DHARMA CONSULTING
Soluciones de Negocios para la Gestión de Proyectos

SCRUMstudy Authorized Training Partner

Project Management Institute R.E.P.

CV-GAP005

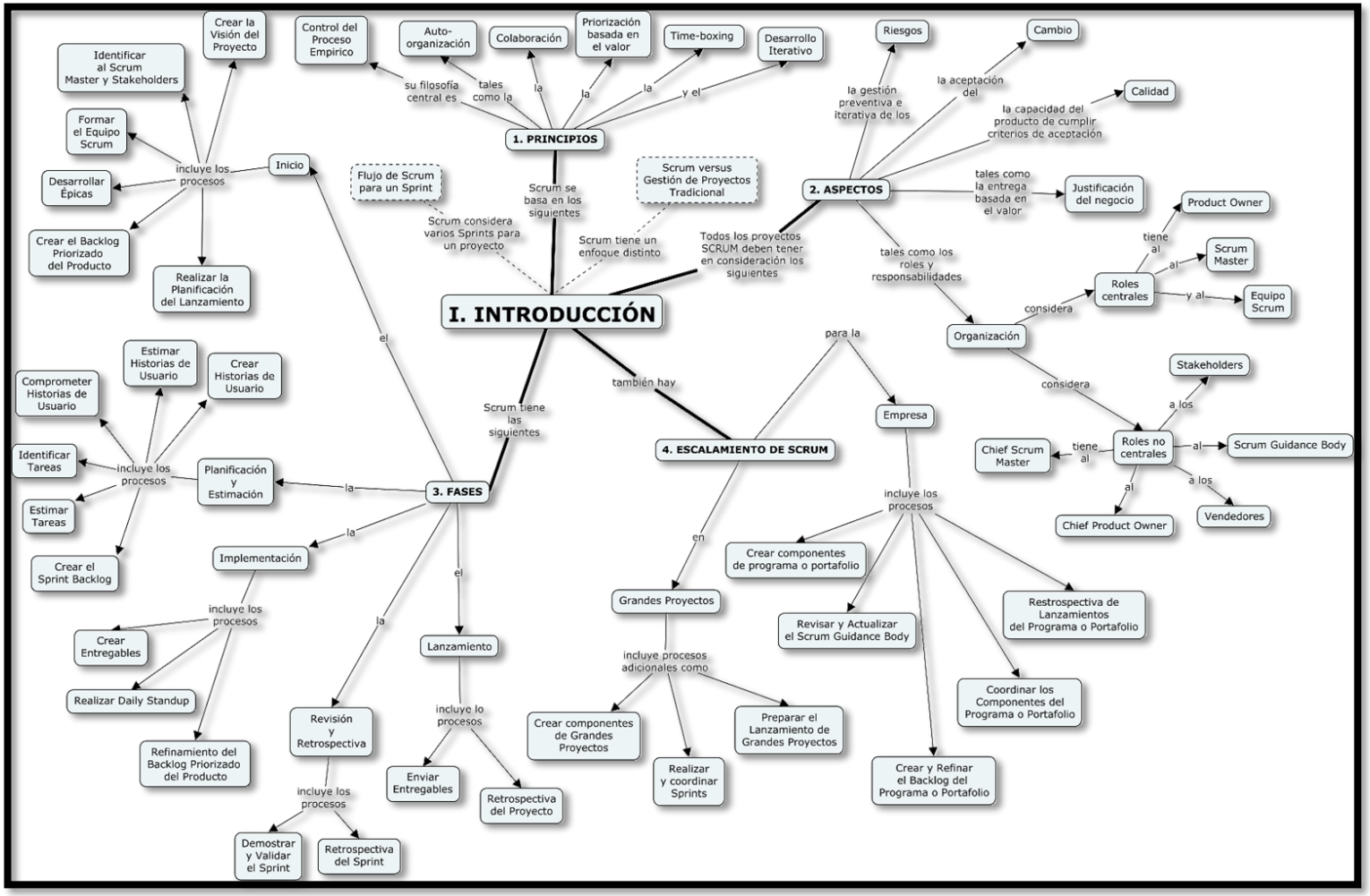
SCRUM PARA LA GESTIÓN ÁGIL DE PROYECTOS (ACORDE A LA GUÍA SBOK™ 3RA EDICIÓN)

MATERIAL DE LECTURA DE LA UNIDAD 1

Contenido:

- Mapa Mental del Capítulo 1: "Introducción" de la *Guía SBOK*™ elaborado por Dharma Consulting.
- Apéndice A: Descripción General de Ágil, extraído de la *Guía SBOK*™ de SCRUMstudy.
- Capítulo 1: "Introducción", extraído de la *Guía SBOK*™ de SCRUMstudy. *© 2017 SCRUMstudy™, una marca de VMEdu, Inc. Todos los derechos reservados*.

MAPA MENTAL: "INTRODUCCIÓN"



FUENTE: *A Guide to the Scrum Body of Knowledge* (SBOK™ GUIDE) – 3rd Edition

APÉNDICE A: DESCRIPCIÓN GENERAL DE ÁGIL

A.1 Introducción

El propósito de este apéndice es familiarizar al lector con el concepto de desarrollo ágil y las diversas metodologías ágiles.

En las siguientes secciones se incluyen:

A.2 Descripción general—Esta sección analiza la definición y los factores detrás del surgimiento de Ágil.

A.3 Manifiesto Ágil—En esta sección se presenta el *Manifiesto Ágil*, sus principios, y la *Declaración de Interdependencia* para proporcionar el contexto histórico de Ágil.

A.4 Métodos Ágil—Esta sección ofrece una breve descripción de las metodologías ágiles específicas tales como:

- **Lean Kanban**
- **Extreme Programming (XP)** – *Programación Extrema*
- **Crystal Methods** – *Métodos Crystal*
- **Dynamic Systems Development Methods (DSDM)** – *Métodos de desarrollo de sistemas dinámicos*
- **Feature Driven Development (FDD)** – *Desarrollo basado en funcionalidades*
- **Test Driven Development (TDD)** – *Desarrollo guiado por pruebas*
- **Adaptive Software Development (ASD)** – *Desarrollo adaptativo de software*
- **Agile Unified Process (AUP)** – *Proceso Unificado Ágil*
- **Domain Driven Design (DDD)** – *Diseño guiado por el dominio*

A.2 Descripción general

La palabra "ágil" generalmente hace referencia a la capacidad de moverse o responder con rapidez y facilidad; ser ágil. En cualquier tipo de disciplina de administración, la agilidad es una calidad, y, por lo tanto, es algo bueno que se debe buscar. Específicamente, la gestión ágil de proyectos implica ser adaptativo durante la creación de un producto, servicio u otro resultado.

Es importante entender que, aunque el desarrollo de métodos ágiles es altamente adaptativo, también es necesario considerar la estabilidad en su proceso de adaptación.

A.2.1 El surgimiento de Ágil

Los rápidos cambios en la tecnología, las demandas y expectativas del mercado han creado mayores desafíos en el desarrollo de productos y servicios mediante el uso de modelos tradicionales de gestión de proyectos. Esto abrió el camino para la conceptualización e implementación de métodos y valores ágiles en muchas organizaciones. Los modelos de desarrollo ágil atienden las deficiencias asociadas con los modelos tradicionales de gestión de proyectos para satisfacer las crecientes demandas ambientales y expectativas que enfrentan las organizaciones. Dado a que los modelos tradicionales de gestión de proyectos en general hacen énfasis en una amplia planificación anticipada y se ajustan al plan una vez que se establece, tales modelos no tuvieron éxito al intentar cumplir la realidad de los frecuentes cambios ambientales.

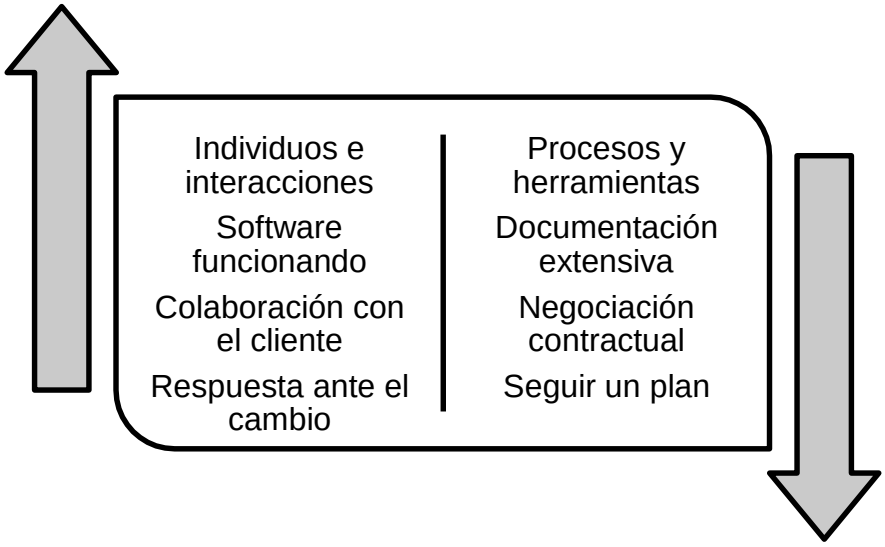
Ágil depende de la planificación adaptiva y del desarrollo y la entrega iterativa. Se enfoca principalmente en el valor de las personas al hacer eficazmente el trabajo. Aunque las metodologías adaptativas e incrementales han existido desde los años cincuenta, únicamente las metodologías que conforman el *Manifiesto Ágil* generalmente se consideran verdaderamente "ágiles".

