Scrum Manager®
Uncovering better ways

Scrum en equipos con IA — Guía rápida

Scrum para equipos que coordinan inteligencia humana y artificial

Versión 1.0 — Marzo 2026

Índice

Sobre esta guía

Las reglas del ajedrez caben en una página: seis piezas, cómo se mueven y poco más. Cualquiera puede aprenderlas en una tarde. Sin embargo, saber jugar al ajedrez es otra cosa: requiere práctica, intuición y un conocimiento que no se puede reducir a reglas escritas.

Con Scrum ocurre algo parecido. Sus elementos fundamentales (roles, artefactos, eventos) son sencillos de aprender. Lo difícil es aplicarlos con criterio en contextos reales. Esta guía adopta ese mismo espíritu: describe los elementos de **Scrum en equipos con IA** de forma breve y directa, como las reglas del juego. Cómo jugar bien es algo que cada equipo tendrá que descubrir practicando.

Scrum en equipos con IA no es un framework nuevo. Es la evolución natural de Scrum para equipos donde humanos y agentes de IA trabajan juntos en la creación de productos. Los principios ágiles —empirismo, entrega de valor, adaptación continua— permanecen intactos. Lo que cambia es cómo se implementan cuando la IA comprime los ciclos de entrega, los equipos son más pequeños y el interlocutor ya no es solo humano.

Esta guía asume que el lector ya conoce Scrum. No se explica qué es un sprint o un backlog: se explica en qué se transforma.

PARTE 1: elementos de Scrum

Roles

AI product owner o Product architect

Evolución del product owner

Responsable de maximizar el valor del producto decidiendo qué merece ser construido y validando que cada incremento resuelve un problema real. En un entorno donde la IA permite construir casi cualquier cosa, la pregunta estratégica ya no es "¿podemos construirlo?" sino "¿deberíamos construirlo, y por qué?".

- **Foco principal.** Visión estratégica, validación continua con usuarios y decisión de qué entra en el backlog.

Qué cambia respecto al product owner clásico:

- El backlog deja de ser una lista de features para convertirse en un conjunto de **hipótesis a validar**. La IA puede construir rápido; el product architect decide qué vale la pena construir.
- Gestiona la **paradoja de la productividad:** el equipo produce más, pero eso solo genera valor si hay validación estratégica detrás.
- Trabaja en estrecha coordinación con los product builders para definir **specs** (no solo historias de usuario) cuando el trabajo lo ejecutará la IA.
- Necesita competencias de ***data literacy*** y diseño de experimentos de validación, además de las competencias clásicas de producto.
- Es el responsable de decidir cuándo una idea pasa del **carril rápido** (*Vibe coding*) al **carril robusto** (*Vibe engineering*).

> **Indicador clave.** El valor del product architect no se mide por cuántos ítems entrega el equipo, sino por cuántos de esos ítems resuelven problemas reales de usuarios.

AI scrum master o Agile enabler

Evolución del scrum master

Facilita el funcionamiento del sistema de trabajo híbrido (humanos + IA), asegurando que el equipo trabaja con eficacia, reflexiona sobre su proceso y evoluciona sus prácticas de forma continua.

- **Foco principal.** Diseño de flujos de colaboración híbridos, gobernanza de IA, coaching del cambio y pensamiento sistémico.

Qué cambia respecto al Scrum Master clásico:

- Ya no facilita sólo interacciones entre personas, sino el **sistema completo humanos + agentes IA:** cómo se divide el trabajo, cómo se revisa, cómo se gobierna.
- **Es responsable de la gobernanza ética y de calidad en el uso de IA:** políticas de uso, límites, auditoría de outputs, rendición de cuentas.
- **Gestiona la confianza del equipo en la IA:** ayuda a encontrar el equilibrio entre confianza ciega (que genera deuda técnica) y desconfianza paralizante (que anula el potencial de la herramienta).
- **Facilita el coaching del cambio cultural** que implica pasar de un equipo exclusivamente humano a un equipo híbrido: resistencias, miedos, curvas de aprendizaje desiguales.
- **Aplica pensamiento sistémico para ver el flujo completo:** dónde se generan cuellos de botella, cómo interactúan los loops de feedback, qué efectos no intencionados produce el uso de IA.

