



El marco de gestión de proyectos de metodología agile Scrum

Autor: Jonathan Carrero

Índice

1. Introducción a las metodologías ágiles	4
1.1. Aparición y descripción general de los métodos ágiles	5
1.2. El <i>Manifiesto agile</i>	7
1.2.1. Principios del Manifiesto agile	8
1.3. Aproximación tradicional vs. agile	10
1.4. Aproximación al marco Scrum	11
1.5. Cómo está organizado este documento	12
2. Gestión de equipos ágiles	12
2.1. Los roles del equipo de proyecto en Scrum	13
2.1.1. El Propietario del Producto o <i>Product Owner</i>	14
2.1.2. El <i>Scrum Master</i>	16
2.1.3. El <i>Development Team</i>	17
2.1.4. Roles auxiliares en Scrum	19
2.1.5. La dinámica de un equipo Scrum	19
3. Los eventos de Scrum	21
3.1. Sprint	21
3.1.1. La cancelación de un sprint	22
3.2. Sprint planning	23
3.2.1. Origen del contenido que realizar en el sprint	23
3.2.2. ¿Qué puede entregarse en este sprint?	26
3.2.3. ¿Cómo se puede entregar el trabajo seleccionado?	26
3.2.4. Objetivo del sprint o <i>sprint goal</i>	27
3.3. Daily meeting	28
3.4. Sprint review	29
3.5. Sprint retrospective	30
4. Artefactos de Scrum	31
4.1. Los artefactos de Scrum	32
4.1.1. Product backlog	32
4.1.2. Ejemplo de aproximación	34
4.1.3. Sprint backlog	36
4.1.4. Impediment backlog	37
4.1.5. Incremento	38
4.2. Herramientas para la gestión	39
4.2.1. Los tableros	40

4.2.2.	Burn-down chart	40
4.2.3.	Jira	41
4.2.4.	Asana	42
4.2.5.	Pivotal Tracker	44
4.2.6.	Active Collab	44
5.	La agilidad organizacional	44
5.1.	Evolucionar hacia la agilidad técnica y organizacional	45
5.1.1.	Agilidad técnica	45
5.1.2.	Agilidad organizacional	46
5.1.3.	La cadena de valor	47

1. Introducción a las metodologías ágiles

En el entorno empresarial actual, la gestión de proyectos de Tecnologías de la Información se ha vuelto más compleja y dinámica que nunca. La necesidad de adaptarse rápidamente a los cambios del mercado, satisfacer las demandas cambiantes de los clientes y entregar productos de alta calidad en plazos ajustados ha llevado a la adopción de enfoques más flexibles y ágiles en la gestión de proyectos. Las metodologías ágiles han emergido como una respuesta efectiva a estos desafíos, revolucionando la forma en que se abordan y ejecutan los proyectos de TI. A continuación, se comentará sobre las metodologías ágiles en la gestión de proyectos de TI, desde su definición hasta sus características, tipos, aspectos relevantes y descripción de las metodologías más utilizadas.

Podríamos decir que las metodologías ágiles son un conjunto de enfoques y prácticas de gestión de proyectos que enfatizan la flexibilidad, la colaboración y la adaptación constante a los cambios. A diferencia de los enfoques tradicionales de gestión de proyectos, que se basan en la planificación exhaustiva y la ejecución secuencial, las metodologías ágiles valoran la interacción humana, la entrega incremental y la respuesta ágil a los cambios en los requisitos del proyecto.

Este conjunto de metodologías se caracterizan por una serie de principios y valores fundamentales:

- **Colaboración y Comunicación:** fomentan la colaboración activa entre los miembros del equipo y los interesados, promoviendo la comunicación constante y la toma de decisiones conjuntas.
- **Flexibilidad y Adaptabilidad:** se adaptan de manera fluida a los cambios en los requisitos y prioridades del proyecto, lo que permite responder rápidamente a las demandas del mercado.
- **Entrega Incremental:** dividen el proyecto en entregas pequeñas y manejables, lo que permite obtener resultados tangibles y valiosos de manera temprana.
- **Feedback Continuo:** se basan en la retroalimentación constante de los interesados y usuarios, lo que permite ajustar y mejorar el producto en cada iteración.
- **Enfoque en el Cliente:** colocan al cliente en el centro del proceso, asegurándose de que el producto final satisfaga sus necesidades y expectativas.
- **Autogestión del Equipo:** empoderan al equipo para tomar decisiones

y autogestionar su trabajo, lo que aumenta la responsabilidad y la motivación.

Por otro lado, existen varias metodologías ágiles, cada una con enfoques y características específicas. Algunas de las más conocidas son:

- **Scrum:** es una de las metodologías ágiles más populares. Se basa en la división del proyecto en iteraciones llamadas *sprints*. Cada sprint tiene una duración fija y al final se entrega un incremento funcional del producto.
- **Kanban:** se centra en el flujo de trabajo visual y limita la cantidad de trabajo en progreso. Las tareas se mueven a través de etapas definidas, lo que ayuda a mantener un flujo constante y a detectar cuellos de botella.
- **Extreme Programming (XP):** se centra en las prácticas de desarrollo de software, como la programación en parejas y las pruebas automatizadas. Prioriza la entrega de software funcional y de alta calidad.
- **Crystal:** es una familia de metodologías ágiles que se adaptan según el tamaño y la complejidad del proyecto. Se centra en la colaboración, la comunicación y la entrega incremental.
- **Lean Software Development:** inspirado en los principios de Lean Manufacturing, se enfoca en eliminar el desperdicio y optimizar el flujo de trabajo. Prioriza la entrega rápida y la mejora continua.

Por cierto, antes de continuar con los siguientes capítulos, puedes echar un vistazo a cómo está organizado este documento en la Sección X.

1.1. Aparición y descripción general de los métodos ágiles

En la gestión de proyectos, el término *ágil* es comúnmente demandado por las empresas en la actualidad. Este término representa la versatilidad y adaptabilidad, bien sea preventiva o reactiva, a las situaciones que acontecen con los proyectos, productos y servicios que ofrece la organización. La agilidad se debe entender como una forma de pensar que aborda herramientas potencialmente beneficiosas para las organizaciones. Sin embargo, se debe tener cuidado porque dominarla y potenciarla no es tan sencillo como parece.

En la actualidad, los cambios, demandas y expectativas de los mercados, además, de los cambios tecnológicos que cada vez son más abrumadores, hacen que las metodologías de gestión de proyectos tradicionales se vean desafiadas y superadas constantemente. Esto se debe a que los modelos más tradicionales tienden a querer controlar y anticipar una planificación completa de los proyectos, la

cual se ve inmediatamente mermada por los frecuentes cambios en el entorno. En respuesta a esta situación, los directores de proyecto han tenido que evolucionar su forma de abordar los proyectos agilizándolos de manera progresiva para suplir los requerimientos que las otras metodologías no atendían correctamente, y así poder cubrir las expectativas del mercado.

Las metodologías ágiles están basadas en las planificaciones iterativas, en las cuales se realizan entregas tempranas cuyo objetivo es incrementar el valor progresivamente. De esta forma se procura poder adaptar el producto a las expectativas del cliente con cada una de las iteraciones. Por ello, es necesaria la comunicación activa y permanente entre todos los interesados. Desde los años cincuenta se han propuesto distintos modelos adaptativos e iterativos. Sin embargo, no se formaliza el término *ágil* hasta que se conformó el *Manifiesto agile*.

Los proyectos ágiles son parte de la gestión de proyectos. En la actualidad, cada vez son más utilizados por las empresas y organizaciones, ya que en este tipo de metodologías los procesos son diferentes a la clásica gestión, y se busca obtener resultados mucho más rápidamente.

Como hemos resaltado anteriormente, desde los años cincuenta se han desarrollado aproximaciones adaptativas e incrementales como respuesta a una situación que en su momento parecía no tener solución. Estas metodologías hoy se alinean en su mayoría con los valores y principios ágiles firmados en el *Manifiesto agile*.

Tarea: busca un timeline de las metodologías ágiles. ¿Cuándo han surgido cada una de ellas? ¿Cuándo empezó a utilizarse el marco Scrum?

Antes de comenzar a definir Scrum, que será el marco de trabajo en el cual se enfocará este primer trimestre, hace falta entender, por lo menos de forma general, cuál es la razón de ser de los proyectos ágiles, pues de esta manera será más sencillo comprender y aplicar esta metodología en una organización.

Los proyectos ágiles son un conjunto de metodologías dirigidas al desarrollo de proyectos que requieren rapidez y flexibilidad en su proceso. En la mayoría de las ocasiones estos proyectos están vinculados con el desarrollo de software y el ámbito digital.

Las metodologías ágiles de gestión de proyectos nacieron, por lo tanto, gracias al dinamismo del mundo empresarial contemporáneo, en el que los métodos tradicionales de gestión no han resultado ser lo suficientemente eficientes. Y, específicamente, tienen su origen en los proyectos informáticos.

Al aplicarse la gestión tradicional a los proyectos relacionados con la informá-

tica, no se solían obtener resultados óptimos, pues estos requieren de cambios constantes y adaptaciones, de una comunicación más continua, y de menos planificación. Son entornos donde al demandarse tantas modificaciones, eventualmente los planes realizados en un primero momento pasan a un segundo plano de importancia.

