Módulo 2. Fundamentos del marco Scrum

El marco *Scrum* se ha consolidado como una de las propuestas más difundidas dentro de los enfoques ágiles para la gestión del trabajo en contextos complejos e inciertos. Su adopción responde a la necesidad de contar con formas de organización que permitan aprender de manera progresiva, ajustar decisiones y generar valor en entornos donde los requisitos y las prioridades cambian con frecuencia. En este sentido, *Scrum* no se presenta como una metodología prescriptiva, sino como un marco que orienta la colaboración, la transparencia y la adaptación continua.

Este módulo se centra en el análisis de *Scrum* desde una doble perspectiva. Por un lado, se abordan sus fundamentos, componentes y principios, con el objetivo de comprender su origen, su propósito y la lógica que sostiene su funcionamiento. Por otro lado, se profundiza en la gestión de valor y en el trabajo colaborativo, aspectos centrales para interpretar cómo *Scrum* articula estrategia y ejecución en el día a día de los equipos ágiles.

A lo largo de la lectura, se examinan los roles, eventos y artefactos que conforman el marco *Scrum*, así como su contribución a la organización del trabajo en ciclos iterativos. Asimismo, se analizan herramientas clave como el *product backlog*, el *sprint backlog* y la *Definition of Done,* entendidas como mecanismos que favorecen la toma de decisiones informadas y la responsabilidad compartida.

Finalmente, el módulo incorpora una mirada comparativa entre *Scrum* y *Kanban*, y aborda la relación con los clientes desde la lógica de los contratos ágiles. El propósito es brindarte un marco conceptual que te permita

comprender cómo *Scrum* facilita la gestión del valor y el trabajo colaborativo en equipos ágiles, más allá de la aplicación mecánica de sus prácticas.

1. El marco Scrum: origen, propósito y fundamentos

***Scrum* se presenta como un marco de trabajo diseñado para abordar problemas complejos mediante soluciones adaptativas, incrementales y colaborativas. A diferencia de los enfoques tradicionales de gestión de proyectos, *Scrum* no prescribe un conjunto rígido de pasos a seguir, sino que establece una estructura mínima que permite a los equipos aprender a partir de la experiencia y ajustar su forma de trabajo de manera continua.**

Según Drumond (s. f.), *Scrum* surge como parte del movimiento ágil y se apoya en la idea de que el conocimiento completo de un

problema no está disponible desde el inicio. En consecuencia, propone ciclos cortos de trabajo que permiten inspeccionar resultados parciales y adaptar decisiones en función de nueva información. Este enfoque resulta especialmente relevante en contextos donde los requisitos cambian con frecuencia o no pueden definirse con precisión desde el comienzo.

Martins (2025) explica que *Scrum* se utiliza principalmente para gestionar el desarrollo de productos complejos, aunque su aplicación se ha extendido a otros ámbitos organizacionales. Su propósito central consiste en facilitar la entrega continua de valor mediante la colaboración entre personas con distintas competencias, promoviendo la transparencia y la responsabilidad compartida.

Desde una perspectiva conceptual, *Scrum* se apoya en el control empírico de procesos. Esto implica tomar decisiones basadas en la observación de lo que ocurre en la práctica y no únicamente en supuestos teóricos. Drumond (s. f.) señala que este control empírico se sostiene en tres pilares: transparencia, inspección y adaptación. Estos pilares orientan la manera en que los equipos observan su trabajo, detectan desvíos y ajustan su comportamiento.

Para ti, como estudiante, resulta importante comprender que *Scrum* no garantiza resultados por sí mismo. Funciona como un

marco que requiere compromiso, reflexión y aprendizaje continuo por parte de quienes lo utilizan. Su efectividad depende de la coherencia entre sus principios y las prácticas reales del equipo.

Roles de *Scrum*: responsabilidades y colaboración

Uno de los elementos distintivos de *Scrum* es la definición clara de roles, cada uno con responsabilidades específicas orientadas a facilitar la colaboración y la entrega de valor. A diferencia de los modelos jerárquicos tradicionales, estos roles no se organizan en función de autoridad formal, sino de responsabilidades diferenciadas dentro de un mismo equipo.