A.3 El Manifiesto Ágil

En febrero del 2001, un grupo de 17 gurús de la informática, desarrolladores de software y administradores, se reunieron para discutir los métodos de desarrollo de software ligero. Formaron la Alianza Ágil (*Agile Alliance*) y lo debates de estas reuniones resultaron después en un *Manifiesto para el desarrollo ágil de software* (*Manifesto for Agile Software Development*). El manifiesto fue escrito por Flowler y Highsmith (2001) y suscrito después por todos los participantes a fin de establecer los lineamientos básicos de cualquier método ágil.

El propósito del Manifiesto Ágil se plasmó de la siguiente forma:

Estamos descubriendo formas mejores de desarrollar
software tanto por nuestra propia experiencia como ayudando a terceros.
A través de este trabajo hemos aprendido a valorar:



Esto es, aunque valoramos los elementos de la derecha,
valoramos más los de la izquierda.

Kent Beck	James Grenning	Robert C. Martin
Mike Beedle	Jim Highsmith	Steve Mellor
Arie van Bennekum	Andrew Hunt	Ken Schwaber
Alistair Cockburn	Ron Jeffries	Jeff Sutherland
Ward Cunningham	Jon Kern	Dave Thomas
Martin Fowler	Brian Marick	

Las cuatro contrapartes del Manifiesto Ágil se elaboran de la siguiente forma:

1. **Individuos e interacciones sobre procesos y herramientas**
   Aunque los procesos y las herramientas ayudan a terminar con éxito un proyecto, son las personas quienes asumen, participan e implementan un proyecto y determinan cuáles procesos y herramientas utilizar. Por lo tanto, los actores clave en cualquier proyecto son las personas, por ello, el énfasis debe estar en ellos y en sus interacciones en vez de los complicados procesos y herramientas.

2. **Software funcionando sobre documentación extensiva**
   Aunque la documentación es necesaria y útil para cualquier proyecto, muchos equipos se centran en la recopilación y el registro de descripciones cualitativas y cuantitativas de los entregables, cuando el valor real que se le entrega al cliente es en forma de un software funcional. Por lo tanto, en vez de la documentación detallada, el enfoque ágil está en la entrega de un software de buen funcionamiento en incrementos a lo largo del ciclo de vida del producto.

3. **Colaboración con el cliente sobre negociación contractual**
   Tradicionalmente a los clientes se les ha visto como participantes externos, involucrados principalmente al inicio y al final del ciclo de vida del producto y cuya relación se basaba en contractos y en su cumplimiento. Ágil cree en un enfoque de valor compartido en el cual los clientes se consideran colaboradores. El equipo de desarrollo y el cliente trabajan unidos para evolucionar y desarrollar el producto.

4. **Responder ante el cambio sobre seguir un plan**
   En el mercado actual —donde los requerimientos del cliente, las tecnologías disponibles y los patrones empresariales cambian constantemente—, es fundamental abordar el desarrollo de productos en una forma adaptativa que permita la incorporación de cambio y rápidos ciclos de vida de desarrollo de producto en vez de enfatizar el seguimiento de planes formados probablemente con información obsoleta.

A.3.1 Principios de Manifiesto Ágil

Los 12 principios del Manifiesto Ágil de Fowler y Highsmith (2001) son los siguientes:

1. Nuestra mayor prioridad es satisfacer al cliente mediante la entrega temprana y continua de software valioso.

2. Aceptamos que los requisitos cambien, incluso en etapas tardías del desarrollo. Los procesos Ágiles aprovechan el cambio para proporcionar ventaja competitiva al cliente.

3. Entregamos software funcional frecuentemente, entre dos semanas y dos meses, de preferencia al menor tiempo posible.

4. Los responsables de negocio y los desarrolladores deben trabajar juntos de forma cotidiana durante todo el proyecto.

5. Los proyectos se desarrollan en torno a individuos motivados. Hay que darles el entorno y el apoyo que necesitan, y confiarles la ejecución del trabajo.

6. El método más eficiente y efectivo de comunicar información al equipo de desarrollo y entre sus miembros es la conversación cara a cara.

7. El software funcionando es la medida principal de progreso.

8. Los procesos ágiles promueven el desarrollo sostenible. Los patrocinadores, desarrolladores y usuarios deben ser capaces de mantener un ritmo constante de forma indefinida.

9. La atención continua a la excelencia técnica y al buen diseño mejora la agilidad.

10. La simplicidad, o el arte de maximizar la cantidad de trabajo no realizado, es esencial.

11. Las mejores arquitecturas, requisitos y diseños emergen de equipos auto-organizados.

12. A intervalos regulares el equipo reflexiona sobre cómo ser más eficaz para después ajustar y perfeccionar su comportamiento según corresponda.

A.3.2 Declaración de interdependencia

La *Declaración de Interdependencia* de gestión de proyectos ágiles fue redactada a principios del 2005 por un grupo de 15 líderes de proyecto como suplemento del Manifiesto Ágil. Enumera seis valores de gestión necesarios para reforzar la mentalidad del desarrollo ágil, particularmente en la gestión de proyectos complejos o inciertos.

La declaración destaca que los equipos del proyecto, los clientes y demás stakeholders son interdependientes y están conectados, lo cual se debe reconocer a fin de lograr el éxito. Los valores por si mismos son también interdependientes.

> *Nosotros...*
>
> ***Aumentamos el retorno sobre la inversión*** *enfocándonos en la aportación continua de valor.*
> ***Entregamos resultados fiables*** *mediante la participación de clientes en las interacciones frecuentes, así como la propiedad compartida.*
> ***Esperamos la incertidumbre*** *y la gestionamos a través de iteraciones, con anticipación y adaptación.*
> ***Damos rienda suelta a la creatividad y la innovación*** *reconociendo que las personas son la fuente última de valor, y creando un ambiente donde pueden ser la diferencia.*
> ***Aumentamos el rendimiento*** *mediante la orientación del grupo a los resultados y la responsabilidad compartida para la efectividad del equipo.*
> ***Mejoramos la eficacia y fiabilidad*** *a través de estrategias, procesos y prácticas específicas.*
>
> Anderson. D., Augustine, S., Avery, C., Cockburn, A., Cohn, M., *et al.* 2005

A.4 Métodos Ágiles

En la década de los noventa y a principios del 2000, se originó y obtuvo fuerza una serie de metodologías ágiles. Aunque difieren en una variedad de aspectos, lo que tienen en común se deriva de su apego al *Manifiesto Ágil*.
A continuación, se describen brevemente los siguientes métodos ágiles:

- **Lean Kanban**
- **Extreme Programming (XP)** – *Programación Extrema*
- **Crystal Methods** – *Métodos Crystal*
- **Dynamic Systems Development Methods (DSDM)** – *Métodos de desarrollo de sistemas dinámicos*
- **Feature Driven Development (FDD)** – *Desarrollo basado en funcionalidades*
- **Test Driven Development (TDD)** – *Desarrollo guiado por pruebas*
- **Adaptive Software Development (ASD)** – *Desarrollo adaptativo de software*
- **Agile Unified Process (AUP)** – *Proceso Unificado Ágil*
- **Domain Driven Development (DDD)** – *Desarrollo guiado por el dominio*

A.4.1 Lean Kanban

El concepto de Lean optimiza el sistema de una organización a fin de producir resultados valiosos con base en sus recursos, necesidades y alternativas, a la vez que se reducen las perdidas. Las pérdidas (*waste*) pudieran ser por la fabricación de algo equivocado, la imposibilidad de aprender, o prácticas que impidan el proceso. Debido a que estos factores tienen una naturaleza dinámica, una organización Lean evalúa la totalidad de su sistema y refina constantemente sus procesos. El fundamento de Lean es la reducción de la duración de cada ciclo (cada interacción), lo cual lleva a un aumento en la productividad, reduciendo retrasos y ayudando a detectar errores en las primeras etapas, disminuyendo en consecuencia la cantidad total del trabajo necesario para finalizar una tarea. Los principios del software Lean se han implementado con éxito en el desarrollo de software.
Kanban literalmente significa "cartel" o "letrero", e implica el uso de ayuda visual para dar seguimiento a la producción. El concepto fue introducido por Taiichi Ohno, considerado como el padre de los Sistemas de Producción Toyota (TPS, por sus siglas en inglés). El uso de ayuda visual es eficaz y se ha convertido en una práctica común. Algunos ejemplos incluyen: tarjetas de tarea, Scrumboards y Burndown Charts. Dichos métodos generaron atención debido a su práctica en Toyota, empresa líder en gestión de procesos. Lean Kanban integra el uso de métodos de visualización según lo prescrito por Kanban aunado a los principios de Lean, creando así un sistema visual de gestión de proceso evolutivo incremental.

A.4.2 Extreme Programming *(Programación Extrema)*

El Extreme Programming (XP), o *Programación Extrema*, creada en la Chrysler Corporation, obtuvo impulso en la década de 1990. La Programación Extrema, o XP, por sus siglas en inglés, evita el aumento radical del costo de software cambiante con el paso del tiempo. Las características claves del XP incluyen el desarrollo incremental, horarios flexibles, códigos de prueba automatizados, comunicación verbal, diseño en evolución constante y la vinculación a corto y largo plazo de todos los involucrados.