1.2. El *Manifiesto agile*

En febrero del 2001, un grupo de 17 profesionales y administradores del sector informático, dedicados al desarrollo de software, crearon la asociación denominada Alianza Ágil (Agile Alliance), dedicada a discutir los métodos de gestión que les proporcionaban mejores resultados. En los debates de estas reuniones se redactó un *Manifiesto para el desarrollo agile de software* (Manifesto for Agile Software Development).

El *Manifiesto ágil* está comprometido con cuatro valores fundamentales que lideran el acercamiento ágil al desarrollo de software. Cada metodología ágil aplica estos cuatro valores de diferentes maneras, pero todos confían en ellos para guiar el desarrollo y la entrega de software que funcione y de alta calidad.

- **Individuos e interacciones por encima de procesos y herramientas:** este es el primer valor en el Manifiesto ágil. Valorar a la gente más que a los procesos o a las herramientas es fácil de entender, porque se trata de gente que responde a las necesidades de negocio y conduce el proceso de desarrollo. Si el proceso o las herramientas conducen el desarrollo, el equipo es menos sensible al cambio y tiene menos probabilidad de conocer las necesidades del cliente. La comunicación es un ejemplo de la diferencia entre valorar los individuos y los procesos. En el caso de los individuos, la comunicación es fluida y ocurre cuando surge una necesidad. En el caso del proceso, la comunicación se programa y requiere de un contenido específico.
- **Software que funcione por encima de una documentación extensiva:** Históricamente, se ha gastado una enorme cantidad de tiempo en documentar el producto para el desarrollo y la última entrega de un producto. Las especificaciones y requisitos técnicos, la propuesta técnica, los documentos de diseño de interfaz, los planes de prueba, los planes de documentación y el visto bueno de cada uno de ellos eran algo fundamental. El listado es enorme y era una de las causas de los grandes retrasos en el desarrollo de un producto. Ágil no elimina la documentación, sino que la reestructura de forma que proporciona al desarrollador lo que necesita

para hacer el trabajo sin empantanarse en minucias. Ágil documenta los requisitos como historias de usuario, que son suficientes para un desarrollador software para empezar la tarea de construir una nueva funcionalidad. El *Manifiesto ágil* valora la documentación, pero valora mucho más el software que funciona.

- **Colaboración con el cliente por encima de la negociación contractual:** La negociación es el periodo en el que el cliente y el encargado del producto trabajan en los detalles de una entrega de producto, con puntos donde los detalles pueden ser renegociados. La colaboración es una criatura completamente diferente. Con modelos de desarrollo tales como waterfall, los clientes negocian los requisitos del producto, a menudo con mucho detalle, antes de empezar cualquier trabajo. Esto significa que el cliente se involucra en el proceso de desarrollo antes de que este empiece y después de que se complete, pero no durante el proceso explícito de desarrollo. El *Manifiesto ágil* describe a un cliente que está comprometido y colabora en el proceso de desarrollo. Esto hace que sea mucho más fácil para el desarrollo el poder satisfacer las necesidades de los clientes. Los métodos ágiles pueden incluir al cliente en demostraciones en intervalos periódicos de tiempo, por lo que tener a un usuario final como parte del equipo y asistiendo a todas las reuniones puede asegurar que el producto cumple con las necesidades de negocio del cliente.
- **Respuesta ante el cambio en lugar de seguir un plan:** El desarrollo de software tradicional considera el cambio como caro, por lo que los cambios tienden a evitarse. La intención es desarrollar de forma detallada y elaborar planes con un conjunto definido de funcionalidades, con unas prioridades y con un gran número de dependencias en la entrega, de forma que el equipo pueda trabajar en la siguiente pieza del puzle. Con ágil, el tiempo de una iteración significa que las prioridades pueden ser empaquetadas de iteración en iteración y se pueden añadir nuevas funcionalidades en la siguiente iteración. El punto de vista de ágil es que los cambios siempre mejoran un proyecto y que proporcionan un valor adicional al producto, por lo que todos los participantes deben estar preparados para realizarlos.

1.2.1. Principios del Manifiesto agile

Fowler y Highsmith completaron el *Manifiesto ágil* (2001) con los siguientes doce principios de gestión del proyecto que deberían guiar a los métodos ágiles

utilizados para el desarrollo de sistemas. La combinación de los cuatro valores fundamentales vistos en el apartado anterior, junto con los doce principios que veremos en este apartado, hicieron que se creara el desarrollo de software ágil tal y como lo conocemos. Estos doce principios se pueden dividir en tres grandes categorías que veremos a continuación. Por cierto, no es necesario aprenderse todos los principios, pero sí conocerlos.

La **primera categoría** se basa en la entrega de software de forma regular.

- Principio 1. Nuestra máxima prioridad es satisfacer al cliente a través de la entrega temprana y continua de software con valor para el cliente.
- Principio 2. Entregar software que funcione con frecuencia, desde un par de semanas a un par de meses, pero con preferencia en la elección de tiempos cortos.
- Principio 3. El software que funciona debe ser la principal medida de progreso.
- Principio 4. Los procesos ágiles promueven el desarrollo sostenible. Los patrocinadores, desarrolladores y usuarios deberían ser capaces de mantener un ritmo constante de desarrollo y de forma continua.

La **segunda categoría** se basa en la comunicación del equipo.

- Principio 5. La gente de negocios y los desarrolladores tienen que trabajar juntos de forma diaria durante el desarrollo del proyecto.
- Principio 6. El método más eficiente y eficaz de transmitir información al equipo de desarrollo y entre sus miembros es una conversación cara a cara.
- Principio 7. Las mejores arquitecturas, requisitos y diseños emergen de equipos autoorganizados.
- Principio 8. Construir un proyecto en torno a individuos motivados. Darles el soporte y el ambiente de trabajo que necesitan y confiar en ellos para que hagan su trabajo.
- Principio 9. A intervalos regulares, el equipo reflexiona sobre cómo llegar a ser más efectivo, y luego ajusta su comportamiento en consecuencia.

Por último, la **tercera categoría** se basa en la excelencia del diseño.

- Principio 10. La atención continua a la excelencia técnica y al buen diseño mejora la agilidad.

- Principio 11. Simplicidad, el arte de maximizar la cantidad de trabajo no realizado es fundamental.
- Principio 12. Los cambios de requisitos son bienvenidos, incluso en fases tardías del desarrollo. Los procesos ágiles aprovechan el cambio para obtener una ventaja competitiva para el cliente.

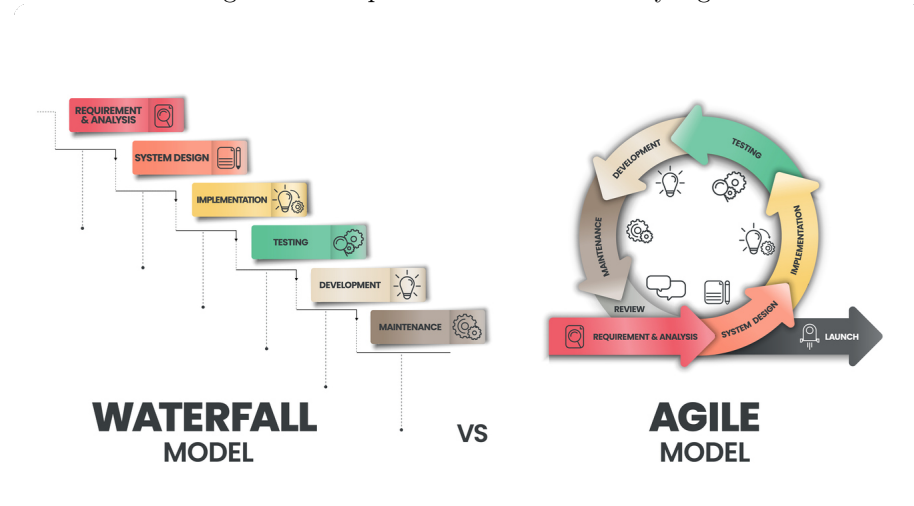
1.3. Aproximación tradicional vs. agile

Frente a la gestión de proyectos tradicional como un ciclo de vida waterfall, una aproximación agile al proyecto comprende cambiar el foco o énfasis del Project Manager, su equipo y los distintos elementos que comprenden sus trabajos y tareas. Se pueden comparar las distintas aproximaciones en función de estos puntos clave suponiendo un desarrollo web.

Tarea: busca una tabla comparativa entre los distintos aspectos de Waterfall y Agile.

En el anterior ejemplo del desarrollo de un sitio web, el cliente que solicita este proyecto preferirá obtener alguna página relacionada con su web lo antes posible, aunque no estén completas todas las páginas o secciones de esta. Sin embargo, si esperamos a entregar el proyecto cuando todo esté completo, durante el tiempo que dure el proyecto, el cliente no tendrá nada que mostrar en su sitio web. La aproximación agile a este proyecto se centra en aportar valor al cliente lo antes posible, mediante la entrega de partes 100 % funcionales del entregable final.