La **facilitación de eventos** sigue siendo parte de su rol, pero es una fracción menor. Las herramientas de IA pueden asistir en la logística de los eventos; el agile enabler aporta el juicio humano que la IA no puede replicar.

> **Indicador clave.** El valor del agile enabler se mide por la madurez y fluidez del sistema de trabajo del equipo, no por la cantidad de reuniones que facilita.

Product builders

Evolución de los developers

Profesionales responsables de construir el incremento de producto, combinando el uso de herramientas no-code, generación de código asistida por IA y comprensión profunda del producto y su arquitectura.

- **Foco principal.** Orquestación de IA, arquitectura, escritura de especificaciones y revisión de código generado.

Qué cambia respecto a los developers clásicos:

- El valor se desplaza de **implementar a orquestar:** diseñar arquitecturas, escribir specs que guíen a la IA, revisar y validar lo que genera, asegurar calidad y seguridad.
- Necesitan competencias de **prompt engineering, context engineering y spec writing** además de las competencias técnicas tradicionales.
- Son responsables de que el código generado por IA cumpla la **Definition of Done reforzada**: análisis estático, no-duplicación, seguridad, revisión cruzada.

- En el **modelo de doble carril**, pueden trabajar en modo exploración rápida (*vibe coding*) o en modo producción (*vibe engineering*), ajustando el nivel de rigor según el carril.
- La capacidad de **revisar críticamente** lo que genera la IA es una competencia central. La IA produce soluciones plausibles pero frecuentemente frágiles; el product builder es quien detecta los errores sutiles.

> **Indicador clave.** El valor de los product builders se mide por la calidad, mantenibilidad y robustez de lo que entregan, no por la cantidad de código producido.

Artefactos

Product backlog

Lista ordenada y emergente de todo lo que podría ser necesario en el producto.

Qué cambia:

- Evoluciona de una lista de features a un **repositorio de hipótesis a validar**. Cada ítem debería poder responder: ¿qué problema resuelve y para quién?
- Contiene dos tipos de ítems que conviven: **historias de usuario** (intención y contexto de negocio para comunicación humana) y **specs SDD** (especificaciones detalladas para guiar el trabajo de agentes IA).
- **No todos los ítems necesitan el mismo nivel de especificación:** un prototipo rápido puede partir de una historia; una funcionalidad crítica para producción requiere una spec.
- El product architect es responsable de su ordenación, con el criterio de **valor validado**, no solo valor percibido.

Sprint backlog

Subconjunto del product backlog seleccionado para el sprint, junto con el plan para entregar el incremento y alcanzar el sprint goal.

Qué cambia:

- Incluye explícitamente la **atribución humano/IA**: qué tareas requieren creatividad o juicio humano y cuáles son ejecutables por agentes IA.
- Las tareas asignadas a agentes IA deben cumplir la **Definition of Ready para IA** (ver «Prácticas, Parte 2») antes de considerarse preparadas para el sprint.
- Las specs funcionan como el **contrato entre el equipo humano y los agentes IA:** definen con precisión qué se espera que la IA genere.

Incremento

Resultado tangible y utilizable del sprint que cumple la Definition of Done.

Qué cambia:

- Con micro-iteraciones dentro del sprint, pueden producirse **múltiples incrementos** en un solo sprint.
- Todo incremento que contenga código generado por IA debe pasar la **DoD reforzada** antes de considerarse "Done".
- Si se emplea un "carril rápido" (vibe coding) para generar prototipos, éstos no se consideran **incrementos liberables:** son artefactos de aprendizaje que sirven para validar hipótesis.

Definition of Done (DoD) reforzada

Criterio compartido de calidad que un incremento debe cumplir para considerarse terminado. En equipos con IA incluye controles específicos para código generado por IA.

Elementos de la DoD reforzada:

- **Análisis estático.** El código generado por IA pasa herramientas de análisis estático antes de integrarse.
- **No-duplicación.** Se verifica que la IA no ha introducido código duplicado (la duplicación en proyectos asistidos por IA tiende a aumentar significativamente).
- **Seguridad.** Revisión de vulnerabilidades específica para código generado por IA.
- **IA supervisando IA.** Se utilizan agentes de IA como revisores automáticos de lo que otros agentes generan, con semáforos y alertas.
- **Revisión humana.** Un Product Builder revisa y valida el código antes de que se integre, con especial atención a los errores sutiles que la IA introduce con apariencia de corrección.