La Programación Extrema valora la comunicación, la retroalimentación, la simplicidad y el valor. Los distintos roles en el enfoque de XP incluyen: el cliente, el desarrollador, el tracker y el coach. Prescribe varias prácticas de codificación, de desarrollo y de negocio, así como eventos y artefactos para lograr un desarrollo eficaz y eficiente. La Programación Extrema ha sido adoptada extensamente debido a sus prácticas de ingeniería bien definidas.

A.4.3 Crystal Methods *(Métodos Crystal)*

Las metodologías Crystal para el desarrollo de software fueron introducidas por Alistair Cockburn a principios de la década de 1990. La intención de los métodos Crystal es centrarse en las personas; ser ligeros y fáciles de adaptar. Debido a que las personas son primordiales, el proceso de desarrollo y las herramientas no son fijas, sino que se ajustan a los requerimientos y características específicas del proyecto. Se utiliza el espectro de colores para decidir sobre la variante de un proyecto. Los factores tales como la comodidad, el dinero a discreción, dinero esencial y la vida, juegan un papel importante para determinar el "peso" de la metodología, lo cual se representa en varios colores del espectro. La familia Crystal se divide en: *Crystal Clear* (claro como el cristal), *Crystal Yellow* (cristal amarillo), *Crystal Orange* (cristal naranja), *Crystal Orange Web* (cristal naranja web), *Crystal Red* (cristal rojo), *Crystal Maroon* (cristal marrón), *Crystal Diamond* (cristal diamante) y *Crystal Sapphire* (cristal zafiro).

Todos los métodos Crystal tienen cuatro roles: patrocinador ejecutivo (*executive sponsor*), diseñador líder (*lead designer*), desarrolladores y usuarios experimentados. Los métodos Crystal recomiendan varias estrategias y técnicas para lograr agilidad. Un ciclo de proyecto Crystal consiste de la implementación del acta constitutiva (*chartering*), ciclo de entrega y cierre (*wrap-up*).

A.4.4 Dynamic Systems Development Methods, DSMS *(Métodos de desarrollo de sistemas dinámicos)*

El framework del sistema de desarrollo de sistemas dinámicos (DSMS, por sus siglas en inglés) fue publicado inicialmente en 1995 y lo administra el Consorcio DSMS. El DSMS fija la calidad y el esfuerzo en términos de costo y tiempo al principio y ajusta los entregables del proyecto para cumplir con los criterios fijos mediante la priorización de los entregables en las categorías: "Debe tener" (*Must have*), "Debería tener" (*Should have*), "Podría tener" (*Could have*) y "No tendrá" (*Won't have*), con el uso de la técnica de priorización MoSCoW. El DSMS es un método orientado en sistemas con seis fases distintas: pre-proyecto; viabilidad; fundamentos; exploración e ingeniería; desplazamiento y evaluación de beneficios.

A.4.5 Feature Driven Development, FDD *(Desarrollo basado en funcionalidades)*

El desarrollo basado en funcionalidades (FDD, por sus siglas en inglés) fue introducido por Jeff De Luca en 1997 y opera bajo el principio de concluir un proyecto mediante su fragmentación en pequeñas funciones valoradas por el cliente que puedan presentarse en menos de dos semanas. El FDD tiene dos principios fundamentales: el desarrollo de software es una actividad humana y es una

funcionalidad valorada por el cliente.

El FDD define seis roles principales: Project Manager, Chief Architect, Developer Manager, Chief Programmers, Class Owners y Domain Experts, aunados a una serie de funciones complementarias. El proceso FDD es iterativo y consiste en el desarrollo de un modelo general; en crear una lista de características y después planificar, diseñar y crear con base en la característica.

A.4.6 Test Driven Development, TDD *(Desarrollo guiado por pruebas)*

Conocido también como "desarrollo primero por pruebas" (del inglés: *Test-First Development*), el desarrollo guiado por pruebas fue introducido por Kent Beck, uno de los creadores de la Programación Extrema. El desarrollo guiado por pruebas es un método de desarrollo de software que implica redactar primero códigos de prueba automáticos y desarrollar la cantidad mínima de código necesario para avanzar después hacia la siguiente prueba. Todo el proyecto se divide en pequeñas características valoradas por el cliente que deben desarrollarse en el ciclo de desarrollo más breve posible. Las pruebas se redactan con base en los requerimientos y especificaciones del cliente. Las pruebas diseñadas en la etapa anterior se utilizan para diseñar y redactar el código de producción.

El desarrollo guiado por pruebas (TDD, por sus siglas en inglés) se puede clasificar en dos niveles: *Acceptance TDD* (ATDD), lo cual requiere de una prueba distinta de aceptación, y *Developer TDD* (DTDD) que implica la redacción de una sola prueba de desarrollo. El TDD se ha popularizado debido a las numerosas ventajas que ofrece como los resultados confiables, la retroalimentación constante y la reducción de tiempo de depuración (*debugging*).

A.4.7 Adaptive Software Development, ASD *(Desarrollo adaptativo de software)*

El desarrollo adaptativo de software (ASD, por sus siglas en inglés) surgió a partir del rápido trabajo de desarrollo de aplicaciones por parte de Jim Highsmith y Sam Bayer. Los aspectos más destacados del ASD son la constante adaptación de procesos al trabajo con el que se cuenta, el suministro de soluciones a los problemas que surgen en los grandes proyectos, así como el desarrollo iterativo e incremental con prototipos continuos.

Al ser un método de desarrollo impulsado por el riesgo y tolerante al cambio, el ASD indica que un plan no puede aceptar incertidumbres y riesgos, ya que esto sería indicativo de un plan deficiente y fallido. El desarrollo adaptativo de software se basa en características y se guía por metas. La primera fase en este tipo de desarrollo es la especulación (a diferencia de la planificación), seguida de las fases de colaboración y aprendizaje.

A.4.8 Agile Unified Process, AUP *(Proceso Unificado Ágil)*

El Proceso Unificado Ágil (AUP, por sus siglas en inglés) evolucionó del Proceso Unificado Racional de IBM (del inglés: *IBM's Rational Unified Process*). El Proceso Unificado Ágil, desarrollado por Scott Ambler, combina las técnicas ágiles probadas y examinadas por la industria, tales como el desarrollo guiado por pruebas (TDD),

la modelación ágil, la gestión ágil de cambios y la refactorización de base de datos, a fin de brindar un producto funcional de la mejor calidad.

El Proceso Unificado Ágil modela sus procesos y técnicas con base en los valores de las herramientas de simplicidad, agilidad, personalización, auto-organización e independencia y se enfoca en actividades de alto valor. Los principios y valores del Proceso Unificado Ágil se ponen en acción en las fases de: Incepción (inicio), Elaboración, Construcción y Transición.

A.4.9 Domain-Driven Design, DDD *(Diseño guiado por el dominio)*

El diseño guiado por el dominio (DDD, por sus siglas en inglés) es un método de desarrollo ágil diseñado para manejar diseños complejos con la implementación vinculada a un modelo evolutivo. Fue conceptualizado en el 2004 por Eric Evans y gira en torno al diseño de un dominio central. La palabra "dominio" se define como un área de actividad en la cual el usuario aplica un programa o funcionalidad. Muchas de estas áreas se procesan en lotes y se diseña un modelo. El modelo consiste en un sistema de abstracciones que se pueden utilizan para diseñar un proyecto general y resolver los problemas relacionadas a los dominios en lote. Los valores centrales del DDD incluyen: el diseño orientado en el dominio, diseño guiado por el modelo, el lenguaje ubicuo y el contexto limitado.

En el DDD se establece un lenguaje ubicuo y se modela el dominio. Después sigue el diseño, el desarrollo y la evaluación. La refinación y refactorización del modelo del dominio se lleva a cabo hasta que sea satisfactorio.

CAPÍTULO 1: INTRODUCCIÓN

La *Guía para el Cuerpo de Conocimiento de Scrum (Guía SBOK™)* proporciona directrices para la aplicación con éxito de Scrum: el desarrollo ágil de productos y el método de entrega de proyectos más popular. Brinda un framework integral que incluye los principios, aspectos y procesos de Scrum.

Scrum, tal como se define en la *Guía SBOK™,* aplica a los siguientes:

- Portafolios, programas y/o proyectos en cualquier industria.
- Productos, servicios o cualquier otro resultado que se les entregarán a los interesados.
- Proyectos de cualquier tamaño y complejidad.

El término "producto" en la Guía SBOK™ puede referirse a un producto, servicio, o cualquier otro entregable. Scrum puede aplicarse en forma efectiva a cualquier proyecto en cualquier industria, desde proyectos pequeños o equipos con tan sólo seis miembros, hasta proyectos grandes y complejos con varios cientos de integrantes.

Este primer capítulo describe la finalidad y el framework de la *Guía SBOK™* y proporciona una introducción a los conceptos claves de Scrum. Contiene un resumen de los principios de Scrum, al igual que los aspectos y los procesos sobre el tema. El capítulo 2 amplía la información sobre los seis principios de Scrum, que son la base del mismo. Los capítulos del 3 al 7 tratan en detalle los cinco aspectos de Scrum que se deben abordar en cualquier proyecto: organización, justificación del negocio, la calidad, el cambio y el riesgo. Los capítulos del 8 al 12 cubren los 19 procesos de Scrum que forman parte de la creación de un proyecto Scrum. Estos procesos forman parte de las cinco fases de Scrum: Inicio; Planificación y estimación; Implementación, Revisión y retrospectiva y Lanzamiento. Estas fases describen a detalle las entradas y salidas asociadas con cada proceso, así como las diferentes herramientas que pueden utilizarse en cada una. Algunas entradas, herramientas y salidas son obligatorias y estas se indican como tales; otras son opcionales dependiendo del proyecto específico, de los requisitos de la organización y/o lineamientos establecidos por el Scrum Guidance Body de la organización (SGB). Los capítulos 13 y 14 son nuevas adiciones a la *Guía SBOK™*, mismas que brindan las directrices para escalar en Scrum en grandes proyectos y escalar Scrum para la empresa.