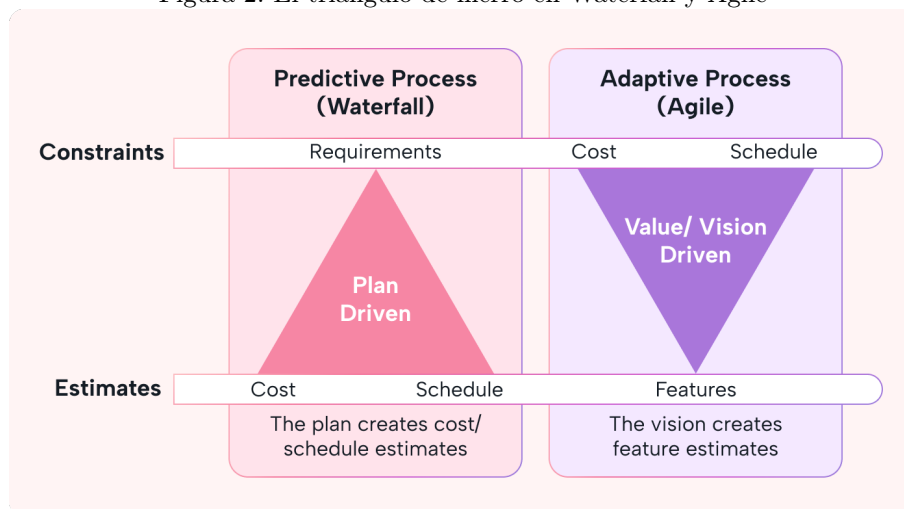
Figura 1: Comparativa entre Waterfall y Agile



1.4. Aproximación al marco Scrum

Un proyecto, tradicionalmente, consiste en un esfuerzo colaborativo con el fin de crear un nuevo producto, servicio o resultado de acuerdo con ciertos parámetros establecidos y cumpliendo un estándar de calidad acordado. Comúnmente, los proyectos vienen acotados por restricciones de tiempo, coste y alcance. Las metodologías ágiles buscan abordar estas restricciones de otra forma:

Figura 2: El triángulo de hierro en Waterfall y Agile



Las metodologías ágiles mantienen fijas las variables coste y tiempo, y, por lo tanto, es el alcance el que varía. Se itera con el cliente para buscar la mejor versión que se pueda entregar. De esta forma, el entorno colaborativo y la retroalimentación forman el eje fundamental de esta forma de pensar y desarrollar proyectos.

Scrum es uno de los métodos ágiles más utilizados. Es un marco de gestión adaptable, iterativo, rápido, flexible y eficaz, diseñado para entregar, rápidamente, valor considerable a lo largo del proyecto. Scrum pretende garantizar transparencia en la comunicación, así como crear un ambiente de responsabilidad colectiva. El marco Scrum 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 tamaño y complejidad.

Ken Schwaber y Jeff Sutherland desarrollaron el concepto de Scrum y su aplicabilidad al desarrollo de software en una presentación hecha en un congreso de programación realizado en 1995 en Austin, Texas. En los últimos años, Scrum se ha convertido en el método de desarrollo de proyectos predilecto de muchas

organizaciones a nivel mundial. Una fortaleza clave de Scrum radica en el uso de equipos interfuncionales, autogestionados y facultados que dividen su trabajo en ciclos de trabajo cortos, llamados *carreras* o *sprints*.

Este marco de trabajo se fundamenta en tres pilares que basados en la teoría del empirismo son fundamentales para su desarrollo.

- **La transparencia:** todo el equipo debe saber dónde estamos y a dónde vamos.
- **La inspección:** se revisa siempre el trabajo realizado para identificar mejoras.
- **La adaptación:** se cambia lo planificado si es necesario para alcanzar los objetivos.

Estos pilares se materializan durante todos los eventos y en todos los artefactos (elementos que conforman el marco) de Scrum.

1.5. Cómo está organizado este documento

Durante el primer punto abordamos una introducción de lo que son las metodologías ágiles. Esto incluye propósito, características, algunas de las metodologías comúnmente utilizadas, por qué surgieron y cuál es su historia, qué son los proyectos ágiles y algunas diferencias entre el enfoque agile y los proyectos tradicionales. Todos estos puntos de aproximación no serán objeto de evaluación de la asignatura.

A continuación, entramos de lleno en el marco Scrum, donde estudiaremos qué es, los elementos que tiene y cuál es el objetivo que persigue. Desde aquí en adelante, todos los conceptos que se aborden sí entrarán en las distintas pruebas de evaluación de la asignatura.

2. Gestión de equipos ágiles

Scrum es una práctica que se basa en los principios Agile, y se utiliza como marco de trabajo en proyectos de software. En Scrum, se emplean métodos empíricos científicos en lugar de los enfoques algorítmicos programados utilizados en proyectos de cascada lineal.

Scrum ha sido utilizado desde principios de los años noventa, pero sigue siendo una metodología relativamente nueva en la gestión de proyectos. No se trata de un proceso o técnica específica para construir productos, sino más bien de

un marco de trabajo que puede emplear varias técnicas y procesos. Según Ken Schwaber y Jeff Sutherland, creadores de Scrum, el enfoque funciona bien con Agile porque ayuda a los miembros del equipo a abordar problemas complejos y a lograr entregas productivas y creativas con el máximo valor posible.

Según *Scrum.org*, el enfoque beneficia tanto al equipo como al gestor del proyecto al demostrar la eficacia relativa de las prácticas de gestión de producto y desarrollo, permitiendo mejoras en el futuro si es necesario. En definitiva, Scrum se considera un enfoque altamente efectivo para la gestión de proyectos complejos y desafiantes. Gracias a esta metodología, podremos ver cómo nuestra productividad en el trabajo se ve reforzada, ya que esta metodología nos obliga a cumplir con determinados plazos de entrega, comentar nuestros éxitos y justificar nuestros fracasos. Debido a esto, esta metodología te hace ser más productivo:

- Aumenta la capacidad de reacción ante cambios repentinos.
- Reduce el tiempo que se tarda en lanzar algo al mercado.
- Se obtiene mayor calidad de software, ya que ayuda a obtener software de gran calidad en cada una de las iteraciones.
- Mejores predicciones de tiempos; como veremos más adelante, la velocidad media del equipo del sprint, de forma que se pueda medir de forma tangible el trabajo realizado por cada sprint.
- Reducción de riesgos. Al estar trabajando en las funcionalidades de mayor prioridad, nos permite despejar riesgos de forma anticipada.

Debes saber que hay tres grandes conjuntos de elementos dentro de esta metodología de trabajo: El equipo Scrum, Los eventos Scrum y Los artefactos Scrum. En esta sección del documento vamos a abordar el primero de ellos.

2.1. Los roles del equipo de proyecto en Scrum

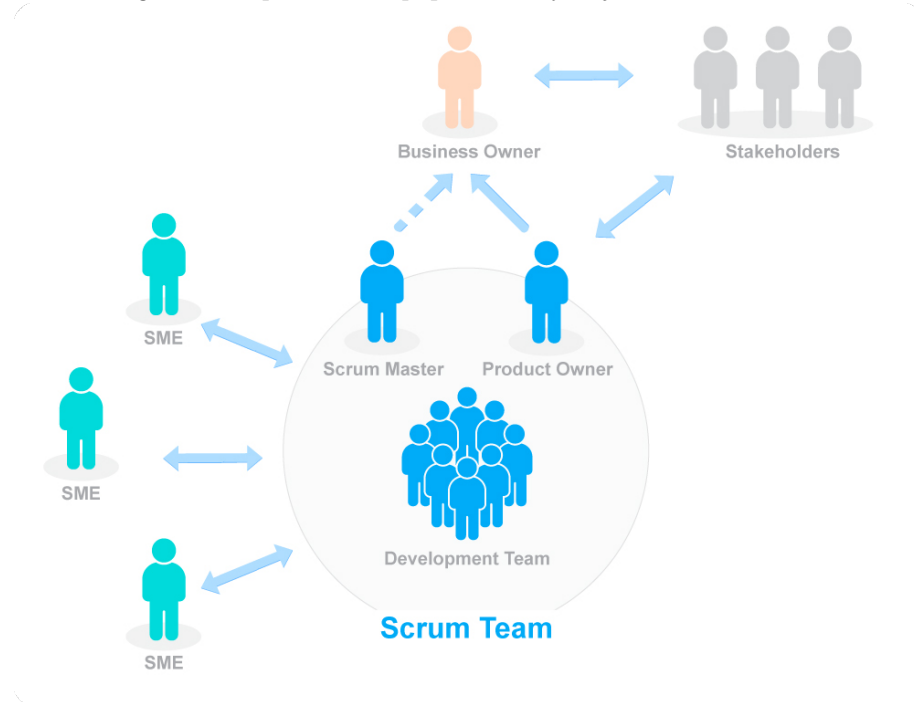
Los roles que desempeñan los integrantes de un equipo Scrum son distintos a los roles de administración de proyectos tradicionales. A diferencia de estos últimos, en un equipo Scrum solo existen tres roles principales y no necesariamente están alineados con los roles de administración. Los tres roles de Scrum son los siguientes:

- **Product Owner:** Orientado al negocio.
- **Scrum Master:** Centrado en que el equipo sea lo más productivo posible.

