

v. 5.0 (2026)



SCRUM MASTER



Scrum Manager®

Scrum Master

Gestión ágil de proyectos, equipos y organizaciones

Versión 5.0 (2026)

Scrum Manager ®

Fecha de la versión: octubre 2026.

Autor de la versión: equipo de Scrum Manager®.

Ilustraciones: María de la Fuente Soro y Ana Andrés Soria.

Uncovering Better Ways SLU es la editora y propietaria de los derechos de distribución, que libera en los términos de la licencia Creative Commons Atribución – No Comercial 4.0 Internacional (CC BY-NC 4.0). Los formadores y centros oficiales de Scrum Manager quedan licenciados bajo los términos CC BY 4.0 para su actividad formativa.

Derechos registrados en Safe Creative. N° de registro: 2610167341120

Índice de contenidos

Índice de contenidos.....	3
Prefacio	5
Introducción	7
Agilidad.....	7
Procesos y gestión predictiva.....	9
El <i>Manifiesto Ágil</i>	10
Los doce principios del <i>Manifiesto</i>	13
Origen de scrum	14
Desarrollo, trabajo y conocimiento.....	15
Gestión predictiva y gestión evolutiva	17
Prácticas, principios y valores	18
La IA en el trabajo del conocimiento.....	19
Lo que cambia y lo que permanece.....	21
Parte I — El ciclo scrum	23
El ciclo scrum.....	24
El empirismo.....	26
Comprometidos e implicados	27
El equipo scrum y la IA	28
El product owner	30
Los desarrolladores.....	31
El scrum master	32
Los artefactos de scrum	33
Unidades de trabajo.....	35
El product backlog	36
El sprint backlog	37
El incremento	38
La Definición de Hecho	39
La Definition of Ready.....	40
Los eventos de scrum.....	41
El sprint.....	42
La reunión de planificación del sprint.....	43
Planificar con la IA en el equipo.....	44

El scrum diario.....	46
La revisión del sprint.....	48
La retrospectiva del sprint	49
Medir el avance.....	50
Estimar.....	52
Producción y resultado.....	55
Parte II — Principios y valores	57
Principios y valores scrum	58
Los principios de scrum.....	59
Los valores de scrum	60
Las personas por encima de los roles	61
Artefactos y eventos al servicio de los principios	62
Métodos de trabajo para la calidad	63
Trabajo en pareja.....	65
Parte III — El oficio del scrum master en equipos con IA.....	67
Un rol que evoluciona	67
Facilitar el sistema de trabajo humano-IA	68
Acuerdos de uso de la IA en el equipo	69
Alfabetización en IA.....	70
Calidad y trabajo de relleno	71
Impedimentos en equipos con IA	72
Aprendizaje y desarrollo de las personas.....	73
Seguridad psicológica y cambio	74
Antipatrones en equipos con IA	75
Scrum con IA fuera del software.....	76
Apéndice	78
Glosario.....	78
Referencias bibliográficas.....	81
Información, recursos y comunidad profesional.....	84
Mejora continua y control de calidad.....	84

Prefacio

Presentamos el temario de la [certificación oficial Scrum Master de Scrum Manager®](#). En él se explica cómo implantar y mejorar *scrum* aplicado a la gestión ágil de proyectos, equipos y organizaciones, en el escenario en que hoy trabajan la mayoría de ellos: con la inteligencia artificial integrada en el día a día.



Nos dirigimos a todas las personas interesadas en el modelo de gestión ágil denominado *scrum*, bien para aplicarlo en su trabajo diario o en el de su equipo, bien para aprender a gestionar de forma ágil proyectos de cualquier tipo.

Aunque este modelo surgió en empresas del sector tecnológico, hoy se encuentra en todo tipo de entornos innovadores, y el factor común es siempre la producción de conocimiento, la inestabilidad y el cambio rápido y constante. El temario es adecuado si se trabaja en cualquiera de las llamadas «empresas del conocimiento» y, sobre todo, si **el valor de lo que se produce depende del talento de personas motivadas**, más que de los procesos y las herramientas que emplean.

Esta versión integra en todo el temario el uso de la inteligencia artificial en los equipos. La IA no ocupa un capítulo aparte, sino que aparece allí donde cambia algo: en los roles, en los artefactos, en los eventos y en el trabajo del propio *scrum master*. La postura de la guía es que los principios y valores de *scrum* se mantienen y cobran incluso más importancia, mientras que muchas prácticas evolucionan, y que cada equipo debe decidir cómo hacerlo en su contexto. Los ejemplos proceden tanto del desarrollo de *software* como de otros equipos del conocimiento: *marketing*, recursos humanos, asesoría legal, educación u operaciones.

El contenido está dividido en cuatro bloques:

- La **introducción** contextualiza el nacimiento de *scrum*, define qué es la agilidad en el contexto empresarial y explica qué ha cambiado con la llegada de la IA. También presenta la diferencia que Scrum Manager® marca entre «prácticas» y «principios y valores» ágiles, clave para lo que viene a continuación.
- La **primera parte** se centra en las **prácticas más extendidas de *scrum***: los roles, los artefactos y los eventos que se han asentado como estándar, y lo que la IA cambia en cada uno.

- La **segunda parte** ahonda en los **principios y valores** de los que surgen esas prácticas, y en algunas técnicas que, sin formar parte del marco estándar, completan el inventario de herramientas de gestión ágil.
- La **tercera parte** trata del **oficio del scrum master en equipos que trabajan con IA**: un rol que evoluciona y que gana nuevos ámbitos de responsabilidad.

Las prácticas que describe la guía no tienen todas el mismo grado de asentamiento. Junto a las que forman parte del conocimiento consolidado de la profesión, la integración de la IA ha traído otras que están ganando adopción con rapidez y otras que acaban de aparecer. Las secciones que tratan de estas últimas se señalan con un recuadro al comienzo, con una de dos etiquetas:

- **Tendencia en consolidación**: práctica que gana adopción con rapidez, pero que aún no es general.
- **Tendencia emergente**: práctica reciente, de alta relevancia y alta volatilidad.

El contenido de esas secciones es información profesional sobre la tendencia del sector en el momento de publicar esta versión (octubre 2026). Conviene conocerlo, pero **no forma parte del examen de certificación**.

Al hablar sobre scrum se manejan muchos anglicismos y términos que adquieren un significado concreto dentro de la gestión ágil. El temario sigue tres convenciones:

- Las palabras inglesas van en cursiva la primera vez que aparecen y en redonda a partir de entonces.
- Los conceptos que cuentan con una traducción asentada se emplean traducidos, con el término original indicado entre paréntesis la primera vez.
- Los nombres de los conceptos del oficio, como scrum, *sprint* o *product owner*, se escriben en minúscula; conservan sus mayúsculas los nombres propios y la Definición de Hecho (*Definition of Done*, DoD).

Introducción

Agilidad

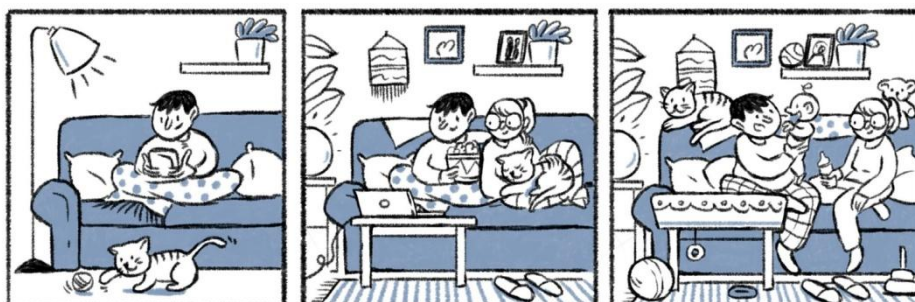
La gestión ágil surgió como antítesis de un modelo de gestión al que haremos referencia con frecuencia: la gestión de proyectos predictiva. Ambas tienen sus virtudes y resultan más útiles en ciertas industrias. La predictiva se centra en planificar, en calcular un presupuesto y en marcar plazos de entrega. Si el proyecto se termina en la fecha acordada, sin exceder el coste y con todas las funcionalidades del plan inicial, se considera un éxito.

Por muy razonable que suene esta estrategia, en las industrias que evolucionan de forma rápida y constante tiene muchos inconvenientes. Esa definición del éxito sirve en un entorno estable, con productos que son resultado de una atención escrupulosa a procesos y protocolos.

La **gestión predictiva** es fruto de la Revolución Industrial y viene del mundo de la construcción, de la automoción y de las fábricas. Si lo que el cliente busca es una casa, esta tendrá que construirse de forma que sea sólida y segura y cumpla con las necesidades de sus habitantes, y en un escenario ideal, dentro del plazo previsto y sin exceder el coste.

Hoy se fabrican y venden productos que no tienen nada que ver con una casa. Pueden ser abstractos, como una película, una *app* móvil o una campaña de comunicación. Se pueden probar cosas nuevas durante el desarrollo y ver de forma empírica lo que funciona y lo que no, hacer ajustes sobre la marcha y partir de un primer esbozo con lo básico para ir creciendo.

El escenario puede cambiar, y una funcionalidad que parecía esencial al principio puede estar desfasada en la fecha de entrega; o un competidor puede lanzar una novedad que obligue a revisar las prioridades. Para ser competitivo se necesita responder rápido en escenarios inciertos, sin requisitos estables, con clientes que necesitan usar el producto cuanto antes y mejorarlo de forma continua, y en productos en los que la innovación es un valor clave.



Estas razones, y otras que veremos, llevaron a cuestionar los modelos de gestión predictiva, que no encajaban con lo que se necesitaba en las llamadas «empresas del conocimiento»: las que desarrollan productos o servicios basándose en el conocimiento más que en las herramientas y los procesos.

En esos mercados es inútil pretender iniciar un proyecto con un plan cerrado. Se necesitan estrategias que entreguen resultados tangibles pronto y que permitan responder a tiempo a los cambios, porque el producto se construye al mismo tiempo que se modifican e introducen nuevos requisitos. Hay clientes que necesitan una primera versión con lo mínimo en cuestión de semanas, y no un producto completo dentro de uno o dos años, para poner en el mercado un concepto nuevo y desarrollar su valor de forma continua.

De dónde venimos y por qué la agilidad se suele asociar con la informática

El conocimiento evoluciona siguiendo un patrón dialéctico de tesis, antítesis y síntesis. A cada tesis le surge una antítesis que pone en evidencia sus problemas y contradicciones. La antítesis también resulta inadecuada de alguna manera, y de la confrontación de las dos surge un tercer momento, la síntesis, que ofrece una nueva comprensión del problema.

Es en este momento que la tesis y antítesis anteriores se reconcilian y trascienden. No obstante, con el tiempo, incluso la síntesis resultará tendenciosa en algún aspecto. Servirá entonces como tesis para un nuevo movimiento dialéctico, y así el proceso continúa de forma zigzagueante en espiral.

— Nonaka y Toyama, 2004

Los marcos de prácticas ágiles no surgieron como una tesis de conocimiento, sino como antítesis del modelo que la ingeniería del software venía desarrollando. Por eso, aunque hoy se aplican en todo tipo de trabajos, su historia está ligada a la informática.

Procesos y gestión predictiva

En 1968, durante la llamada «crisis del software», la OTAN celebró la primera conferencia centrada en analizar los problemas de la programación. Se puso de manifiesto la necesidad de una disciplina que aplicara un enfoque sistemático y cuantificable al desarrollo, la operación y el mantenimiento de los sistemas informáticos, y así surgió la «ingeniería del software» (Naur y Randell, 1969). Esta primera estrategia, la tesis, se basó en dos pilares:

- **Ingeniería de procesos.** En la producción industrial existía un principio de calidad contrastado con éxito: «la calidad del resultado depende de la calidad de los procesos empleados». Dicho de otra forma, no hace falta gente brillante; mientras los procesos sean de calidad, el resultado también lo será.
- **Gestión predictiva.** Un tipo de gestión que se centra en garantizar el cumplimiento de agendas y presupuestos.

Mientras la disciplina se perfeccionaba a través de modelos de procesos y cuerpos de conocimiento para la gestión de proyectos (MIL-Q9858, ISO 9000, ISO 9000-3, ISO 12207, SPICE, SW-CMM...), en la industria del software surgían dudas sobre esta estrategia. Desde mediados de los noventa hasta 2005-2010 han sido habituales las posturas radicales entre los defensores de los modelos de procesos (tesis) y los de los marcos ágiles (antítesis).

La diferencia entre un atracador de bancos y un teórico de CMM es que con el atracador se puede negociar.

— Orr, 2002

La evaluación en CMM depende más de una buena presentación en papel que de la calidad real del producto de software. Tiene que ver más con el seguimiento a ciegas de una metodología que con el desarrollo y puesta en producción de un sistema en el panorama tecnológico.

— Orr, 2002

Si uno pregunta a un ingeniero de software típico si cree que CMM se puede aplicar a los métodos ágiles, responderá o con una mirada de sorpresa o con una carcajada histérica.

— Turner y Jain, 2002

No estaba claro que la planificación predictiva fuese apropiada para cualquier proyecto. En la práctica, los criterios del éxito no siempre son el cumplimiento de fechas, costes y funcionalidades preestablecidas. Además, se cuestionaba que el software, como otros trabajos basados en el conocimiento, pudiera producirse con patrones de procesos industriales.

Se empezó a aceptar que el conocimiento tácito de quien realiza el trabajo puede aportar más al valor del resultado que la tecnología y los procesos empleados. Lo veremos con detalle en «Desarrollo, trabajo y conocimiento».

El Manifiesto Ágil

Estamos poniendo al descubierto mejores métodos para desarrollar software, haciéndolo y ayudando a otros a que lo hagan. Con este trabajo hemos llegado a valorar: a los individuos y su interacción, por encima de los procesos y las herramientas; el software que funciona, por encima de la documentación exhaustiva; la colaboración con el cliente, por encima de la negociación contractual; la respuesta al cambio, por encima del seguimiento de un plan. Aunque hay valor en los elementos de la derecha, valoramos más los de la izquierda.

— *Manifiesto por el Desarrollo Ágil de Software, 2001*

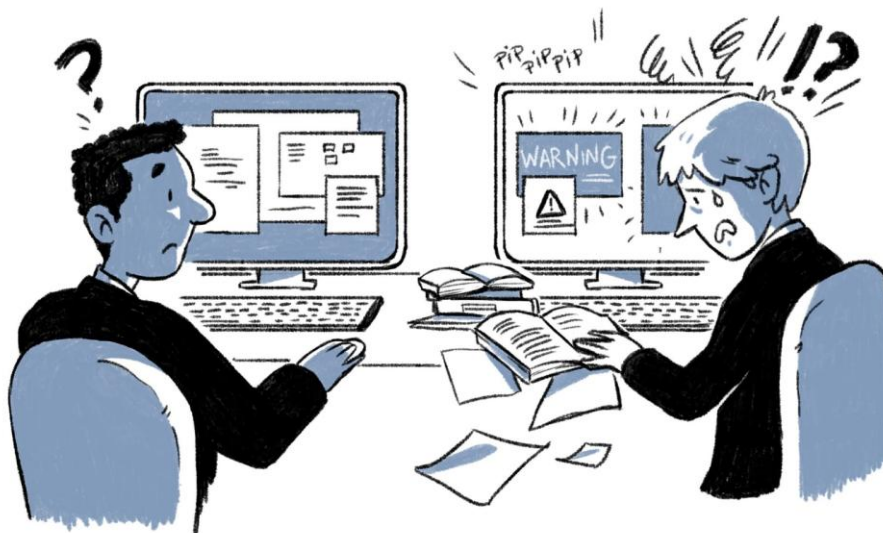
En febrero de 2001, diecisiete profesionales del software se reunieron en la estación de esquí de Snowbird, en Utah, convocados por Robert C. Martin. La reunión tenía un precedente en el encuentro que Kent Beck, autor del libro que presentó la metodología *Extreme Programming* (Beck, 1999), había organizado en Oregón la primavera anterior (Highsmith, 2001). Todos tenían algo en común: eran críticos con los modelos de producción basados en procesos.

En la reunión se acuñó el término «métodos ágiles» para los que estaban surgiendo como alternativa a las metodologías formales (CMM-SW, precursor de CMMI; PMI; SPICE, precursor de ISO 33000...), que consideraban pesadas y rígidas por su carácter normativo y su dependencia de planificaciones detalladas previas al desarrollo. Los participantes resumieron en cuatro postulados lo que conocemos como *Manifiesto Ágil*, los valores sobre los que se asientan estos métodos, y establecieron además doce principios.

«Valoramos más a los individuos y su interacción que a los procesos y las herramientas.» Es el postulado más importante. Los procesos ayudan como guía de operación, y las herramientas adecuadas mejoran la eficiencia, pero hay tareas que requieren talento y necesitan personas motivadas que lo aporten. En el desarrollo ágil los procesos son una ayuda para guiar el trabajo, y la calidad del resultado depende sobre todo del conocimiento de las personas.



«**Valoramos más el software que funciona, por encima de la documentación exhaustiva.**» El *Manifiesto* no considera inútil la documentación, solo la innecesaria. Los documentos registran y comunican información relevante y a veces son obligatorios por razones legales, pero su relevancia debe ser menor que la del producto. Observar prototipos y partes terminadas aporta un *feedback* que genera ideas imposibles de concebir al principio, mientras que un documento de requisitos muy detallado antes de empezar suele ser una pérdida de tiempo y levanta barreras entre departamentos.

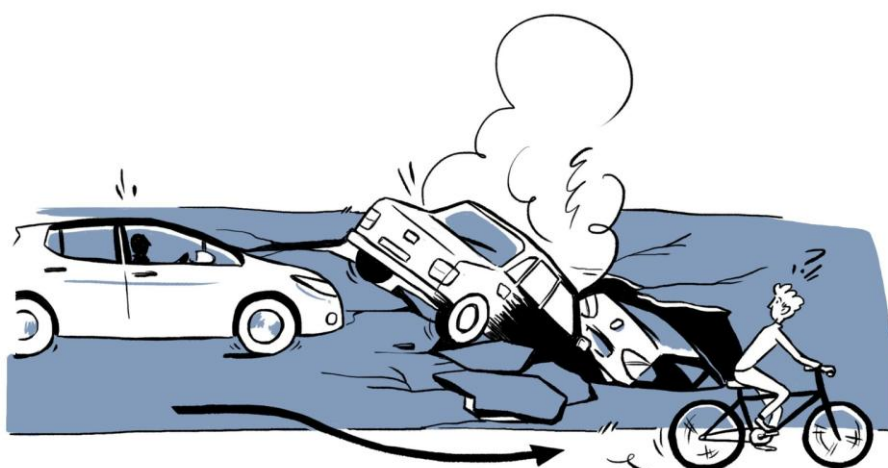


«**Valoramos más la colaboración con el cliente que la negociación contractual.**» El objetivo de un proyecto ágil es proporcionar de forma continua el mayor valor posible al producto, más que controlar que se cumplen los planes iniciales. En productos en

evolución continua no se puede fijar en un documento cerrado cómo será el resultado final; es más eficiente tomar feedback directo mientras se desarrolla y redefinir los requisitos de lo que queda. Para ello, el cliente acompaña al equipo durante el proceso.



«**Valoramos más la respuesta al cambio que el seguimiento de un plan.**» Los valores principales de la gestión ágil son la anticipación y la adaptación, frente a la planificación y el control de la gestión ortodoxa. En productos de requisitos inestables, en los que el cambio es inherente, la capacidad de respuesta resulta más valiosa que la de asegurar el cumplimiento de un plan.