Martins (2025) señala que *Scrum* define tres roles principales: *product owner*, *Scrum master* y equipo de desarrollo. Estos roles conforman lo que se denomina *Scrum* Team, y su interacción resulta central para el funcionamiento del marco.

El *product owner* es responsable de maximizar el valor del producto. Según Drumond (s. f.), esta responsabilidad implica gestionar y priorizar el *product backlog*, asegurando que el trabajo del equipo se alinee con las necesidades del cliente y los objetivos

del negocio. El *product owner* actúa como nexo entre las partes interesadas y el equipo, pero no dirige técnicamente el trabajo.

El *Scrum master* cumple un rol facilitador. Martins (2025) explica que su función principal consiste en asegurar que *Scrum* sea comprendido y aplicado correctamente. Esto incluye acompañar al equipo en la adopción de prácticas ágiles, eliminar impedimentos que afecten el trabajo y promover la mejora continua. El *Scrum master* no actúa como jefe del equipo, sino como un apoyo para su funcionamiento.

El equipo de desarrollo está conformado por profesionales que trabajan de manera colaborativa para crear el incremento del producto. Drumond (s. f.) destaca que estos equipos son autoorganizados y multifuncionales, lo que significa que deciden internamente cómo realizar el trabajo y cuentan con las competencias necesarias para hacerlo.

Desde el punto de vista del trabajo en equipo, esta estructura de roles favorece la responsabilidad compartida y reduce la dependencia de decisiones centralizadas. Para ti, comprender esta lógica implica reconocer que *Scrum* redefine la manera en que se distribuyen las responsabilidades dentro de los equipos.

Además de su origen en el movimiento ágil, el marco *Scrum* se distingue por proponer una forma particular de abordar la complejidad. Martins (2025) explica que *Scrum* resulta especialmente adecuado cuando los problemas no pueden resolverse mediante soluciones lineales o completamente previsibles. En estos contextos, intentar definir todos los requisitos desde el inicio suele conducir a retrabajos, demoras y pérdida de valor. *Scrum* responde a esta situación mediante ciclos cortos de aprendizaje que permiten ajustar decisiones de manera progresiva.

El propósito de *Scrum* no consiste en eliminar la incertidumbre, sino en gestionarla de forma explícita. Drumond (s. f.) señala que el marco asume que el conocimiento sobre el producto se construye durante el proceso de desarrollo y no antes de comenzarlo. Por este motivo, *Scrum* enfatiza la inspección frecuente de los resultados parciales y la adaptación del plan en función de lo aprendido. Esta lógica contrasta con enfoques predictivos, donde los cambios suelen percibirse como desviaciones indeseadas.

Desde el punto de vista de sus fundamentos, *Scrum* se apoya en la idea de equipos pequeños, estables y con un alto grado de autonomía. Martins (2025) destaca que la autoorganización no implica ausencia de reglas, sino la existencia de un marco claro que permite a los equipos decidir cómo alcanzar los objetivos definidos. Esta combinación de estructura mínima y libertad operativa busca favorecer la responsabilidad colectiva y la colaboración.

Otro fundamento relevante del marco *Scrum* es su orientación a la entrega incremental de valor. Cada *sprint* produce un incremento que permite evaluar avances concretos y obtener retroalimentación temprana. Drumond (s. f.) explica que esta entrega frecuente reduce el riesgo asociado a grandes desarrollos prolongados, ya que posibilita detectar problemas o desalineaciones en etapas tempranas del proceso.

Scrum también propone una redefinición del éxito del proyecto. En lugar de medirlo únicamente por el cumplimiento de un plan inicial, se pone el foco en la capacidad del equipo para responder a cambios y entregar valor de manera sostenida. Martins (2025) advierte que esta perspectiva requiere un cambio cultural, tanto en los equipos como en las organizaciones, ya que implica aceptar que la planificación tiene límites en entornos complejos.

Para ti, como estudiante, comprender estos fundamentos permite interpretar *Scrum* más allá de sus componentes visibles. El marco no se reduce a eventos, roles o artefactos, sino que expresa una manera de concebir el trabajo, el aprendizaje y la toma de decisiones en contextos inciertos. Este entendimiento resulta clave para aplicar *Scrum* de forma coherente y evitar interpretaciones superficiales centradas únicamente en la mecánica del proceso.