- **Development Team:** Centrado en el desarrollo de funcionalidades.

Adicionalmente, encontramos algunos acrónimos importantes como Stakeholders, Expertos en la materia (SME) y Business Owner.

Figura 3: Esquema del equipo Scrum y flujo de comunicación



2.1.1. El Propietario del Producto o *Product Owner*

Esta persona tiene la responsabilidad de definir el trabajo y priorizar las tareas correspondientes en el proyecto. Es crucial que tenga un profundo conocimiento de los objetivos del proyecto, los del cliente, del mercado y de la organización en su conjunto. Este miembro guía al equipo a lo largo de todo el proceso de desarrollo del proyecto.

El product owner realiza esta misión a través del *product backlog*, que no es más que un listado de funcionalidades del producto ordenadas por prioridad. Por ejemplo, si estuviéramos desarrollando un producto de marketing, este rol es el usuario principal del sistema, el cual suele ser alguien del negocio (analista de negocio, gestor de producto, experto en digital, etc.) con un conocimiento sólido de los usuarios, el mercado, la competencia y las tendencias futuras del sistema que se está desarrollando.

¿Cuál es la diferencia entre el dueño del producto y un gerente de proyectos? La

principal diferencia radica en que el propietario del producto participa activamente durante todo el ciclo de vida del proyecto en lugar de simplemente definir el trabajo al comienzo mediante una declaración de alcance, como lo haría un gerente de proyecto tradicional. El propietario del producto debe revisar y cambiar las prioridades a medida que cambien las necesidades, y debe comunicarlo al equipo Scrum de manera oportuna.

A diferencia de un comité o grupo de personas, el dueño del producto es una única persona con responsabilidades específicas en el proyecto, que incluyen:

- Describir los elementos que conforman la lista del producto.
- Ordenar los elementos en la lista del producto para alcanzar los objetivos de la mejor manera posible.
- Maximizar el valor del trabajo realizado por el equipo de desarrollo.
- Garantizar que la lista del producto sea viable, transparente y clara para todos, para que se pueda visualizar con claridad el trabajo que el equipo abordará a continuación.
- Asegurar que el equipo de desarrollo comprende los elementos de la lista del producto a un nivel adecuado.

Algo curioso es que el PO debe hacer de protector del equipo de desarrollo; esto consiste en que managers de otras áreas no puedan pedirles tareas directamente sin pasar por su filtro. La razón de ser del protector no es restar autonomía al equipo de desarrollo, sino evitar que se le soliciten tareas que no sean prioritarias al producto o que les hagan perder el foco del *sprint*(iteración).

Al mismo tiempo, un buen PO será aquel encargado de maximizar y optimizar el producto. Además, como se ha mencionado, es el que se encarga de que la visión del producto esté clara para todo el equipo Scrum y de esta forma lograr mantener el foco y entregar la mejor versión de la iteración. Entre sus aspectos clave podríamos mencionar:

- Validar los incrementos frecuentemente con los interesados del proyecto y el cliente en particular y, así, recibir información en tiempo real para ajustar las debidas modificaciones.
- El product owner debe estar siempre enfocado en priorizar las funcionalidades más valiosas; sin embargo, no debe dejar a un lado las funcionalidades de baja prioridad ya que estas pueden terminar desmotivando al equipo o a cierta parte de los interesados.

- El product owner tiene la última palabra en la priorización de las funcionalidades y las interdependencias entre ellas; sin embargo, este debe escuchar las recomendaciones del equipo Scrum para obtener el mejor resultado.

2.1.2. El *Scrum Master*

El rol del Scrum master es garantizar que el equipo Scrum trabaje de acuerdo con los principios, prácticas y reglas de la metodología, y que estos se comprendan y adopten completamente.

El Scrum master es un líder que trabaja para el beneficio de todo el proyecto, un experto que guía al equipo y asegura que el dueño del producto y el equipo de desarrollo trabajen dentro del marco de Scrum.

¿Cómo crees que consigue que el equipo pueda hacer el trabajo de la forma más eficiente posible?

- Eliminando los impedimentos que nos podamos encontrar durante el proyecto.
- Facilitando las reuniones de Scrum para que sean productivas y se consigan sus objetivos.
- Enseñando al equipo a autogestionarse, es decir, guiando al equipo con preguntas para que descubra por sí mismo una solución.
- Trabajando con el product owner para asegurar que el product backlog está en buena forma y preparado para el siguiente sprint.

El Scrum master es responsable de diferentes servicios y responsabilidades que varían según los otros dos roles principales en el proyecto, es decir, el dueño del producto y el equipo de desarrollo. A continuación, se presentan algunas de las tareas que se espera que la persona con este rol desempeñe:

- Entender la planificación del producto en un entorno empírico y encontrar las técnicas necesarias para gestionar efectivamente la lista del producto. El Scrum master se debe asegurar de que todos los miembros implicados conocen el marco de trabajo.
- Guiar al equipo de desarrollo para que este se autoorganice, sea multifuncional y elimine los impedimentos que pueden ocurrir en el proceso. Este es un proceso más profundo que va más allá de conocer las técnicas y herramientas de Scrum.

- Asegurar que el dueño del producto sepa cómo priorizar la lista del producto y que el equipo entienda la necesidad de contar con elementos claros y concisos para el trabajo. Esta posición intermedia entre el dueño del producto y el equipo de desarrollo precisa una gestión más horizontal, al servicio de todo el equipo.
- Facilitar los eventos de Scrum que sean requeridos.
- Planificar la implementación de Scrum en la organización, además de liderar y guiarla en el proceso de adopción del marco de trabajo.
- Eliminar los impedimentos que pueden ocurrir en el proceso. Algunos de estos podrían ser: pérdida de miembros del equipo, cambios imprevistos en la composición del equipo, dudas técnicas, falta de formación y habilidades, problemas con las herramientas de desarrollo, presión por parte de la dirección de la empresa, conflicto entre miembros del equipo, reuniones sin importancia a las que el DT debe acudir, etc.

2.1.3. El *Development Team*

El equipo de desarrollo está compuesto por profesionales encargados de crear y entregar un producto finalizado, que potencialmente se puede poner en producción, al final de cada sprint. Solo los miembros del equipo de desarrollo participan en la creación del producto.

Los equipos de desarrollo son estructurados por la organización para coordinar y gestionar su propio trabajo, optimizando así la eficiencia y efectividad de su trabajo en conjunto.

El equipo de desarrollo es responsable de llevar a cabo el trabajo real del proyecto, llevándolo más allá de la planificación propuesta y convirtiéndolo en algo tangible. Cada miembro del equipo tiene habilidades específicas que, junto con las de otros miembros, se combinan para abordar cualquier actividad que se presente.

El equipo trabaja de manera colaborativa, actúa colectivamente y se esfuerza por descubrir cómo lograr sus objetivos. Si bien el dueño del producto establece las prioridades y el proceso de Scrum guía el trabajo, y es supervisado por el Scrum master, todas las demás responsabilidades son compartidas por el equipo de desarrollo.

Si abordamos el tamaño de un equipo Scrum, el tamaño óptimo del equipo de desarrollo es aquel que es lo suficientemente pequeño para mantener la agilidad,

pero lo suficientemente grande para completar una cantidad significativa de trabajo. Si el equipo de desarrollo tiene menos de tres miembros, puede haber limitaciones en cuanto a las habilidades necesarias para un sprint, lo que puede retrasar la entrega de un incremento que se pueda poner en producción. Además, tener un equipo demasiado pequeño puede reducir la interacción y producir ganancias de productividad limitadas.

- El tamaño del equipo de desarrollo debe estar entre tres y nueve personas, sin tomar en consideración al product owner ni al Scrum master.
- Si se da el caso de requerir que el equipo sea mayor a nueve personas, se debe reconsiderar la estrategia para dividir en más equipos.

Por otro lado, tener más de nueve miembros en el equipo puede requerir demasiada coordinación. Los equipos de desarrollo grandes pueden generar demasiada complejidad como para ser gestionados adecuadamente mediante un proceso empírico. El tamaño del equipo no se define solo en función de los roles de dueño del producto y Scrum master, a menos que también estén contribuyendo a trabajar en la lista de pendientes del sprint.

Al mismo tiempo, podríamos decir que los equipos de desarrollo presentan las siguientes características:

- **Son auto-organizados:** El equipo de desarrollo es completamente autónomo. Ni el Scrum master indica al equipo cómo convertir los elementos de la lista del producto en incrementos de funcionalidad que potencialmente se puedan desplegar. El equipo es responsable de llevar a cabo su trabajo y de cómo lo realizan.
- **Todos los miembros se consideran en la misma posición:** Dentro del equipo de desarrollo, todos los miembros son desarrolladores y no se les asigna otra función o título, independientemente del trabajo que realicen. No se permiten subequipos, incluso si se necesitan considerar dominios particulares como pruebas o análisis de negocio. Esta regla no tiene excepciones.
- **Son multifuncionales:** El equipo dispone de las habilidades adecuadas sin depender de agentes externos, para proporcionar el incremento de producto esperado.
- **La responsabilidad se comparte:** Si bien cada miembro del equipo de desarrollo puede tener habilidades especializadas y desenvolverse mejor en algunas áreas, la responsabilidad final recae en el equipo en su conjunto.