Historias de usuario + Specs (SDD)

Los dos formatos de descripción del trabajo que conviven en equipos con IA.

Historias de usuario

- **Formato:** "Como [usuario], quiero [acción] para [beneficio]".
- **Capturan intención y contexto de negocio.** Son el vehículo de comunicación entre humanos.
- **Siguen siendo valiosas** para conectar al equipo con el usuario y con el "por qué" de lo que se construye.

Specs (Spec-Driven Development)

Especificaciones detalladas que sirven como **instrucción directa para agentes IA**: contexto técnico, restricciones, criterios de aceptación verificables, esquemas de datos, contratos de API.

- Son **documentos vivos** que iteran como un backlog: se refinan, adaptan y evolucionan.
- No sustituyen a las historias de usuario: **las complementan**. La historia captura la intención; la spec la traduce a un lenguaje procesable por la IA.
- **Riesgo a gestionar:** si las specs se vuelven rígidas y exhaustivas, se regresa al modelo de cascada. La spec es documentación operativa, no un contrato inamovible.

Eventos

Sprint

Duración. 1 semana (referencia orientativa; cada equipo ajusta según su contexto).

Qué cambia:

- **La cadencia se acorta** porque la IA comprime los tiempos de entrega. Sprints de 2 semanas pueden sentirse lentos cuando el equipo puede generar incrementos en días.
- Dentro del sprint se ejecutan **micro-iteraciones:** ciclos rápidos de ideación → generación (con IA) → validación que producen incrementos parciales.
- El sprint sigue siendo el **contenedor de planificación, reflexión y adaptación.** Las micro-iteraciones aceleran la ejecución; el sprint proporciona el ritmo de inspección.

Sprint planning

Duración. Proporcionalmente más corta que en Scrum clásico, dado el sprint más corto y el equipo más pequeño.

Qué cambia:

- La fase más crítica es la **atribución de tareas humano/IA**: decidir explícitamente qué requiere creatividad o juicio humano y qué puede ejecutar un agente IA.
- Para las tareas asignadas a agentes IA, se verifican las condiciones de la **Definition of Ready para IA**: ¿tiene la IA el contexto suficiente (prompts, documentación, schema)?
- Se definen o refinan las **specs** para las tareas que ejecutará la IA.
- Se establece el **sprint goal** con mayor intencionalidad: con la capacidad de la IA para construir casi cualquier cosa, el objetivo estratégico es más importante que nunca para evitar dispersión.

Check-in diario

Evolución de la daily scrum

Formato. Ligero o asíncrono. En equipos de 2-4 personas que trabajan juntos continuamente, una reunión formal de 15 minutos puede ser desproporcionada.

Qué cambia:

- El foco se desplaza de "qué hice ayer / qué haré hoy" a la **gestión de desviaciones**: ¿los agentes IA están produciendo lo esperado? ¿Hay outputs que se desvían de la spec? ¿Algún agente ha reportado baja confianza en un módulo?
- Se revisan los ***logs* de output** de los agentes IA para detectar problemas tempranos.
- **Puede ser asíncrono** (un canal compartido con actualizaciones breves) con sincronización presencial solo cuando se detecten desviaciones o bloqueos.

Sprint review

Qué cambia:

- Evoluciona de "demostrar lo construido" a **compartir aprendizajes validados**. El equipo puede presentar un experimento del carril rápido que falló como resultado valioso si generó aprendizaje.
- Es una **co-presentación humano + IA**: el product builder que se responsabiliza del incremento lo presenta y contextualiza, con transparencia sobre qué generó la IA y cómo.
- El product architect utiliza la review para evaluar si los incrementos validan las hipótesis del backlog.

Retrospectiva

Qué cambia:

- Además de la dinámica humana del equipo, incluye explícitamente la **evaluación de los flujos de colaboración con IA**: ¿estamos dándole buen contexto? ¿Revisamos con el rigor adecuado? ¿La DoD reforzada está funcionando? ¿Estamos acumulando deuda técnica?
- Es el evento donde el equipo **inspecciona y adapta su relación con la IA**. Es ahora más importante que nunca: los equipos están aprendiendo a trabajar con una tecnología nueva y necesitan un espacio deliberado para reflexionar.
- Puede incluir revisión de **métricas de salud del código:** duplicación, complejidad ciclomática, cobertura de tests, ratio de refactoring vs. código nuevo.