SCRUM Y EL TRABAJO EN EQUIPO ÁGIL

Más allá de sus componentes técnicos, ***Scrum*** propone una forma particular de concebir el trabajo en equipo. Drumond (s. f.) señala que su énfasis en la colaboración, la autoorganización y la mejora continua responde a la necesidad de gestionar el conocimiento distribuido entre las personas.

Martins (2025) explica que los equipos ***Scrum*** funcionan mejor cuando existe confianza, comunicación abierta y responsabilidad compartida. Estas condiciones no se imponen mediante reglas formales, sino que se construyen a través de prácticas coherentes y reflexión constante.

Los eventos, roles y artefactos de ***Scrum*** funcionan como soportes para este trabajo colaborativo. Owen (2024) destaca que la regularidad de los eventos facilita la creación de rutinas que sostienen el aprendizaje del equipo.

Para el estudiante, comprender ***Scrum*** desde esta perspectiva implica reconocer que no se trata únicamente de una técnica de gestión, sino de un marco que redefine la forma en que las personas trabajan juntas para enfrentar problemas complejos.

Eventos de Scrum y ciclos de trabajo iterativos

Scrum organiza el trabajo a través de eventos que estructuran ciclos iterativos denominados *sprint*s. Estos eventos no son reuniones administrativas aisladas, sino instancias diseñadas para facilitar la planificación, la inspección y la adaptación del trabajo.

Luzhko (2025) define el *sprint* como una iteración de duración fija, generalmente de una a cuatro semanas, durante la cual el equipo desarrolla un incremento del producto. Cada *sprint* tiene un objetivo claro y produce un resultado potencialmente utilizable.

Esta estructura permite dividir problemas complejos en partes manejables.

El ciclo comienza con la planificación del *sprint*. Según Drumond (s. f.), en este evento el equipo define qué trabajo se realizará y cómo se llevará a cabo. La planificación permite alinear expectativas y establecer un objetivo compartido para el ciclo de trabajo.

Durante el *sprint*, el equipo realiza reuniones diarias conocidas como *daily Scrum*. Owen (2024) explica que estas reuniones breves tienen como propósito sincronizar el trabajo del equipo, identificar impedimentos y ajustar el plan diario. No se trata de reportar avances a una autoridad, sino de facilitar la coordinación interna.

Al finalizar el *sprint*, se realiza la revisión del *sprint*. Luzhko (2025) señala que este evento permite inspeccionar el incremento desarrollado y obtener retroalimentación de las partes interesadas. La revisión favorece la adaptación del *product backlog* en función de lo aprendido.

Finalmente, la retrospectiva del *sprint* ofrece un espacio para reflexionar sobre el proceso de trabajo. Owen (2024) destaca que en esta instancia el equipo analiza qué funcionó, qué puede mejorarse y qué acciones concretas implementará en el próximo ciclo.

Para el estudiante, estos eventos ilustran cómo *Scrum* integra planificación, ejecución y reflexión en un proceso continuo de aprendizaje.

Artefactos de Scrum y transparencia del trabajo

Los artefactos de *Scrum* cumplen la función de hacer visible el trabajo y facilitar la toma de decisiones informadas. A través de ellos, el equipo y las partes interesadas pueden comprender qué se está haciendo, qué se ha logrado y qué queda por hacer.

Martins (2025) identifica tres artefactos principales: *product backlog*, *sprint backlog* e incremento. Cada uno cumple un rol específico dentro del marco.

El *product backlog* es una lista ordenada de todo el trabajo necesario para el producto. Drumond (s. f.) señala que su contenido evoluciona a lo largo del tiempo, reflejando nuevas necesidades, aprendizajes y cambios de contexto. Su gestión recae en el *product owner*, aunque su refinamiento involucra al equipo.

El *sprint backlog* representa el trabajo seleccionado para un *sprint* específico. Según Martins (2025), este artefacto permite al equipo visualizar su compromiso y realizar ajustes durante el ciclo de trabajo. Su transparencia facilita la coordinación y la autoorganización.

El incremento es el resultado del trabajo realizado durante el *sprint*. Luzhko (2025) explica que este incremento debe cumplir con criterios de calidad definidos previamente, lo que permite evaluar objetivamente el progreso.