2.1.4. Roles auxiliares en Scrum

Los roles auxiliares en los equipos Scrum son aquellos que no tienen un rol formal y no se involucran frecuentemente en el proceso; sin embargo, deben ser tomados en cuenta. En la gestión ágil de proyectos, es importante comunicar a los expertos del negocio y otros interesados, que se denominan stakeholders, cuyo aporte es la retroalimentación con respecto a la salida del proceso a fin de revisar y planear cada sprint.

Se podría decir que los roles auxiliares son dos:

- **Stakeholders:** Son las personas que hacen posible el proyecto y para quienes este producirá el beneficio acordado que justifica su desarrollo. Su participación directa solo es necesaria durante las revisiones del sprint. Son los clientes, proveedores y vendedores.
- **Administradores:** También se les dice managers, y son los responsables de establecer el entorno para el desarrollo del proyecto.

2.1.5. La dinámica de un equipo Scrum

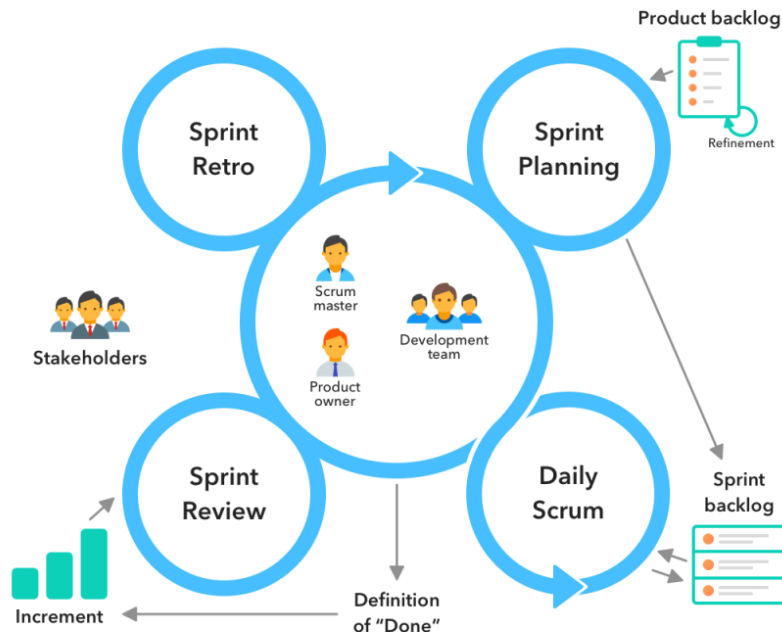
La dinámica del equipo Scrum se puede equiparar con la dinámica que se mantiene en un equipo de rugby, más específicamente con una jugada llamada «melé».

La melé es una estrategia en la que los jugadores de ambos equipos se agrupan en una formación en la cual lucharán por obtener el balón que se introduce por el centro, es esa escena clásica que vemos en la mayoría de películas y series de televisión.

A simple vista, esta jugada no parece ameritar mucha técnica o coordinación, pues es difícil apreciar el esfuerzo cuando se es un simple espectador, pero los jugadores de ambos equipos sí reconocen su complejidad: si un miembro del equipo se viene abajo, se cae toda la melé; en consecuencia, los jugadores deben estar bien coordinados, apoyarse en sus compañeros para empujar al mismo tiempo y, con ello, avanzar a la misma velocidad.

Esto mismo sucede con Scrum, cada uno de los roles que mencionamos anteriormente forma parte de la melé, y pone su trabajo al servicio del resto del equipo, que sería el proyecto, para obtener el mismo objetivo, que en el rugby es el balón, pero en la gestión de proyectos son los beneficios que se obtienen.

Figura 4: Workflow en Scrum



Tarea: vamos a formar grupos de 3-4 personas. Esta actividad consiste en que nos vamos a poner en la piel de un PO y un SM. Lo primero que tenéis que hacer es pensar en alguna idea que pueda desarrollarse utilizando Scrum. Por ejemplo, la construcción de una casa no sería una buena idea. Pero al mismo tiempo, tampoco es necesario que sea un desarrollo software... Piensa un poco antes de ponerte manos a la obra. Para obtener ideas puedes utilizar ChatGPT, no hay problema (sólo para obtener algunas ideas. Recuerda que en el examen no tendrás ningún tipo de ayuda). Cuando tengas la idea, en primer lugar te vas a poner en la piel del PO para crear un product backlog. Piensa en las funcionalidades o tareas que deberían llevarse a cabo. Recuerda que esta lista debe estar ordenada por prioridad, por lo que te tocará debatir con el resto de miembros del grupo. Una vez que tengáis dicha lista, ahora os vais a poner en la piel del SM para crear un *impediments backlog*, o lo que es lo mismo: una lista de impedimentos que podrían surgir a la hora de desarrollar dicho producto (si no recuerdas qué es, vuelve a leer la sección 2.1.2). Cuando todo el mundo haya terminado, compartiremos con el resto de compañeros de clase el trabajo realizado.

3. Los eventos de Scrum

Cada vez más, las empresas están adquiriendo marcos de trabajo *agile* para acelerar ciertos proyectos que, por sus características, incertidumbre y necesidad de adaptación a entornos cambiantes, se desarrollan mejor con estas metodologías.

Las características principales de estos marcos ya las hemos introducido, y como bien sabes son en relación con sus ciclos de vida más cortos e iterativos, frente a un ciclo de vida tradicional o en cascada.

Dentro de los marcos de trabajo ágiles, en Scrum existen tres roles fundamentales: product owner, scrum master y equipo de desarrolladores. Estos, para su correcto funcionamiento, trabajan dentro de un conjunto de eventos, que le dan la estructura y secuencia en los procesos a todo el desarrollo del scrum.

En el desarrollo de esta unidad profundizaremos en los aspectos fundamentales y en propósitos de cada uno de los eventos del marco de trabajo:

- Sprint.
- Sprint review.
- Sprint retrospective.
- Daily sprint.
- Sprint planning.

3.1. Sprint

El sprint es la base del scrum, se lleva a cabo en periodos de tiempo relativamente cortos, aproximadamente de un mes como máximo, y un promedio de tres o dos semanas. Este es el momento de crear un incremento de producto, utilizable y potencialmente liberable. Lo ideal es que cada sprint siempre tenga la misma duración.

Figura 5: Elementos en Scrum

Roles	Artifacts	Events
<ul style="list-style-type: none">• Product owner• Development team• Scrum master	<ul style="list-style-type: none">• Increment• Product backlog• Sprint backlog	<ul style="list-style-type: none">• Sprint planning• Sprint review• Sprint retrospective• Daily scrum

Al igual que un proyecto en general, el sprint se utiliza para lograr algo; por lo tanto, tiene una definición de lo que se va a desarrollar, un diseño y un plan flexible que guiarán su construcción, el trabajo y el producto resultante. Una vez que se han alcanzado los objetivos, un nuevo sprint comenzará.

Durante la realización del sprint se debe tener en cuenta lo siguiente:

- No es correcto realizar modificaciones que pongan en peligro el objetivo que se ha de conseguir.
- No se pueden disminuir los objetivos de calidad.
- El alcance puede ser renegociado con el dueño del producto y el equipo de desarrollo a medida que se descubren nuevas condiciones, pero el tiempo máximo del sprint no es negociable.

3.1.1. La cancelación de un sprint

Un sprint puede ser cancelado antes de que el bloque de tiempo pautado llegue a su fin, pero solo el dueño del producto tiene la autoridad para cancelarlo. Los interesados, el equipo de desarrollo o del scrum master pueden influenciar sobre el dueño del producto, pero la decisión final está totalmente bajo su juicio. A modo de curiosidad, si un sprint se cancela, las funcionalidades desarrolladas que estén terminadas (done) pueden ser aceptadas por el product owner y ser puestas a disposición del cliente.

Lo más común es que un sprint se cancele si este queda obsoleto antes de completarse, si la compañía cambia de dirección y enfoque, o si cambian las condiciones del mercado y la tecnología, lo cual no es un motivo de sorpresa en un mundo tan dinámico y competitivo como el actual.

Entre los aspectos más relevantes en la cancelación del sprint se pueden destacar los siguientes:

- Los elementos terminados (done) pueden ser puestos a disposición del cliente y el resto de ellos volverá a la pila del producto.
- Los elementos que no se han podido completar deben volver a la pila del producto, donde se evaluará su prioridad o se descartarán de no ser requeridos para los objetivos del producto.
- Si un sprint es cancelado, se debe poner a disposición del cliente aquello que se encuentre terminado (done) e iniciar inmediatamente el nuevo sprint. Las cancelaciones de sprint consumen recursos, ya que todos deben

reagruparse en otra reunión de planificación para empezar otro proceso. Además, generan malestar en el equipo scrum en general.