Los doce principios del *Manifiesto*

Además de los cuatro postulados que acabamos de ver, el *Manifiesto Ágil* establece doce principios. Se escribieron pensando en el software, pero su sentido vale para cualquier producto del conocimiento:

1. Nuestra principal prioridad es satisfacer al cliente a través de la entrega temprana y continua de software de valor.
2. Son bienvenidos los requisitos cambiantes, incluso si llegan tarde al desarrollo. Los procesos ágiles se dobligan al cambio como ventaja competitiva para el cliente.
3. Entregar con frecuencia software que funcione, en periodos de un par de semanas hasta un par de meses, con preferencia en los periodos breves.
4. Las personas del negocio y los desarrolladores deben trabajar juntos de forma cotidiana a través del proyecto.
5. Construcción de proyectos en torno a individuos motivados, dándoles la oportunidad y el respaldo que necesitan y procurándoles confianza para que realicen la tarea.
6. La forma más eficiente y efectiva de comunicar información de ida y vuelta dentro de un equipo de desarrollo es mediante la conversación cara a cara.
7. El software que funciona es la principal medida del progreso.
8. Los procesos ágiles promueven el desarrollo sostenido. Los patrocinadores, desarrolladores y usuarios deben mantener un ritmo constante de forma indefinida.
9. La atención continua a la excelencia técnica y el buen diseño enaltecen la agilidad.
10. La simplicidad como arte de maximizar la cantidad de trabajo que no se hace, es esencial.
11. Las mejores arquitecturas, requisitos y diseños emergen de equipos que se autoorganizan.
12. En intervalos regulares, el equipo reflexiona sobre la forma de ser más efectivo y ajusta su conducta en consecuencia.

Veinticinco años después, varios de estos principios han ganado actualidad. Cuando una parte del trabajo la produce la IA, la medida del progreso sigue siendo el resultado que funciona y no el volumen producido (principio 7), la excelencia técnica y la simplicidad (principios 9 y 10) protegen de acumular trabajo que nadie ha revisado, y la reflexión periódica del equipo (principio 12) es el mecanismo que permite ajustar cómo se integra la IA. Volveremos sobre ellos a lo largo de la guía.

Origen de scrum

Scrum es un modelo de desarrollo ágil caracterizado por:

- Equipos autónomos y autogestionados que comparten su conocimiento de forma abierta y aprenden juntos.
- Una estrategia de desarrollo incremental, en lugar de la planificación completa del producto.
- Basar la calidad del resultado en el conocimiento tácito de las personas y su creatividad, y no en la calidad de los procesos empleados.
- Solapar las diferentes fases del desarrollo, en lugar de realizarlas una tras otra en un ciclo secuencial o «de cascada».

El origen de la palabra se encuentra en un ámbito muy alejado de la gestión de proyectos, el deporte. En *rugby*, «scrum» es la formación en la que ambos equipos, agazapados y atenazados entre sí, empujan para obtener el balón sin tocarlo con la mano.

Para lo que a nosotros nos interesa, tenemos que desplazarnos al Japón de los años ochenta, cuando los investigadores Ikujiro Nonaka y Hirotaka Takeuchi dieron una dimensión nueva al término. Identificaron una forma de desarrollo en las empresas de manufactura que obtenían los mejores resultados de innovación y tiempo de salida al mercado (Fuji Xerox, Canon, Honda, NEC, Epson, Brother, 3M y Hewlett-Packard) y compararon su trabajo en equipos autogestionados con el avance en formación de los jugadores de rugby (Takeuchi y Nonaka, 1986).

Aunque esta forma de trabajo surgió en la manufactura de productos tecnológicos, a partir de 1995 se empezó a aplicar también al software. Ese año, Ken Schwaber presentó en la conferencia OOPSLA (*Object-Oriented Programming, Systems, Languages & Applications*) una metodología de desarrollo de software basada en un ambiente scrum, con ese mismo término (Schwaber, 1995). Aquel primer marco tenía fases y artefactos (*pregame, game, postgame, planning, sprints, wrap...*); algunos se han mantenido, pero las reglas del juego han cambiado mucho desde entonces.

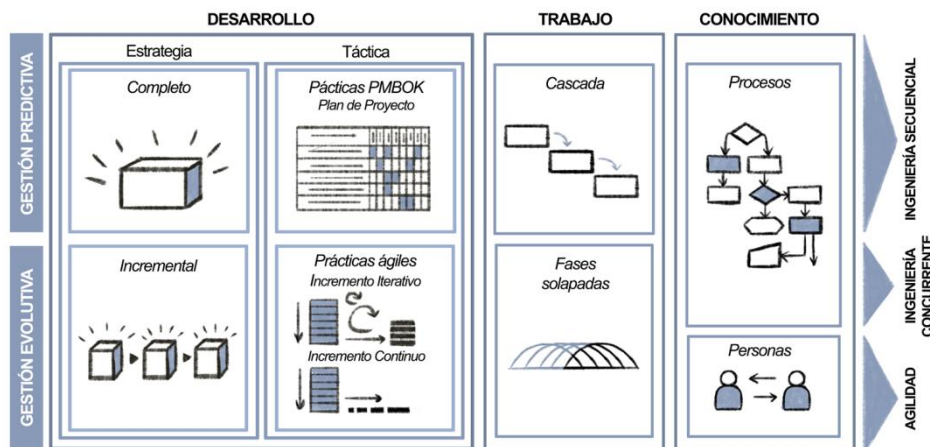
No existe una autoridad que determine lo que es scrum y lo que no. Ha cambiado y seguirá evolucionando con las aportaciones de la comunidad de profesionales, que define las prácticas que resultan más útiles. Entre las incorporadas con el tiempo están las retrospectivas, el refinamiento del *product backlog*, la *Definition of Ready* (DoR) y los mapas de historias (*story maps*), y la integración de la IA es el cambio más reciente de esa evolución.

El espíritu original se mantiene: las prácticas deben ayudar a los equipos a autogestionarse y a mantener un flujo de avance continuo, produciendo resultados de forma iterativa y frecuente. Scrum Manager® usa el término «scrum» con el significado original que le dieron Nonaka y Takeuchi.

Desarrollo, trabajo y conocimiento

Desde los años ochenta se han desarrollado tantos modelos y prácticas para mejorar la calidad y la eficiencia de los proyectos que el panorama puede resultar abrumador. En este capítulo vamos a trascender las etiquetas y a resumir la base de esos marcos, los principios que subyacen y las estrategias con que se desarrollan.

Usaremos como coordenadas tres conceptos (desarrollo, trabajo y conocimiento) y dos modelos de gestión ya mencionados, la gestión predictiva y la gestión evolutiva. Con estas cinco ideas se despeja el aparente laberinto de marcos de trabajo, porque todos ellos se pueden situar en este diagrama.



Desarrollo

El desarrollo de un proyecto puede ser completo o incremental.

- **En un desarrollo completo**, la descripción de lo que se desea obtener está disponible al inicio; es completa y detallada y sirve de base para estimar, planificar la agenda y los recursos, y gestionar después el cumplimiento de lo previsto.
- **En un desarrollo incremental**, la descripción completa no está disponible al inicio, sino que se complementa y evoluciona durante el desarrollo. Se puede gestionar con dos tácticas:
 - **Desarrollo incremental continuo**, con técnicas que logran un flujo continuo de funcionalidades o partes del producto que se entregan al cliente según se terminan.
 - **Desarrollo iterativo**, con técnicas de tiempo prefijado (*timeboxing*) que mantienen la producción de incrementos a un ritmo fijo. Es el que emplea el marco estándar de scrum, que llama sprint a cada iteración (→«El sprint»); al final de cada una se produce al menos un incremento del producto, una parte entregable y lista para usarse.

Trabajo

La forma de trabajar puede ser secuencial («en cascada») o concurrente.

- **El trabajo secuencial divide el trabajo en fases**, y cada fase comienza cuando termina la anterior. El ejemplo más conocido es el ciclo de cascada de la ingeniería del software: requisitos, análisis, diseño, codificación, pruebas e implantación.
- **El trabajo concurrente solapa en el tiempo las diferentes fases**, de modo que todas se revisan de forma simultánea y continua.

Conocimiento

Los modelos pueden situar el conocimiento en los procesos o en las personas.

- **En una producción basada en procesos**, el conocimiento es explícito y la calidad del resultado reside, en mayor medida, en los procesos y la tecnología empleados.
- **En una producción basada en las personas**, el conocimiento es tácito y la calidad del resultado depende de la experiencia y el talento de quienes hacen el trabajo.

La diferencia se ve al comparar la cena preparada con un robot de cocina con la que cocina una persona de forma libre. Cualquiera puede preparar la primera siguiendo las instrucciones, y el resultado será el mismo sin importar su habilidad. En la segunda prima el talento, y un aficionado no prepara lo mismo que un chef. Ambos se benefician de buenas herramientas, como una sartén que no se pegue o un cuchillo bien afilado, pero son solo una ayuda.



El conocimiento tácito es personal, específico del contexto, y por tanto difícil de formalizar y comunicar. El conocimiento explícito o «codificado», por otro lado, es conocimiento que puede transmitirse con lenguaje formal y sistemático.

— Nonaka y Takeuchi, 1995

Esta distinción será útil más adelante para entender qué aporta la IA a un equipo y qué no puede aportar (—«La IA en el trabajo del conocimiento»).

Gestión predictiva y gestión evolutiva

Con las tres coordenadas anteriores podemos describir los dos grandes modelos de gestión y situar scrum entre ellos.

Ingeniería secuencial

La ingeniería secuencial, también llamada gestión predictiva, tiene como objetivo ofrecer resultados predecibles. Un proyecto tiene éxito si desarrolla el producto previsto sin exceder el plazo ni los recursos acordados. Su desarrollo es completo y emplea prácticas de planificación tradicional. En el mundo del software, sus principales referentes son PMI e IPMA y los modelos de procesos CMMI, ISO 33000 o SPICE, todos con ingeniería secuencial y producción basada en procesos.

Gestión evolutiva: ingeniería concurrente y agilidad

La gestión evolutiva tiene como objetivo entregar cuanto antes un producto mínimo viable e incrementar su valor de forma continua. Solapa las fases de trabajo y desarrolla de forma incremental, con iteraciones breves y cíclicas o con un flujo constante.

Puede llevarse a cabo con producción basada en procesos (ingeniería concurrente) o en personas (agilidad). La distinción es importante, porque sin ella se llega a considerar agilidad la simple aplicación de las reglas estándar de scrum, o el uso de técnicas visuales *kanban* para mantener un flujo continuo.

Ingeniería concurrente	Agilidad
Emplea recursos propios de la gestión ágil: solapamiento de fases de desarrollo, equipos multidisciplinares e iteraciones frecuentes de mejora.	Reduce o elimina tareas administrativo-burocráticas que no aportan valor al producto o al sistema de desarrollo.
Se centra en la calidad de los procesos.	Propia de las empresas del conocimiento. Se centra en el conocimiento tácito de las personas, en la cultura y el talento.

Vídeo explicativo: [Gestión predictiva y evolutiva](#) (—«Referencias bibliográficas»).

Vídeo explicativo: [Objetivo de la gestión predictiva y de la gestión ágil](#) (→ «Referencias bibliográficas»).

Scrum

Resumiendo, volvemos a encontrarnos con las características de scrum que vimos al hablar de su origen, y vemos cómo encajan en la gestión ágil:

- Utiliza una estrategia de desarrollo incremental, iterativo con tiempo prefijado o continuo.
- Solapa las diferentes fases del desarrollo.
- Basa la calidad del resultado en el conocimiento tácito de las personas y su creatividad.
- Trabaja con equipos autónomos y autogestionados, que comparten su conocimiento y aprenden juntos. De ahí el nombre y la metáfora de «avanzar en scrum».

Prácticas, principios y valores

Cuando se empieza a trabajar con scrum, como con cualquier otra herramienta, es recomendable leer el manual de Scrum Master y seguir las instrucciones, es decir, adoptar el marco estándar. En la primera parte explicamos en qué consiste y cuáles son los roles, artefactos y eventos que lo configuran. El marco es una plantilla, pero no un molde. **No son reglas** que deban seguirse al pie de la letra para garantizar velocidad o calidad, sino un buen **punto de partida** para empezar a trabajar de forma iterativa.

Conviene no engañarse: **si nuestro foco está puesto en el proceso, en seguir las reglas, y no en el conocimiento tácito y el talento de las personas, no se está haciendo agilidad, sino ingeniería concurrente**. La agilidad requiere confianza en las capacidades del equipo. Las herramientas y los procesos deben ayudar a gestionar el trabajo, no a las personas, porque el objetivo final es que los equipos sean capaces de **autogestionarse**.

Esta advertencia vale también para la IA, la herramienta más potente que ha entrado nunca en un equipo. Usarla para automatizar el cumplimiento de las reglas, o para producir más sin pensar mejor, aleja al equipo de la agilidad en lugar de acercarlo.

Cuando se alcanza un flujo de avance iterativo, se puede ir más allá del marco estándar. Llega el momento de desaprender las prácticas y de apoyarse en los principios y valores de scrum para adaptar el marco y otras técnicas a las características del proyecto o del equipo. En la mayoría de empresas ágiles estas prácticas se adaptan, y la integración de la IA obliga a todas a hacerlo.

La primera parte desarrolla las técnicas más extendidas de scrum, el marco estándar que puede encontrarse aquí y en otros manuales. En la segunda explicaremos cómo quitar los ruedines de la bici, que vienen muy bien al principio pero pueden entorpecer a la larga, y en la tercera veremos qué exige hoy el oficio de scrum master.



La IA en el trabajo del conocimiento

Desde 2022, la inteligencia artificial generativa ha entrado en la mayoría de los trabajos del conocimiento y, desde 2024, se han extendido los agentes capaces de ejecutar tareas completas con poca intervención humana. Hoy la IA redacta borradores de campañas e informes, resume documentación legal, analiza datos, propone diseños y escribe código. El resultado es un cambio en la forma de producir valor, que afecta a todos los equipos aunque no a todos por igual.

Volvamos al robot de cocina del capítulo «Desarrollo, trabajo y conocimiento». La IA es, en buena medida, conocimiento explícito a una escala desconocida hasta ahora: reproduce y combina con enorme rapidez lo que puede expresarse con lenguaje formal. Lo que no aporta por sí sola es el conocimiento tácito, es decir, el juicio sobre qué merece hacerse, la comprensión del contexto y la responsabilidad sobre el resultado. Esa es la parte que sigue en manos de las personas.

El cuello de botella se desplaza

Durante toda la historia de la gestión ágil, el punto estrecho del trabajo ha estado casi siempre en la ejecución. Escribir el código, redactar el documento o producir la pieza costaba tiempo y esfuerzo, y la gestión se organizaba en torno a ese esfuerzo para aprovecharlo, estimarlo y medirlo.

Cuando ejecutar se abarata, el cuello de botella se traslada a otras actividades: **decidir qué merece hacerse, revisar lo que se ha producido y validar que resuelve un problema real.** Un equipo de marketing puede generar en una tarde veinte variantes de una campaña, pero alguien tiene que elegir cuáles encajan con la marca y comprobar que no prometen lo que el producto no cumple. Un equipo de software puede generar en

horas el código de una funcionalidad, pero alguien tiene que revisarlo, integrarlo y asegurarse de que es la funcionalidad que hacía falta.

La IA amplifica el sistema de trabajo

La IA no es neutral. Magnifica las condiciones que ya existen en el equipo: donde hay buenas prácticas, entrega frecuente y criterios de calidad claros, acelera la entrega de valor; donde hay procesos frágiles y poca claridad sobre qué construir, produce más trabajo sin validar y agrava los problemas. El estudio DORA de 2025, con unos 5.000 profesionales del software, situó el uso de la IA en el 90 % y concluyó que su efecto principal es el de un amplificador de las fortalezas y debilidades que la organización ya tiene (DORA, 2025).

Por eso los mecanismos que permiten ver pronto qué está pasando y corregir a tiempo ganan valor con la IA en lugar de perderlo, y son precisamente los de scrum: trabajar en incrementos pequeños, inspeccionar con frecuencia el resultado y la forma de trabajar, y adaptarse a lo que se descubre.

Productividad percibida y productividad medida

La sensación de ir más deprisa no siempre coincide con la realidad. En un estudio experimental de 2025, un grupo de desarrolladores expertos tardó un 19 % más en completar sus tareas con IA que sin ella, mientras creía haber ido un 20 % más rápido (Becker y otros, 2025). Los propios autores advierten de que el resultado corresponde a un contexto concreto y a las herramientas de aquel momento, pero la lección es general: la productividad hay que medirla, porque la percepción engaña.

Tampoco se trata de negar la ganancia. En un experimento con equipos de innovación de producto, una persona que trabajaba con IA alcanzó una calidad comparable a la de una pareja que trabajaba sin ella (Dell'Acqua y otros, 2025). La IA aporta capacidad real, y **la cuestión para un equipo es convertir esa capacidad en valor y no solo en volumen.**

El análisis detallado de este cambio, sus efectos sobre la gestión ágil y los criterios para integrar la IA en cualquier tipo de equipo tiene temario propio en Scrum Manager: [Agile Bedrock](#).

Lo que cambia y lo que permanece

Ante el impacto de la IA, la comunidad ágil se ha dividido. Una postura sostiene que las prácticas de scrum han llegado al final de su recorrido: los ciclos de varias semanas serían lentos, los equipos se reducen y los roles cambian o desaparecen. La otra sostiene que la IA multiplica la agilidad y que sus principios son más necesarios que nunca, porque la velocidad exige más reflexión.

La postura de Scrum Manager toma de cada una lo que tiene de acertado. **Los principios ágiles mantienen su vigencia; muchas prácticas necesitan una transformación profunda**, y esa transformación la hace cada equipo según su contexto, sin esperar a que alguien se la prescriba. En este capítulo adelantamos qué permanece y qué cambia, y dónde lo trataremos.

Lo que permanece

Permanece el **empirismo**: la transparencia sobre el estado real del trabajo, la inspección frecuente y la adaptación a lo que se descubre. Permanece la **entrega temprana y continua de valor**, y con ella la idea de que el progreso se mide por resultados útiles y no por el volumen de lo producido.

Permanece también la **responsabilidad humana**. La IA puede proponer, redactar o ejecutar, pero las decisiones sobre el producto y la responsabilidad sobre el resultado corresponden siempre a personas. Y permanecen los **valores del Manifiesto****, que la IA vuelve más críticos. La conversación entre personas sigue siendo insustituible, y cuando construir lo equivocado es rápido y barato, la colaboración con el cliente es la mejor defensa. Responder al cambio, por su parte, sigue exigiendo momentos de reflexión que ninguna herramienta proporciona por sí sola.

Lo que cambia

Cambia la composición del trabajo del equipo, que incorpora la IA como una capacidad más bajo responsabilidad humana (—«El equipo scrum y la IA»). Cambian los criterios de calidad, porque lo que produce la IA necesita revisión explícita antes de darse por terminado (—«La Definición de Hecho»). Cambian algunos aspectos de los eventos, de la medición del avance y del propio rol de scrum master, que trataremos en la primera y la tercera parte.

Ninguno de estos cambios es igual en todos los equipos. El efecto de la IA es máximo en el desarrollo de software, donde puede generar y probar código, y más gradual en otros trabajos del conocimiento, donde suele asistir en la redacción, el análisis o la síntesis mientras la persona sigue haciendo el trabajo central. Un equipo de recursos humanos o de asesoría legal, además, trabaja a menudo con información sensible y con exigencias regulatorias que limitan qué puede delegarse.

Dónde se profundiza

Esta guía trata de scrum y del oficio de scrum master. Dos temarios de Scrum Manager desarrollan aspectos que aquí solo resumimos:

- [Agile Bedrock](#): los fundamentos de la gestión ágil en la era de la IA, la integración de la asistencia inteligente en cualquier tipo de equipo y los criterios para decidir la forma de trabajar según el contexto.
- [Product Owner](#): el oficio de decidir qué construir, el descubrimiento de producto, la gestión del product backlog y la medición del valor.