PARTE 2: Prácticas de la comunidad

Prácticas emergentes que la comunidad profesional está explorando y desarrollando. No son prescriptivas: son opciones que cada equipo evalúa según su contexto.

Doble carril (*Double-Track*)

Un modelo de trabajo con dos velocidades simultáneas.

Carril rápido — Vibe coding

- Orientado a crear **prototipos no liberables** a producción.
- **Propósito:** validar ideas con usuarios rápidamente usando IA y herramientas no-code.
- La calidad del código es secundaria; lo que importa es la **velocidad de aprendizaje**.
- Es el espacio del **"vibe coding"**: exploración libre con IA.

Carril robusto — Vibe engineering

- Asegura **estándares de ingeniería, seguridad y calidad** para lo que va a producción.
- Se aplican **Spec-Driven Development, DoD reforzada y QA**.
- Es el espacio donde el código generado por IA **se somete al rigor de producción**.

Transición entre carriles

- **La transición es bidireccional:** una idea validada en vibe coding pasa a vibe engineering para producción, pero también puede volver al carril rápido si necesita revalidación.
- **El product architect decide** cuándo una idea merece la transición al carril robusto.
- **La velocidad del carril rápido puede ser 10x-100x mayor que la del robusto:** gestionar esta diferencia es clave.

Estimación en tres niveles

Es un modelo de estimación adaptado a equipos híbridos donde los story points clásicos (diseñados para medir esfuerzo cognitivo humano) no reflejan la realidad del trabajo con IA.

Nivel 1 — Tareas automatizadas (*Zero-Point*)

- Trabajo que la IA ejecuta de forma autónoma o casi autónoma.
- No se estiman en story points. Se miden por **coste computacional** (tokens, tiempo de ejecución) y **tiempo de revisión humana** necesario.

- **Ejemplo:** generar tests unitarios a partir de una spec, crear scaffolding de un componente.

Nivel 2 — Tareas humanas (*Standard*)

- Trabajo que requiere creatividad, juicio o conocimiento que la IA no puede aportar.
- Se estiman con los métodos habituales (story points, tallas de camiseta, etc.).
- **Ejemplo:** definir la estrategia de producto, diseñar la experiencia de usuario, negociar con stakeholders.

Nivel 3 — Tareas de revisión e integración (R&I)

- Trabajo humano dedicado a **revisar, validar e integrar** lo que genera la IA.
- Es el tipo de tarea más fácil de subestimar y el que genera más cuellos de botella.
- Se estima explícitamente porque un agente IA puede generar código en minutos, pero la revisión humana puede llevar horas.
- **Ejemplo:** revisar y validar un módulo generado por IA, integrar outputs de diferentes agentes, resolver conflictos entre código generado y arquitectura existente.

> **Por qué importa.** Si se planifica el sprint solo con la velocidad de generación de la IA, se crea un cuello de botella masivo en la revisión humana. Estimar las tareas R&I explícitamente es lo que evita esto.

Definition of Ready para IA

Es el criterio que debe cumplir un ítem del backlog antes de ser asignado a un agente IA para su ejecución en el sprint.

Criterios

- La **spec** está completa y ha sido validada por un product builder.
- El agente IA tiene acceso al **contexto necesario:** documentación de API, esquema de base de datos, estándares de arquitectura, patrones del proyecto.
- Los **prompts técnicos** están escritos con la precisión suficiente para que el agente genere output relevante.
- Los **criterios de aceptación** son verificables y testeables.
- Se ha estimado el **tiempo de revisión humana** (tarea R&I) y se ha incluido en la planificación del sprint.

> **Por qué importa.** Si un agente IA no tiene el contexto suficiente, su output será irrelevante o erróneo, generando retrabajo. La Definition of Ready para IA es el equivalente a hacer un buen onboarding a un miembro nuevo del equipo.

Sprint planning como atribución humano/IA

Es una evolución del sprint planning donde la actividad central es decidir explícitamente qué trabajo hace cada tipo de inteligencia (humana o artificial).