3.2. Sprint planning

El objetivo del sprint planning es definir lo que se va a hacer en el sprint y cómo se va a hacer. Su time-box es de ocho horas como máximo para sprints de un mes; si la duración del sprint es menor, por ejemplo, dos semanas, la reunión de planificación también se reduce.

A modo de ejemplo, un sprint planning meeting para un sprint de duración de dos semanas puede ser realizado incluso entre 60 y 45 minutos. Esto se consigue si las funcionalidades están bien definidas y son priorizadas para que puedan ser estimadas fácilmente por el equipo scrum.

Participan todos los miembros del equipo scrum, pero es el scrum master quien se asegura de que el evento tenga lugar y los asistentes entiendan su propósito, así como de que la reunión no exceda el tiempo de duración.

La reunión de planificación responde a las siguientes preguntas:

- ¿Qué puede entregarse en el incremento resultante del sprint que comienza?
- ¿Cómo se conseguirá hacer el trabajo necesario para entregar el incremento?

3.2.1. Origen del contenido que realizar en el sprint

Aunque el desarrollo de los backlogs pertenecientes al framework de scrum se tratará con detalle en las siguientes secciones, sí es interesante ubicarlos para entender mejor la finalidad del sprint planning meeting y el origen de la actividad que se puede entregar en un sprint.

En la presente nos centraremos en los artefactos referentes al marco de trabajo scrum, compuesto por los siguientes:

- Product backlog.
- Sprint backlog.

Tarea: busca en Internet qué herramientas podríamos utilizar para hacer una buena gestión de product y sprint backlog.

Product backlog

El product backlog no es más que un listado de funcionalidades priorizadas, que

contienen pequeñas descripciones de todas las funcionalidades deseadas en el producto. La persona que prioriza y trabaja en el product backlog es el product owner, aunque, típicamente, un equipo scrum y el product owner son los encargados de identificar funcionalidades que pueden ser parte del producto. Luego, el product owner es el encargado de priorizarlas acorde con lo que los clientes están demandando del producto.

Cuando se empieza con un producto desde cero, una vez que el equipo scrum y el product owner han identificado funcionalidades y el product owner las ha priorizado, ya es casi seguro que tendremos más funcionalidades de las que van a caber en el primer sprint, por lo que el equipo scrum ya puede empezar a trabajar.

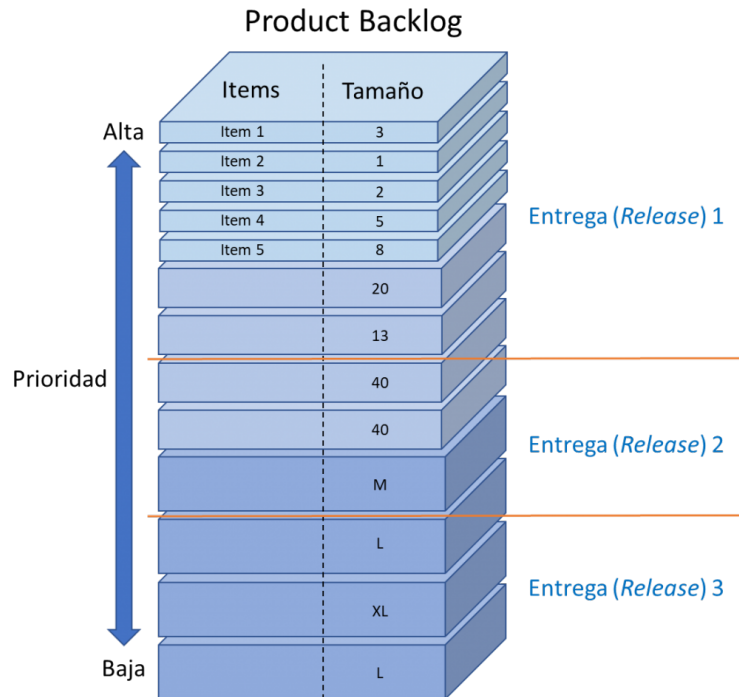
Desde este punto en adelante, el product backlog estará siempre vivo, y crece y cambia a lo largo de la vida del proyecto y a medida que recibimos feedback de la funcionalidad recibida por parte de los clientes.

El product backlog suele estar formado por los siguientes tipos de elementos:

- **Funcionalidades:** son propiamente las historias de usuario.
- **Bugs:** tanto los reportados por el cliente como los encontrados por el equipo de calidad una vez que una funcionalidad se encuentra en producción.
- **Épicas:** las épicas son historias de usuario muy grandes que no pueden finalizarse en un sprint, por lo que necesitan que se descompongan en otras historias de usuario con un tamaño mucho más adecuado para ser gestionadas con los principios y técnicas ágiles: estimación y seguimiento cercano.
- **Trabajo de investigación o adquisición de conocimientos:** muchas veces, para realizar una historia de usuario (o varias), hay que tener un profundo conocimiento de a qué nos enfrentemos, y scrum aconseja valorar el trabajo que vas a realizar en materia de estudio/investigación.

Otra de las claves de las funcionalidades del product backlog es que las de máxima prioridad (las que más arriba están de todo en el backlog, por tanto, las que entrarán en el siguiente sprint) tienen que estar estimadas, es decir, que deben tener una valoración sobre el peso o el tamaño que tiene esa historia de usuario.

Figura 6: Estructura de un product backlog



Tarea: investiga un poco más acerca de qué son las historias de usuario. Una vez lo tengas claro, forma grupos de 2-3 personas y haz un listado de historia de usuario sobre algún proyecto concreto. Para finalizar, busca información sobre qué son los puntos de historia.

Sprint backlog

El sprint backlog es una lista de tareas identificadas por el equipo para que se completen durante el sprint. Durante la reunión del sprint planning, el equipo selecciona un número de elementos del backlog e identifica las tareas necesarias para completar cada historia de usuario. La mayoría de los equipos suelen estimar también el número de horas que cada tarea requerirá a un miembro del equipo para completarla.

Es fundamental que el equipo seleccione los elementos y estime un buen sprint backlog, ya que sus miembros son los que se comprometen a completar las tareas y demostrar esas funcionalidades al product owner al final del sprint.

Tarea: llegados a este punto te recomiendo que hagas una tabla del tipo product backlog vs. sprint backlog donde veas todas las diferencias entre ambos conceptos.

3.2.2. ¿Qué puede entregarse en este sprint?

En esta sección de la planificación se deben definir y acordar aquellos elementos de la pila del producto (product backlog) que dentro de un sprint se pueden completar o llegar a su fase done.

Durante el sprint se supone que los miembros del equipo deben actualizar el sprint backlog con la nueva información disponible, y esto se suele hacer una vez al día, de forma que esta información esté disponible para el daily meeting y para poder hablar de ello. De esta forma, el scrum master puede verificar con estos datos el progreso que el equipo lleva de cada tarea de esa historia de usuario.

Recuerda que los equipos scrum se suelen demorar hasta cinco sprints en lograr que sus planificaciones se acerquen a lo que se entrega al final de este. Esto quiere decir que los primeros sprints serán de reconocimiento, para afinar detalles y llegar al rendimiento óptimo del equipo.

Al mismo tiempo, algunos aspectos relevantes que se han de considerar durante la planificación del sprint son los siguientes:

- Para seleccionar las funcionalidades que se van a desarrollar se deben considerar el tamaño de los elementos y su prioridad. Además, se debe entender la cantidad de días laborables que existen dentro del sprint en conjunto al rendimiento del equipo en los incrementos pasados.
- La planificación es uno de los eventos fundamentales del sprint y debe hacerse tomando en consideración los datos del rendimiento real, «el día a día del proyecto». Es un error intentar planificar basándose en condiciones ideales.

3.2.3. ¿Cómo se puede entregar el trabajo seleccionado?

Esta pregunta se responde una vez se ha realizado la primera parte de la planificación; el equipo de desarrollo establece cómo desarrollará los elementos que han sido seleccionados para el sprint.

Para esto se tienen en consideración los siguientes aspectos:

- El equipo debe descomponer en unidades pequeñas las funcionalidades; el objetivo de esto es que se puedan manejar actividades con duración de un día o menos y, así, monitorear el avance de sprint.
- La labor de cómo desarrollar el sprint y todas las funcionalidades seleccionadas.

nadas corresponde al equipo de desarrollo; este, de forma autoorganizada, elabora el camino que se ha de seguir.

- El product owner es el responsable de transmitir al equipo de desarrollo lo que se espera de cada uno de los elementos que desarrollarán en el sprint. De esta forma, el equipo puede entender cuánto se debe cubrir y hasta qué punto son capaces de desarrollarlo en solo incremento.
- Si el equipo de desarrollo detecta, al desglosar las actividades, que no puede cubrir todos los elementos seleccionados, es un buen momento para eliminar las funcionalidades que puedan esperar y elaborar una planificación más real.

3.2.4. Objetivo del sprint o *sprint goal*

El objetivo del sprint es una meta establecida que puede ser alcanzada mediante la implementación de la lista de producto. El sprint goal, como también se le dice, es creado durante la reunión de planificación del sprint, y ofrece al equipo de desarrollo cierta flexibilidad con respecto a la funcionalidad implementada en el proceso de creación.