Este capítulo está dividido en las siguientes secciones:

1.1 Información general de Scrum

1.2 ¿Por qué utilizar Scrum?

1.3 Propósito de la *Guía SBOK™*

1.4 Framework de la *Guía SBOK™*

1.5 Scrum vs. Gestión tradicional de proyectos

1.1 Información general de Scrum

Un proyecto Scrum consiste en un esfuerzo de colaboración para crear un nuevo producto, servicio u otro resultado tal como se define en la Declaración de la visión del proyecto (*Project Vision Statement*). Los proyectos se ven afectados por limitaciones de tiempo, costos, alcance, calidad, recursos, capacidades organizacionales y demás limitaciones que dificultan su planificación, ejecución, administración y, por último, su éxito. Sin embargo, la implementación exitosa de los resultados de un proyecto terminado le proporciona ventajas económicas considerables a una organización. Por lo tanto, es importante que las organizaciones seleccionen e implementen un método adecuado de gestión de proyectos.

Scrum es uno de los métodos ágiles más populares. Es un framework adaptable, iterativo, rápido, flexible y eficaz, diseñado para ofrecer un valor considerable en forma rápida a lo largo del proyecto. Scrum garantiza transparencia en la comunicación y crea un ambiente de responsabilidad colectiva y de progreso continuo. El framework de Scrum, tal como se define en la *Guía SBOK™,* está estructurado de tal manera que es compatible con el desarrollo de productos y servicios en todo tipo de industrias y en cualquier tipo de proyecto, independientemente de su complejidad.

Una fortaleza clave de Scrum radica en el uso de equipos interfuncionales (*cross-functional*), auto- organizados y empoderados que dividen su trabajo en ciclos de trabajo cortos y concentrados llamados *Sprints.* La figura 1-1 proporciona una visión general de flujo de un proyecto Scrum.

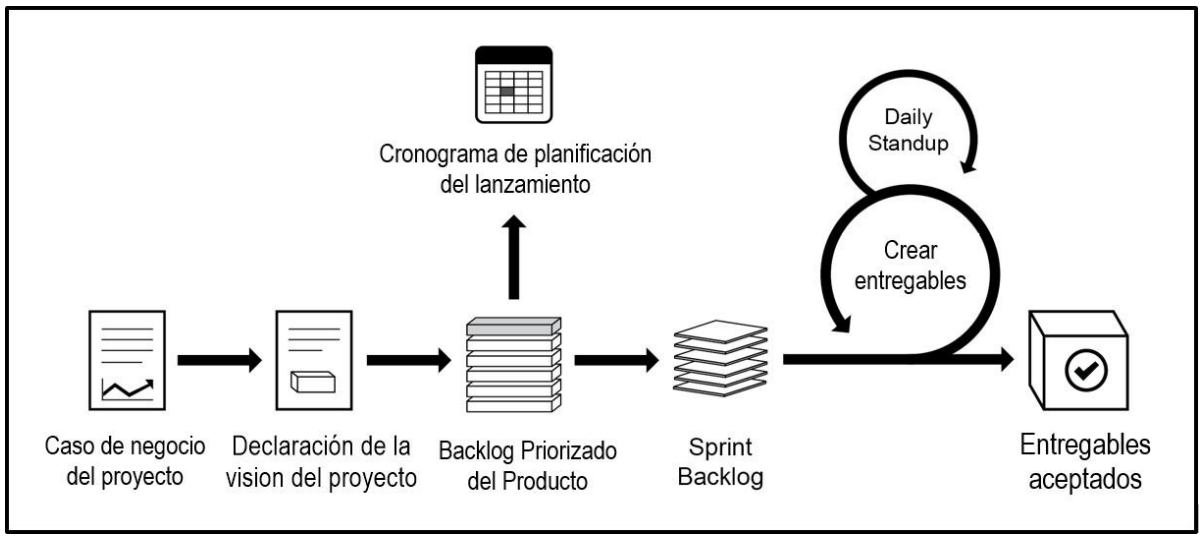


Figura 1-1: Flujo de Scrum para un sprint

El ciclo de Scrum empieza con una reunión de stakeholders, durante la cual se crea la visión del proyecto. Después, el Product Owner desarrolla una Backlog Priorizado del Producto (*Prioritized Product Backlog)* que contiene una lista requerimientos del negocio y del proyecto por orden de importancia en forma de una historia de usuario. Cada sprint empieza con una reunión de planificación del sprint (*Sprint Planning Meeting*) durante la cual se consideran las historias de usuario de alta prioridad para su inclusión en el sprint. Un sprint generalmente tiene una duración de una a seis semanas durante las cuales el Equipo Scrum trabaja en la creación de entregables

(del inglés *deliverables*) en incrementos del producto. Durante el sprint, se llevan cabo Daily Standups muy breves y concretos, donde los miembros del equipo discuten el progreso diario. Haca el final del sprint, se lleva a cabo una Reunión de Revisión del Sprint (*Sprint Review Meeting*) en la cual se proporciona una demostración de los entregables al Product Owner y a los stakeholders relevantes. El Product Owner acepta los entregables sólo si cumplen con los criterios de aceptación predefinidos. El ciclo del sprint termina con una Reunión de Retrospectiva del Sprint (*Retrospect Sprint Meeting*), donde el equipo analiza las formas de mejorar los procesos y el rendimiento a medida que avanzan al siguiente sprint.

1.1.1 Breve historia de Scrum

A mediados de la década de los 80s, Hirotaka Takeuchi y Ikujiro Nonaka definieron una estrategia de desarrollo de producto flexible e incluyente donde el equipo de desarrollo trabaja en unidad para alcanzar un objetivo común. Describieron un método innovador para el desarrollo de productos al que llamaron enfoque holístico o "rugby", "donde un equipo intenta llegar hasta el final como una unidad, pasando el balón hacia atrás y adelante". Basaron su enfoque en los estudios de casos de diversas industrias de fabricación. Takeuchi y Nonaka propusieron que el desarrollo de productos no debe ser como una carrera de relevos secuencial, sino que debería ser análogo al del juego de rugby, donde el equipo trabaja en conjunto, pasando el balón hacia atrás y hacia adelante a medida que se desplaza en unidad por el campo. El concepto de rugby de un "Scrum" (donde un grupo de jugadores se junta para reiniciar el juego) se introdujo en este artículo para describir la propuesta de los autores de que el desarrollo de productos debe implicar "mover al Scrum campo abajo".

Ken Schwaber y Jeff Sutherland desarrollaron el concepto de Scrum y su aplicabilidad al desarrollo de software durante una presentación en la Conferencia internacional sobre programación, lenguajes y aplicaciones orientadas a objetos (*Object-Oriented Programming, Systems, Languages & Applications,* o OOPSLA) en 1995 en Austin, Texas. Desde entonces, varios practicantes, expertos y autores de Scrum han seguido perfeccionando la conceptualización y framework de Scrum. En los últimos años, Scrum ha aumentado en popularidad, y es hoy en día el método de desarrollo de proyectos predilecto de muchas organizaciones a nivel mundial.

1.2 ¿Por qué utilizar Scrum?

Algunas de las ventajas principales del uso de Scrum en cualquier proyecto son:

1. **Adaptabilidad**—El control del proceso empírico y el desarrollo iterativo hacen que los proyectos sean adaptables y abiertos a la incorporación del cambio.

2. **Transparencia**—Todos los radiadores de información tales como un Scrumboard y el Sprint Burndown Chart se comparten, lo cual conduce a un ambiente de trabajo abierto.

3. **Retroalimentación continua**—La retroalimentación continua se proporciona a través de los procesos de *Realizar Daily Standup* y *Demostrar y validar el sprint*.

4. **Mejora continua**—Los entregables se mejoran progresivamente sprint por sprint a través del proceso de *Refinar el Backlog Priorizado del Producto.*

5. **Entrega continúa de valor**—Los procesos iterativos permiten la entrega continua de valor tan frecuentemente como el cliente lo requiere a través del proceso de *Envío de entregables*.

6. **Ritmo sostenible**—Los procesos Scrum están diseñados de tal manera que las personas involucradas pueden trabajar a un ritmo sostenible que, en teoría, puede continuar indefinidamente.

7. **Entrega anticipada de alto valor**—El proceso de *Crear el Backlog Priorizado del Producto* asegura que los requisitos de mayor valor del cliente sean los primeros en cumplirse.

8. **Proceso de desarrollo eficiente**—El *Time-boxing* y la reducción al mínimo del trabajo que no es esencial conducen a mayores niveles de eficiencia.

9. **Motivación**—Los procesos de *Realizar Daily Standup* y *Retrospectiva del sprint* conducen a mayores niveles de motivación entre los empleados.

10. **Resolución de problemas de forma más rápida**—La colaboración y co-ubicación de equipos interfuncionales conducen a la resolución de problemas con mayor rapidez.

11. **Entregables efectivos**—El proceso de *Crear el Backlog Priorizado del Producto*, y las revisiones periódicas después de la creación de entregables aseguran entregas eficientes al cliente.