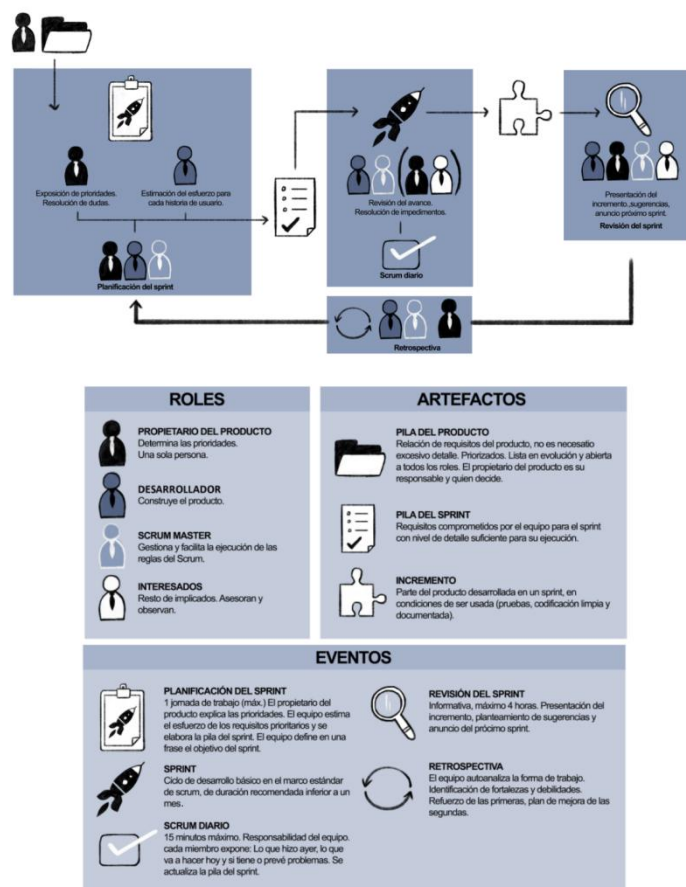
Parte I — El ciclo scrum

Aprendiendo las prácticas estándar.



El ciclo scrum

En esta primera parte veremos el marco estándar de scrum: sus roles, sus artefactos y sus eventos, y lo que la IA cambia en cada uno de ellos.



Los componentes del ciclo estándar de scrum son:

- El equipo scrum, compuesto por tres roles: product owner, desarrolladores y scrum master.
- Los artefactos: product backlog, *sprint backlog* e incremento.
- Los eventos: sprint, reunión de planificación del sprint, scrum diario, revisión del sprint y retrospectiva del sprint.

Se comienza con la visión general del resultado que se desea y, a partir de ella, se especifican y detallan las funcionalidades que se quieren obtener en primer lugar.

Cada ciclo de desarrollo o iteración (sprint) termina con la entrega de una o varias partes operativas del producto (incrementos). **La duración de un sprint no excede de cuatro semanas.** Lo habitual es que todos los sprints tengan la misma duración y marquen una cadencia, aunque esta puede **evolucionar o ajustarse**.

Vídeo explicativo: [El marco de desarrollo scrum](#) (—«Referencias bibliográficas»).

Scrum gestiona de forma empírica la evolución del proyecto con estas tácticas.

Revisión de las iteraciones

Al final de cada sprint se revisa el resultado con todos los implicados en el proyecto. Por tanto, la duración del sprint es el plazo máximo para descubrir planteamientos erróneos o mejorables y malentendidos sobre lo que se está construyendo.

Desarrollo incremental

No se trabaja con diseños o abstracciones. El desarrollo incremental ofrece al final de cada iteración una parte operativa del producto, que se puede usar, inspeccionar y evaluar.

Solapamiento de fases

Durante la construcción se depuran el diseño y la arquitectura, que no se cierran en una primera fase del proyecto. Las fases que el desarrollo en cascada realiza una tras otra se solapan en scrum y avanzan de forma simultánea.

Autogestión

La gestión predictiva asigna a un gestor de proyecto la responsabilidad de dirigirlo. En scrum los equipos se autogestionan, con un ámbito de decisión suficiente para adoptar las resoluciones que consideren oportunas, lo que agiliza la toma de decisiones y permite responder con rapidez a los imprevistos. Con la IA, ese ámbito de decisión incluye también cómo se integra la IA en el trabajo del equipo.

Colaboración

Todos los miembros del equipo colaboran de forma abierta con los demás, según sus capacidades y no según su rol o su puesto. Mediante la autogestión y la colaboración, el equipo gestiona con solvencia la labor que en otros modelos realizaría un gestor de proyectos.

El empirismo

Scrum se basa en el **empirismo**, la idea de que el conocimiento procede de la experiencia y de que las decisiones se toman a partir de lo que se observa. En lugar de confiar en un plan detallado elaborado al principio, el equipo avanza en ciclos cortos, mira el resultado y ajusta el rumbo. Este modo de trabajar se sostiene sobre tres pilares.

Transparencia

El estado real del trabajo, del producto y del proceso debe ser visible para quienes toman decisiones sobre ellos. Sin transparencia no hay inspección fiable, porque se inspeccionaría una imagen distorsionada de la realidad. Los artefactos de scrum existen, en buena medida, para hacer visible el trabajo: lo que queda por hacer, lo que se está haciendo y lo que ya está terminado.

Con la IA, la transparencia incluye un aspecto nuevo, la **procedencia** del trabajo. El equipo y las personas interesadas deben poder saber qué partes ha producido la IA, quién las ha revisado y con qué criterio. Un informe que parece terminado pero que nadie ha comprobado es un ejemplo de falta de transparencia, aunque esté a la vista de todos.

Inspección

El equipo examina con frecuencia el producto y la forma de trabajar para detectar desviaciones y problemas. La inspección se produce en los eventos de scrum, y su valor depende de que sea frecuente y de que se haga sobre información fiable.

La IA multiplica la información disponible y puede ayudar a analizarla: resumir el avance, detectar patrones o señalar riesgos. Pero generar datos no equivale a inspeccionar. La inspección exige que alguien se detenga, mire con criterio y saque conclusiones, y la velocidad de producción tiende a empujar al equipo a saltarse ese momento.

Adaptación

Cuando la inspección revela que algo se desvía de lo aceptable, el equipo ajusta el producto o su forma de trabajar cuanto antes. La adaptación es lo que convierte el aprendizaje en resultados, y requiere que el equipo tenga autonomía para decidir los cambios.

Los tres pilares se refuerzan entre sí y son, como vimos en la introducción, lo que la IA vuelve más necesario. Cuanto más deprisa se produce, más importante es ver pronto qué está pasando y corregir a tiempo.

Comprometidos e implicados

Durante el desarrollo de un proyecto intervienen muchas personas que aportan valor con distintos niveles de compromiso y responsabilidad. Para distinguirlas se suele hablar de «comprometidos» e «implicados».

- **Comprometidos:** intervienen directamente en la construcción del producto o el desarrollo del servicio y responden de su resultado. Son los miembros del equipo scrum.
- **Implicados:** otras partes interesadas que aportan información, recursos o decisiones de su ámbito, pero que no responden del resultado del sprint. Por ejemplo, la dirección, el área comercial, el marketing, quienes operan el sistema que se desarrolla o el soporte a usuarios.

La distinción no es una jerarquía de importancia. Los implicados son imprescindibles, porque sin su información el equipo no sabría qué construir ni si lo construido sirve. Lo que la distinción aclara es quién asume el compromiso con el objetivo del sprint y quién decide cómo se hace el trabajo. Las personas implicadas pueden asistir a algunos eventos y aportar su visión, pero no dirigen el trabajo del equipo.

En la práctica, la frontera no siempre es nítida, y a veces un implicado se incorpora al equipo durante un tiempo o un miembro del equipo pasa a otro proyecto. Lo importante es que cada persona sepa en qué posición está en cada momento, porque de ello dependen sus responsabilidades.

La distinción sirve también para situar a la IA, que interviene en la construcción del producto, pero no puede comprometerse ni responder de un resultado. Lo veremos a continuación.

El equipo scrum y la IA

La IA ya participa en el trabajo de muchos equipos: redacta, resume, analiza, propone y, en algunos casos, ejecuta tareas completas. Esto plantea una pregunta práctica para el marco: dónde se sitúa la IA dentro del equipo scrum.

La respuesta de esta guía es que **la IA es una capacidad del equipo, bajo la responsabilidad de las personas que lo forman**. Amplía lo que el equipo puede hacer, como lo amplía cualquier herramienta potente, pero no ocupa un rol ni asume responsabilidades. El product owner, el scrum master y los desarrolladores son siempre personas, y responden del resultado del trabajo, lo haya producido quien lo haya producido.

La responsabilidad no se delega

La IA puede proponer, redactar o ejecutar, pero no puede rendir cuentas. Cuando un agente comete un error, la responsabilidad no se diluye en un «lo hizo la IA»: debe estar claro qué persona responde de lo que la IA ha producido. En scrum esa responsabilidad es compartida por el equipo, que acepta como terminado un trabajo solo cuando cumple los criterios que el propio equipo ha acordado (—«La Definición de Hecho»).

Esta idea, que la IA propone y la persona decide y responde, recorre toda la guía. Es también la que permite usar la IA con confianza, porque la confianza se apoya en saber quién ha revisado qué.

Cuatro decisiones explícitas

Integrar la IA es una decisión de autogestión del equipo, no una configuración de la herramienta. Sea cual sea el uso que le dé, todo equipo debe acordar de forma explícita cuatro cuestiones:

1. **Qué hace la IA:** qué tareas se le delegan y cuáles se reservan a las personas.
2. **Quién revisa:** qué persona revisa lo que produce y con qué criterio.
3. **Cómo se integra:** en qué momento del flujo de trabajo interviene y cómo se incorpora su resultado.
4. **Qué límites tiene:** qué no puede hacer, a qué información no puede acceder y cuándo hay que detenerla.

Un equipo de asesoría legal, por ejemplo, puede decidir que la IA prepare la primera versión de un resumen de jurisprudencia, que lo revise siempre la persona responsable del asunto antes de enviarlo, que se use solo en la fase de investigación y que nunca reciba datos personales de los clientes. Cuando estas decisiones quedan sin tomar, aparecen tareas que nadie supervisa y que se convierten en zonas de riesgo.

Los acuerdos no se toman una vez para siempre. El equipo los revisa en la retrospectiva a medida que aprende qué funciona (—«La retrospectiva del sprint»), y el scrum master le ayuda a mantenerlos vivos (—«Acuerdos de uso de la IA en el equipo»).

La IA como miembro del equipo

Tendencia emergente (2026). El contenido de esta sección describe un debate abierto en la profesión. No entra en el examen de certificación.

Una parte de la comunidad propone considerar a los agentes de IA como miembros del equipo con su propio trabajo asignado, siempre que al menos una persona responda de su resultado. Algunos equipos ya les asignan elementos del sprint backlog y les aplican la misma Definición de Hecho que al resto del trabajo.

Otras voces consideran que la analogía es imperfecta, porque la IA no se compromete, no aprende del equipo como lo hace una persona ni participa de verdad en la reflexión sobre la forma de trabajar. Desde esta perspectiva, es una herramienta muy potente al servicio de los comprometidos.

Al margen de cómo termine el debate, las consecuencias prácticas son las mismas: las cuatro decisiones explícitas y la responsabilidad humana sobre el resultado.

El product owner

En el equipo scrum se distinguen tres roles: el product owner, los desarrolladores y el scrum master. Empezamos por el primero.

El product owner es quien toma las decisiones del cliente sobre el producto, y su responsabilidad es **maximizar el valor** de lo que construye el equipo. Para simplificar la comunicación y la toma de decisiones, este rol recae en una única persona. Si el cliente es una organización grande o con varios departamentos, puede organizar su comunicación interna como considere oportuno, pero en el equipo solo se integra una persona, que lo representa y tiene el conocimiento y las atribuciones necesarias para decidir.

En los desarrollos internos suele asumir este rol el *product manager* o la persona responsable del área que necesita el producto; en los desarrollos para clientes externos, quien dirige el proceso de adquisición en el cliente. Según las circunstancias, el product owner puede delegar tareas en el equipo o en alguien de su confianza, pero mantiene sus responsabilidades:

- Desarrollar y administrar el product backlog.
- Exponer la visión del producto y el valor esperado de cada sprint, y participar en la reunión de planificación de cada sprint (→«La reunión de planificación del sprint»).

Es quien está a cargo del product backlog. Decide en última instancia cómo será el producto y el orden en que se construyen los incrementos: qué se pone, qué se quita y qué prioridad tiene cada elemento. Conoce el plan del producto, sus posibilidades, la inversión y el retorno esperado, y comparte esa visión con el equipo.

Tiene conocimiento experto del negocio del cliente. Analiza de forma constante el entorno (la evolución del mercado, la competencia, las alternativas) y combina esa información con la que surge del equipo durante el desarrollo: sugerencias, alternativas técnicas y evaluación de cada incremento. Necesita conocer scrum y, en circunstancias ideales, haber trabajado antes con los mismos desarrolladores. La organización respeta sus decisiones y no modifica las prioridades ni los elementos del product backlog.

Enlace de interés: [Las 5 claves de un buen product owner, Scrum Manager Podcast](#) (→«Referencias bibliográficas»).

El product owner ante la IA

Cuando construir se abarata, el punto estrecho del trabajo se desplaza a decidir qué merece construirse, y ese es el territorio del product owner. Su rol gana importancia con la IA, siempre que evolucione de lo táctico a lo estratégico: un product owner que se limita a redactar y ordenar elementos del backlog hace justo la parte del trabajo que la IA más puede asistir.

Su aportación se concentra en mantener la visión, en diseñar cómo se validará lo que se construye y en decidir dónde aporta valor la IA y dónde solo genera volumen. Si el equipo puede producir diez propuestas de campaña en un sprint, el product owner necesita una forma de saber cuáles funcionan, o el equipo producirá mucho sin aprender nada. El oficio de product owner tiene temario propio en Scrum Manager: [Product Owner](#).

Los desarrolladores

Los desarrolladores son los profesionales que construyen el incremento de cada sprint. Aunque scrum los llame así, no son siempre programadores. El término engloba a todas las personas cuyo trabajo contribuye directamente a construir el incremento: redactores, diseñadores, analistas, técnicos o especialistas de cualquier área, según el producto.

El tamaño del equipo scrum no tiene una norma fija. Contando al product owner, al scrum master y a los desarrolladores, y según las circunstancias de la organización, del proyecto y del uso que el equipo haga de la IA, puede ser adecuado un equipo de tres o cuatro personas, o uno mayor, **hasta un máximo de diez**. Más allá, resulta difícil mantener la comunicación directa y se intensifican los roces de la dinámica de grupos.

Hackman y Vidmar (1970) sometieron a grupos de entre dos y siete miembros a tareas de producción, discusión y resolución de problemas, y encontraron que la insatisfacción con el tamaño del grupo crecía de forma aproximadamente lineal con este, con un punto óptimo en torno a 4,6 miembros.

Son profesionales multifuncionales que trabajan de forma solidaria con responsabilidad compartida. Alguno puede estar especializado en un área concreta, pero la responsabilidad del resultado recae sobre todos por igual. Lo que distingue a un «equipo» de un «grupo de trabajo» es precisamente esa responsabilidad compartida:

- En **un grupo de trabajo**, cada persona tiene asignadas sus responsabilidades, sigue un proceso o unas pautas de ejecución y responde de forma individual.
- En **un equipo**, las personas comparten un propósito común, conseguir el mayor valor posible para la visión del cliente, y se autogestionan para trabajar y responder de forma conjunta. No hay un gestor que asigne y coordine el trabajo; son los desarrolladores quienes se coordinan, colaboran con el product owner en el product backlog y comparten el objetivo de cada sprint y la responsabilidad de lograrlo.

Los desarrolladores ante la IA

Con la IA, el valor del trabajo de los desarrolladores se desplaza en parte de ejecutar a **especificar, orquestar y revisar**. Saber qué pedir a la IA, con qué contexto, cómo evaluar lo que produce, cuándo rechazarlo y cómo integrarlo pasa a ser una competencia central del equipo.

Esto no hace prescindible el conocimiento del oficio; al contrario, evaluar un trabajo producido por la IA exige más criterio que producirlo uno mismo. Un informe financiero generado en minutos puede contener un error de cálculo que solo detecta quien conoce bien la materia, y lo mismo ocurre con un contrato, un diseño o un fragmento de código. Por eso la experiencia de las personas sigue siendo la principal garantía de calidad del equipo, y por eso conviene cuidar que los perfiles con menos experiencia la adquieran (→ «Aprendizaje y desarrollo de las personas»).

Los equipos que trabajan con IA tienden a ser más pequeños, porque cada persona puede abarcar más trabajo. Un equipo pequeño se coordina con menos esfuerzo, pero tiene menos diversidad de perspectivas y depende más de cada persona. El tamaño adecuado es una decisión del contexto, dentro del máximo de diez personas para el equipo scrum completo.

El scrum master

El scrum master es responsable de la **eficacia del equipo scrum**. Lo consigue ayudando al equipo y a la organización a entender y aplicar scrum, y creando las condiciones para que el equipo mejore de forma continua su manera de trabajar. Su fin es que el equipo y el cliente sean capaces de organizarse y trabajar con autonomía.

Es un rol de servicio, que se ejerce en tres direcciones:

- **Al equipo:** le ayuda a autogestionarse, a centrarse en crear incrementos de valor y a resolver los impedimentos que frenan su avance; gestiona las dificultades de dinámica de grupo y procura que los eventos se celebren y resulten útiles.
- **Al product owner:** le ayuda a gestionar el product backlog, a expresar con claridad los objetivos y a relacionarse con las personas interesadas.
- **A la organización:** forma y asesora sobre scrum, ayuda a planificar su implantación y elimina los obstáculos entre el equipo y el resto de la organización.

El scrum master no dirige el trabajo del equipo ni asigna tareas. Tampoco conduce el scrum diario, que es una reunión de los desarrolladores (→«El scrum diario»): su papel es facilitar cuando hace falta y ocuparse después de los impedimentos que hayan surgido.

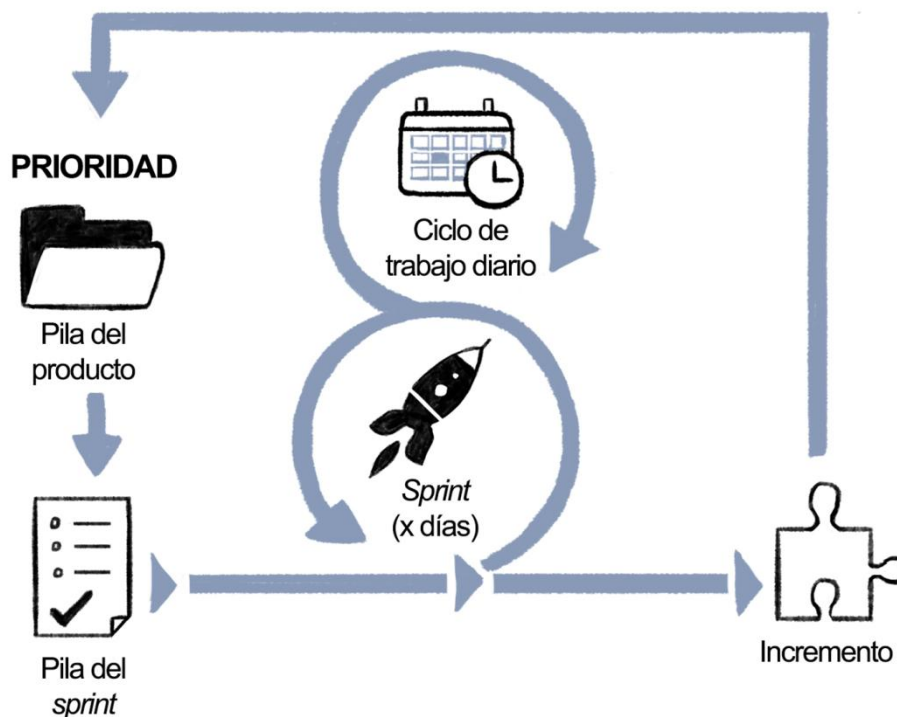
Un rol que crece con la IA

La integración de la IA amplía el ámbito de responsabilidad del scrum master. Además de facilitar el trabajo entre personas, ahora ayuda al equipo a diseñar cómo se reparte el trabajo con la IA, a acordar sus límites y a revisar con rigor lo que produce. También protege los momentos de reflexión frente a la presión de la velocidad y cuida de las personas en un cambio que genera incertidumbre.

Algunas tareas operativas del rol, como preparar agendas, tomar notas o analizar métricas, pueden apoyarse en la IA. Lo que la IA no puede hacer es leer el estado de un equipo, crear seguridad psicológica o acompañar un cambio de forma de trabajar, y ahí está el centro del oficio. Lo desarrollaremos en la tercera parte (→ «Un rol que evoluciona»).