A medida que el equipo de desarrollo trabaja, se mantiene el objetivo del sprint en mente, y con el fin de satisfacerlo se implementan la funcionalidad y la tecnología.

Algunos aspectos que se deben considerar para el sprint goal son los siguientes:

- Se debe tener un objetivo claramente definido que esté sobre todas las tareas y que, además, englobe las funcionalidades que se van a desarrollar durante el sprint.
- El objetivo debe ser el centro del trabajo y foco de todo el equipo; para lograr esto se debe evitar dispersar tareas que no contribuyen a su consecución.
- Si en determinado caso, durante el desarrollo del sprint, el equipo de desarrollo determina que no es capaz de alcanzar el objetivo, lo correcto es renegociar el sprint backlog con el product owner y alcanzar una solución beneficiosa.

Por norma general, el sprint goal se utiliza para informar a los stakeholders sobre el progreso del sprint, sobre lo que está trabajando el equipo, ya que muchas veces los clientes quieren saber o incluso necesitan escuchar en detalle lo que el equipo está trabajando.

3.3. Daily meeting

El scrum diario o daily scrum es una reunión con un bloque de tiempo de quince minutos para que el equipo de desarrollo sincronice sus actividades y cree un plan para las siguientes 24 horas, partiendo de una inspección del trabajo realizado desde el último scrum diario y haciendo una proyección acerca del trabajo que podría completarse antes del siguiente.

El scrum diario se realiza a la misma hora y en el mismo lugar todos los días para reducir la complejidad. Durante la reunión, cada miembro del equipo de desarrollo da respuesta a las siguientes preguntas:

- ¿Qué hice ayer que ayudó al equipo de desarrollo a lograr el objetivo del sprint?
- ¿Qué haré hoy para ayudar al equipo de desarrollo a lograr el objetivo del sprint?
- ¿Veo algún impedimento que evite que el equipo de desarrollo o yo logremos el objetivo del sprint?

El equipo de desarrollo usa el scrum diario para medir y evaluar el progreso hacia el objetivo del sprint y para evaluar qué tendencia sigue este progreso hacia la finalización del trabajo contenido en la lista.

Esta es la forma en que se optimizan las posibilidades de que el equipo de desarrollo cumpla con los objetivos propuestos. El scrum master se asegura de que el equipo de desarrollo mantenga la reunión, pero solo el equipo es responsable de dirigir el scrum diario.

Un dato curioso es que, a veces, hay situaciones en las que, bien sea por el volumen de trabajo, bien por la complejidad o por alguna situación de estrés, los equipos eligen hacer dos daily meetings (una al inicio de la mañana y otra al inicio de la tarde). Esto tiene un impacto sumamente positivo, ya que ayuda a los miembros del equipo a mantenerse en sincronía.

El scrum diario mejora la comunicación, elimina la necesidad de mantener otras reuniones, identifica y suprime impedimentos relativos al desarrollo, resalta y promueve la toma de decisiones rápidas, y mejora el nivel de conocimiento del equipo de desarrollo.

Esta daily o reunión diaria de sincronización del equipo es una reunión fundamental para facilitar la transferencia de información y la colaboración entre los miembros del equipo con el fin de aumentar su productividad. Cada miembro

del equipo puede echar un vistazo a las tareas que están realizando los demás, para que, de esta forma, al final de la reunión, se puedan hacer las adaptaciones necesarias y, así, cumplir con el compromiso que el equipo adquirió al principio del sprint de presentar todo ese trabajo al cliente.

Por lo tanto, la reunión daily es una de las reuniones más importantes para un equipo scrum; en ella se comprenden las tareas en las que están trabajando

3.4. Sprint review

Esta revisión del sprint (también llamada reunión del refinamiento del backlog o backlog refinement meeting) es la reunión que ayuda a preparar el product backlog para la siguiente reunión de sprint planning. A esta reunión deben acudir tanto el equipo como el scrum master y el product owner

Recuerda que lo más importante que se obtiene de la revisión del sprint es la retroalimentación que se recibe, ya que esto permite evaluar las prioridades ya fijadas para el backlog.

Al final del sprint se lleva a cabo una revisión para inspeccionar el incremento y adaptar la lista de producto si fuese necesario. Durante esta etapa, el equipo scrum y los interesados colaboran.

La revisión consiste en una reunión informal, no una reunión de seguimiento, y la presentación del incremento tiene como objetivo facilitar la retroalimentación. También tiene un bloque de tiempo estimado que no excede de las cuatro horas para sprints de un mes; si son más cortos, el tiempo de la reunión también se reduce.

En la revisión se hace lo siguiente:

- Incluir nuevas historias de usuario de acuerdo con las nuevas necesidades del cliente.
- Eliminar historias de usuario que ya no parecen relevantes.
- Reevaluar las prioridades de las historias de usuario.
- Asignar estimación a las historias de usuarios que necesitan ser apropiadamente priorizadas.
- Corregir estimaciones, ya que se posee nueva información.
- Dividir épicas en historias de usuario.

En resumen, la intención principal de esta reunión es asegurar que el backlog se mantiene poblado con elementos que son relevantes, detallados y estimados con un grado apropiado a su prioridad, y de acuerdo con el conocimiento actual del proyecto y sus objetivos.

3.5. Sprint retrospective

La retrospective (retrospectiva del sprint) es una reunión liderada por el scrum master en la que el equipo discute el sprint que acaba de concluir y en la que se intenta determinar lo que se podría cambiar para que el siguiente sprint fuese más productivo. En la revisión del sprint se mira lo que el equipo está construyendo, mientras que las retrospectivas ponen más énfasis en cómo lo están construyendo.

Por lo tanto, la retrospectiva del sprint tiene lugar después de la revisión y antes de la siguiente reunión de planificación. Se trata de una reunión restringida a un bloque de tiempo de horas para sprints de un mes.

Cualquier retrospectiva debería dar respuesta a tres puntos básicos de discusión:

- ¿Qué funcionó bien durante el sprint?
- ¿Qué salió mal durante el sprint?
- ¿Qué podríamos hacer de forma diferente para mejorar?

De esta forma, el equipo tiene un mecanismo que le permite evolucionar y mejorar de manera continua durante toda la vida del proyecto. Es muy importante que todos los roles tengan la oportunidad de expresar sus opiniones en un ambiente abierto, honesto y constructivo. Por supuesto, esta reunión también ayuda a obtener feedback del equipo sobre el trabajo y el progreso del equipo.

Los elementos clave de la retrospectiva son los siguientes:

- Se hace una revisión del sprint finalizado con el objetivo de evaluar a las personas, las relaciones y las herramientas utilizadas.
- Se revisan aquellos eventos que han salido bien y se identifican las oportunidades de mejora; es una buena estrategia que permite visualizar las fallas dentro de los logros alcanzados.
- No solo se deben identificar las mejoras, sino que se debe crear un plan para implementarlas al equipo y que rindan frutos en los futuros sprints.
- Las mejoras de los procesos se hacen al final de cada sprint, de esta manera se asegura que el equipo siempre realiza mejoras en la forma en la que

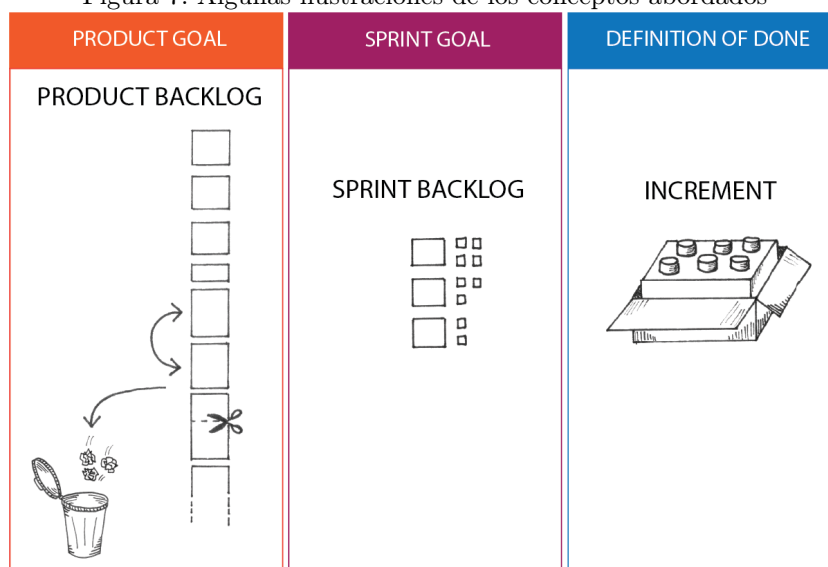
funciona.

- Todos los roles toman parte en esta reunión.
- Todos los miembros identifican qué salió bien y lo que podría mejorarse.
- Se discute sobre el proceso que han seguido y dan sugerencias para mejorarlo.
- Se discute sobre otras ideas que podrían mejorar su productividad.
- El scrum master prioriza acciones y lecciones aprendidas basadas en la dirección del equipo.

