la IA; conviven con specs según el riesgo.



El backlog convive: historia suelta para prototipar, especificación precisa para producción.

Introducción

Las historias de usuario nacieron como herramienta de comunicación entre humanos: breves, capturan la intención y dejan espacio para la conversación. Pero no son la forma más precisa de instruir a una IA generativa, que interpreta literalmente y no infiere el contexto que no se le da. Este capítulo desarrolla la convivencia —no la sustitución— entre historias y especificaciones.

13.1 La tensión fundamental

Para humanos, las historias funcionan bien: un desarrollador puede preguntar, inferir contexto y proponer alternativas. Para la IA son ambiguas: necesita instrucciones más precisas. La solución no es elegir entre una y otra, sino encontrar la convivencia adecuada. Una especificación para IA incluye típicamente contexto (qué existe ya, qué restricciones aplican), requisitos funcionales (inputs y outputs esperados, casos borde), requisitos no funcionales (rendimiento, seguridad), criterios de aceptación verificables y ejemplos concretos.

13.2 La convivencia según el riesgo

No todo necesita el mismo nivel de especificación: un prototipo de validación puede partir de una historia; una funcionalidad crítica para producción requiere una spec detallada. El nivel de formalización depende del riesgo. La spec complementa la conversación, no la sustituye: el formato «Conversation, Card, Confirmation» sigue siendo válido, y la spec traduce esa conversación a un lenguaje procesable por la IA. Las specs son documentos vivos que iteran con el mismo espíritu que un backlog ágil.

El riesgo de regresión a waterfall. Si las especificaciones se vuelven demasiado rígidas, se regresa al modelo de cascada. Especificaciones excesivamente formalizadas ralentizan los ciclos de cambio. La clave: la spec no es un contrato inamovible, sino una hipótesis que se refina con el feedback del código generado. El nexo con SDD (capítulo 18) es directo.

Lo que debes saber hacer

Al dominar este tema, un profesional es capaz de:

- Explicar por qué las historias de usuario son ambiguas para la IA aunque funcionen para humanos.
- Enumerar los componentes de una especificación para IA.
- Decidir el nivel de formalización (historia vs. spec) según el riesgo del elemento.
- Reconocer y evitar la regresión a waterfall por exceso de rigidez en las specs.

Errores y antipatrones frecuentes

Sustituir todas las historias por specs. No todo lo necesita; un prototipo de validación se frena con una spec completa.

Tratar la spec como contrato inamovible. Es una hipótesis que se refina; rigidizarla devuelve al equipo a la cascada.

Prescindir de la conversación porque ya hay spec. La spec traduce la conversación, no la elimina; la conversación alinea intenciones.

Para profundizar

· Scrum Manager — SDD (State of the art)

· GitHub Spec Kit

BLOQUE D · ARTEFACTOS Y EVENTOS ADAPTADOS

Capítulo 14. Definition of Done para código generado por IA

EN CONSOLIDACIÓN

Sin controles deliberados, la IA acelera la acumulación de deuda técnica.

El código de IA solo está hecho tras pasar una DoD ampliada: contrapeso a la velocidad.

Introducción

La Definition of Done necesita adaptarse específicamente cuando parte del código es generado por IA. Los datos de la industria muestran que, sin controles deliberados, la IA acelera la acumulación de deuda técnica: la duplicación de código aumentó, el refactoring cayó, y la confianza de los desarrolladores en el código generado descendió. La DoD es el mecanismo para contrarrestarlo.

14.1 Una DoD ampliada

Una Definition of Done adaptada al código de IA incluye criterios específicos en cuatro áreas:

- **Revisión humana obligatoria:** todo código generado ha sido revisado por al menos un desarrollador que entiende lo que hace —no solo que «funciona»— y verifica que cumple los estándares.
- **Análisis de calidad:** pasa los análisis estáticos, no aumenta la duplicación por encima del umbral, la complejidad está dentro de límites y no hay vulnerabilidades detectadas.
- **Testing:** tests unitarios cubren la funcionalidad, los de integración pasan y los de aceptación definidos en la spec pasan.
- **Documentación:** el código está documentado según los estándares y los cambios se reflejan en la arquitectura si aplica.