12. **Centrado en el cliente**—El poner énfasis en el valor del negocio y tener un enfoque de colaboración con los stakeholders asegura un framework orientado al cliente.

13. **Ambiente de alta confianza**—Los procesos de *Realizar Daily Standup* y la *Retrospectiva del Sprint* promueven la transparencia y colaboración, dando lugar a un ambiente de trabajo de alta confianza que garantiza una baja fricción entre los empleados.

14. **Responsabilidad colectiva**—El proceso de *Comprometer Historias de Usuarios* permite que los miembros del equipo hagan suyo el proyecto y su trabajo lleve a una mejor calidad.

15. **Alta velocidad**—Un framework de colaboración permite a los equipos interfuncionales altamente cualificados alcanzar su potencial y una alta velocidad.

16. **Ambiente innovador**—Los procesos de *Retrospectiva de Sprint* y *Retrospectiva del Proyecto* crean un ambiente de introspección, aprendizaje y capacidad de adaptación que conllevan a un ambiente de trabajo innovador y creativo.

1.2.1 Escalabilidad de Scrum

Para ser eficaces, los equipos Scrum idealmente deben tener de seis a diez miembros. Esta práctica pudiera ser la razón de la idea errónea de que el framework de Scrum sólo puede utilizarse para proyectos pequeños. Sin embargo, este framework puede aumentarse fácilmente para utilizarse de manera eficaz en grandes proyectos, programas y portafolios. En situaciones donde el tamaño del Equipo Scrum excede diez personas, se pueden formar diversos equipos Scrum para trabajar en el proyecto. El enfoque lógico de las directrices y los principios de este framework pueden utilizarse para gestionar proyectos de cualquier tamaño, que abarcan grandes geografías y organizaciones. Los proyectos grandes pueden tener múltiples equipos Scrum que trabajan forma paralela, por lo que es necesario sincronizarse y facilitar el flujo de información y mejorar la comunicación. Los proyectos grandes y complejos generalmente se implementan como parte de un programa o portafolio.
Los detalles sobre la escalabilidad en Scrum para grandes proyectos se proporcionan en el Capítulo 13 y lo relacionado a escalar Scrum para la empresa se cubre en el Capítulo 14.

1.3 Propósito de la Guía SBOK™

En los últimos años ha sido evidente que las organizaciones que utilizan Scrum como el framework de ejecución de proyectos preferido, consistentemente obtienen altos retornos sobre la inversión. El enfoque de Scrum en la entrega impulsada por el valor ayuda a que los equipos de Scrum rindan resultados durante el proyecto tan pronto como les sea posible.

La *Guía SBOK*™ ha sido desarrollada como medio para crear una guía necesaria para organizaciones y profesionales de gestión de proyectos que deseen implementar Scrum, así como para quienes ya lo hacen y deseen mejorar sus procesos. Se basa en la experiencia adquirida de miles de proyectos a través de una variedad de organizaciones e industrias. En su desarrollo se tomaron en cuenta las aportaciones de muchos expertos en Scrum y profesionales de gestión de proyectos.

La *Guía SBOK*™ es especialmente valiosa:

- para los miembros del equipo principal de Scrum, incluyendo los siguientes:
  - ° Product Owners que deseen comprender plenamente el framework de Scrum y particularmente las inquietudes del cliente o stakeholder relacionadas a la justificación del negocio, a la calidad, el cambio y los aspectos de riesgo asociados con los proyectos Scrum.
  - ° Scrum Masters que quieran aprender su rol específico en la supervisión de la aplicación del framework de Scrum en proyectos de este tipo.
  - ° Miembros del Equipo Scrum que deseen comprender mejor los procesos de Scrum y las herramientas asociadas que pueden utilizarse para crear el producto o servicio del proyecto.
- como una guía integral para todos los practicantes de Scrum que trabajan en proyectos Scrum en cualquier organización o industria.
- como fuente de referencia para cualquier persona que interactúe con el equipo principal de Scrum, incluyendo, pero sin limitarse, al Portfolio Product Owner, Portfolio Scrum Master, Program Product Owner, Program Scrum Master, Scrum Guidance Body y Stakeholders (patrocinador, cliente y usuarios).
- como un manual para cualquier persona que no tenga experiencia previa o conocimiento del framework de Scrum, pero quiera aprender más sobre el tema.

El contenido de la *Guía SBOK*™ también es útil para las personas que se preparan para tomar los siguientes exámenes de certificación de SCRUMstudy™:

- Scrum Developer Certified (SDC™)
- Scrum Master Certified (SMC™)
- Scaled Scrum Master Certified (SSMC™)
- SCRUMstudy Agile Master Certified (SAMC™)
- Scrum Product Owner Certified (SPOC™)
- Scaled Scrum Product Owner Certified (SSPOC™)
- Expert Scrum Master Certified (ESMC™)

1.4 Framework de la Guía SBOK™

La *Guía SBOK™* se divide en las siguientes tres áreas:

1. Los **principios** que se contemplan en el capítulo 2 amplían la información sobre los seis principios que constituyen el fundamento sobre el que se basa Scrum.

2. Los **aspectos** que se cubren en los capítulos del 3 al 7 describen los cinco aspectos que se consideran importantes para todos los proyectos Scrum.

3. Los **procesos** que se cubren en los capítulos del 8 al 12 incluyen los diecinueve procesos fundamentales de Scrum y sus entradas, herramientas y salidas asociadas. Los capítulos 13 y 14 abordan procesos adicionales específicos sobre escalar Scrum en grandes proyectos y escalar Scrum para la empresa.

La figura 1-2 ilustra el framework de la *Guía SBOK™,* que muestra que los principios, aspectos y procesos interactúan entre sí y son de igual importancia al tratar de obtener una mejor comprensión del framework de Scrum.

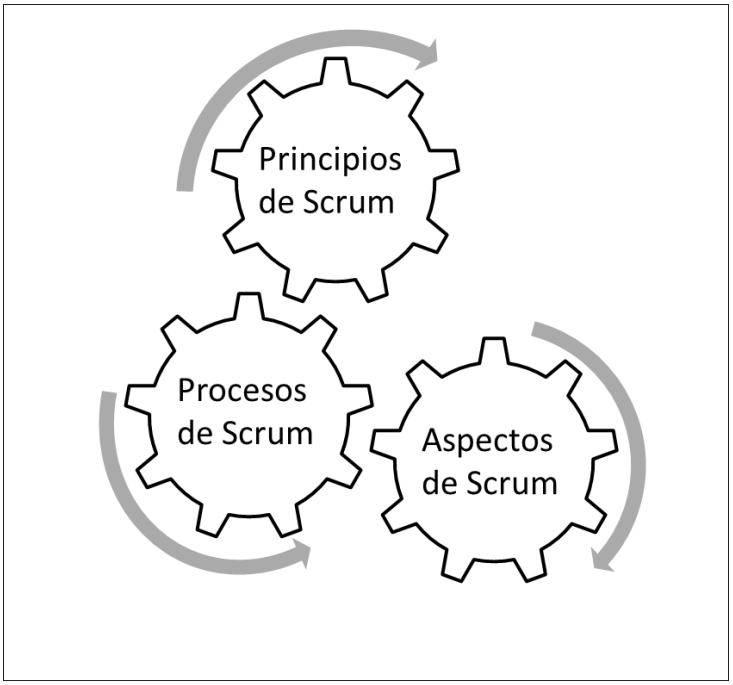


Figura 1-2: Framework de la *Guía SBOK™*

1.4.1 ¿Cómo utilizar la *Guía SBOK*™?

La *Guía SBOK*™ puede utilizarse como una referencia y guía de conocimiento tanto por practicantes de Scrum con experiencia y demás profesionales de desarrollo de productos y servicios, como por personas sin experiencia previa o conocimiento de Scrum o de métodos de gestión de proyectos. Los contenidos se organizan para facilitar la consulta de los tres roles principales del Equipo Scrum: Scrum Master, Product Owner y Equipo Scrum.

Los capítulos que abarcan los seis principios de Scrum (capítulo 2) y los cinco aspectos de Scrum (capítulos del 3 al 7), incluyen una guía de roles. Esta guía brinda información sobre los roles del equipo principal de Scrum (Scrum Core Team).

A fin de facilitar la mejor aplicación del framework de Scrum, la *Guía SBOK*™ ha diferenciado claramente entre las entradas, las herramientas y las salidas obligatorias de las opcionales. Las entradas, herramientas y salidas que se indican con asteriscos (*) son obligatorias, o consideradas importantes para el éxito, mientras que las que no tienen asteriscos son opcionales. Se recomienda que las personas que empiezan a aprender sobre Scrum se enfoquen principalmente en las entradas, las herramientas y las salidas obligatorias, mientras que los profesionales con más experiencia deben leer todos los capítulos del proceso a fin de beneficiarse de las entradas, herramientas y salidas sugeridas como mejores prácticas opcionales.

Scrum es un framework y no pretende ser prescriptivo, lo cual significa que hay espacio para la flexibilidad en su aplicación. Todos los procesos fundamentales de Scrum detallados en el SBOK (capítulos del 8 al 12) son obligatorios para cada proyecto Scrum, pero se aplicarían con base en las necesidades específicas de la organización, del proyecto, del producto o el equipo. Los procesos adicionales aplicarían solamente cuando se escale Scrum en grandes proyectos (capítulo 13) o cuando se escale Scrum para las empresas (capítulo 14).