Los artefactos de scrum

Los artefactos de scrum son sus herramientas, sus bloques de construcción elementales. Ayudan a los roles durante los eventos y, sobre todo, hacen **transparente** el trabajo: lo que se quiere conseguir, lo que se está haciendo y lo que ya está terminado.



Artefactos más extendidos

El marco estándar se apoya en tres artefactos clave, cada uno asociado a un compromiso que le da sentido:

- **Product backlog:** la lista ordenada de todo lo que puede necesitar el producto. Al principio refleja el producto mínimo viable (MVP) o la visión inicial, y crece y evoluciona durante el desarrollo. Su compromiso es la **visión del producto**, el resultado al que se quiere llegar.
- **Sprint backlog:** el trabajo que los desarrolladores van a realizar durante el sprint. Su compromiso es el **objetivo del sprint**, el valor que el sprint debe aportar.
- **Incremento:** el resultado del sprint, una parte del producto que funciona, probada y lista para usarse. Su compromiso es la **Definición de Hecho**, los criterios que debe cumplir para considerarse terminado.

Enlace de interés: [Cómo crear un backlog, Scrum Manager Podcast](#) (→«Referencias bibliográficas»).

Otros artefactos

Junto a los tres anteriores, muchos equipos emplean otros artefactos que veremos más adelante:

- *Gráfico de avance* (burn down chart)*: muestra el trabajo pendiente del sprint y su evolución (→«Medir el avance»).
- *Gráfico de producto* (burn up chart)*: muestra cuánto se ha construido del producto y la previsión de lo que queda (→«Medir el avance»).
- **Definition of Ready (DoR):** acuerdo sobre cuándo un elemento del product backlog está listo para entrar en un sprint (→«La Definition of Ready»).
- **Definición de Hecho (DoD):** acuerdo sobre los criterios para considerar terminado un trabajo (→«La Definición de Hecho»).

En equipos que trabajan con IA, todos estos artefactos cumplen una función añadida. Son el lugar donde queda registrado qué trabajo se ha delegado en la IA y quién lo ha revisado, y donde las personas interesadas pueden comprobarlo.

Enlace de interés: [Definition of Done, Scrum Manager Podcast](#) (→«Referencias bibliográficas»).

Unidades de trabajo

La «unidad de trabajo» es el elemento que el equipo utiliza para gestionar el trabajo dentro de un sprint. Las más usadas son las historias de usuario, las historias técnicas y las tareas.

- **Historia de usuario:** una descripción corta, simple y específica del valor que el producto debe proporcionar al usuario final, escrita desde su perspectiva. Nace de las conversaciones entre los desarrolladores y el product owner para alinear la visión, y sirve para recordar lo hablado.
- **Historia técnica:** similar a la historia de usuario, pero para trabajos que no ve directamente el usuario final, como reorganizar un sistema existente o implantar medidas de seguridad.
- **Tarea:** una unidad más pequeña, una actividad que hay que completar para cumplir una historia, como redactar un texto, diseñar una pantalla o realizar pruebas.

Cuando una historia es demasiado grande y está poco definida se la suele llamar «épica» (*epic*). Las épicas son requisitos ágiles, pero por su tamaño y falta de detalle no sirven como unidades de trabajo. Desarrollar un nuevo sistema de gestión de incidencias, o lanzar la campaña de captación de un nuevo curso, son ejemplos de épica.

Enlace de interés: [Cómo se crea una historia de usuario, Scrum Manager Podcast](#) (—«Referencias bibliográficas»).

Enlace de interés: [5 características de una buena historia de usuario, Scrum Manager Podcast](#) (—«Referencias bibliográficas»).

Al empezar a trabajar con scrum es habitual descomponer las historias en tareas, porque permite ver el avance día a día y ayuda a los equipos con menos experiencia a no quedarse atascados. Con la experiencia, descomponer cada historia puede volverse tedioso; si el equipo gestiona bien su trabajo directamente con historias, obligarle a descomponerlas es microgestión innecesaria. Usar unas unidades u otras depende del contexto.

Especificar con ejemplos

Las historias de usuario nacieron para comunicar a personas, que pueden preguntar, inferir el contexto y proponer alternativas. La IA, en cambio, interpreta lo que se le da de forma literal y no completa el contexto que falta. Cuando una parte del trabajo la va a producir la IA, el equipo necesita expresar con más precisión qué se espera.

La forma más útil de hacerlo es **especificar con ejemplos**: acompañar la historia de casos concretos que muestran el resultado esperado y que sirven para comprobar si se ha conseguido. Para la historia «como persona candidata, quiero recibir respuesta a mi solicitud para saber si sigo en el proceso», un equipo de recursos humanos puede añadir ejemplos como «si la solicitud no cumple el requisito de titulación, la respuesta lo indica y

ofrece las otras ofertas abiertas del área» o «la respuesta nunca menciona datos de otras personas candidatas».

Los ejemplos convierten la intención en criterios verificables, tanto para quien revisa como para la IA que produce. El nivel de detalle depende del riesgo: una prueba rápida para aprender puede partir de la historia sola, y lo que va a llegar a usuarios reales necesita criterios precisos. La especificación complementa la conversación, pero no la sustituye.

El product backlog

El product backlog es el inventario de todo lo que se quiere incorporar al producto a través de los sucesivos sprints: funcionalidades, mejoras, cambios técnicos y corrección de errores. Representa lo que esperan el cliente, los usuarios y las demás partes interesadas, y todo lo que implique trabajo para el equipo debe estar reflejado en él. Algunos ejemplos de sus elementos:

- «Ofrecer a los usuarios la consulta de los archivos publicados por un determinado miembro de la plataforma».
- «Consultar los pedidos realizados por un vendedor en un rango de fechas».
- «Preparar la guía de acogida para las personas que se incorporan al departamento».

Su característica esencial es que **contiene información viva**, en continua evolución. Más que un documento de requisitos, es una herramienta que facilita la comunicación con el equipo. Al empezar el proyecto contiene unos pocos elementos, los mejor entendidos en ese momento, y se amplía y modifica a medida que avanza el desarrollo. Basta con que tenga elementos suficientes para un primer sprint.

El product owner lo mantiene ordenado según el valor y la urgencia de cada elemento. El grado de concreción de los elementos es proporcional a su prioridad: los más prioritarios deben estar lo bastante detallados para incluirse en el sprint, y los demás pueden esperar.

El refinamiento

A la labor de ordenar, detallar y preestimar los elementos se la llama **refinado** o **preparación**. Es una tarea colaborativa entre el product owner y los desarrolladores, que pueden reunirse para ello en cualquier momento, sin que consuma más del 10 % de la capacidad de trabajo del equipo. Las estimaciones corresponden a los desarrolladores, porque son quienes harán el trabajo (—«Estimar»).

Para mantener el product backlog conviene usar medios simples, conocidos y compartidos por todo el equipo: notas adhesivas en un tablero ordenadas por prioridad o una herramienta de gestión con la que todos estén familiarizados. Es un radiador de información y una herramienta para la comunicación directa.

El product backlog ante la IA

Cuando construir es rápido y barato, el riesgo de llenar el producto de cosas que nadie necesita aumenta. Por eso muchos equipos tratan cada elemento del product backlog como una **hipótesis**: una apuesta sobre qué problema resuelve y para quién, que hay que validar. Cada elemento debería poder responder a una pregunta sencilla: qué problema resuelve y cómo sabremos si lo ha resuelto.

La IA puede ayudar a redactar, agrupar o detectar elementos duplicados, pero la decisión sobre qué entra y en qué orden sigue siendo del product owner. Los métodos para descubrir y validar qué merece construirse se desarrollan en el temario [Product Owner](#) de Scrum Manager.

El sprint backlog

El sprint backlog es la lista de todas las unidades de trabajo necesarias para lograr el objetivo del sprint. En él las historias se descomponen, si el equipo lo considera útil, en unidades del tamaño adecuado para seguir el avance cada día e identificar riesgos y problemas sin procesos de gestión complejos.

Sprint #3 - Inicio: 6-mar-24			Objetivo del sprint  INCREMENTO Parte del producto desarrollada en un sprint, en condiciones de ser usada (pruebas, codificación limpia y documentada).											
SPRINT BACKLOG			ESFUERZO PENDIENTE											
Unidad de trabajo	Estado	Asignada	L	M	M	J	V	L	M	M	J	V		
Descripción	En curso		16	8	8	4								
Descripción	Terminada		12	8	4									
Descripción	En curso		8	12	11	12								
Descripción	Pendiente		8	8	8	8								
...														

Todo el equipo colabora en su confección durante la reunión de planificación del sprint. Si los desarrolladores dividen las historias en tareas, se aconseja que ninguna dure más de un día de trabajo. Si el product backlog es territorio del product owner, **el sprint backlog es territorio de los desarrolladores**, que son los únicos que pueden modificarlo durante el sprint.

Proporciona comunicación visual directa, y sobre él los desarrolladores revisan cada día el avance. Lo ideal es que esté a la vista de todos, en un tablero físico o en una herramienta colaborativa, con el formato más cómodo para el equipo y con esta información:

- El objetivo del sprint.
- La lista de historias o tareas.

- La persona que ha asumido cada unidad de trabajo.
- El estado de cada una y cuánto queda para completarla.

El sprint backlog debe estar alineado con el objetivo del sprint, que marca un hito en el avance hacia la visión del producto.

El sprint backlog ante la IA

Cuando parte del trabajo lo produce la IA, el sprint backlog hace visible cómo se reparte. Muchos equipos indican en cada unidad de trabajo qué partes producirá la IA y qué persona las revisará, de modo que la atribución forma parte de la transparencia del sprint.

Sobre todo, el sprint backlog refleja que **revisar también es trabajo**. La IA puede producir en minutos lo que una persona tardará horas en revisar e integrar, y si el plan del sprint solo cuenta lo que se produce, la revisión se convierte en un cuello de botella invisible. Las tareas de revisión e integración se anotan y se planifican como cualquier otra («Planificar con la IA en el equipo»).

El incremento

El incremento es la parte del producto construida en un sprint que está en condiciones de entregarse al cliente: terminada, probada y operativa. En un sprint pueden producirse **uno o varios incrementos**, y todos deben cumplir la Definición de Hecho.

No se consideran incrementos los prototipos, los módulos sueltos ni las partes pendientes de pruebas. Idealmente, cada elemento del product backlog se refiere a algo entregable y no a trabajos internos del tipo «diseño de la base de datos» o «redacción del índice».

El primer sprint puede ser una excepción. Se suele llamar «sprint cero» cuando tiene objetivos como contrastar la plataforma, el diseño o las herramientas y métodos de trabajo previstos, algo necesario al comenzar algunos proyectos.

Si la parte desarrollada requiere documentación, o procesos de validación y verificación documentados, estos también tienen que estar realizados para considerar terminado (*done*) el incremento, es decir, entregable al cliente.

Un prototipo no es un incremento

La IA permite producir prototipos en muy poco tiempo, y esa es una de sus grandes aportaciones: probar una idea con usuarios cuesta hoy una fracción de lo que costaba. Pero un prototipo sirve para aprender y no está preparado para entregarse. Le faltan la fiabilidad, la seguridad, el cumplimiento de las normas y el mantenimiento que no se ven en una demostración.

Los prototipos son útiles, pero se presentan como lo que son. Cuando una idea validada con un prototipo pasa a formar parte del producto, se construye o se revisa con los criterios completos de la Definición de Hecho.

La Definición de Hecho

La **Definición de Hecho (DoD)** es el acuerdo del equipo sobre los criterios que debe cumplir un trabajo para considerarse terminado. Es el compromiso del incremento: un incremento no está terminado hasta que cumple todos sus criterios, y lo que no los cumple vuelve al product backlog. Debe ser conocida y compartida por todo el equipo y por las personas interesadas.

La DoD da transparencia, porque todos saben qué significa «terminado» sin necesidad de preguntarlo. También da calidad, porque los criterios no se negocian sprint a sprint. Los fija el equipo, a partir de las exigencias de la organización, y los revisa a medida que madura. Para un equipo de software puede incluir que el código esté revisado, probado e integrado; para un equipo de comunicación, que el texto esté revisado por una segunda persona, verificadas sus fuentes y aprobado por el área legal cuando corresponda.

La Definición de Hecho ante la IA

Con la IA, la Definición de Hecho es el principal contrapeso a la velocidad. La IA produce trabajo con apariencia de terminado que puede contener errores sutiles, datos inventados o soluciones que funcionan hoy y causan problemas mañana. Sin criterios explícitos, ese trabajo se acumula sin que nadie lo detecte.

Por eso los equipos que trabajan con IA hacen su DoD más exigente, no menos. El criterio central es la **revisión humana**: todo trabajo producido por la IA ha sido revisado por una persona que entiende lo que hace, y no solo que funciona o que parece correcto. Cuando la revisión se apoya en la experiencia de quien revisa, la DoD protege también contra el sesgo de dar por bueno lo que produce una máquina.

La Definición de Hecho reforzada

Tendencia en consolidación (2026). El contenido de esta sección describe prácticas que ganan adopción con rapidez, pero aún no son generales. No entra en el examen de certificación.

Muchos equipos añaden a su DoD criterios específicos para el trabajo producido por la IA, que se conocen como **DoD reforzada**. Los más frecuentes son:

- **Verificación independiente:** lo que produce la IA lo comprueba alguien o algo distinto de quien lo produjo, porque la IA tiende a dar por bueno su propio trabajo.
- **Procedencia visible:** queda registrado qué partes produjo la IA y quién las revisó.

- **Comprobaciones automáticas:** en software, análisis automáticos de calidad y seguridad; en otros equipos, comprobación de fuentes, de datos personales o de coherencia con la guía de estilo.

Algunos equipos usan una segunda IA como primer filtro antes de la revisión humana, que nunca sustituye.

Hecho y resultado logrado

Tendencia emergente (2026). El contenido de esta sección describe una propuesta reciente en la profesión. No entra en el examen de certificación.

Algunos autores proponen distinguir entre el trabajo **terminado**, que cumple la DoD, y el **resultado logrado**, cuando hay evidencia de que ese trabajo ha producido el beneficio que se buscaba. La propuesta responde a un riesgo que la IA agrava, el de terminar mucho trabajo sin saber si sirve para algo.

La Definition of Ready

La **Definition of Ready (DoR)** es el acuerdo del equipo sobre cuándo un elemento del product backlog está preparado para entrar en un sprint. Un elemento preparado está definido y preestimado, puede completarse en un único sprint y tiene establecidos los criterios para considerarlo terminado, así como la persona que los verificará.

La DoR evita que entren en el sprint elementos que el equipo no entiende o no puede terminar, y que el sprint se dedique a aclarar lo que debería haberse aclarado antes. Se aplica sobre todo a los elementos más prioritarios del product backlog, durante el refinamiento.

Conviene no burocratizarla. Una DoR con demasiados requisitos frena la entrada de trabajo valioso y convierte la preparación en un trámite. El objetivo es que el equipo y el product owner estén de acuerdo en que el elemento está listo, no rellenar una lista de comprobación.

Preparar el trabajo que hará la IA

Tendencia en consolidación (2026). El contenido de esta sección describe prácticas que ganan adopción con rapidez, pero aún no son generales. No entra en el examen de certificación.

Cuando una parte del trabajo se va a delegar en la IA, muchos equipos amplían su DoR con criterios específicos:

- La IA tiene acceso al **contexto** que necesita: la documentación, los criterios de estilo, los datos y los antecedentes del trabajo.

- Los criterios de aceptación son verificables, idealmente con ejemplos (—«Unidades de trabajo»).
- Se ha previsto quién revisará el resultado y cuánto tiempo llevará la revisión.

Es el equivalente a preparar bien la incorporación de una persona nueva al equipo. Si la IA no tiene el contexto suficiente, su resultado será irrelevante o erróneo y generará trabajo de corrección.

Los eventos de scrum

Los eventos constituyen la rutina de trabajo en scrum. Cada uno es una oportunidad formal de inspeccionar y adaptar, y todos tienen una duración máxima (*time-box*) que evita que se alarguen sin necesidad. Celebrarlos con regularidad crea una cadencia que reduce la necesidad de otras reuniones.

- **Sprint:** el contenedor de todos los demás eventos y el periodo en que se construye el incremento (—«El sprint»).
- **Reunión de planificación del sprint:** marca el inicio del sprint y determina su objetivo y el trabajo necesario para conseguirlo.
- **Scrum diario:** una reunión breve en la que los desarrolladores revisan el avance hacia el objetivo y ajustan su plan para el día.
- **Revisión del sprint:** la inspección del incremento con las personas interesadas y la adaptación del product backlog si es necesario.
- **Retrospectiva del sprint:** la reflexión del equipo sobre su forma de trabajar y el plan de mejoras para la siguiente iteración.

Un evento que no forma parte del marco tradicional pero que se ha extendido es la **reunión de refinamiento del backlog**, que ya vimos al hablar del product backlog. No hay consenso sobre cuándo celebrarla; es frecuente que tenga lugar antes de la reunión de planificación, para llegar a ella con los elementos ya detallados y estimados.

Las duraciones máximas que daremos para los eventos corresponden a un sprint de cuatro semanas. En sprints más cortos, los eventos suelen ser proporcionalmente más breves.

Con la IA, los eventos no pierden sentido, sino que cambian de foco. En cada uno el equipo decide algo sobre la IA: en la planificación, qué se le delega; en el scrum diario, cómo va lo que ha producido; en la revisión, si lo construido es lo correcto; y en la retrospectiva, cómo funciona la colaboración con ella.

Enlace de interés: [Eventos de scrum, Scrum Manager Podcast](#) (—«Referencias bibliográficas»).

El sprint

El sprint es el evento clave de scrum para mantener un ritmo de avance continuo: un periodo de tiempo acotado, **de cuatro semanas como máximo**, durante el que se construyen uno o varios incrementos del producto. Lo habitual es que todos los sprints de un equipo duren lo mismo, de modo que marquen una cadencia. Cada sprint tiene un **objetivo del sprint**, el valor que debe aportar, que orienta las decisiones del equipo durante su desarrollo.

Al empezar a trabajar con scrum es recomendable considerar el sprint como el contenedor de todos los demás eventos:

- Marca el ritmo de avance diario, que se visualiza y comparte en el scrum diario.
- Marca un ritmo fijo para comprobar el desarrollo del producto en las reuniones de planificación y revisión.
- A ese mismo ritmo se celebran las retrospectivas, para reflexionar y mejorar.

En implementaciones más maduras es posible considerar que el ámbito del sprint es solo la construcción del incremento, dejando a un lado las reuniones. Esto da más flexibilidad, por ejemplo, para realizar sprints de duraciones diferentes o para separar la frecuencia de las retrospectivas de la de los sprints.

La duración del sprint

Elegir la duración es un equilibrio. Un sprint largo da tiempo para construir incrementos más grandes, pero retrasa el momento de inspeccionar y aumenta el riesgo de construir durante semanas en la dirección equivocada. Un sprint corto ofrece feedback más frecuente, pero aumenta el peso relativo de los eventos.

Con la IA, muchos equipos sienten que el sprint se les queda largo, porque pueden producir incrementos en días. La pregunta útil no es cuánto puede producir el equipo, sino cada cuánto necesita inspeccionar y adaptar. La disponibilidad de las personas interesadas para revisar, y la capacidad del propio equipo para revisar con rigor lo que produce, siguen poniendo un límite al ritmo, por mucho que la IA acelere la producción.

Cadencias en tensión

Tendencia en consolidación (2026). El contenido de esta sección describe prácticas que ganan adopción con rapidez, pero aún no son generales. No entra en el examen de certificación.

Los equipos que trabajan con IA están explorando varias opciones de cadencia:

- **Microiteraciones dentro del sprint:** el sprint se mantiene como contenedor de planificación y revisión, con ciclos cortos de producción y validación en su interior.

- **Sprints más cortos**, de una semana o menos, con más feedback pero con más peso relativo de los eventos.
- **Flujo continuo** con técnicas kanban, sin sprints de duración fija, con el riesgo de perder los momentos estructurados de reflexión.

Ninguna es mejor en todos los casos. Lo importante es que la cadencia sea una decisión consciente del equipo, revisada en la retrospectiva, y que, sea cual sea, conserve los momentos de inspección y adaptación.