14.2 Métricas de salud y la IA que supervisa IA

Además de la DoD, los equipos incorporan métricas continuas como contrapeso a la velocidad: ratio de código generado vs. refactorizado, tendencia de duplicación, cobertura de tests del código generado y tiempo de revisión por línea. Se revisan en retrospectivas para detectar tendencias. Un modelo emergente es usar una IA como revisor automático de lo que otra genera —un primer filtro de calidad antes de la revisión humana—, que no la sustituye pero la hace más eficiente. El Scrum Guide Expansion Pack introduce además la «Definition of Outcome Done», que añade a «terminado» la pregunta de si el trabajo produjo un impacto real.

Lo que debes saber hacer

Al dominar este tema, un profesional es capaz de:

- Justificar la necesidad de adaptar la DoD con los datos de degradación de calidad.
- Enumerar los criterios de una DoD ampliada para código generado por IA.

- Definir métricas de salud del código y revisarlas en retrospectiva.
- Explicar el modelo de «IA supervisando IA» y sus límites.

Errores y antipatrones frecuentes

Aceptar código de IA porque «funciona». Funcionar no es cumplir estándares ni estar libre de duplicación o vulnerabilidades; hay que entenderlo.

No medir la salud del código. La deuda técnica acelerada por la IA crece invisible si no se monitoriza.

Confiar la revisión solo a otra IA. El filtro automático ayuda, pero no sustituye el juicio humano sobre el código.

Para profundizar

· Scrum Guide Expansion Pack — Definition of Outcome Done

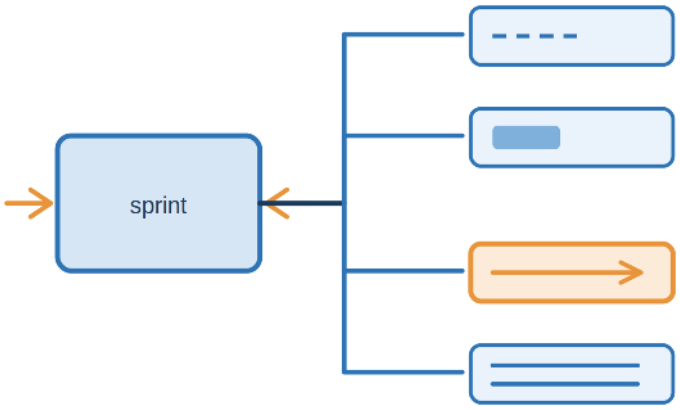
· OWASP Top 10 for LLM Applications

BLOQUE D · ARTEFACTOS Y EVENTOS ADAPTADOS

Capítulo 15. La cadencia del sprint en tensión

EN CONSOLIDACIÓN

La velocidad de la IA tensiona el sprint de dos semanas; hay cuatro opciones en exploración.



La velocidad de la IA tensiona el sprint: cuatro cadencias en exploración, como decisión consciente.

Introducción

El sprint de dos semanas es el estándar de facto, pero no siempre lo fue: las primeras iteraciones de scrum usaban ciclos de uno o dos meses, y fue la comunidad quien fue acortando la cadencia. Con la IA, esa cadencia entra de nuevo en tensión, porque los equipos pueden generar incrementos en horas o días. Este capítulo presenta las opciones que los equipos exploran, sin prescribir ninguna.

15.1 Las cuatro opciones

1. **Mantener sprints con micro-iteraciones internas.** El sprint sigue como contenedor de planificación y revisión, con ciclos cortos dentro. Preserva los puntos de sincronización; puede sentirse artificial si la velocidad real es mucho mayor.
2. **Sprints más cortos.** Una semana o menos. Mayor frecuencia de feedback; el overhead de planning/review puede consumir proporción excesiva del tiempo.
3. **Flujo continuo (Kanban).** Abandonar el sprint como contenedor temporal con límites de WIP. Máxima flexibilidad; riesgo de perder los momentos de reflexión estructurada.
4. **Cadencia dual.** Un ritmo rápido para prototipado (días) y otro más pausado para producción y validación (semanas). Adapta la cadencia al tipo de trabajo; añade complejidad de gestión.