1.4.2 Principios de Scrum

Los principios de Scrum son las pautas básicas para aplicar el framework de Scrum y deben implementarse en forma obligatoria en todos los proyectos Scrum. Los seis principios de Scrum que se presentan en el capítulo 2 son los siguientes:

1. Control del proceso empírico (*Empirical Process Control*)
2. Auto-organización (*Self-organization*)
3. Colaboración (*Collaboration*)
4. Priorización basada en valor (*Value-based Prioritization*)
5. Time-boxing
6. Desarrollo iterativo (*Iterative Development*)

La figura 1-3 ilustra los seis principios de Scrum.

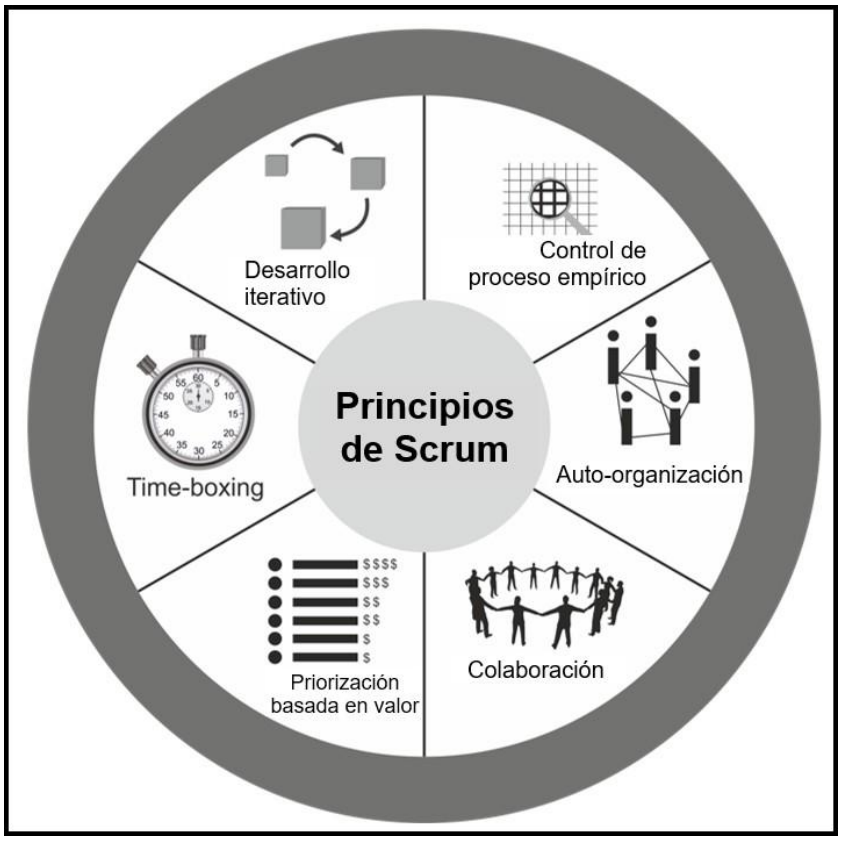


Figura 1-3: Principios de Scrum

Los principios de Scrum se pueden aplicar a cualquier tipo de proyecto en cualquier organización y deben cumplirse a fin de garantizar la aplicación efectiva del framework de Scrum. Los principios de Scrum no están abiertos a la discusión ni pueden modificarse, y deben aplicarse tal como se especifica en la *Guía SBOK™.* El mantener los principios intactos y usarlos apropiadamente infunde confianza en el framework de Scrum respecto al cumplimiento de los objetivos del proyecto. Los aspectos y procesos de Scrum, sin embargo, pueden modificarse para cumplir con los requisitos del proyecto o la organización.

1. **Control del proceso empírico**—Este principio enfatiza la filosofía central de Scrum con base a las tres ideas principales de transparencia, inspección y adaptación.

2. **Auto-organización**—Este principio se enfoca en los trabajadores de hoy en día, que entregan un valor considerablemente mayor cuando se auto-organizan, lo cual resulta en equipos que poseen un gran sentido de compromiso y responsabilidad; a su vez, esto produce un ambiente innovador y creativo que es más propicio para el crecimiento.

3. **Colaboración**—Este principio se centra en las tres dimensiones básicas relacionadas con el trabajo colaborativo: conocimiento, articulación y apropiación. También fomenta la gestión de proyectos como un proceso de creación de valor compartido con equipos que trabajan e interactúan conjuntamente para ofrecer el mayor valor.

4. **Priorización basada en valor**—Este principio pone de relieve el enfoque de Scrum para ofrecer el máximo valor de negocio, desde el principio del proyecto hasta su conclusión.

5. **Time-boxing**—Este principio describe cómo el tiempo se considera una restricción limitante en Scrum, y cómo este se utiliza para ayudar a manejar eficazmente la planificación y ejecución del proyecto. Los elementos del time boxing en Scrum incluyen sprints, Daily Standups, reuniones de planificación del sprint y reuniones de revisión del sprint.

6. **Desarrollo iterativo**—Este principio define el desarrollo iterativo y hace énfasis en cómo gestionar mejor los cambios y crear productos que satisfagan las necesidades del cliente. También delinea las responsabilidades del Product Owner y las de la organización relacionadas con el desarrollo iterativo.

1.4.3 Aspectos de Scrum

Los aspectos de Scrum deben abordarse y gestionarse durante todo un proyecto Scrum. Los cinco aspectos de Scrum que se presentan en los capítulos del 3 al 7 son los siguientes:

1.4.3.1 Organización

Entender los roles y responsabilidades definidos en un proyecto Scrum es muy importante a fin de asegurar la implementación exitosa de Scrum.
Los roles de Scrum se dividen en dos amplias categorías:

1. **Roles centrales**—Los roles centrales son aquellos que se requieren obligadamente para crear el producto o servicio del proyecto. Las personas a quienes se les asignan los roles centrales están plenamente comprometidas con el proyecto y son las responsables del éxito de cada iteración del mismo, así como del proyecto en su totalidad.

   Estos roles incluyen:

   - El **Product Owner** es la persona responsable de lograr el máximo valor empresarial para el proyecto. Este rol también es responsable de la articulación de requisitos del cliente y de mantener la justificación del negocio para el proyecto. El Product Owner representa la voz del cliente.

   - El **Scrum Master** es un facilitador que asegura que el Equipo Scrum cuente con un ambiente propicio para completar el proyecto con éxito. El Scrum Master guía, facilita y enseña las prácticas de Scrum a todos los involucrados en el proyecto; elimina los impedimentos que pueda tener el equipo y se asegura de que se estén siguiendo los procesos de Scrum.

   - El **Equipo Scrum** es el grupo o equipo de personas responsables de entender los requisitos especificados por el Product Owner y de crear los entregables del proyecto.

2. **Roles no centrales**—Los roles no centrales son los que no son necesariamente obligatorios para el proyecto Scrum, y estos pueden incluir a miembros de los equipos que estén interesados en el proyecto. No tienen ningún rol formal en el equipo del proyecto, y pueden interactuar con el equipo, pero pueden no ser responsables del éxito del proyecto. Los roles no centrales deben tenerse en cuenta en cualquier proyecto de Scrum.

   Los roles no centrales incluyen los siguientes:

   - **Stakeholder(s)** es un término colectivo que incluye a clientes, usuarios y patrocinadores, que con frecuencia interactúan con el equipo principal de Scrum, e influyen en el proyecto a lo largo de su desarrollo. Lo más importante es que el proyecto produzca beneficios colaborativos para los stakeholders.

   - El **Scrum Guidance Body** (SGB) es un rol opcional, que generalmente consiste en un conjunto de documentos y/o un grupo de expertos que normalmente están involucrados en la definición de los objetivos relacionados con la calidad, las regulaciones gubernamentales, la seguridad y otros parámetros claves de la organización. El SGB guía el trabajo llevado a cabo por el Product Owner, el Scrum Master y el Equipo Scrum.

   - Los **vendedores**, incluyendo a individuos u organizaciones externas, ofrecen productos y/o servicios que no están dentro de las competencias centrales de la organización del proyecto.

La figura 1-4 ilustra la estructura de la organización Scrum.

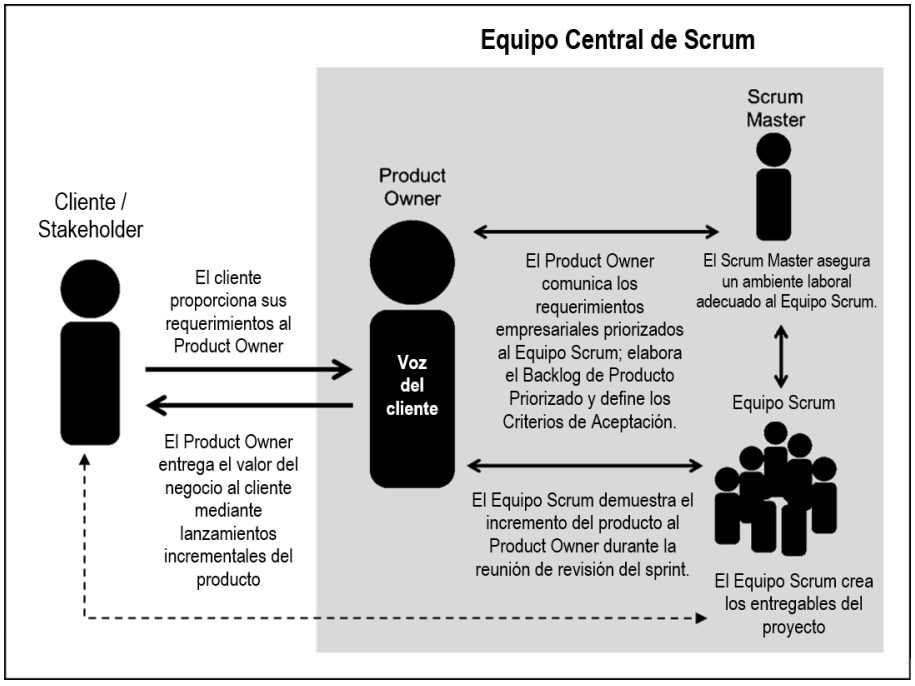


Figura 1-4: Organización en Scrum

El aspecto de organización de Scrum aborda también los requisitos de estructura del equipo para implementar Scrum en grandes proyectos, programas y portafolios.