La reunión de planificación del sprint

La reunión de planificación marca el inicio de cada sprint. En ella se toman como base las prioridades y necesidades de negocio del cliente para determinar el objetivo del sprint y el trabajo que se va a realizar. Asisten el product owner y los desarrolladores, y pueden participar otras personas a las que el equipo invite para aportar información. Su duración es proporcional a la del sprint y no supera las ocho horas en un sprint de cuatro semanas.

Precondiciones	Entradas
Los elementos de mayor prioridad del product backlog están preparados, con un nivel de concreción suficiente y una estimación previa. Los desarrolladores conocen las tecnologías y el negocio lo suficiente para estimar y para comprender lo que expone el product owner.	Product backlog. Incrementos anteriores (excepto en el sprint cero). Capacidad del equipo para el sprint y rendimiento reciente. Circunstancias de negocio del cliente y valor que espera obtener el product owner.

La reunión responde a tres preguntas.

¿Por qué es valioso este sprint?

El product owner expone en qué forma el producto puede incrementar su valor con el resultado del sprint, presenta los elementos de mayor prioridad y explica los cambios relevantes del product backlog desde la reunión anterior. La exposición está abierta a preguntas, sugerencias y alternativas.

El equipo define entonces el **objetivo del sprint**, una frase que sintetiza el valor que se va a entregar. Elaborarlo en común garantiza que todo el equipo comprende y comparte la finalidad del trabajo, y durante el sprint sirve de criterio para tomar decisiones.

¿Qué se puede hacer en el sprint?

Los desarrolladores seleccionan los elementos del product backlog que pueden completar, teniendo en cuenta su capacidad y su rendimiento reciente. Durante la selección pueden refinarse los elementos que necesiten más concreción. Esta reunión es un punto caliente de scrum para la fertilización cruzada de ideas.

¿Cómo se conseguirá el incremento?

Los desarrolladores deciden cómo convertirán los elementos seleccionados en un incremento terminado, y componen el sprint backlog. Pueden descomponer los elementos en tareas de un día o menos, algo recomendable en equipos con menos experiencia porque ayuda a visualizar el avance y a afinar la intuición sobre lo que implica cada tipo de trabajo. El product owner atiende dudas y comprueba que se comprende y comparte el objetivo.

Funciones del scrum master en la planificación

El scrum master ayuda a que la reunión sea útil y se mantenga dentro de su duración. En particular, procura que:

1. Haya un product backlog preparado por el product owner.
2. Se mantenga el diálogo entre el product owner y los desarrolladores y se llegue a un acuerdo sobre lo que incluirá el sprint.
3. La descomposición y la estimación del trabajo sean realistas, incluidas las tareas de análisis, investigación, revisión o apoyo.
4. Al final de la reunión estén claros los elementos que se van a ejecutar, el objetivo del sprint y el sprint backlog inicial.

Planificar con la IA en el equipo

En un equipo que trabaja con IA, la reunión de planificación incorpora una decisión más: cómo se reparte el trabajo del sprint entre las personas y la IA. No es una cuestión técnica, sino de autogestión, y aplica las cuatro decisiones explícitas que vimos en «El equipo scrum y la IA» a cada sprint concreto.

El objetivo del sprint gana peso

Cuando la IA permite producir casi cualquier cosa con rapidez, el riesgo de dispersarse aumenta. Un objetivo del sprint claro es la mejor defensa, porque ayuda a decidir qué no se va a hacer aunque se pueda hacer. Un equipo de marketing que puede generar cincuenta piezas en una semana necesita saber más que nunca cuál es el propósito de la campaña y qué piezas lo sirven.

La revisión se planifica

La parte del trabajo que más se tiende a subestimar es la revisión y la integración de lo que produce la IA. La IA puede generar en minutos lo que una persona tardará horas en comprobar, y si el plan del sprint solo cuenta lo que se produce, el equipo descubrirá a mitad del sprint que no tiene tiempo para revisar lo producido. Por eso la capacidad de revisión es un dato de la planificación tan importante como la capacidad de producción, y las tareas de revisión se anotan en el sprint backlog.

Atribución humano-IA

Tendencia en consolidación (2026). El contenido de esta sección describe prácticas que ganan adopción con rapidez, pero aún no son generales. No entra en el examen de certificación.

Muchos equipos hacen explícita en la planificación la **atribución** de cada unidad de trabajo: qué requiere juicio o creatividad humana, qué puede producir la IA bajo revisión y qué persona será responsable de cada parte. Para las tareas que se delegan en la IA, comprueban además que se cumplen los criterios de preparación específicos (→«La Definition of Ready»). La atribución no es definitiva: si durante el sprint resulta que la IA no produce lo esperado, el equipo reasigna el trabajo.

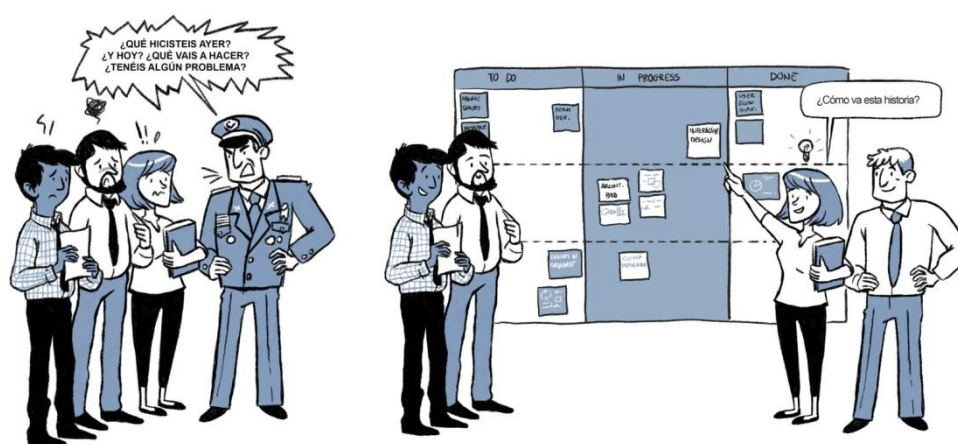
El scrum diario

En scrum, los desarrolladores supervisan la evolución de cada sprint en **reuniones diarias muy breves, de 15 minutos como máximo**. Se las conoce también como «reunión de pie» (*stand-up meeting*) o, coloquialmente, *daily*. Pueden celebrarse en persona o en remoto, siempre con acceso al tablero o a la herramienta que muestra el avance del sprint. En ellas los desarrolladores sincronizan su trabajo y ajustan el plan para las siguientes 24 horas.

Entradas	Resultados
Sprint backlog y gráfico de avance o tablero kanban con la información de la reunión anterior. Información del avance de cada desarrollador.	Sprint backlog y gráfico de avance o tablero actualizados. Impedimentos identificados.

Formato de la reunión

Los desarrolladores conducen esta reunión. No es una reunión de control, sino un espacio para que el equipo colabore, comparta el estado del trabajo y detecte las dificultades. Pueden asistir otras personas, aunque no intervienen. El papel del scrum master es facilitar si hace falta y ocuparse de resolver los impedimentos después de la reunión.



Un formato frecuente al empezar es que cada desarrollador explique lo que ha logrado desde la reunión anterior, lo que va a hacer hasta la siguiente y si tiene o prevé algún impedimento. Este formato de participación individual puede tener efectos negativos en

la motivación y en la dinámica del equipo, y se recomienda **evolucionar hacia un formato que se centre en el trabajo, no en las personas**.

En ese formato, los desarrolladores observan el estado del trabajo, elemento a elemento. Viendo qué hay en curso, asumen el trabajo pendiente según el orden de prioridad y se aseguran de que no se formen cuellos de botella. Así se reduce la duración de la reunión, aumenta la autonomía del equipo y mejora la colaboración; es más frecuente, por ejemplo, que varias personas decidan trabajar juntas en el elemento prioritario que corre riesgo de retrasarse.

El scrum diario ante la IA

Cuando parte del trabajo lo produce la IA, el scrum diario incorpora una pregunta más: si lo que ha producido la IA desde la reunión anterior es lo esperado y quién lo está revisando. Las desviaciones, el trabajo producido que espera revisión y los bloqueos por falta de contexto son ahora impedimentos tan relevantes como los de siempre.

Formatos asíncronos

Tendencia en consolidación (2026). El contenido de esta sección describe prácticas que ganan adopción con rapidez, pero aún no son generales. No entra en el examen de certificación.

En equipos muy pequeños que trabajan juntos de forma continua, una reunión formal puede resultar desproporcionada. Algunos sustituyen parte de los scrum diarios por actualizaciones breves en un canal compartido, y se reúnen solo cuando aparecen desviaciones o bloqueos. El riesgo es perder la conversación que detecta los problemas antes de que se escriban, por lo que conviene revisar en la retrospectiva si el formato sigue cumpliendo su función.

La revisión del sprint

La revisión del sprint se celebra al final del sprint para inspeccionar el incremento y recoger sugerencias. Asisten todo el equipo scrum y las personas interesadas que lo deseen. Su duración no supera las cuatro horas en un sprint de cuatro semanas.

Esta reunión marca a intervalos regulares el ritmo de construcción y la trayectoria del producto, y permite que las personas interesadas aporten sugerencias para seguir evolucionando. Los desarrolladores presentan lo que planificaron, lo que han terminado, lo que no y los impedimentos encontrados, de modo que las expectativas sean realistas desde el principio.

Entradas	Resultados
Incremento terminado.	Feedback para el product owner. Información para mejorar el valor del producto. Product backlog revisado.

Formato de la reunión

Es una reunión informal, de trabajo, en la que se muestra el incremento en funcionamiento y se conversa sobre él. Según el producto, puede incluir también documentación de usuario o técnica. Un protocolo recomendado es este:

1. Los desarrolladores exponen el objetivo del sprint, los elementos que incluía y los que se han terminado.
2. Muestran el funcionamiento de lo construido y se abre un turno de preguntas y sugerencias, que genera información valiosa para el product owner y para el equipo.
3. Con lo aprendido, el equipo y las personas interesadas comentan qué conviene hacer a continuación, y el product owner adapta el product backlog.

La revisión del sprint ante la IA

Con la IA, la pregunta central de la revisión se desplaza de si lo construido está bien hecho a si **se está construyendo lo correcto**. Cuando producir es barato, el riesgo principal es entregar mucho que no sirve, y la revisión es el momento de comprobarlo con quienes lo van a usar. Por eso conviene incorporar a la revisión datos sobre cómo se están usando los incrementos anteriores, y no solo mostrar los nuevos.

También conviene evitar la sobreproducción. Presentar más de lo que las personas interesadas pueden evaluar con atención satura la reunión y empobrece el feedback. Y la transparencia sobre la procedencia se aplica aquí como en el resto del trabajo: el equipo explica qué partes ha producido la IA y cómo se han revisado.

La retrospectiva del sprint

La retrospectiva se celebra tras la revisión de cada sprint y antes de la planificación del siguiente. Su duración no supera las tres horas en un sprint de cuatro semanas. Es un ejemplo de práctica que no estaba en el marco original de scrum, pero que se ha consolidado con el tiempo. En ella el equipo scrum reflexiona sobre su forma de trabajar, identifica fortalezas y puntos débiles y planifica acciones de mejora.

Por celebrarse al final del sprint, a veces se confunde con la revisión del sprint, pero conviene tratarlas por separado porque sus objetivos son diferentes. La revisión analiza **qué** se está construyendo, el producto; la retrospectiva se centra en **cómo** se trabaja.

Participa todo el equipo scrum, incluido el product owner, que conviene que se considere equipo más que cliente. Su perspectiva aporta mucho valor, porque es quien mantiene el foco en construir lo que el cliente necesita. Si no fuera participativo, el scrum master actúa como facilitador para lograr su implicación.

Más necesaria que nunca

Con la IA, la retrospectiva es el evento que más riesgo corre de suprimirse y el que más se necesita. Es el espacio natural para evaluar cómo funciona la colaboración con la IA:

- si los acuerdos sobre qué se delega y quién revisa siguen siendo adecuados;
- si la revisión es suficientemente rigurosa o se está acumulando trabajo sin revisar;
- si la Definición de Hecho necesita adaptarse;
- si la mayor producción se está traduciendo en más valor.

Hay un patrón que se repite en los equipos a los que la IA no les sale bien. Adoptan la herramienta, la producción sube y eliminan eventos para ir más deprisa. A partir de ahí pierden la capacidad de detectar los problemas, que se acumulan hasta que un incidente los hace visibles. El paso decisivo es el tercero, eliminar los momentos de reflexión, y los equipos que prosperan hacen lo contrario.

Adaptar el formato, no suprimirla

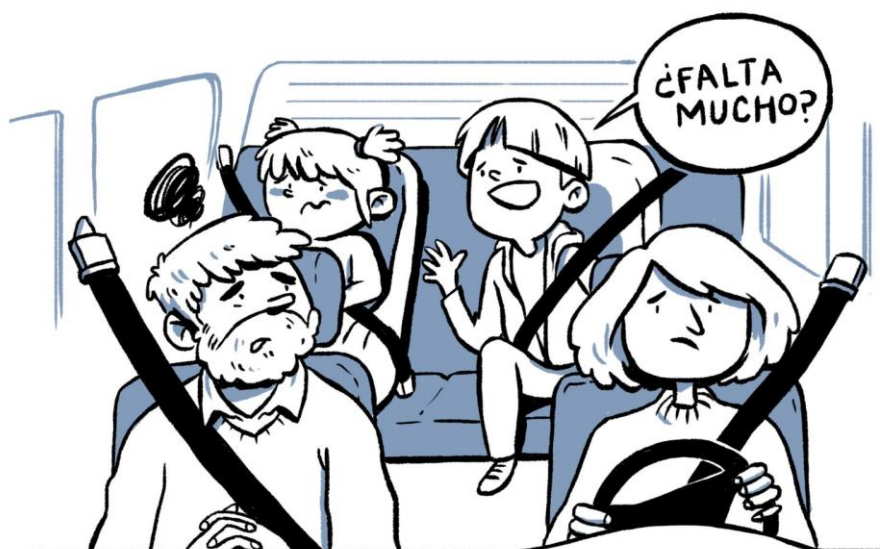
Si una retrospectiva larga no funciona, se puede probar con una más breve, pero no con ninguna. En flujos de trabajo muy rápidos, la retrospectiva del sprint puede complementarse con retrospectivas breves y frecuentes, con retrospectivas temáticas sobre el uso de la IA o con retrospectivas de incidente cuando algo ha salido mal. Lo que no se negocia es que el equipo dedique tiempo a inspeccionar y adaptar su forma de trabajar.

Medir el avance

En scrum se mide para tomar decisiones, no por medir. **Cuantas menos métricas, mejor:** medir es costoso, y cada indicador debe servir para entregar más valor al cliente. Los objetivos de la medición del avance son planificar los sprints de forma realista, marcar el ritmo, sincronizar equipos y prever fechas de entrega. Para todos ellos, scrum mide **el trabajo que queda**, no el que se ha hecho.

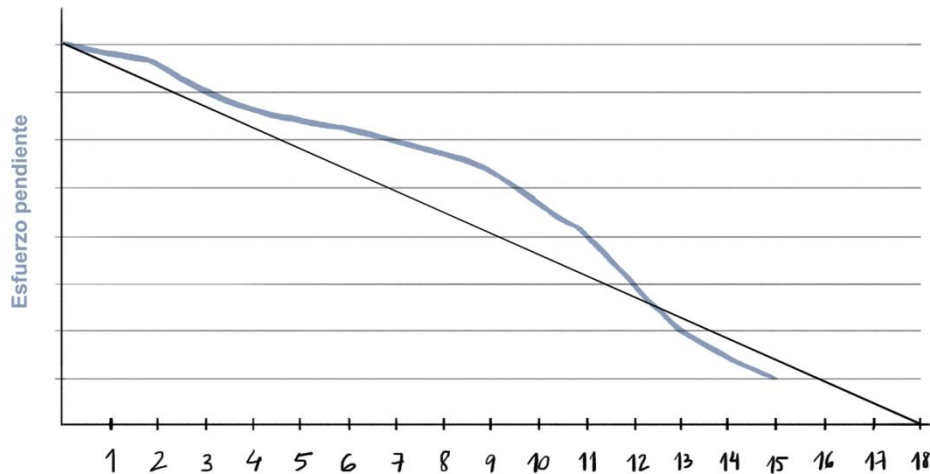
¿Cuánto queda?

Medir el trabajo hecho es fácil, pero no sirve para calcular el avance. «Esto debía costar una semana y han pasado tres días, así que quedan cuatro» suena lógico, pero rara vez se cumple, porque surgen imprevistos o atajos. Por eso el avance se determina por el trabajo pendiente, como bien saben los niños cuando preguntan en un viaje, una y otra vez, si falta mucho.



Gráficos de avance y de producto

El **gráfico de avance** (burn down) lo actualizan los desarrolladores durante el sprint, a ser posible cada día, para detectar pronto las desviaciones. En el eje vertical representa el trabajo pendiente, y en el horizontal, los días del sprint. Si se trabaja con historias en lugar de tareas, se actualiza cuando se completa cada historia. Si la línea se mantiene durante varios días por encima de la diagonal del avance ideal, el sprint está subestimado; si cae más deprisa, se terminará antes.

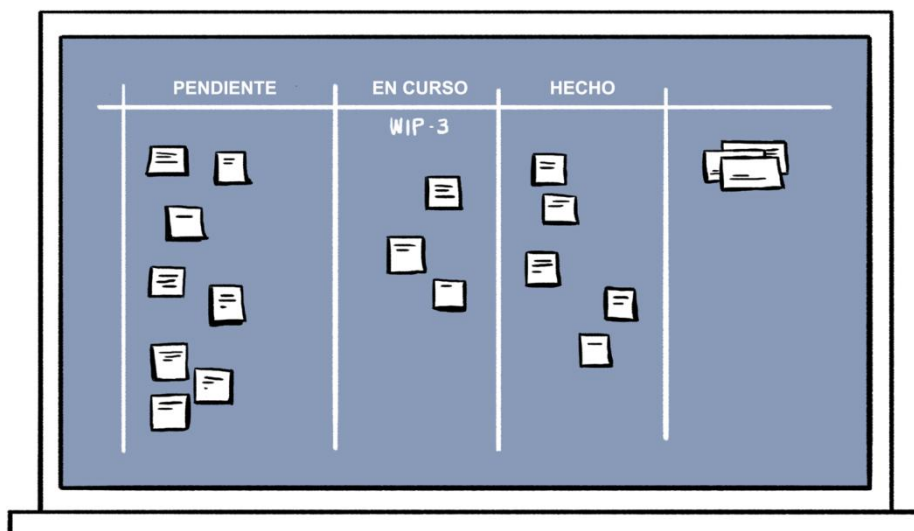


El **gráfico de producto** (burn up) es una herramienta de previsión del product owner. Representa lo construido y lo que queda del product backlog a lo largo de los sprints, con previsiones pesimista, realista y optimista. Es un documento vivo que muestra las previsiones tras cada cambio del product backlog, no un plan estable.

Vídeo explicativo: [Gráfico de avance «Burn down chart»](#) y [Gráfico de producto «Burn up chart»](#) (—«Referencias bibliográficas»).

Kanban y el flujo de trabajo

Kanban es una técnica para gestionar de forma visual un flujo continuo de trabajo, con o sin sprints. El equipo sitúa las unidades de trabajo en un tablero cuyas columnas indican su estado («pendiente», «en curso», «terminado» y otras que necesite el equipo), y así se ve de inmediato dónde se acumula el trabajo.



La pieza clave es el límite de trabajo en curso, o **WIP** (*work in process*): el número máximo de unidades que pueden estar a la vez en una columna. Un WIP de 3 en «en curso» significa que el equipo no trabaja en más de tres elementos a la vez. Limitar el

trabajo en curso obliga a terminar antes de empezar algo nuevo y reduce el **tiempo de ciclo**, lo que tarda un elemento en recorrer el tablero desde que se empieza hasta que se termina.

La revisión como restricción

Con la IA, el tablero suele mostrar un fenómeno nuevo: el trabajo se acumula en la columna de revisión. La IA produce más de prisa de lo que las personas pueden comprobar, y ese es ahora el punto estrecho del flujo. Muchos equipos añaden una columna «en revisión» con su propio límite WIP, para que la producción no supere la capacidad de revisar y la acumulación quede a la vista. Medir el tiempo de ciclo completo, incluida la revisión, da una imagen más fiel del avance que contar lo producido.