15.2 El riesgo común y la pregunta rectora

Eliminar la cadencia por completo conlleva un riesgo real: perder los momentos de reflexión estructurada, especialmente la retrospectiva. Paradójicamente, es ahora cuando más se necesita reflexionar sobre el proceso, justo cuando la velocidad invita a omitirlo. La pregunta para cada equipo: ¿nuestro sprint actual es un contenedor útil que da ritmo y reflexión, o un cuello de botella que frena la entrega real? Lo importante es que sea decisión consciente, no inercia.

Lo que debes saber hacer

Al dominar este tema, un profesional es capaz de:

- Describir las cuatro opciones de cadencia y el equilibrio de cada una.
- Recordar que la duración del sprint siempre fue una convención evolutiva, no un dogma.

- Identificar el riesgo común de eliminar la cadencia: perder la reflexión estructurada.
- Plantear la pregunta rectora para decidir la cadencia de su equipo de forma consciente.

Errores y antipatrones frecuentes

Cambiar la cadencia por inercia o moda. La decisión debe ser consciente y ligada al contexto, no a lo que hace el sector.

Adoptar flujo continuo sin disciplina de reflexión. Sin los momentos estructurados del sprint, es fácil perder la inspección y adaptación.

Mantener el sprint por costumbre cuando frena la entrega. Si el contenedor estorba, conservarlo por inercia también es un error.

Para profundizar

· Scrum Guide Expansion Pack

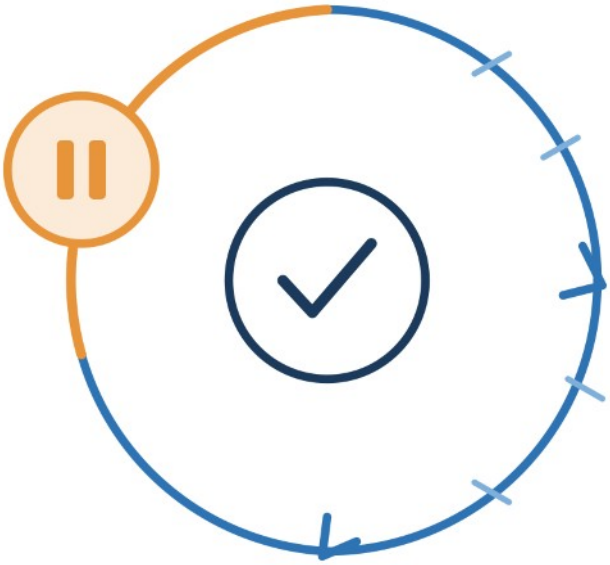
· Scrum Manager BoK — El manifiesto ágil

BLOQUE D · ARTEFACTOS Y EVENTOS ADAPTADOS

Capítulo 16. La retrospectiva: más necesaria que nunca

ESTABLECIDO

El evento que más riesgo corre de eliminarse y el que más se necesita.



La retrospectiva es la parada de reflexión que sostiene el empirismo frente a la presión de velocidad.

Introducción

La revisión y la retrospectiva son los eventos de inspección y adaptación por excelencia. Con la IA, su importancia aumenta, no disminuye. Paradójicamente, la retrospectiva es el evento que más riesgo corre de ser eliminado por la presión de velocidad, y sin embargo es el que más se necesita.

16.1 Por qué es más necesaria

La retrospectiva es el espacio natural para evaluar cómo funciona la colaboración con la IA, detectar si se acumula deuda técnica, plantear si la velocidad de producción se traduce en valor real, cuestionar si los roles siguen teniendo sentido y decidir si hay que adaptar la DoD. Los equipos que la eliminan para «ir más rápido» pierden el mecanismo que les permite inspeccionar y adaptar su forma de trabajar con IA. Esto es, literalmente, renunciar al empirismo.

16.2 El patrón que distingue éxito y fracaso

Un patrón se repite en los equipos que fracasan con IA: adoptan herramientas, la velocidad sube, eliminan eventos «para ir más rápido», pierden la capacidad de detectar problemas, acumulan deuda sin darse cuenta, la calidad se degrada y un incidente grave acaba exponiendo todo. El punto crítico es el tercer paso: eliminar los momentos de reflexión. Los equipos que prosperan hacen lo contrario: mantienen o intensifican los eventos de reflexión y los usan para ajustar cómo trabajan con IA.