1.4.3.2 Justificación del negocio

Es importante que una organización lleve a cabo una evaluación adecuada del negocio antes de iniciar cualquier proyecto. Esto ayuda a los tomadores de decisiones clave a entender la necesidad de cambio en la empresa o de un nuevo producto o servicio, la justificación para seguir adelante con un proyecto y su viabilidad.

En Scrum, la justificación del negocio se basa en el concepto de entrega impulsada por el valor (*Value- driven Delivery*). Una de las características claves de cualquier proyecto es la incertidumbre sobre los resultados. Es imposible garantizar el éxito de un proyecto, independientemente del tamaño o la complejidad del mismo. Considerando esta inseguridad de alcanzar el éxito, Scrum busca iniciar la entrega de resultados lo antes posible en el proyecto. Esta entrega temprana de resultados, y por lo tanto de valor, proporciona una oportunidad para la reinversión y demuestra el valor del proyecto a los stakeholders interesados.

La adaptabilidad de Scrum permite que los objetivos y procesos del proyecto cambien si cambia su justificación del negocio. Es importante señalar que, si bien el Product Owner es el responsable principal de la justificación del negocio, otros miembros del equipo también contribuyen considerablemente.

1.4.3.3 Calidad

En Scrum, la calidad se define como la capacidad con la que cuenta el producto o los entregables para cumplir con los criterios de aceptación y de alcanzar el valor de negocio que el cliente espera.
Para garantizar que un proyecto cumpla con los requisitos de calidad, Scrum adopta un enfoque de mejora continua mediante el cual el equipo aprende de sus experiencias y de la participación de los stakeholders para mantener constantemente actualizado el Backlog Priorizado del Producto con cualquier cambio en los requisitos. El Backlog Priorizado del Producto nunca se completa sino hasta el cierre o conclusión del proyecto. Cualquier cambio en los requisitos debe reflejar los cambios en el entorno empresarial, ya sean internos o externos, y permitirle al equipo trabajar continuamente y adaptarse para lograr dichos requerimientos.

Debido a que Scrum requiere que el trabajo se realice en incrementos durante los sprints, esto hace que los errores o defectos se noten con más facilidad mediante pruebas de calidad repetitivas y no simplemente cuando el producto final o servicio esté casi terminado. Por otra parte, las tareas relacionadas a la calidad (por ejemplo, desarrollo, pruebas y documentación) se completan como parte del mismo sprint por el mismo equipo. Esto asegura que la calidad sea inherente a cualquier entregable que se crea como parte de un sprint. A tales entregables de proyectos Scrum, que son potencialmente enviables, se les conoce como "terminado".

Por lo tanto, la mejora continua con pruebas repetitivas optimiza la probabilidad de alcanzar los niveles esperados de calidad en un proyecto Scrum. Las discusiones constantes entre el equipo principal de Scrum y los stakeholders (incluyendo los clientes y los usuarios), junto con incrementos reales del producto que se entregan al final de cada sprint, aseguran que la brecha entre las expectativas de los clientes del proyecto y los verdaderos entregables se reduzca constantemente.

El Scrum Guidance Body también puede proporcionar directrices sobre la calidad que pueden ser de interés para todos los proyectos Scrum en la organización.

1.4.3.4 Cambio

Cada proyecto, independientemente del método o framework que se utilice, está expuesto a cambios. Es importante que los miembros del equipo del proyecto entiendan que los procesos de desarrollo de Scrum están diseñados para aceptar el cambio. Las organizaciones deben tratar de maximizar los beneficios que se deriven de los cambios y minimizar cualquier impacto negativo a través de procesos de gestión de cambio diligentes, según los principios de Scrum.

Un principio fundamental de Scrum es su reconocimiento de que **a)** los stakeholders (clientes, usuarios y patrocinadores) cambian de opinión acerca de lo que quieren y lo que necesitan durante un proyecto (a esto se le conoce en ocasiones como: "requisitos volátiles" o *requirements churn*); y, **b)** que es muy difícil, si no es que imposible, que los stakeholders definan todos los requisitos al inicio del proyecto.

Los proyectos Scrum aceptan los cambios mediante el uso de sprints breves e iterativos que incorporan la retroalimentación del cliente en cada entregable del sprint. Esto permite que el cliente interactúe regularmente con los miembros del Equipo Scrum, que vea los entregables a medida que estén listos y que cambie los requisitos si es necesario antes del siguiente sprint.

Asimismo, los equipos de gestión de programa o portafolio pueden responder a las solicitudes de cambio pertenecientes a los proyectos Scrum aplicables a su nivel.

1.4.3.5 Riesgo

El riesgo se define como un evento incierto o serie de eventos que pueden afectar los objetivos de un proyecto y pueden contribuir a su éxito o fracaso. A los riegos que pueden tener un impacto positivo en el proyecto se les conoce como oportunidades, mientras que las amenazas son riesgos que pudieran afectar negativamente al proyecto. La gestión de riesgos debe hacerse de forma preventiva, y es un proceso iterativo que debe comenzar al inicio del proyecto y continuar a lo largo del ciclo de vida del mismo. El proceso de gestión de riesgos debe seguir algunos pasos estandarizados para asegurar que estos se identifiquen y evalúen, y que se determine un curso adecuado de acción y se proceda en consecuencia.

Los riesgos deben ser identificados, evaluados y atendidos con base a dos factores: la probabilidad de ocurrencia de cada riesgo y el posible impacto en el caso de tal ocurrencia. Los riesgos con una alta probabilidad y valor de impacto (que se calcula

multiplicando ambos factores) deben ser atendidos primero que aquellos con un valor relativamente bajo. En general, una vez que se detecta un riesgo, es importante entender el mismo en relación con las causas probables y los posibles efectos.

1.4.4 Procesos de Scrum

Los procesos de Scrum abordan las actividades específicas y el flujo de un proyecto de Scrum. En total hay diecinueve procesos fundamentales de Scrum que aplican a todos los proyectos. Estos procesos se agrupan en cinco fases y se presentan en los capítulos del 8 al 12 de la *Guía SBOK*™ tal como se muestra en la Tabla 1-1.

Capítulo	Fase	Procesos fundamentales de Scrum
8	Inicio	1. Crear la visión del proyecto 2. Identificar al Scrum Master y Stakeholder(s) 3. Formar Equipos Scrum 4. Desarrollar épica(s) 5. Crear el Backlog Priorizado del Producto 6. Realizar la planificación de lanzamiento
9	Planificación y estimación	7. Crear historias de usuario 8. Estimar historias de usuario 9. Comprometer historias de usuario 10. Identificar tareas 11. Estimar tareas 12. Crear el Sprint Backlog
10	Implementación	13. Crear entregables 14. Realizar Daily Standup 15. Refinar el Backlog Priorizado del Producto
11	Revisión y retrospectiva	16. Demostrar y validar el sprint 17. Retrospectiva del sprint
12	Lanzamiento	18. Enviar entregables 19. Retrospectiva del proyecto

Tabla 1-1: Resumen de los procesos fundamentales de Scrum

Estas fases describen a detalle cada proceso, incluyendo sus entradas, herramientas y salidas asociadas. En cada proceso, algunas entradas, herramientas y salidas son obligatorias (las que tienen un asterisco [*]), mientras que otras son opcionales. La inclusión de las entradas, herramientas y/o salidas opcionales dependerá del proyecto en particular, de la organización o la industria. Las entradas, herramientas y salidas señaladas con un asterisco son consideradas obligatorias o importantes para la implementación exitosa de Scrum en cualquier organización.

Para proyectos Scrum a grande escala que requieren de una coordinación entre múltiples equipos, existen tres procesos adicionales de Scrum que se definen en el capítulo 13: Escalar Scrum en grandes proyectos (*Scaling Scrum for Large Projects*). Existen también procesos específicos definidos cuando se implementa Scrum al nivel empresarial, lo cual se aborda en el capítulo 14: Escalar Scrum para la empresa (*Scaling Scrum for the Enterprise*). Estos procesos adicionales de Scrum se resumen en la tabla 1-2.

Capítulo	Aplicabilidad	Procesos adicionales de Scrum
13	Scrum para grandes proyectos	1. Crear componentes de grandes proyectos 2. Realizar y coordinar sprints 3. Preparar el lanzamiento de grandes proyectos
14	Scrum para la empresa	1. Crear componentes de programa o portafolio 2. Revisar y actualizar el Scrum Guidance Body 3. Crear y refinar el backlog del programa o portafolio 4. Coordinar los componentes del programa o portafolio 5. Retrospectiva de los lanzamientos del programa o portafolio

1.4.4.1 Inicio

1. *Crear la visión del proyecto*—En este proceso se revisa el caso de negocio del proyecto (*Project Business Case*) a fin de crear una Declaración de la visión del proyecto, que servirá de inspiración y proporcionará un enfoque para todo el proyecto. En este proceso se identifica al Product Owner.
2. *Identificar al Scrum Master y Stakeholder(s)*—En este proceso se identifica al Scrum Master y stakeholders utilizando criterios de selección específicos.