Estimar

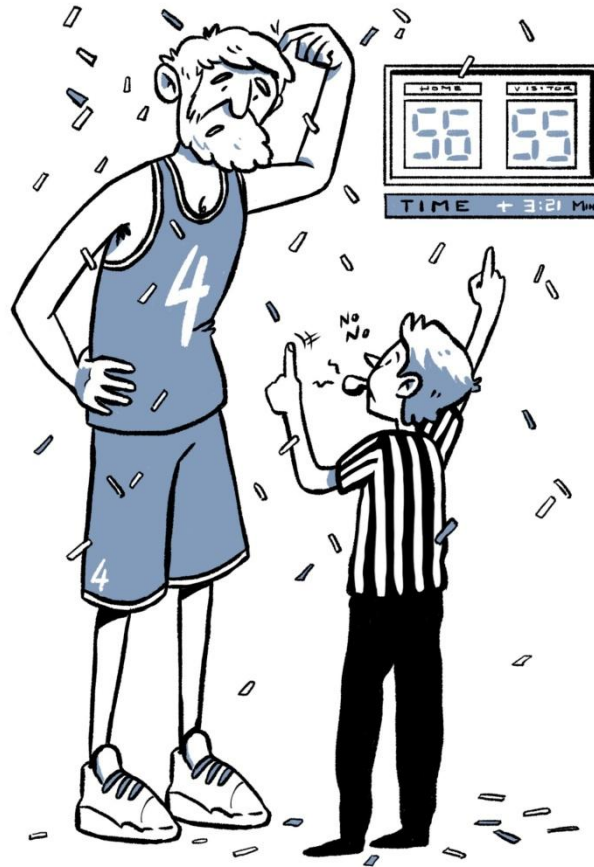
Estimar sirve para planificar sprints realistas y prever entregas, pero el trabajo del conocimiento rara vez tiene una única solución, y su duración varía mucho según quién lo haga y en qué circunstancias. Por eso la estimación ágil prefiere **unidades relativas** y la estiman siempre quienes van a hacer el trabajo.

Puntos de historia

La unidad relativa más habitual es el **punto de historia**: el tamaño de un trabajo comparado con otro ya conocido. En un equipo de programación, un punto puede equivaler a preparar una pantalla de acceso; en uno de diseño gráfico, a maquetar un tríptico. Si una persona estima cuántos «huevos fritos» cuesta planchar una camisa, la respuesta dependerá de su habilidad con cada tarea, y por eso la estimación la hace quien va a trabajar.

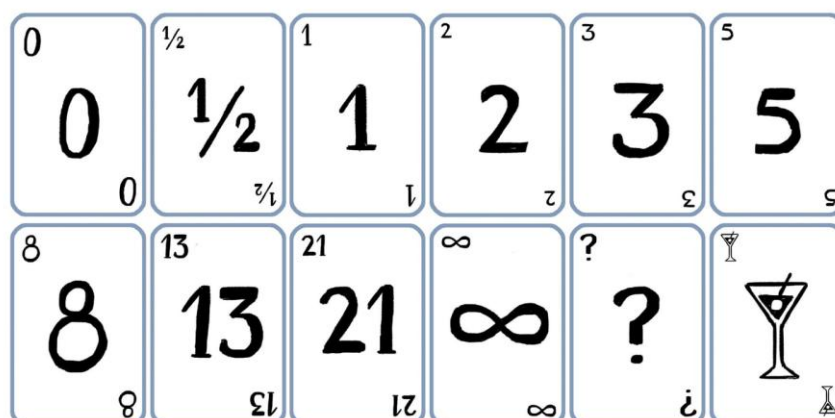


La **velocidad** es la cantidad de trabajo, en puntos, que el equipo completa en un sprint, y sirve para prever cuánto cabe en el siguiente. Al estimar en tiempo conviene distinguir el **tiempo ideal** (el que costaría la tarea concentrados y sin interrupciones) del **tiempo real**, igual que los 40 minutos de juego de un partido de baloncesto ocupan más de una hora de reloj.



Técnicas de estimación

La **estimación de póquer** (*planning poker*), ideada por James Grenning, es la técnica más extendida. Cada participante elige en secreto una carta con su estimación, normalmente de una baraja con la sucesión de Fibonacci, en la que los valores se separan más cuanto mayor es la estimación porque también crece la incertidumbre. Se muestran a la vez y, si las estimaciones difieren mucho, quienes han dado las más extremas explican sus razones antes de repetir. Evita las discusiones circulares y hace participar a todos.



La **estimación en la pared** sitúa notas con los elementos del product backlog sobre una pared: los desarrolladores fijan la posición horizontal según el tamaño, y el product owner, la vertical según la prioridad.

Vídeo explicativo: [Estimación en la pared](#) (—«Referencias bibliográficas»).

Los puntos de historia se malinterpretan con facilidad cuando se usan para comparar equipos o como medida de productividad, porque invitan a inflar las estimaciones. Por eso algunos equipos usan alternativas más sencillas, como las **tallas de camiseta** (XS, S, M, L, XL) o solo tres medidas (demasiado grande, demasiado pequeño o adecuado). Otros, con un flujo de trabajo maduro, dejan de estimar (**#NoEstimates**) y prevén las entregas a partir del ritmo real al que terminan el trabajo.

Enlace de interés: [Estimar o no estimar, Scrum Manager Podcast](#) (—«Referencias bibliográficas»).

Estimar con la IA en el equipo

La IA altera la estimación por dos lados. Por una parte, el trabajo que produce la IA no encaja bien en unidades pensadas para el esfuerzo humano: una tarea que antes costaba ocho puntos puede producirse en minutos. Por otra, la revisión de ese trabajo sí requiere esfuerzo humano, y es la parte que más se tiende a olvidar.

La consecuencia práctica es que el equipo estima también la revisión y la integración, y no solo la producción. La previsión de entregas a partir de datos de ritmo real, en lugar de estimaciones, se desarrolla en el temario [Product Owner](#) de Scrum Manager.

Estimación en tres niveles

Tendencia en consolidación (2026). El contenido de esta sección describe prácticas que ganan adopción con rapidez, pero aún no son generales. No entra en el examen de certificación.

Algunos equipos distinguen tres tipos de trabajo al estimar: el que produce la IA de forma casi autónoma, que no se estima en puntos sino por el tiempo de revisión que exigirá; el trabajo humano, que se estima como siempre; y el trabajo de revisión e integración, que se estima de forma explícita porque es el que genera los cuellos de botella.

Producción y resultado

Con la IA, la cantidad de trabajo que produce un equipo puede crecer mucho sin que crezca el valor que entrega. Por eso conviene distinguir con claridad entre **producción** (lo que el equipo genera: documentos, funcionalidades, piezas, líneas de código) y **resultado** (el cambio que eso produce en los usuarios y en el negocio).

Velocidad no es agilidad

La agilidad es la capacidad de responder al cambio mientras se entrega valor, y no la velocidad de producción. Un equipo puede ser muy rápido y poco ágil: entrega mucho, pero sin saber si es lo correcto ni poder cambiar de rumbo cuando la evidencia lo pide. Y un equipo puede producir menos y ser muy ágil, porque entrega lo validado, responde pronto al mercado y mejora su forma de trabajar.

No toda velocidad es igual. La que importa es la de validación (cuánto se tarda en saber si una idea funciona), la de aprendizaje y la de adaptación. La que puede ser ilusoria es la de producción, si lo producido no se usa, no sirve o tiene que rehacerse.

Productividad percibida y medida

Como vimos en la introducción, la sensación de ir más deprisa con la IA no siempre coincide con la realidad. Las personas tienden a sobrestimar lo que la herramienta les ahorra y a no contar el tiempo que dedican a revisar y corregir. Por eso la productividad se mide con datos del trabajo terminado y no con impresiones, y se mide en el equipo, no en cada persona.

Medir resultados

Las métricas útiles en un equipo con IA miden el flujo y el resultado, no la actividad. Algunas preguntas que ayudan a elegir las son estas:

- ¿Entregamos más valor o solo más trabajo?
- ¿Podemos cambiar de rumbo cuando la evidencia lo pide?
- ¿Aprendemos al mismo ritmo al que producimos?
- ¿La calidad se mantiene o se degrada?

Medir cuánto se usa la IA, o cuánto se produce con ella, no dice nada sobre el valor. Celebrar el volumen es una de las formas más rápidas de caer en la trampa de producir mucho sin avanzar. La medición del valor del producto se desarrolla en el temario [Product Owner](#) de Scrum Manager.

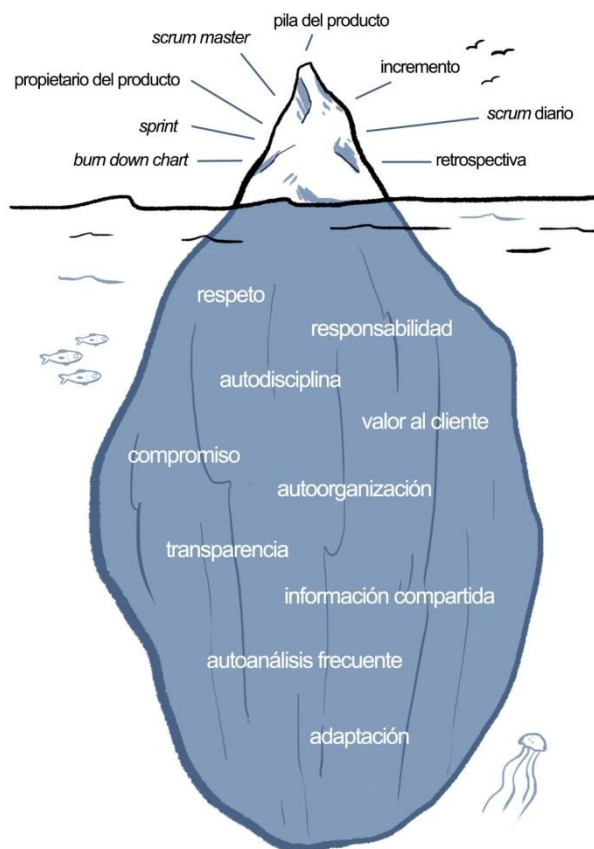
Parte II — Principios y valores

Interiorizando y adaptando.



Principios y valores scrum

Hay algo que no se suele decir en los cursos de scrum: el marco que acabamos de estudiar, las prácticas y herramientas de la primera parte, no funciona sin unos principios y valores alineados con él.



La razón es que las prácticas de scrum no son procesos, es decir, pautas que garanticen resultados las ejecute quien las ejecute. Scrum se asienta sobre el conocimiento tácito de las personas y sobre unos valores de la organización. Las prácticas son las ramas del árbol y, sin unas buenas raíces, no dan frutos. Conviene recordarlo de vez en cuando para no centrarse solo en las herramientas.

Hay empresas con los principios tan interiorizados que desarrollan su propio modelo de agilidad, con prácticas distintas de las que hemos visto, y obtienen productos innovadores y de calidad. Y hay empresas que, aplicando todas las prácticas del scrum estándar, solo consiguen personas desmotivadas y estresadas y resultados mediocres.

Se necesita flexibilidad para adaptar lo aprendido a la realidad de cada empresa y proyecto. Para hacer una buena gestión hay que conocer las herramientas y ajustarlas al equipo, aprovechando lo que funcione y modificando o desechando lo que no, en lugar de pretender que las personas se adapten a métodos que no les convencen. La IA es el

caso más claro: no hay una forma correcta de integrarla que sirva a todos los equipos, y cada uno debe encontrar la suya apoyándose en los principios.

Los principios y valores pueden dividirse, aunque sea por pura semántica, según estén detrás de las prácticas o de la cultura de la organización. Los **principios** son el soporte de las prácticas; los **valores**, de la cultura. En esta parte veremos unos y otros, cómo los sostienen los roles, artefactos y eventos, y algunas prácticas más que, sin formar parte del marco estándar, se utilizan con frecuencia.

Los principios de scrum

La finalidad de los artefactos, los eventos y las demás técnicas ágiles es que el trabajo se base en estos ocho principios. Junto a cada uno anotamos lo que la IA cambia en su aplicación.

Entrega de valor

La entrega temprana y continua de valor al cliente, para lo que es necesario que este colabore con el equipo y que su visión se comparta y comprenda. Con la IA, este principio exige distinguir con más cuidado que nunca entre producir y entregar valor, porque el volumen de lo producido puede crecer mucho sin que crezca el valor.

Mejora continua

En agilidad se reflexiona con frecuencia sobre los métodos de trabajo, cuestionando su efectividad y adaptándolos, y el mismo esfuerzo autocrítico se aplica a los productos y servicios. La integración de la IA es hoy uno de los principales objetos de esa mejora, y la retrospectiva su lugar natural.

Desarrollo iterativo e incremental

El producto no se construye conforme a un plan inicial detallado y completo, sino que arranca de un mínimo viable sobre el que se añaden incrementos. La IA acorta los ciclos de producción, pero no el tiempo que necesitan los usuarios para probar y opinar, y el ritmo de los incrementos debe respetar ese tiempo.

Ritmo de trabajo sostenible

Un ritmo de trabajo que evite la ley de Parkinson (el trabajo se expande hasta llenar el tiempo disponible para terminarlo) y la presión de descubrir los retrasos demasiado tarde. Con la IA aparece un riesgo nuevo para la sostenibilidad, la carga mental de revisar sin descanso lo que la IA produce, que el equipo debe vigilar.

Atención continua a la excelencia

El uso de técnicas que garanticen la calidad de los productos y servicios y permitan detectar los errores con antelación o en el momento de producirse. Cuando una parte del trabajo la produce la IA, este principio se concreta en una Definición de Hecho exigente y en una revisión rigurosa.

Operativa visible

La información se comparte con claridad para facilitar la colaboración, identificar pronto los impedimentos y permitir que todo el equipo conozca el estado del producto y aporte ideas. Con la IA, la operativa visible incluye la procedencia del trabajo: qué ha producido la IA y quién lo ha revisado.

Cadencia y sincronización global

Este principio es más relevante cuando se sincronizan varios equipos que trabajan en productos o servicios relacionados. Se busca que la frecuencia de reuniones y las fechas de entrega sean previsibles, por ejemplo sincronizando los sprints de los diferentes equipos.

Personas sobre procesos

La inteligencia colectiva del equipo, su conocimiento tácito, es la responsable de la calidad del producto. La IA aporta conocimiento explícito a gran escala, pero el juicio sobre qué es valioso, la comprensión del contexto y la responsabilidad del resultado siguen siendo de las personas.

Los valores de scrum

La cultura de una empresa es la suma de sus características organizativas y de gobierno, que pueden acelerar o frenar el desarrollo de la agilidad. En organizaciones cuya cultura se asienta en valores propios del trabajo industrial, basado en procesos, los principios ágiles dan resultados mucho más modestos que en las organizaciones con valores que los potencian. Estos son los seis valores que Scrum Manager® considera propios de una cultura ágil:

- Asertividad.
- Valoración del talento.
- Claridad.
- Confianza.
- Estructura desjerarquizada.
- Propósito común.

Además, es necesario el soporte directivo: que la dirección de la empresa tenga una cultura afín, esté implicada y apoye a las personas con formación y recursos suficientes.

Los valores ante la IA

La IA pone a prueba varios de estos valores. La **confianza** se vuelve más compleja, porque el equipo tiene que aprender a confiar en la IA en su justa medida, sin la confianza ciega que acepta lo que produce sin revisarlo ni la desconfianza que renuncia a su potencial. La **claridad** exige que las decisiones sobre la IA (qué se delega, quién revisa, qué límites tiene) sean explícitas y conocidas por todos.

La **valoración del talento** adquiere un sentido nuevo, porque el talento que importa es el que la IA no aporta: el juicio, la experiencia y la capacidad de trabajar con otras personas. Y el **soporte directivo** incluye ahora dar al equipo formación sobre la IA, criterios claros sobre su uso y la tranquilidad de que su integración no es una amenaza encubierta para las personas.

Los principios y valores ágiles a nivel de organización tienen temario propio en Scrum Manager: AgiLevel (<https://agilevel.com>).

Las personas por encima de los roles

Los artefactos y eventos ayudan a desarrollar los principios ágiles, pero estos tienen que ser asumidos primero por las personas. Sin personas con talento, comprometidas y conscientes de sus responsabilidades, las prácticas no sirven de nada, y sin una cultura con los valores que hemos visto no se desarrollan ni el talento ni el compromiso.

Ken Schwaber lo expresó de forma provocadora en una charla de 2006 (Schwaber, 2006). Con un equipo brillante y bien equipado, scrum permite construir un incremento en cada iteración; pero scrum también funciona con un equipo que lo hace todo mal, porque al final de cada iteración muestra con crudeza dónde se está. El marco hace visible la realidad, pero no la cambia por sí mismo: lo que la cambia son las personas.

El product owner y los principios

El product owner es quien hace posible la **entrega temprana y continua de valor**. Gestiona y ordena el product backlog con una actitud proactiva, introduce cambios cuando lo considera necesario y comunica con claridad la visión del cliente para que todo el equipo se alinee con ella. Su actitud es clave para que la comunicación sea honesta y fluida.

Los desarrolladores y los principios

Los desarrolladores son responsables de producir incrementos en cada iteración y de mantener un **ritmo de trabajo sostenible**. Comprometidos con su trabajo, aplican una mirada crítica para mejorar sus resultados y sus métodos, y son conscientes de que lo que cuenta es su inteligencia colectiva, y no solo el talento de cada uno.

El scrum master y los principios

El scrum master ayuda a que el marco funcione y a que el equipo mejore, resolviendo los impedimentos que frenan su avance. Su presencia como rol dedicado es más aconsejable en equipos con poca experiencia o en organizaciones grandes, con rotación o incorporación frecuente de personas.

En equipos pequeños, estables y con una agilidad consolidada, las responsabilidades de funcionamiento y mejora del marco pueden estar interiorizadas y asumidas por todo el equipo. En esos casos, el scrum master suele evolucionar hacia el servicio a varios equipos o hacia la transformación de la organización (→«Un rol que evoluciona»).

Artefactos y eventos al servicio de los principios

Los artefactos y los eventos no son un fin en sí mismos. Cada uno existe para hacer posibles uno o varios principios, y conocer esa relación es lo que permite adaptarlos sin perder su sentido. Cuando un equipo modifica un artefacto o un evento, por ejemplo para integrar la IA, la pregunta que debe hacerse es si el cambio sigue sirviendo a los mismos principios.

Los artefactos

- **Product backlog.**
 - Entrega de valor: permite asumir los cambios del negocio del cliente y centrar el esfuerzo en lo que más valor aporta en cada momento.
 - Desarrollo iterativo e incremental: a diferencia de un documento de requisitos cerrado, es un documento vivo cuyos elementos y prioridades cambian.
 - Operativa visible: es un radiador de información con el que el product owner y el equipo comparten la visión del producto.
- **Sprint backlog.**
 - Desarrollo iterativo e incremental: delimita el trabajo de un sprint y marca el pulso del avance.
 - Operativa visible: es una herramienta de comunicación interna que muestra de un vistazo el estado del sprint y el reparto del trabajo, también el que hace la IA.
- **Incremento.**
 - Entrega de valor: presentar al final de cada sprint una parte terminada y lista para usarse permite comprobar si se está generando valor.
 - Atención continua a la excelencia: la Definición de Hecho garantiza que lo terminado cumple los criterios de calidad.