Cómo funciona

1. El product architect presenta los ítems priorizados del product backlog para el sprint.
2. El equipo **clasifica cada tarea** según los tres niveles de estimación: automatizada, humana o revisión/integración.
3. Para las tareas automatizadas, se verifica la **Definition of Ready para IA**: ¿tiene el agente todo lo que necesita?
4. Se escriben o refinan las **specs** necesarias.
5. Se planifica el sprint incluyendo explícitamente las **tareas R&I**, no solo las de generación.
6. Se establece el **sprint goal** con foco estratégico.

> **Riesgo a evitar.** Llenar el sprint solo con tareas de generación por IA sin reservar capacidad suficiente para revisión humana. El sprint se colapsa cuando todo el código está "generado" pero nadie lo ha podido revisar.

IA como *teammate*

Es un enfoque (propuesto por Scrum.org) para integrar la IA como compañero de equipo, no como herramienta pasiva. Se estructura en cuatro pasos:

1. ***Model management.*** Seleccionar y gestionar los modelos de IA adecuados para cada tipo de tarea del equipo.
2. ***Context management.*** Proporcionar a la IA el contexto necesario (estándares, arquitectura, historial, patrones del proyecto) para que sus outputs sean relevantes. Es el equivalente al onboarding de un nuevo miembro.
3. ***Prompt engineering.*** Desarrollar la competencia de comunicación efectiva con la IA como disciplina del equipo. Incluye la creación de **bibliotecas de prompts** reutilizables.
4. **Governance management.** Establecer políticas de uso, límites de gasto (tokens, API), auditoría de outputs y rendición de cuentas. Definir quién es responsable cuando la IA comete un error.

> **Cuándo usarlo.** Es un enfoque más conservador que el *double-track* o el SDD. Es un buen **punto de partida** para equipos que están empezando a integrar IA y quieren hacerlo de forma ordenada sin cambiar radicalmente su forma de trabajar.

Specification by Example (SbE)

Consiste en crear tests basados en ejemplos concretos y verificables que sirven como especificación del comportamiento esperado.

Por qué es especialmente relevante en equipos con IA

- **La IA no puede "corregir sus propios deberes".** Si le pides que genere código y luego le pides que lo revise, tenderá a validar lo que ella misma generó.
- **La SbE proporciona expectativas objetivas y externas** contra las que verificar el output de la IA.
- **Los ejemplos concretos reducen la ambigüedad** que lleva a la IA a generar soluciones "casi correctas pero no del todo".