16.3 La review y la frecuencia de la retro

La review también muda su foco: de «¿está bien construido?» a «¿estábamos construyendo lo correcto?», e incorpora datos de uso de features anteriores, no solo demostración de las nuevas, evitando la sobreproducción que satura a los interesados. En flujos muy rápidos, la retrospectiva de fin de sprint puede complementarse con mini-retrospectivas, retrospectivas temáticas sobre el uso

de IA o retrospectivas de incidentes cuando algo sale mal con código generado.

Adaptar el formato, no eliminar. Si la retrospectiva de dos horas no funciona, se prueba con una. Pero no con cero. Tratar los eventos de reflexión como inamovibles —no se cancelan por «falta de tiempo»— es lo que protege el empirismo frente a la presión de la velocidad.

Lo que debes saber hacer

Al dominar este tema, un profesional es capaz de:

- Explicar por qué la retrospectiva se vuelve más necesaria, no menos, con la IA.
- Describir el patrón del equipo que fracasa con IA y su punto crítico.
- Reformular el foco de la review de «¿bien construido?» a «¿lo correcto?».
- Diseñar variantes de retrospectiva (temática, de incidente, mini) según la cadencia.

Errores y antipatrones frecuentes

Eliminar la retrospectiva para ganar velocidad. Es el punto crítico del patrón de fracaso: sin reflexión, los problemas se acumulan invisibles.

Convertir la review en demostración de volumen. Presentar más features de las que se pueden evaluar con atención no aporta; el foco es el valor.

Cancelar los eventos por «falta de tiempo». Tratarlos como prescindibles es renunciar a la inspección y adaptación.

Para profundizar

· Scrum Guide Expansion Pack — empirical control

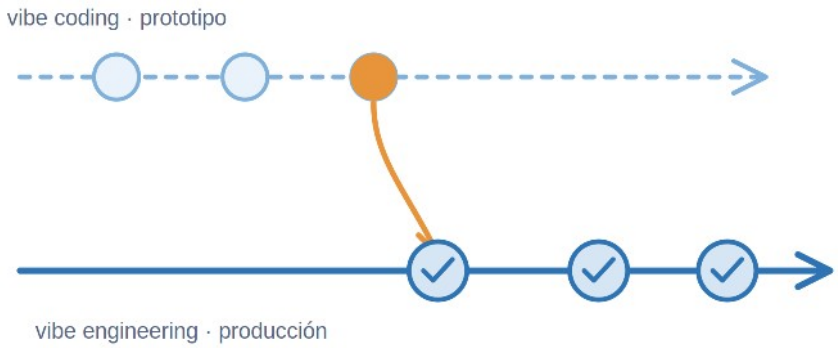
· Scrum Manager BoK — Retrospectiva

BLOQUE E · MODELOS EMERGENTES DE TRABAJO

Capítulo 17. Doble carril: vibe coding y vibe engineering

EN CONSOLIDACIÓN

Separar un flujo rápido para prototipos de un flujo robusto para producción.



Dos carriles: uno rápido para prototipar y otro robusto para producción; lo validado se reescribe.

Introducción

El modelo de doble carril propone separar explícitamente dos flujos de trabajo con objetivos y estándares diferentes. Es la respuesta a una realidad: no todo el trabajo requiere el mismo rigor, y forzar un único estándar frena unas cosas o compromete otras.

17.1 Los dos carriles

Vibe coding: el carril rápido. Flujo ultrarrápido para crear prototipos no liberables a producción. Su propósito es validar ideas con usuarios rápidamente. La calidad del código es secundaria; lo que importa es la velocidad de aprendizaje. Prototipos desechables, experimentación libre, mínima revisión formal.

Vibe engineering: el carril robusto. Flujo que asegura mínimos de ingeniería, seguridad y calidad para que el producto sea robusto y mantenible. Aquí se aplican las prácticas de SDD, se refuerza la DoD y se ejecuta el QA. Código destinado a producción, specs detalladas, revisión exhaustiva.