Los eventos

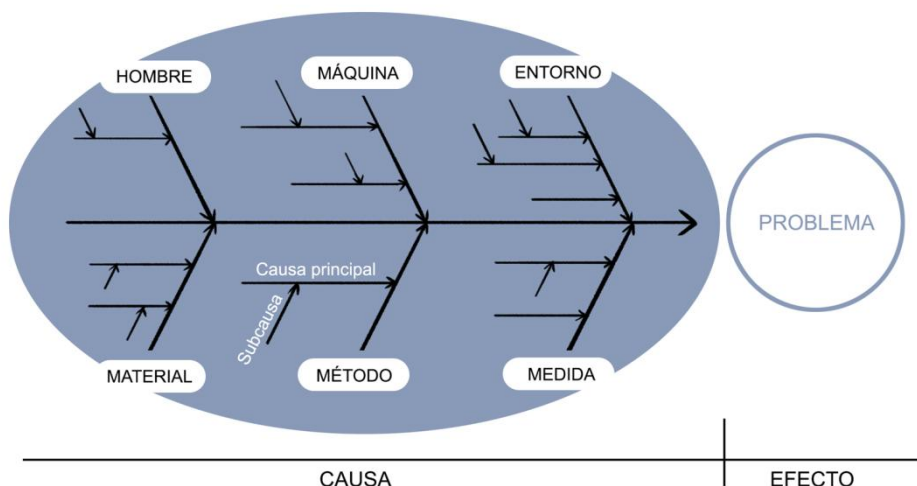
- **Sprint.**
 - Mejora continua: sus iteraciones breves ofrecen momentos regulares para reflexionar sobre el producto y la forma de trabajar.
 - Ritmo de trabajo sostenible: marca el pulso del avance.
 - Cadencia y sincronización global: hace previsibles las reuniones y las entregas y permite sincronizar equipos.
- **Reunión de planificación del sprint.**
 - Entrega de valor: el equipo y el product owner profundizan juntos en la visión y acuerdan el objetivo del sprint.
- **Scrum diario.**
 - Operativa visible: el equipo comparte el estado de su trabajo, también el que ha producido la IA, y detecta los impedimentos.
- **Revisión del sprint.**
 - Entrega de valor: el equipo y las personas interesadas comprueban si lo construido es lo que se necesita.
 - Mejora continua: sus conclusiones ayudan a orientar el siguiente sprint.
- **Retrospectiva del sprint.**
 - Mejora continua: el equipo analiza su forma de trabajar, incluida su colaboración con la IA, para decidir qué mantener y qué cambiar.

Métodos de trabajo para la calidad

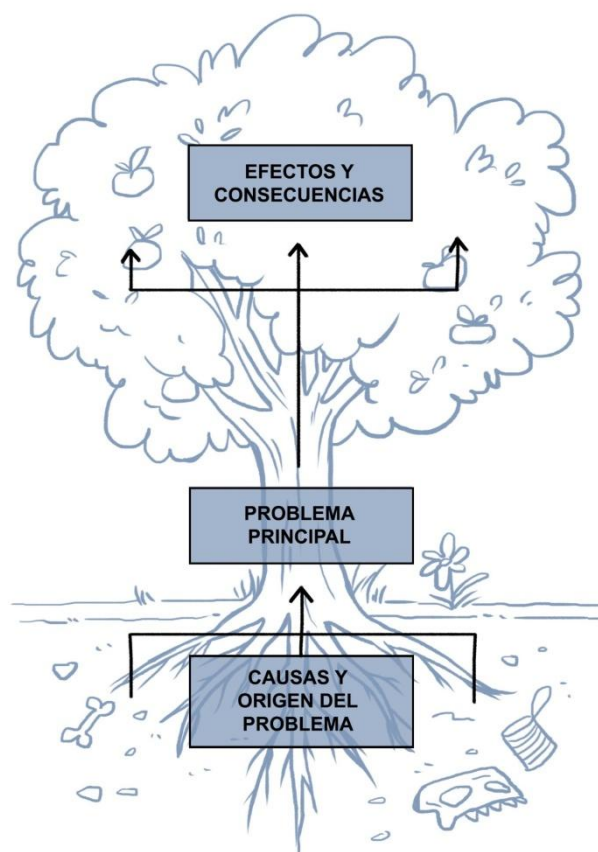
La comunidad profesional aporta muchas prácticas que completan el marco estándar. Ya vimos las de control del avance y estimación en la primera parte; aquí presentamos algunas centradas en la calidad del trabajo. No nos interesa explicar su operativa en profundidad, sino presentarlas como ejemplos y animar a investigar y experimentar con ellas.

Diagramas para retrospectivas: espina de pez y árbol

El **diagrama de espina de pez**, también llamado «de Ishikawa» o «de causa-efecto», representa las relaciones de causa y efecto que actúan sobre un problema. El problema ocupa el eje central y, en torno a él, se enumeran sus posibles causas y subcausas. Procede de la gestión de la calidad y es muy útil en las retrospectivas.



Los **diagramas de árbol** ayudan a llegar a conclusiones desde el análisis de una situación problemática. Usan los elementos del árbol (raíces, tronco, ramas, frutos) para representar las causas del problema, el problema mismo y sus consecuencias, o los medios para resolverlo y sus resultados. No existe un único tipo de diagrama de árbol ni un sistema estándar.



Técnicas a prueba de errores

Consisten en aplicar modos de trabajo que impiden que se produzcan errores. La agilidad emplea técnicas *poka-yoke* y dispositivos de control *andon*, dos conceptos que proceden de la manufactura *lean*. Las técnicas *poka-yoke* sirven para hacer imposible el error humano, como los enchufes que no pueden conectarse al revés, o para que el error salte a la vista cuando se produce, como hacen los correctores ortográficos. Los sistemas *andon* son indicadores visuales, con luces o gráficos, que alertan de inmediato al equipo de un fallo en su entorno de trabajo.

En programación es habitual el desarrollo guiado por pruebas (*test-driven development*, TDD), que consiste en escribir primero las pruebas que debe superar el código y después el código.

Especificar con ejemplos como prueba

Las técnicas a prueba de errores cobran nueva importancia con la IA. Los ejemplos que acompañan a una historia (—«Unidades de trabajo») pueden servir a la vez de instrucción y de prueba: se escriben antes de producir el trabajo y se usan después para comprobar si lo producido los cumple, igual que en el desarrollo guiado por pruebas.

La ventaja es que la comprobación no depende de quien ha producido el trabajo. Esto importa especialmente con la IA, que tiende a dar por bueno lo que ella misma ha producido. Un equipo de comunicación puede fijar antes de encargar un texto a la IA los ejemplos de lo que debe y no debe decir, y comprobar después el borrador contra ellos.

Trabajo en pareja

El trabajo en pareja consiste en asignar a dos personas una misma tarea, que realizan de forma simultánea y conjunta, normalmente alternando la ejecución y la supervisión. Mientras una la lleva a cabo, la otra observa y hace las indicaciones oportunas. En programación se conoce como programación en pareja (*pair programming*).

Es adecuado cuando la calidad del resultado depende sobre todo del conocimiento de quien lo realiza. Por eso los trenes circulan con maquinista y ayudante cuando la tecnología no puede evitar por completo el fallo humano, y nos sentimos más seguros sabiendo que en la cabina de un avión hay dos pilotos trabajando en pareja.

La pareja persona-IA

Con la IA, muchas personas trabajan ya en una pareja de otro tipo, en la que la IA produce y la persona supervisa. La analogía es útil, pero tiene un límite importante: en la pareja persona-IA, la supervisión es siempre de la persona, porque la IA no puede responder del resultado ni detectar con fiabilidad sus propios errores. Quien trabaja con la IA mantiene el papel de supervisor activo: lee con atención, cuestiona las propuestas y comprueba los datos, en lugar de limitarse a aceptar lo que aparece en pantalla.

La revisión cruzada

La pareja de personas sigue teniendo su lugar, y en algunos casos lo gana. Cuando una persona trabaja con IA, que otra persona revise el resultado aporta una mirada independiente que ni la IA ni quien la ha usado tienen. Además, el trabajo en pareja entre una persona con experiencia y otra que empieza es una de las mejores formas de transmitir el conocimiento tácito, el que la IA no puede enseñar (→«Aprendizaje y desarrollo de las personas»).

Parte III — El oficio del scrum master en equipos con IA

Acompañando la evolución.

Un rol que evoluciona

En la primera parte vimos que el scrum master es responsable de la eficacia del equipo scrum. En esta tercera parte vamos a ver qué significa esa responsabilidad cuando el equipo trabaja con IA, y qué ámbitos nuevos incorpora el oficio.

El rol de scrum master ha cambiado varias veces desde que existe scrum. Empezó muy centrado en que el equipo conociera y aplicara las reglas del marco, y con el tiempo se ha ido desplazando hacia la facilitación, el *coaching* del equipo y el cambio en la organización. La IA es un paso más en esa evolución, y no el final del rol.

Lo que la IA asume

Una parte de las tareas operativas del scrum master puede apoyarse en la IA: preparar agendas, tomar notas y resumir las reuniones, sugerir dinámicas para una retrospectiva, analizar las métricas de flujo o detectar patrones en los comentarios del equipo. Un scrum master que defina su valor solo por esas tareas verá cómo la IA lo hace prescindible.

En algunas organizaciones el título de scrum master ha perdido presencia y sus funciones se reparten entre otros puestos, como las personas responsables de la entrega o de los equipos técnicos. Pero las funciones siguen siendo necesarias, porque los equipos siguen necesitando a alguien que cuide su forma de trabajar, y la IA hace esa forma de trabajar más compleja.

Lo que crece

Lo que la IA no puede hacer es precisamente el centro del oficio: leer el estado de un equipo, detectar un conflicto latente, crear seguridad psicológica, ver las conexiones entre problemas que parecen independientes y acompañar a las personas en un cambio. A ello se suman ámbitos nuevos que trataremos en esta parte:

- facilitar el sistema de trabajo entre personas e IA;
- ayudar al equipo a acordar el uso de la IA y sus límites;
- promover la alfabetización en IA del equipo;
- cuidar la calidad frente al trabajo que solo parece terminado;
- resolver los impedimentos propios de los equipos con IA;
- proteger el aprendizaje de las personas y su bienestar en el cambio.

Nuevos nombres para el rol

Tendencia emergente (2026). El contenido de esta sección describe una tendencia reciente en el mercado laboral. No entra en el examen de certificación.

Algunas organizaciones han empezado a usar títulos nuevos para funciones próximas a las del scrum master en equipos con IA, como facilitador del flujo, orquestador de IA o habilitador ágil (*agile enabler*). Ninguno se ha asentado como estándar, y sus definiciones varían de una organización a otra. Lo que tienen en común es el foco en el sistema de trabajo entre personas e IA, que es el que esta guía atribuye al scrum master.

Facilitar el sistema de trabajo humano-IA

El scrum master siempre ha facilitado la interacción entre personas. En un equipo con IA, facilita además el **sistema de trabajo** completo: cómo se reparte el trabajo entre las personas y la IA, cómo fluye de una a otra, dónde se revisa y cómo se integra. Para ello necesita una mirada de conjunto, que vea el flujo de principio a fin en lugar de cada tarea por separado.

Lotes pequeños

La IA invita a producir en grandes cantidades: un informe completo en lugar de una sección, cincuenta piezas de campaña en lugar de cinco. Pero los lotes grandes son más difíciles de revisar, ocultan los errores y retrasan el feedback. Las investigaciones sobre equipos de software que trabajan con IA señalan el trabajo en lotes pequeños como una de las condiciones para que la IA se traduzca en mejores resultados (DORA, 2025).

El scrum master ayuda al equipo a mantener los lotes pequeños aunque la herramienta permita producir más: dividir el trabajo en unidades que se puedan revisar con atención, integrarlas con frecuencia y pedir opinión pronto.

La revisión como punto estrecho

Cuando la IA acelera la producción, el punto estrecho del flujo se desplaza a la revisión. Si el equipo no lo gestiona, el trabajo producido se acumula a la espera de revisión, las personas revisan con prisa y la calidad se resiente. El scrum master ayuda a hacer visible esa acumulación, con una columna de revisión en el tablero y su propio límite de trabajo en curso, y a que el equipo ajuste su ritmo de producción a su capacidad de revisar (→ «Medir el avance»).

Efectos no buscados

Toda intervención en un sistema produce efectos que nadie había previsto. Un equipo que delega en la IA la redacción de las actas de reunión puede descubrir que ya nadie las lee, porque son demasiado largas. Uno que genera más propuestas puede saturar a las personas que tienen que valorarlas. El scrum master está atento a estas consecuencias y las lleva a la retrospectiva para que el equipo decida qué hacer con ellas.

Acuerdos de uso de la IA en el equipo

Los equipos ágiles funcionan con **acuerdos de trabajo**: compromisos explícitos sobre cómo colaboran, que el propio equipo establece y revisa. La integración de la IA necesita sus propios acuerdos, que concretan las cuatro decisiones explícitas de «El equipo scrum y la IA». El scrum master ayuda al equipo a tomarlas, a ponerlas por escrito y a mantenerlas vivas.

Qué cubren los acuerdos

Unos acuerdos de uso de la IA suelen recoger:

- **Qué se delega:** las tareas en que el equipo usa la IA y las que reserva a las personas.
- **Cómo se revisa:** quién revisa cada tipo de trabajo producido por la IA y con qué criterios, en coherencia con la Definición de Hecho.
- **Qué información se le da:** qué datos pueden usarse con la IA y cuáles no, en especial los personales, los confidenciales y los de clientes.
- **Cómo se hace visible:** cómo se indica qué partes del trabajo ha producido la IA.
- **Qué herramientas:** cuáles están autorizadas por la organización y en qué condiciones.

Un equipo de recursos humanos, por ejemplo, puede acordar que la IA ayude a redactar ofertas de empleo y a resumir la normativa, pero que nunca intervenga en la valoración de candidaturas, y que ningún dato personal salga de las herramientas autorizadas por la organización.

El marco de la organización

Los acuerdos del equipo se sitúan dentro de las políticas de la organización sobre la IA y de la normativa que le sea aplicable. El equipo decide dentro de ese marco, y el scrum master ayuda a conocerlo y a resolver las situaciones en las que el marco no responde a una necesidad del equipo, llevándolas a quien corresponda en la organización. La integración de la IA en las distintas áreas de una organización se trata en el temario [Agile Bedrock](#) de Scrum Manager.

Límites para los agentes

Tendencia en consolidación (2026). El contenido de esta sección describe prácticas que ganan adopción con rapidez, pero aún no son generales. No entra en el examen de certificación.

Cuando el equipo usa agentes que ejecutan tareas completas, los acuerdos incluyen además sus **límites de actuación**: a qué sistemas e información pueden acceder, qué acciones pueden realizar sin aprobación humana y cuáles no, y cómo se detiene un agente cuando su actuación se desvía de lo esperado. Estos límites se revisan con más frecuencia que el resto de los acuerdos, porque las capacidades de los agentes cambian con rapidez.

Alfabetización en IA

Para usar la IA con criterio, todas las personas del equipo necesitan un conocimiento básico de cómo funciona, qué puede hacer y qué no. Esta **alfabetización en IA** no consiste en saber programarla, sino en entender lo suficiente para decidir cuándo usarla y cómo revisar lo que produce.

Qué necesita saber el equipo

Los contenidos básicos de la alfabetización en IA para un equipo son:

- **Sus capacidades y sus límites:** qué tipo de tareas resuelve bien y en cuáles falla con más frecuencia.
- **Sus errores característicos:** puede inventar datos o referencias con total apariencia de verdad, reproducir sesgos y dar respuestas distintas a la misma pregunta.
- **El uso de la información:** qué ocurre con los datos que se le proporcionan y qué riesgos tiene para la privacidad y la confidencialidad.
- **La confianza calibrada:** confiar en la IA en la medida en que se ha comprobado su fiabilidad para cada tipo de tarea, ni más ni menos.

Una obligación en la Unión Europea

En la Unión Europea, el Reglamento de Inteligencia Artificial (Unión Europea, 2024) obliga desde febrero de 2025 a las organizaciones que usan sistemas de IA a ocuparse de la alfabetización en IA de su personal. Tras su modificación de 2026 (Unión Europea, 2026), la obligación consiste en adoptar medidas que apoyen esa alfabetización, teniendo en cuenta la formación y la experiencia de cada persona y el contexto en que se usa la IA. El scrum master es un agente natural de esas medidas dentro del equipo.

Evitar el experto único

Un riesgo frecuente es que una sola persona del equipo se convierta en la experta en IA a la que todos consultan. El equipo depende entonces de ella, y el conocimiento no se reparte. El scrum master favorece que el aprendizaje sea de todo el equipo: compartir lo que funciona en el scrum diario o en la retrospectiva, practicar en pareja y dedicar tiempo a experimentar juntos.

Calidad y trabajo de relleno

La IA produce con facilidad trabajo que parece terminado: informes bien redactados, presentaciones vistosas o código que funciona en una primera prueba. Pero parecer terminado no es estar terminado. Se ha llamado **trabajo de relleno** (*workslop*) al trabajo producido con IA que tiene buena apariencia pero no hace avanzar la tarea, porque está incompleto, le falta contexto o contiene errores (Niederhoffer y otros, 2025).

El coste se traslada

Lo característico del trabajo de relleno es que su coste no lo paga quien lo produce, sino quien lo recibe. Un informe generado en cinco minutos puede costar una hora a la persona que tiene que leerlo, descubrir que no responde a la pregunta y rehacerlo. Además, quien lo envía pierde credibilidad ante sus compañeros, lo que deteriora la confianza en el equipo.

En un equipo scrum, el trabajo de relleno se manifiesta como incrementos que no superan la revisión, elementos que vuelven atrás o personas interesadas que dejan de confiar en lo que se les presenta. Es una forma de deuda que se acumula en silencio.

Cómo se previene

La prevención combina varias prácticas que ya conocemos:

- una **Definición de Hecho** que exija revisión humana de lo producido por la IA;
- **criterios de aceptación con ejemplos**, que permiten comprobar si el trabajo cumple su propósito;
- la **procedencia visible**, para que quien recibe el trabajo sepa cómo se ha producido;
- y un acuerdo de equipo claro: quien entrega un trabajo responde de él, lo haya producido como lo haya producido.

El scrum master ayuda a que el equipo hable abiertamente de este fenómeno en la retrospectiva, sin culpabilizar a nadie, y a que ajuste sus acuerdos en consecuencia.

Impedimentos en equipos con IA

Una de las responsabilidades clásicas del scrum master es resolver los impedimentos que frenan el avance del equipo. La IA no los elimina, sino que añade otros nuevos, y el scrum master tiene que aprender a reconocerlos.

Los impedimentos nuevos

Estos son algunos de los impedimentos más frecuentes en los equipos que trabajan con IA:

- **La revisión acumulada:** el trabajo producido espera a ser revisado y frena el flujo.
- **El contexto insuficiente:** la IA no tiene acceso a la información que necesita y produce resultados irrelevantes que hay que rehacer.
- **Las herramientas no autorizadas o mal configuradas:** el equipo no puede usar la IA que necesita, o la usa sin las garantías que exige la organización.
- **La deuda acumulada:** el trabajo producido deprisa y revisado a la ligera genera problemas que aparecen más tarde.
- **La dependencia:** el equipo depende de una herramienta o un proveedor concreto, o de la única persona que sabe usarlos.
- **El coste:** el uso de la IA tiene un coste económico que puede crecer sin que nadie lo vigile.

Cómo se abordan

Como cualquier otro impedimento, los nuevos se hacen visibles, se priorizan y se resuelven, por el propio equipo cuando está a su alcance o por el scrum master cuando dependen de otras áreas de la organización. Algunos tienen solución técnica, como dar a la IA acceso a la documentación que necesita; otros requieren decisiones de la organización, como autorizar una herramienta o fijar un presupuesto.

Muchos de estos impedimentos no se ven a simple vista. El scrum master los busca en las señales del flujo (trabajo que se acumula, elementos que vuelven atrás, plazos que se alargan) y en las conversaciones del equipo, y los lleva a la retrospectiva cuando su causa está en la forma de trabajar.

Aprendizaje y desarrollo de las personas

La IA plantea un reto para el aprendizaje de las personas que conviene conocer. Revisar con criterio lo que produce la IA exige experiencia, y la experiencia se adquiere haciendo el trabajo. Si la IA hace el trabajo y las personas solo lo revisan, ¿cómo adquieren la experiencia necesaria para revisarlo bien?

El riesgo para quienes empiezan

El riesgo es mayor para las personas que se incorporan a la profesión. Antes aprendían haciendo las tareas más sencillas, que hoy delega la IA con facilidad, y preguntando a sus compañeros con más experiencia, a quienes ahora sustituye a menudo la consulta a la IA. Un equipo que no presta atención a esto puede encontrarse dentro de unos años sin personas capaces de revisar el trabajo que la IA produce.