Métricas adaptadas

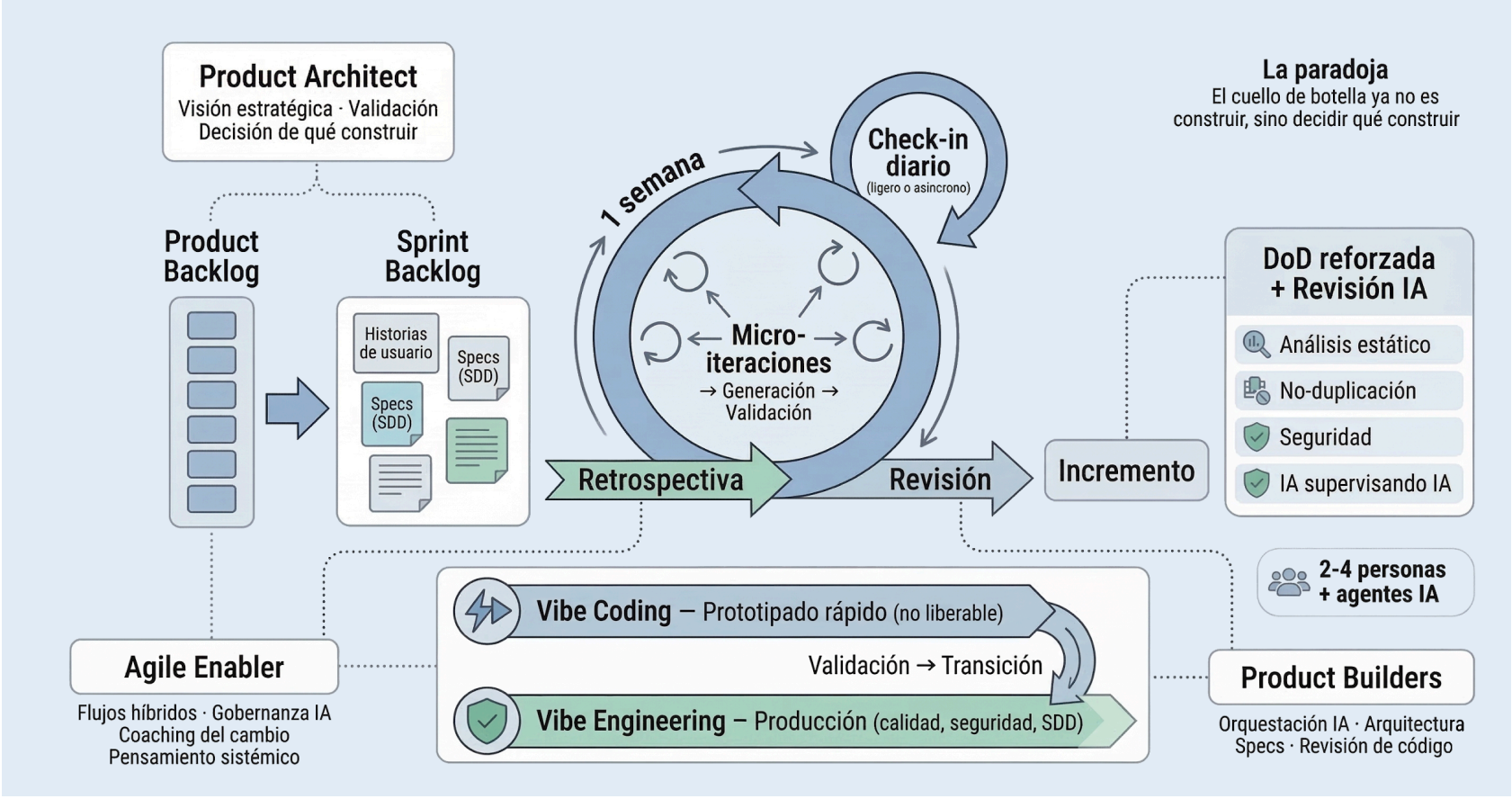
Métricas que pierden relevancia

Velocity (*story points* por sprint) deja de ser significativa cuando parte del trabajo lo ejecuta IA. No mide el valor entregado.

Métricas que ganan relevancia

- ***Cycle time:*** tiempo desde que un ítem entra en desarrollo hasta que está en producción.
- ***Lead time:*** tiempo desde que se identifica una necesidad hasta que se entrega valor al usuario.
- **Ratio de código refactorizado vs. código nuevo:** indicador de salud técnica (la IA tiende a generar código nuevo en lugar de mejorar el existente).
- **Duplicación de código:** indicador crítico en proyectos asistidos por IA.
- **Tiempo de revisión humana por tarea IA:** indicador del cuello de botella real.
- **Hipótesis validadas por sprint:** indicador del valor estratégico entregado, no solo del volumen producido.

Infografía de referencia



Bibliografía

Dell'Acqua, F. et al. (2025). “The Cybernetic Teammate”, *Harvard Business School Working Paper* No. 25-043.

Fernandes, D. (2025). “The AI Teammate Framework”, Scrum.org.

GitHub Blog (2025). *Spec-driven development with AI*.

Isobe, Y. (2025). *Agile in the Age of AI: A Practitioner's Guide to Evolving Scrum*.

Jocham, R. & Sutherland, J. (2026). *AI and Scrum: An Evidence-Informed Integration Guide*, v2026.1.

Liu Shangqi / Thoughtworks (2025). *Spec-Driven Development*.

Schwaber, K. & Sutherland, J. (2020). *The Scrum Guide*.

Scrum.org

- (2025). *The Agile AI Manifesto*.
- (2026). *AI Augmented Scrum Framework*.

Sutherland, J., Coleman, J. & Jocham, R. (2025). *Scrum Guide Expansion Pack*.

Takeuchi, H. & Nonaka, I.

- (1986). "The New New Product Development Game", *Harvard Business Review*.
- (1995). *The Knowledge-Creating Company*, Oxford University Press.