3. *Formar Equipos Scrum*—En este proceso se identifican a los miembros del Equipo Scrum. Normalmente, el Product Owner es el responsable principal de la selección de los miembros del equipo, pero con frecuencia lo hace en colaboración con el Scrum Master.

4. *Desarrollar épica(s)*—En este proceso la Declaración de visión del proyecto sirve como base para el desarrollo de épicas. Se pueden llevar a cabo reuniones de grupos de usuarios para hablar sobre las épicas adecuadas.

5. *Crear el Backlog Priorizado del Producto*—En este proceso se refinan y se crean las épicas, y después se priorizan para crear un Backlog Priorizado del Producto para el proyecto. A este punto también se establecen los criterios de terminado.

6. *Realizar la planificación del lanzamiento*—En este proceso el equipo principal de Scrum revisa las historias de usuario en el Backlog Priorizado del Producto para desarrollar un cronograma de planificación del lanzamiento, que es esencialmente un programa de implementación por fases que se puede compartir con los stakeholders del proyecto. En este proceso también se determina la duración del sprint.

1.4.4.2 Planificación y estimación

7. *Crear historias de usuario*—En este proceso se crean las historias de usuario y los criterios de aceptación de las historias de usuario. Las historias de usuario generalmente las escribe el Product Owner y están diseñadas para

asegurar que los requisitos del cliente estén claramente representados y puedan ser plenamente comprendidos por todos los stakeholders. Se pueden llevar a cabo ejercicios de redacción de historias de usuario, lo cual incluyan a los miembros del Equipo Scrum, resultando en la creación de dichas historias. Estas se incorporan al Backlog Priorizado del Producto.

8. *Estimar historias de usuario*—En este proceso, el Product Owner aclara las historias de usuario para que el Scrum Master y el Equipo Scrum puedan estimar el esfuerzo necesario para desarrollar la funcionalidad descrita en cada historia de usuario.

9. *Comprometer historias de usuario*—En este proceso, el Equipo Scrum se compromete a entregar al Product Owner las historias de usuario aprobadas para un sprint. El resultado de este proceso serían las historias de usuario comprometidas.

10. *Identificar tareas*—En este proceso, las historias de usuario comprometidas se desglosan en tareas específicas y se compilan en una lista de tareas.

11. *Estimar tareas*—En este proceso, el equipo principal de Scrum estima el esfuerzo necesario para cumplir con cada tarea en la lista de tareas. El resultado de este proceso es una: Effort Estimated Task List.

12. *Crear el Sprint Backlog*—En este proceso, el equipo principal de Scrum elabora un Sprint Backlog que contiene todas las tareas a ser completadas en un sprint como parte de la Reunión de Planificación del Sprint.

1.4.4.3 Implementación

13. *Crear entregables*—En este proceso, el Equipo Scrum trabaja en las tareas en el Sprint Backlog para crear los entregables del sprint. Generalmente se utiliza un Scrumboard para dar seguimiento a las actividades que se llevan a cabo. Las asuntos o problemas que enfrenta el equipo Scrum pudieran actualizar se en un Impediment Log (o registro de impedimentos).

14. *Realizar Daily Standup*—En este proceso, se lleva a cabo diariamente una reunión altamente focalizada con un time-box, conocida como Daily Standup. Es aquí donde los miembros del Equipo Scrum se actualizan el uno al otro referente a sus progresos y sobre los impedimentos que pudieran enfrentar.

15. *Refinamiento del Backlog Priorizado del Producto*—En este proceso, el Backlog Priorizado del Producto se actualiza y se refina continuamente. Se puede considerar realizar una reunión de revisión del Backlog Priorizado del Producto, en la que se analiza cualquier cambio o actualización al backlog y se incorpora a dicho backlog según sea necesario.

1.4.4.4 Revisión y retrospectiva

16. *Demostrar y validar el sprint*—En este proceso, el Equipo Scrum muestra los entregables del sprint al Product Owner y a los stakeholders relevantes en una Reunión de Revisión del Sprint. El propósito de esta reunión es asegurar que se obtenga la aprobación y aceptación del Product Owner

respecto a los entregables elaborados en el sprint.

17. *Retrospectiva del sprint*—En este proceso, el Scrum Master y el Equipo Scrum se reúnen para analizar las lecciones aprendidas durante todo el Sprint. Esta información se documenta en forma lecciones aprendidas que pueden aplicarse a futuros sprints. Frecuentemente, como resultado de esta discusión, puede haber mejoras aceptadas (*Agreed Actionable Improvements*) o recomendaciones actualizadas por parte del Scrum Guidance Body.

1.4.4.5 Lanzamiento

18. *Enviar entregables*—En este proceso, los entregables aceptados se entregan o se envían a los stakeholders relevantes. Un documento denominado *Working Deliverables Agreement* (Acuerdo de entregables funcionales) documenta la conclusión satisfactoria del sprint.

19. *Retrospectiva del proyecto*—En este proceso, mismo que concluye el proyecto, los stakeholders y miembros del equipo principal de Scrum se reúnen para hacer una retrospectiva del proyecto e identificar, documentar e internalizar las lecciones aprendidas. A menudo, estas lecciones llevan a la documentación de Agreed Actionable Improvements, que se implementarán en futuros proyectos.

1.4.4.6 Scrum para grandes proyectos

Crear componentes de grandes proyectos—Este proceso define la forma en la que los Product Owners trabajan en conjunto y de cómo varios equipos de Scrum trabajan juntos. También se identifican componentes comunes, así como recursos comunes y especializados.

Realizar y coordinar sprints—Este proceso generalmente solo es relevante en grandes proyectos y aborda aspectos específicos que deben ser considerados durante cada sprint. De ser necesario, se pueden llevar a cabo reuniones de Scrum de Scrums a fin de coordinar los esfuerzos entre los distintos equipos de Scrum.

Preparar el lanzamiento de grandes proyectos—En algunos proyectos grandes, pudiera tener sentido empresarial llevar a cabo un sprint especial con anticipación a un lanzamiento a fin de preparar el lanzamiento del producto (que será decidido por el equipo del proyecto con base en las necesidades del negocio). Este proceso aborda dicho sprint preparatorio.

1.4.4.7 Scrum para la empresa

Crear componentes de programa o portafolio—En este proceso, el Program Product Owner o el Portfolio Product Owner, así como los stakeholders clave identifican componentes comunes y recursos necesarios para el programa o portafolio. Los criterios mínimos de terminado se definen y se identifican a todos los stakeholders.

Revisar y actualizar el Scrum Guidance Body—En este proceso, las recomendaciones del Scrum Guidance Body se revisan constantemente por

parte de sus miembros y se actualizan cuando sea necesario. En este proceso, también se atienden los cambios en los integrantes del Scrum Guidance Body.

Crear y refinar el backlog de programa o portafolio—En este proceso, se elabora, se actualiza y se refina el backlog del programa o portafolio. Se pueden hacer recomendaciones de mejoramiento por parte del Scrum Guidance Body y se pueden modificar los plazos con base en los cambios en los requerimientos y/o procesos del proyecto en el programa o portafolio.

Coordinar los componentes del programa o portafolio—En este proceso se coordinan los componentes del programa o portafolio. Se atienden las dependencias entre proyectos; se discuten los impedimentos comunes y se comparten las mejores prácticas. En ocasiones, se hacen recomendaciones de mejoramiento por parte del Scrum Guidance Body.

Retrospectiva de lanzamientos del programa o portafolio—En este proceso, el Program Product Owner o el Portfolio Product Owner y los stakeholders clave se reúnen para hacer una retrospectiva sobre el lanzamiento de un programa o portafolio e internalizar las lecciones aprendidas. Por lo general dichas lecciones aprendidas llevan a mejoras aceptadas (*Agreed Actionable Improvements*) para ser implementadas a futuro.

1.5 Scrum vs. Gestión tradicional de proyectos

La tabla 1-2 resume muchas de las diferencias entre los modelos tradicionales de gestión de proyectos.

	Scrum	Gestión tradicional de proyectos
El énfasis está en	Las personas	Los procesos
Documentación	Sólo mínima; según se requiera	Integral
Estilo de procesos	Iterativo	Lineal
Planificación por adelantado	Baja	Alta
Priorización de requerimientos	Según el valor del negocio y regularmente actualizada	Fijo en el plan de proyecto
Garantía de calidad	Centrada en el cliente	Centrada en el proceso
Organización	Auto-organizada	Gestionada
Estilo de gestión	Descentralizado	Centralizado
Cambio	Actualizaciones al Backlog Priorizado del Producto	Sistema formal de gestión del cambio
Liderazgo	Liderazgo colaborativo y servicial	Mando y control
Medición del rendimiento	El valor del negocio	Conformidad con el plan
Retorno sobre la inversión (RSI)	Al comienzo y a lo largo del proyecto	Al final del proyecto
Participación del cliente	Alta durante todo el proyecto	Varía dependiendo del ciclo de vida del proyecto

Tabla 1-2: Scrum vs. Gestión tradicional de proyectos