Lo que puede hacer el equipo

El scrum master ayuda al equipo a cuidar el aprendizaje de forma deliberada:

- reservar algunas tareas para que las personas con menos experiencia las realicen sin IA, o con ella pero con supervisión cercana;
- practicar el **trabajo en pareja** entre personas con distinta experiencia, la mejor forma de transmitir el conocimiento tácito;
- revisar juntos el trabajo de la IA, explicando en voz alta por qué algo está bien o mal;
- dedicar tiempo a aprender, también a aprender a usar mejor la IA.

Conservar el conocimiento tácito

La agilidad se basa en el conocimiento tácito de las personas, y ese conocimiento se forma en la práctica y en la relación con otros profesionales. Cuidar el aprendizaje del equipo no es una actividad secundaria, sino la condición para que el equipo siga siendo capaz de aportar el juicio que la IA no tiene.

Seguridad psicológica y cambio

La integración de la IA es un cambio profundo en la forma de trabajar, y como todo cambio genera incertidumbre. Algunas personas la viven con entusiasmo; otras, con preocupación por su puesto, por la pérdida de sentido de su trabajo o por no saber adaptarse. El scrum master cuida de que el equipo pueda hablar de todo ello con franqueza.

Seguridad psicológica

La **seguridad psicológica** es la confianza de que en el equipo se puede hablar, preguntar, discrepar o reconocer un error sin temor a consecuencias negativas. Con la IA es más necesaria que nunca. Las personas necesitan poder decir que no entienden cómo funciona una herramienta, que han cometido un error al revisar lo que producía o que les preocupa el futuro de su puesto.

Sin seguridad psicológica, los problemas se ocultan: nadie reconoce que ha dado por bueno un trabajo sin revisarlo, ni que no sabe usar la herramienta que todos parecen dominar.

Las tensiones del cambio

El cambio genera tensiones que el scrum master ayuda a gestionar:

- **El miedo al reemplazo**, que se atenúa con información clara de la organización sobre sus intenciones y con la participación del equipo en las decisiones sobre la IA.
- **La carga mental** de revisar continuamente trabajo ajeno, que puede resultar más agotadora que producir el propio.
- **Las curvas de aprendizaje desiguales**, que pueden crear brechas entre quienes adoptan la IA con rapidez y quienes necesitan más tiempo.
- **Los conflictos de criterio**, entre quienes quieren usar la IA para todo y quienes desconfían de ella.

Experimentos pequeños y reversibles

La mejor forma de avanzar en un cambio con tanta incertidumbre es experimentar a pequeña escala. El equipo prueba un uso concreto de la IA durante uno o dos sprints, observa los resultados y decide en la retrospectiva si lo mantiene, lo ajusta o lo abandona. Los experimentos pequeños reducen el riesgo, dan a las personas tiempo para adaptarse y convierten la integración de la IA en un proceso empírico, coherente con todo lo que hemos visto en esta guía.

Antipatrones en equipos con IA

Un antipatrón es una forma de actuar que parece razonable pero produce malos resultados de forma sistemática. Conocer los antipatrones más frecuentes ayuda al scrum master a reconocerlos pronto y a ayudar al equipo a corregirlos. Estos son algunos de los que aparecen en los equipos que trabajan con IA.

Antipatrones del flujo de trabajo

- **La velocidad ilusoria:** el equipo celebra que produce más, sin comprobar si entrega más valor.
- **Producir sin revisar:** el trabajo de la IA se da por bueno porque parece correcto o porque funciona en una primera prueba.
- **La revisión que no se planifica:** el sprint se llena de producción y no queda tiempo para revisar.
- **Sobreproducir para la revisión del sprint:** se presenta más de lo que las personas interesadas pueden valorar.

Antipatrones de los eventos

- **Suprimir la retrospectiva:** se eliminan los momentos de reflexión para ir más deprisa, justo cuando más se necesitan.
- **Acortar el sprint por inercia:** se cambia la cadencia porque la IA permite producir más rápido, y no porque el equipo necesite inspeccionar con más frecuencia.
- **El scrum diario sin conversación:** la reunión se sustituye por un resumen automático que nadie comenta.

Antipatrones de la responsabilidad y el aprendizaje

- **«Lo hizo la IA»:** la responsabilidad del resultado se diluye en la herramienta.
- **Delegar el juicio:** se encarga a la IA decidir lo que corresponde a las personas, como qué es valioso o qué riesgo es aceptable.
- **Acuerdos implícitos:** cada persona usa la IA a su manera, sin acuerdos de equipo.
- **El experto único:** todo el conocimiento sobre la IA se concentra en una persona.
- **Medir la adopción:** se mide cuánto se usa la IA en lugar de qué resultados produce.

Ante cualquiera de ellos, la respuesta del scrum master es la de siempre: hacerlo visible, llevarlo a la retrospectiva y ayudar al equipo a decidir cómo corregirlo.

Scrum con IA fuera del software

La mayor parte de lo que se publica sobre la IA en los equipos ágiles procede del desarrollo de software. Sin embargo, scrum se aplica en todo tipo de equipos del conocimiento, y la IA ha llegado también a ellos. Este capítulo recoge lo que cambia cuando el equipo no desarrolla software.

Dónde aporta más la IA

La IA rinde más en las tareas de redacción, síntesis, análisis de información e ideación, que son centrales en muchos equipos no técnicos. En un experimento con profesionales de innovación de producto, las personas que trabajaban con IA alcanzaron una calidad comparable a la de parejas que trabajaban sin ella (Dell'Acqua y otros, 2025). En un estudio con agentes de atención al cliente, la IA mejoró la productividad media, sobre todo la de las personas con menos experiencia (Brynjolfsson y otros, 2025).

Su aportación es menor en el trabajo que depende de la relación personal, del conocimiento profundo de un contexto concreto o de decisiones con consecuencias importantes para las personas.

Lo que cambia respecto al software

Los equipos que no desarrollan software tienen dos diferencias que condicionan el uso de la IA:

- **No suelen tener comprobaciones automáticas**, como las pruebas que verifican el código. La calidad depende casi por completo de la revisión humana, lo que hace todavía más importantes la Definición de Hecho y los criterios con ejemplos.
- **Trabajan a menudo con información sensible y con exigencias regulatorias**. En la Unión Europea, por ejemplo, los sistemas de IA usados en la selección o evaluación de personas se consideran de alto riesgo, con obligaciones específicas.

Ejemplos por áreas

- Un equipo de **marketing** usa la IA para generar variantes de una campaña; el valor está en elegir y validar con datos cuáles funcionan.
- Un equipo de **recursos humanos** la usa para redactar comunicaciones y resumir normativa, pero mantiene fuera de ella las decisiones sobre personas.
- Un equipo de **asesoría legal** la usa para la investigación previa, con revisión obligatoria de cada referencia, porque la IA puede inventarlas.
- Un equipo **docente** la usa para preparar materiales y actividades, y revisa su adecuación al nivel y a los objetivos de aprendizaje.

Lo que todavía no sabemos

La evidencia sobre el uso de la IA en equipos scrum que no desarrollan software es todavía escasa: la mayoría de los estudios analizan tareas o profesiones concretas, no equipos ágiles. Por eso conviene que cada equipo trate su forma de integrar la IA como un experimento y aprenda de su propia experiencia. El análisis de cómo cambia el impacto de la IA según el tipo de trabajo se desarrolla en el temario [Agile Bedrock](#) de Scrum Manager.

Apéndice

Glosario de los términos del temario, referencias bibliográficas, recursos y comunidad profesional de Scrum Manager, y control de calidad de la formación.

Glosario

Acuerdos de uso de la IA: acuerdos del equipo sobre qué se delega en la IA, quién revisa, qué información se le da y qué límites tiene.

Agilidad: gestión evolutiva basada en las personas; se centra en su conocimiento tácito, la cultura y el talento, y elimina las tareas que no aportan valor.

Alfabetización en IA: conocimiento básico de cómo funciona la IA, qué puede hacer, cuáles son sus errores característicos y cómo revisar lo que produce.

Antipatrón: forma de actuar que parece razonable pero produce malos resultados de forma sistemática.

Artefactos: herramientas de scrum que hacen transparente el trabajo: product backlog, sprint backlog e incremento.

Comprometidos: quienes intervienen directamente en la construcción del producto y responden de su resultado; los miembros del equipo scrum.

Conocimiento explícito: conocimiento codificado, que puede transmitirse con lenguaje formal y sistemático.

Conocimiento tácito: conocimiento personal y específico del contexto, difícil de formalizar y comunicar.

Definición de Hecho (Definition of Done, DoD): acuerdo del equipo sobre los criterios que debe cumplir un trabajo para considerarse terminado.

Definition of Ready (DoR): acuerdo sobre cuándo un elemento del product backlog está listo para entrar en un sprint.

Desarrollador: cada profesional que construye el incremento, no solo quien programa; forman un equipo multifuncional y autogestionado.

Empirismo: forma de trabajar que toma las decisiones a partir de lo que se observa, basada en la transparencia, la inspección y la adaptación.

Épica: historia demasiado grande y poco definida (epic) para servir como unidad de trabajo.

Equipo scrum: el formado por el product owner, los desarrolladores y el scrum master; diez personas como máximo.

Especificar con ejemplos: acompañar una historia de casos concretos del resultado esperado, que sirven de instrucción y de prueba.

Estimación de póquer: técnica ideada por James Grenning para estimar con cartas el tamaño de las unidades de trabajo.

Eventos: sprint, reunión de planificación del sprint, scrum diario, revisión del sprint y retrospectiva del sprint.

Gestión evolutiva: gestión que entrega pronto un producto mínimo viable e incrementa su valor de forma continua.

Gestión predictiva: gestión que busca resultados predecibles en plazo, coste y alcance; también, ingeniería secuencial.

Gráfico de avance: gráfico (burn down chart) del trabajo pendiente del sprint frente a sus días.

Gráfico de producto: gráfico (burn up chart) del product owner con lo construido y la previsión de lo que queda.

Historia de usuario: descripción breve del valor que el producto debe proporcionar al usuario, escrita desde su perspectiva.

Implicados: partes interesadas que aportan al proyecto sin responder del resultado del sprint, como la dirección o el soporte a usuarios.

Incremento: parte del producto terminada, probada y operativa; en un sprint puede haber uno o varios.

Kanban: técnica para gestionar de forma visual un flujo continuo de trabajo sobre un tablero.

Objetivo del sprint: el valor que debe aportar un sprint; orienta las decisiones del equipo durante su desarrollo.

Procedencia: información sobre qué partes de un trabajo ha producido la IA y quién las ha revisado.

Product backlog: inventario ordenado de todo lo que puede necesitar el producto; lo mantiene el product owner.

Product owner: quien toma las decisiones del cliente y responde de maximizar el valor del producto.

Puntos de historia: unidad relativa para estimar el tamaño del trabajo, por comparación con un trabajo conocido.

Refinamiento: labor de ordenar, detallar y preestimar el product backlog; no más del 10 % de la capacidad del equipo.

Retrospectiva del sprint: reunión en la que el equipo scrum reflexiona sobre su forma de trabajar y planifica mejoras.

Reunión de planificación del sprint: reunión que inicia el sprint y fija su objetivo y el trabajo para conseguirlo.

Revisión del sprint: reunión al final del sprint para inspeccionar el incremento con las personas interesadas.

Scrum: modelo de desarrollo ágil con equipos autogestionados, desarrollo incremental, fases solapadas y calidad basada en el conocimiento tácito.

Scrum diario: reunión de 15 minutos como máximo en la que los desarrolladores sincronizan su trabajo.

Scrum master: responsable de la eficacia del equipo scrum, al servicio del equipo, del product owner y de la organización.

Seguridad psicológica: confianza en que en el equipo se puede preguntar, discrepar o reconocer un error sin temor.

Sprint: periodo de cuatro semanas como máximo en el que se construyen uno o varios incrementos.

Sprint backlog: lista del trabajo necesario para lograr el objetivo del sprint; es territorio de los desarrolladores.

Tarea: unidad de trabajo más pequeña que una historia, de un día o menos.

Tiempo de ciclo: lo que tarda una unidad de trabajo desde que se empieza hasta que se termina.

Tiempo ideal: tiempo de trabajo sin distracciones ni impedimentos; el tiempo real suele ser mayor.

Trabajo de relleno: trabajo producido con IA que parece terminado pero no hace avanzar la tarea (workslop).

Velocidad: cantidad de trabajo, en puntos, que el equipo completa en un sprint.

WIP: número máximo de unidades de trabajo que pueden estar a la vez en una fase (work in process).

Referencias bibliográficas

Las obras y los recursos de referencia del temario, en orden alfabético.

- Beck, Kent (1999). *Extreme Programming Explained: Embrace Change*, Addison-Wesley.
- Beck, Kent y otros (2001). *Manifiesto por el Desarrollo Ágil de Software*: <https://agilemanifesto.org/iso/es/manifesto.html>
- Becker, Joel; Rush, Nate; Barnes, Elizabeth y Rein, David (2025). *Measuring the Impact of Early-2025 AI on Experienced Open-Source Developer Productivity*, METR.
- Brynjolfsson, Erik; Li, Danielle y Raymond, Lindsey (2025). «Generative AI at Work», *The Quarterly Journal of Economics*, 140(2), pp. 889-942.
- Cockburn, Alistair (2006). *Agile Software Development: The Cooperative Game* (2.^a ed.), Addison-Wesley.
- Dell'Acqua, Fabrizio y otros (2025). «The Cybernetic Teammate: A Field Experiment on Generative AI Reshaping Teamwork and Expertise», *NBER Working Paper* 33641, National Bureau of Economic Research.
- DORA (2025). *State of AI-assisted Software Development 2025*, Google Cloud.
- Grenning, James (2002). *Planning Poker, or How to Avoid Analysis Paralysis While Release Planning*, Renaissance Software Consulting.
- Hackman, J. Richard y Vidmar, Neil (1970). «Effects of size and task type on group performance and member reactions», *Sociometry*, 33(1), pp. 37-54.
- Highsmith, Jim (2001). «History: The Agile Manifesto»: <https://agilemanifesto.org/history.html>
- Ishikawa, Kaoru (1985). *What Is Total Quality Control? The Japanese Way*, Prentice Hall.
- Kerth, Norman L. (2001). *Project Retrospectives: A Handbook for Team Reviews*, Dorset House.
- Naur, Peter y Randell, Brian (eds.) (1969). *Software Engineering: Report on a Conference Sponsored by the NATO Science Committee, Garmisch, Alemania, 7-11 de octubre de 1968*, NATO Scientific Affairs Division.
- Niederhoffer, Kate; Rosen Kellerman, Gabriella; Lee, Angela; Liebscher, Alex; Rapuano, Kristina y Hancock, Jeffrey T. (2025). «AI-Generated "Workslop" Is Destroying Productivity», *Harvard Business Review*.
- Nonaka, Ikujiro y Takeuchi, Hirotaka (1995). *The Knowledge-Creating Company: How Japanese Companies Create the Dynamics of Innovation*, Oxford University Press.

- Nonaka, Ikujiro y Toyama, Ryoko (2004). «Knowledge Creation as a Synthesizing Process». En Hirotaka Takeuchi e Ikujiro Nonaka (eds.), *Hitotsubashi on Knowledge Management*, pp. 91-124, John Wiley & Sons.
- Orr, Ken (2002). *CMM versus Agile Development: Religious Wars and Software Development*, Cutter Consortium, Agile Project Management Executive Report, vol. 3, n.º 7.
- Parkinson, Cyril Northcote (1955). «Parkinson's Law», *The Economist*, 19 de noviembre.
- Schwaber, Ken (1995). «SCRUM Development Process», *OOPSLA '95 Workshop on Business Object Design and Implementation*, Springer.
- Schwaber, Ken (2006). *Scrum Et Al.*, Google Tech Talks.
- Scrum Manager Podcast (2023). *Cómo crear un backlog*, episodio 1x02: <https://open.spotify.com/episode/7hGoEqV2uZL47uZZN6GXfk?si=2eb790bebe644aa8>
- Scrum Manager Podcast (2023). *Cómo se crea una historia de usuario*, episodio 1x03: <https://open.spotify.com/episode/07cSgNf9znGHU93kAm2qaU?si=9813747ed602458b>
- Scrum Manager Podcast (2023). *Eventos de scrum*, episodio 1x04: <https://open.spotify.com/episode/7basfleLN9L5A6FMOGHAF3?si=7d5e6974f32f49af>
- Scrum Manager Podcast (2023). *Definition of Done*, episodio 1x06: <https://open.spotify.com/episode/0GjNQprSs7wbQ8s6G0IGXu?si=2d94f9fd7d1f4770>
- Scrum Manager Podcast (2023). *Estimar o no estimar*, episodio 1x10: <https://open.spotify.com/episode/5qa41nYkK9JSmKVg0E5qt0?si=14aafb0778a44d09>
- Scrum Manager Podcast (2024). *Las 5 claves de un buen product owner*, episodio 2x02: <https://open.spotify.com/episode/5OwC380a6pWE1iV59Mchqo?si=80e27682603a42b4>
- Scrum Manager Podcast (2024). *5 características de una buena historia de usuario*, episodio 2x07: <https://open.spotify.com/episode/5PMBIIhu0cuW9GkA2xNGV5?si=6e579095f4e542d5>
- Scrum Manager YouTube (2017). *Gráfico de producto burn up*: <https://www.youtube.com/watch?v=jdyodO-uFBk>

- Scrum Manager YouTube (2017). *Objetivo de la gestión predictiva y de la gestión ágil*:
<https://www.youtube.com/watch?v=57pcynW3pQo>
- Scrum Manager YouTube (2018). *Gestión predictiva y evolutiva*:
<https://www.youtube.com/watch?v=vQrFunXuURc>
- Scrum Manager YouTube (2021). *Gráfico de avance «Burn down chart»*:
<https://www.youtube.com/watch?v=hveuhx7rZgw>
- Scrum Manager YouTube (2021). *Estimación en la pared*:
<https://www.youtube.com/watch?v=4O2BRg9kDU4>
- Scrum Manager YouTube (2021). *Scrum (es)*:
<https://www.youtube.com/watch?v=SIzfzLRGYBA>
- Takeuchi, Hirotaka y Nonaka, Ikujiro (1986). «The New New Product Development Game», *Harvard Business Review*, 64(1), pp. 137-146.
- Turner, Richard y Jain, Apurva (2002). «Agile Meets CMMI: Culture Clash or Common Cause?», en *Extreme Programming and Agile Methods — XP/Agile Universe 2002*, Lecture Notes in Computer Science, vol. 2418, Springer.
- Unión Europea (2024). Reglamento (UE) 2024/1689 del Parlamento Europeo y del Consejo, de 13 de junio de 2024, por el que se establecen normas armonizadas en materia de inteligencia artificial (Reglamento de Inteligencia Artificial), *Diario Oficial de la Unión Europea*.
- Unión Europea (2026). Reglamento (UE) 2026/1744 del Parlamento Europeo y del Consejo, de 8 de julio de 2026, por el que se modifican los Reglamentos (UE) 2024/1689, (UE) 2018/1139 y (UE) 2023/1230 en lo que respecta a la simplificación de la aplicación de normas armonizadas en materia de inteligencia artificial (Ómnibus digital sobre IA), *Diario Oficial de la Unión Europea*.

Información, recursos y comunidad profesional

Scrum Manager mantiene un ecosistema de recursos abiertos y de comunidad para seguir aprendiendo después de este temario:

[Club Agile →](#)[Skill Arena →](#)[Academy →](#)[BoK/Wiki →](#)[Comunidad →](#)[Tiras de cómic →](#)[Acerca de la certificación Scrum Manager® →](#)[Preguntas frecuentes sobre cursos y exámenes Scrum Manager® →](#)

Síguenos en  [LinkedIn](#)

Mejora continua y control de calidad

El control de calidad de Scrum Manager se basa en las valoraciones de sus estudiantes. Con tu opinión nos ayudas a mantener el nivel de nuestros materiales, cursos, centros y profesores.

Si has participado en una actividad de formación auditada por Scrum Manager®, te rogamos y agradecemos que valores la calidad de la formación recibida. La información que se recoge es anónima. Puedes enviarnos tus comentarios desde el «Área de miembros»: <https://scrummanager.com>