Desde una perspectiva pedagógica, estos artefactos permiten comprender cómo *Scrum* promueve la transparencia como condición para la inspección y la adaptación. Para ti, representan herramientas que articulan trabajo individual y colectivo dentro de un marco compartido.

CONTINUAR

2. Gestión de valor y trabajo colaborativo en equipos ágiles

Backlog y gestión de valor: *product backlog* y *sprint backlog*

La gestión de valor constituye uno de los ejes centrales del marco *Scrum* y de los enfoques ágiles en general. A diferencia de los modelos tradicionales, donde el éxito suele medirse por el cumplimiento de un plan previamente definido, *Scrum* propone evaluar el progreso a partir del valor entregado de manera incremental. En este contexto, los *backlogs* se convierten en herramientas clave para organizar, priorizar y transparentar el trabajo.

UNIR (2025) define el *product backlog* como una lista ordenada y dinámica que contiene todo aquello que podría ser necesario para el producto. No se trata de un documento cerrado ni exhaustivo desde el inicio, sino de un artefacto vivo que evoluciona a medida que se incorpora nueva información, se obtiene retroalimentación y cambian las prioridades del entorno.

Esta característica lo diferencia de los enfoques tradicionales de definición de requisitos.

Reaburn (2025) explica que el *product backlog* cumple una función estratégica, ya que permite visualizar el trabajo pendiente y orientar las decisiones hacia la generación de valor. Los elementos que lo componen suelen expresarse en términos de funcionalidades, mejoras o correcciones, priorizadas según su impacto esperado. Esta priorización no responde únicamente a criterios técnicos, sino que integra consideraciones de negocio y de usuario.

La responsabilidad principal sobre el *product backlog* recae en el *product owner*, quien debe asegurar que el contenido sea comprensible, visible y ordenado. UNIR (2025) señala que esta gestión no implica trabajar de manera aislada, sino en diálogo constante con el equipo y las partes interesadas. El refinamiento del *product backlog* se convierte así en un espacio de colaboración y aprendizaje compartido.

Por su parte, el *sprint backlog* representa el subconjunto de elementos seleccionados para un *sprint* determinado. Reaburn (2025) indica que este artefacto permite al equipo visualizar su compromiso para el ciclo de trabajo y organizar las tareas necesarias para alcanzar el objetivo del *sprint*. A diferencia del

product backlog, el *sprint backlog* es gestionado directamente por el equipo de desarrollo.

Desde una perspectiva pedagógica, comprender la diferencia entre ambos *backlogs* resulta fundamental para entender cómo *Scrum* articula estrategia y ejecución. El *product backlog* orienta la dirección del producto, mientras que el *sprint backlog* traduce esa dirección en acciones concretas de corto plazo. Esta relación favorece la alineación entre valor esperado y trabajo realizado.

Definition of Done y calidad en *Scrum*

La noción de calidad ocupa un lugar central en la gestión ágil del trabajo. En *Scrum*, la calidad no se evalúa únicamente al final del proceso, sino que se integra de manera explícita en cada incremento del producto. En este marco, la *Definition of Done* se presenta como un acuerdo compartido que establece cuándo un trabajo puede considerarse terminado.

Atlassian (s. f.) define la *Definition of Done* como un conjunto de criterios claros y verificables que deben cumplirse para que un incremento sea aceptable. Estos criterios pueden incluir aspectos técnicos, funcionales y de documentación mínima necesaria. Su

propósito consiste en reducir ambigüedades y generar una comprensión común dentro del equipo.

Scrumio (s. f.) advierte que la ausencia de una *Definition of Done* explícita suele generar malentendidos respecto al estado real del trabajo. Un elemento marcado como “hecho” puede no estar listo para su uso si no se han acordado previamente los estándares de calidad. Por ello, la *Definition of Done* actúa como un mecanismo de transparencia.

Desde el punto de vista del trabajo colaborativo, la *Definition of Done* fortalece la responsabilidad compartida. Atlassian (s. f.) señala que este acuerdo no pertenece a una sola persona, sino al equipo en su conjunto. Todos los miembros son responsables de cumplirlo y de revisarlo cuando cambian las condiciones del entorno o del producto.