17.2 Por qué la separación y cómo se transita

El concepto de «dual-track agile» (discovery + delivery) existe desde hace años, pero la IA lo hace más explícito y necesario, porque la velocidad del vibe coding puede ser muy superior a la del vibe engineering. Sin separación, ocurre una de dos cosas: el estándar de producción frena la experimentación, o el código de «prueba rápida» contamina producción y acumula deuda. La transición es clave: una idea validada en vibe coding no se traslada directamente; se reescribe —o se regenera con mejores specs— en vibe engineering. El prototipo demuestra que la idea funciona; el código de producción la implementa correctamente.

Lo que debes saber hacer

Al dominar este tema, un profesional es capaz de:

- Describir los dos carriles, su propósito y sus estándares.
- Explicar por qué la IA hace más necesaria la separación entre experimentación y producción.
- Gestionar la transición entre carriles sin que el prototipo contamine producción.
- Decidir qué trabajo va a cada carril según su finalidad.

Errores y antipatrones frecuentes

Llevar el prototipo de vibe coding directo a producción. El código de prueba sin el rigor del carril robusto acumula deuda técnica.

Aplicar el estándar de producción a la experimentación. Frena la validación rápida que es el propósito del carril ligero.

No distinguir qué carril corresponde a cada trabajo. Mezclar finalidades hace que uno de los dos estándares estorbe siempre.

Para profundizar

· Scrum Manager — SDD (State of the art)

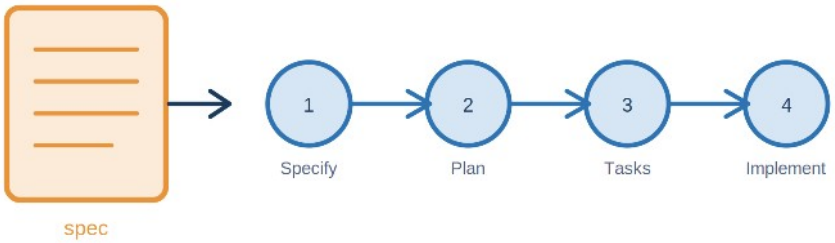
· Scrum Manager — Scrum en la era de la IA

BLOQUE E · MODELOS EMERGENTES DE TRABAJO

Capítulo 18. Spec-Driven Development como modelo de trabajo

EN CONSOLIDACIÓN

La especificación como fuente de verdad de la que se deriva el código.



La especificación es la fuente de verdad de la que se deriva el código, en cuatro fases.

Introducción

El Spec-Driven Development es el modelo más estructurado para trabajar con IA generativa: utiliza especificaciones bien elaboradas como prompts para que los agentes generen código ejecutable. La especificación se convierte en la fuente de verdad; el código es un derivado. Este capítulo lo resume como modelo de trabajo dentro de scrum; el tema tiene desarrollo propio y completo en su State of the art específico.

18.1 El flujo de cuatro fases

Herramientas como GitHub Spec Kit formalizan cuatro fases iterativas: Specify (se describe a alto nivel qué construir y por qué, y el agente genera una spec detallada), Plan (el agente propone un plan de implementación), Tasks (spec y plan se descomponen en tareas pequeñas, revisables de forma aislada) e Implement (el agente ejecuta y el desarrollador revisa cambios focalizados, no bloques masivos). A diferencia de la documentación tradicional, las specs de SDD son artefactos ejecutables: la implementación no puede desviarse sin provocar fallos.

18.2 Niveles de adopción y estado

No todos los equipos adoptan SDD al mismo nivel: spec-first (escribir specs antes de codificar), spec-driven con validación automatizada (las specs incluyen escenarios que se ejecutan), y spec-as-source (la spec reemplaza al código como artefacto principal). Es realista reconocer que es una práctica que aún madura: los agentes no siguen todas las instrucciones de forma fiable, y escribir buenas specs es una competencia que se desarrolla con el tiempo. Requiere experimentación en proyectos de bajo riesgo, paciencia y revisión humana.

Relación con este tema. SDD aparece aquí como uno de los modelos emergentes de trabajo con IA, en su faceta de encaje en el equipo ágil. Su desarrollo técnico completo —método, puertas, escritura de specs, ecosistema— es objeto de un State of the art y un documento de desarrollo propios en Skill Arena.

Lo que debes saber hacer

Al dominar este tema, un profesional es capaz de:

- Describir el flujo de cuatro fases de SDD y qué produce cada una.
- Explicar qué significa que la spec es la fuente de verdad y el código un derivado.