La *Definition of Done* también se vincula con la mejora continua. Scrumio (s. f.) explica que, a medida que el equipo madura, los criterios pueden volverse más exigentes, incorporando nuevos aprendizajes. De este modo, la calidad no

se mantiene estática, sino que evoluciona junto con el equipo y el producto.

Para ti, como estudiante, este concepto permite comprender que la calidad en *Scrum* no depende exclusivamente de controles externos, sino de acuerdos explícitos que guían el trabajo diario y facilitan la toma de decisiones colectivas.

Scrum y *Kanban*: diferencias, complementariedades y métricas de flujo

Scrum y *Kanban* son dos enfoques ágiles que comparten principios generales, pero difieren en su estructura y en la manera de gestionar el trabajo. Comprender sus diferencias y posibles combinaciones resulta relevante para analizar cómo las organizaciones adaptan las prácticas ágiles a distintos contextos.

Martínez Pueyo (2025) explica que *Scrum* se basa en iteraciones de duración fija, con roles y eventos definidos, mientras que *Kanban* propone un flujo continuo de trabajo sin iteraciones obligatorias. Esta diferencia impacta en la forma en que se planifica, ejecuta y revisa el trabajo.

Kanban se centra en la visualización del flujo y en la limitación del trabajo en curso. Miseviciute (2024) señala que esta visualización permite identificar cuellos de botella y mejorar la eficiencia del sistema. En lugar de comprometer trabajo para un período fijo, _Kanban_ prioriza la entrega continua.

Desde una perspectiva comparativa, *Scrum* ofrece una estructura más prescriptiva, lo que puede resultar útil para equipos que están comenzando a trabajar de manera ágil. *Kanban*, en cambio, ofrece mayor flexibilidad y suele adaptarse con facilidad a contextos donde existen procesos ya establecidos.

La combinación de ambos enfoques ha dado lugar a prácticas híbridas. Miseviciute (2024) describe el uso de *Kanban* dentro de equipos *Scrum* como una forma de mejorar la visualización del trabajo y las métricas de flujo, sin abandonar la estructura de *sprint*s. Esta integración permite aprovechar fortalezas de ambos enfoques.

Las métricas de flujo, como el tiempo de ciclo o el trabajo en progreso, aportan información relevante para la toma de decisiones. Martínez Pueyo (2025) indica que estas métricas ayudan a comprender cómo fluye el trabajo y a identificar oportunidades de mejora. En contextos *Scrum*, pueden complementar métricas tradicionales asociadas al *sprint*.

Para el estudiante, analizar *Scrum* y *Kanban* desde esta perspectiva permite comprender que los marcos ágiles no son excluyentes, sino adaptables. La elección y combinación de

prácticas depende del contexto, del tipo de trabajo y del nivel de madurez del equipo.

Contratos ágiles y colaboración con clientes

En los enfoques ágiles, la relación con el cliente se concibe de manera diferente a la lógica contractual tradicional. Mientras que los contratos clásicos suelen basarse en un alcance cerrado, definido exhaustivamente al inicio del proyecto, los contratos ágiles parten del reconocimiento de que el contexto, las necesidades y las soluciones evolucionan con el tiempo. En este sentido, los contratos ágiles buscan habilitar la adaptación continua y la colaboración sostenida entre las partes.

Desde una perspectiva ágil, el contrato deja de funcionar únicamente como un instrumento de control para convertirse en un marco de acuerdos flexibles. Este tipo de contratos prioriza la entrega progresiva de valor, la revisión periódica de prioridades y la incorporación sistemática de aprendizaje. En lugar de fijar de manera rígida todas las funcionalidades, se establecen mecanismos para redefinir el alcance en función de la información que emerge durante el desarrollo del producto.

La gestión del *product backlog* desempeña un rol central en esta lógica contractual. UNIR (2025) señala que el *product backlog* actúa como un artefacto dinámico que refleja las necesidades actuales del cliente y del negocio. En contextos de contratos ágiles, este artefacto se convierte en el principal punto de referencia para negociar prioridades, revisar avances y acordar próximos pasos. De este modo, la colaboración se materializa en decisiones concretas sobre qué construir y cuándo hacerlo.