- Distinguir los tres niveles de adopción de SDD.
- Situar el estado de madurez del modelo y qué requiere su adopción.

Errores y antipatrones frecuentes

Adoptar SDD al máximo nivel de golpe. Conviene empezar por spec-first en proyectos de bajo riesgo y madurar gradualmente.

Confiar ciegamente en que el agente sigue la spec. Aún no lo hace de forma totalmente fiable; la revisión humana sigue siendo necesaria.

Confundir las specs ejecutables con documentación tradicional. Las de SDD se ejecutan y verifican; no son documentos que se archivan.

Para profundizar

· Scrum Manager — SDD (State of the art)

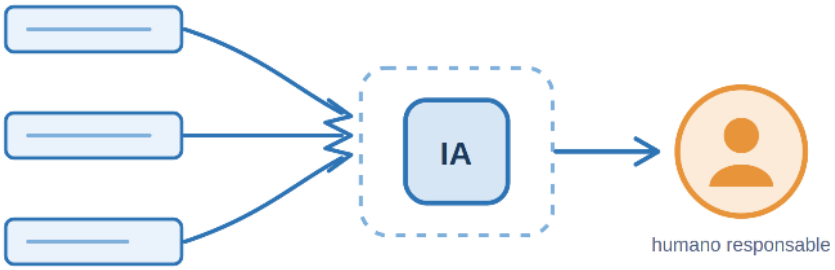
· GitHub Spec Kit

BLOQUE E · MODELOS EMERGENTES DE TRABAJO

Capítulo 19. IA como teammate: onboarding y gestión de contexto

EMERGENTE

Un enfoque gradual que trata a la IA como un compañero al que se hace onboarding.



Se hace onboarding a la IA: contexto, límites e integración, con un humano siempre responsable.

Introducción

El framework «IA como teammate», impulsado desde la comunidad de Scrum.org, propone un enfoque más conservador y gradual para integrar la IA en equipos scrum. Es compatible con las prácticas habituales y puede adoptarse de forma incremental, lo que lo hace accesible para equipos que no quieren reescribir su marco de trabajo.

19.1 El concepto central

En lugar de tratar la IA como una herramienta más, se la trata como un compañero al que se hace «onboarding»: se le proporciona contexto, se le establecen límites y se le integra en el flujo. El elemento clave es la gestión de contexto (context management): igual que un nuevo miembro humano necesita conocer los estándares del equipo, la arquitectura del sistema y el historial de decisiones, la IA necesita ese mismo contexto para que sus outputs sean relevantes. Este onboarding es diferente al de un humano, pero es igualmente necesario.

19.2 Adopción gradual y responsabilidad

La virtud del framework es que no exige una transformación radical: se integra sobre las prácticas existentes y se profundiza a medida que el equipo gana confianza. Como en todo modelo de trabajo con IA, un humano mantiene siempre la responsabilidad del resultado. Conecta con las decisiones de equipo del capítulo 3 (qué puede hacer la IA, quién la revisa, qué límites tiene) y con la cuestión de la IA como rol del capítulo 12.

Por qué es «emergente». El framework concreto es reciente y su adopción aún incipiente. Es la propuesta más conservadora del bloque, pero su formalización y sus resultados están todavía formándose; conviene seguir su evolución en cada revisión del mapa.

Lo que debes saber hacer

Al dominar este tema, un profesional es capaz de:

- Explicar el concepto de tratar a la IA como un teammate al que se hace onboarding.
- Describir la gestión de contexto y por qué es el elemento clave del framework.
- Justificar la adopción gradual como ventaja frente a una transformación radical.
- Relacionar el framework con las decisiones de equipo sobre la IA y la responsabilidad humana.

Errores y antipatrones frecuentes

Omitir el onboarding de contexto. Sin estándares, arquitectura e historial, los outputs de la IA son irrelevantes, igual que los de un recién llegado sin información.

Esperar una transformación radical donde el framework propone gradualidad. Su valor está precisamente en integrarse sobre lo existente.

Delegar la responsabilidad del resultado en la IA «teammate». La analogía del compañero no traslada la rendición de cuentas, que sigue siendo humana.

Para profundizar

· Scrum.org — recursos

· Scrum Guide Expansion Pack — AI as actor