Reaburn (2025) explica que la transparencia del trabajo facilita una relación más equilibrada entre el equipo y el cliente. Al contar con visibilidad sobre el estado del producto, las funcionalidades desarrolladas y el trabajo pendiente, las conversaciones se basan en evidencia y no en supuestos. Esta transparencia reduce conflictos asociados a expectativas no alineadas y favorece acuerdos informados.

La colaboración con clientes en *Scrum* también se apoya en la entrega frecuente de incrementos. Cada incremento permite validar hipótesis, obtener retroalimentación y ajustar el rumbo del producto. Desde esta perspectiva, el contrato ágil no garantiza resultados específicos desde el inicio, sino la existencia de un proceso que permite aprender y adaptarse de manera controlada. Esta lógica resulta especialmente relevante en entornos donde la incertidumbre es elevada.

Asimismo, los contratos ágiles requieren un cambio en los roles tradicionales. El cliente deja de ocupar un lugar exclusivamente evaluador al final del proceso y se involucra de manera continua en la toma de decisiones. Esta participación activa implica asumir responsabilidades compartidas respecto a la priorización del trabajo y a la definición de valor. UNIR (2025) destaca que esta interacción constante fortalece la alineación entre expectativas y resultados.

No obstante, la colaboración efectiva no se produce de manera automática. Reaburn (2025) advierte que resulta necesario establecer acuerdos claros sobre formas de trabajo, criterios de aceptación y canales de comunicación. En este punto, la *Definition of Done* cumple una función relevante, ya que permite acordar qué significa que un incremento esté terminado y sea aceptable para ambas partes. Estos acuerdos contribuyen a sostener la confianza y a reducir ambigüedades.

Desde el punto de vista de la gestión de valor, los contratos ágiles favorecen un enfoque incremental. En lugar de comprometer grandes entregas a largo plazo, se acuerdan ciclos cortos de trabajo que permiten evaluar resultados y redefinir prioridades. Esta modalidad reduce el riesgo de invertir recursos en funcionalidades que no generan valor y facilita la adaptación a cambios del entorno.

Para ti, como estudiante, comprender la lógica de los contratos ágiles implica reconocer que la colaboración con clientes no es un complemento del marco *Scrum*, sino una condición para su funcionamiento. Gestionar valor en equipos ágiles requiere acuerdos flexibles, transparencia y disposición al aprendizaje conjunto. Los contratos ágiles ofrecen un marco que habilita estas prácticas, alineando la relación contractual con los principios de adaptación y mejora continua.

CONTINUAR

Referencias

Altassian. (s. f.). *What is the Definition of Done?.* https://www.atlassian.com/agile/project-management/definition-of-done

Drumond, C. (s. f.). *¿Qué es scrum? Desglose del marco de trabajo ágil*. Atlassian. https://www.atlassian.com/es/agile/scrum

Luzhko, Y. (2025). *¿Qué son los Sprints y las Iteraciones en Scrum?*. PayPro Global. https://payproglobal.com/es/respuestas/que-son-los-sprints-y-las-iteraciones-en-scrum/

Martínez Pueyo, J. (2025). *¿Qué diferencias hay entre Scrum y Kanban?.* IEBS. https://www.iebschool.com/hub/scrum-vs-kanban-diferencias-y-recomendaciones-de-uso/

Martins, J. (2025). *Scrum: conceptos clave y cómo se aplica en la gestión de proyectos.* Asana. https://asana.com/es/resources/what-is-scrum

Miseviciute. D. (2024). *Kanban, Scrum y Scrumban – Cómo se diferencian*. Teamhood. https://teamhood.com/es/kanban-4/kanban-scrum-y-scrumban-como-se-diferencian/

Owen, G. (2024). *The 5 Scrum events explained*. TechTarget. https://www.techtarget.com/searchsoftwarequality/tip/The-scrum-events-explained

Reaburn. A. (2025). *Qué es product backlog y guía para hacer uno con ejemplo*. Assana. https://asana.com/es/resources/product-backlog

Scrumio. (s. f.). *La Definición de Hecho (Definition of Done).* https://www.scrumio.com/scrum/definicion-de-hecho

UNIR. (2025). *El product backlog: qué es y por qué es clave en la gestión de proyectos.* https://www.unir.net/revista/empresa/product-backlog/

CONTINUAR