En este evento, el scrum master juega un papel muy importante, pues es quien alienta al equipo para que mejore sus prácticas, a fin de hacerlas más efectivas y amenas para el siguiente sprint.

Durante cada retrospectiva, el equipo scrum debe planificar las formas de aumentar la calidad del producto mediante la adaptación de la definición de «terminado».

Figura 7: Algunas ilustraciones de los conceptos abordados



4. Artefactos de Scrum

Dentro de las metodologías ágiles se utiliza un término que no es tan común pero cuya aplicación realmente es bastante sencilla de entender: artefactos. Estos no

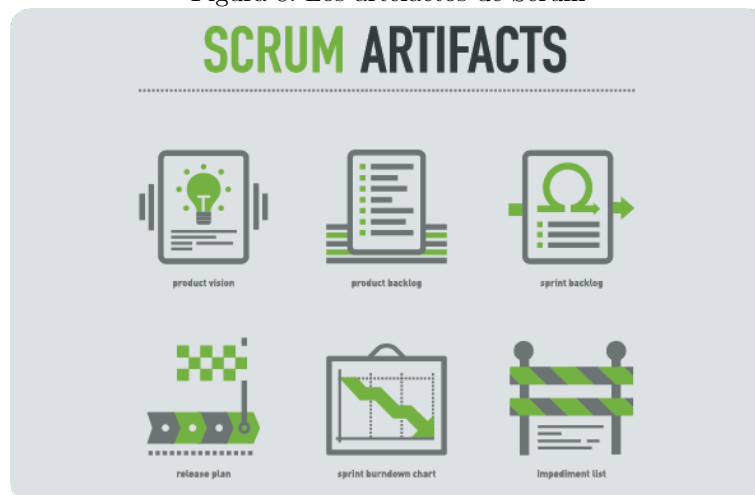
son más que la representación del trabajo, de distintas formas que fortalecen los pilares de transparencia, inspección y adaptación.

En la presente sección nos centraremos en los artefactos referentes al marco de trabajo Scrum, compuesto por los siguientes:

- Product backlog.
- Sprint backlog.
- Impediment backlog.
- Incremento.

Además, se desarrollarán temas de gran importancia como lo pueden ser las herramientas utilizadas en las métricas, la definición de done y algunos artefactos empleados en otros marcos de trabajo de metodologías ágiles

Figura 8: Los artefactos de Scrum



4.1. Los artefactos de Scrum

Los artefactos definidos por Scrum están diseñados específicamente para maximizar la transparencia de la información clave, que es necesaria para asegurar que todos tengan el mismo entendimiento.

4.1.1. Product backlog

A continuación, veremos todos los backlogs que pertenecen al framework de scrum. Recordad que la lista de producto es una lista ordenada y detallada que

describe todo lo que podría ser necesario en el producto, y es la única fuente de requisitos para cualquier cambio que haya de realizarse.

Empezando por el primero, el product backlog no es más que un listado de funcionalidades priorizadas, que contienen pequeñas descripciones de todas las funcionalidades deseadas en el producto. La persona que prioriza y trabaja en el product backlog es el product owner, aunque, típicamente, un equipo scrum y el product owner son los encargados de identificar funcionalidades que pueden ser parte del producto. Luego, el product owner es el encargado de priorizarlas acorde con lo que los clientes están demandando del producto.

Cuando se empieza con un producto desde cero, una vez que el equipo scrum y el product owner han identificado funcionalidades y el product owner las ha priorizado, ya es casi seguro que tendremos más funcionalidades de las que van a caber en el primer sprint, por lo que el equipo scrum ya puede empezar a trabajar.

Desde este punto en adelante, el product backlog estará siempre vivo, y crece y cambia a lo largo de la vida del proyecto y a medida que recibimos feedback de la funcionalidad recibida por parte de los clientes.

Entre las características asociadas a la lista del producto o product backlog se encuentran:

- La persona a cargo de la lista del producto (product backlog) es el dueño del producto, es quien se encarga de su contenido, orden y disponibilidad.
- El product backlog nunca está completo, en él figuran en la parte superior aquellos elementos que se encuentran definidos y tienen prioridad de desarrollo. En la parte inferior se agregan los elementos que se van definiendo o conociendo a medida que avanza el proyecto.
- El product backlog es un artefacto vivo que evoluciona constantemente durante las iteraciones del producto y solo dejará de cambiar cuando no se desarrolle más el producto.
- Los elementos del product backlog, a medida que se acercan al tope de la pila (cuando ya van a ser desarrollados), deben tener definidos: descripción, orden, estimación, valor, criterio de *done*.

Refinamiento del product backlog

El refinamiento del product backlog es el proceso mediante el cual se detallan y organizan los elementos que componen el product backlog. Es un proceso que se

realiza entre el product owner y el equipo de desarrollo, y se lleva a cabo sobre aquellos elementos que están próximos a desarrollarse.

Entre las razones por las que se debe realizar el refinamiento figuran:

- Mediante estos procesos se consolidan las características mencionadas anteriormente de los elementos del product backlog: descripción, orden, estimación, valor y criterio de *done*.
- Los elementos que se encuentran en la parte superior del product backlog (próximos a ser realizados o de alta prioridad) son aquellos que se encuentran más refinados. Esto permite que esos elementos puedan pasar a un estado de done dentro del plazo del sprint.

Pilas (elementos del product backlog)

A los elementos del product backlog se les denomina también «pilas». Este es un concepto asociado a las pilas informáticas, en el cual los elementos se desapilan por la parte superior y son estos los de mayor prioridad.

Estas pilas serán las que pasarán por el proceso de refinado según las características que describimos en el punto anterior.

- Siguiente sprint: completamente refinado.
- Segundo sprint arriba: bien o medianamente refinado.
- Tercer sprint arriba: casi sin refinar.

Esto se debe a que el refinado es un proceso que solo debe consumir el 10 % de la capacidad de trabajo del equipo de proyecto y, además, que las características cambiantes suponen un retrabajo en los elementos.

4.1.2. Ejemplo de aproximación

Supongamos que un cliente español desea usar una aplicación cuyo nicho de mercado es el americano. Desafortunadamente, esta aplicación solo la tenemos disponible en inglés y nuestro usuario español, como cliente, nos propone que incluyamos el español en la aplicación.

Cuando creemos la historia de usuario para incluir esta nueva funcionalidad (nuevo idioma español en la aplicación) en nuestro product backlog, lo haremos más o menos de la siguiente forma:

Como un usuario español, necesito que el producto esté en idioma español de forma que los usuarios puedan entender en todo momento la aplicación que están usando.

Una vez se ha escrito el título de la historia de usuario, nuestra misión es añadir lo que anteriormente se llamaba condiciones de satisfacción y que hoy en día se llama criterios de aceptación (acceptance criteria).

En la descripción de la historia de usuario, nuestra misión es incluir todos los criterios de aceptación u objetivos mínimos que esa funcionalidad debe cumplir y, por lo tanto, es fundamental que sean redactados de manera clara, concisa y sin subestimar su impacto, para, de esta forma, abarcar todos los distintos escenarios de negocio, desde el fundamental hasta las alternativas y las excepciones.

Criterios de aceptación (AC):

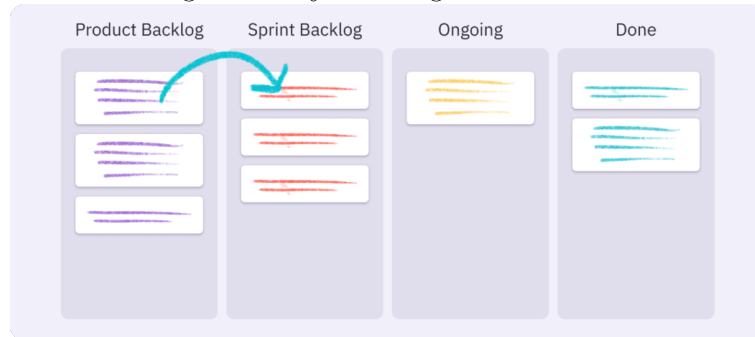
- En la página principal existe un desplegable donde se puede elegir el idioma que tiene la aplicación (inglés y español).
- Una vez que se seleccione un idioma, se cambiará automáticamente para toda la interfaz.
- Todas las cadenas tienen que estar traducidas al castellano.
- Todas las funcionalidades funcionarán a la perfección en castellano.

Posteriormente, dicha historia de usuario se divide en tareas, de forma que el equipo pueda ir cogiendo esas tareas desde el product backlog, llevarlas al sprint backlog, e ir las completando a medida que el sprint avanza.

Tareas a realizar:

- Diseñar la interfaz de usuario con los dos idiomas disponibles (por defecto, el inglés).
- Programar el cambio automático de un lenguaje al otro al elegir el idioma español.
- Verificar que todo el producto está en español una vez seleccionado el español de España.
- Realizar pruebas de regresión del producto al cambiar al idioma español.
- Automatizar las pruebas.
- Reunión con [quien sea] para hablar sobre [lo que sea].

Figura 9: Flujo entre algunos artefactos



4.1.3. Sprint backlog

El sprint backlog es una lista de tareas identificadas por el equipo para que se completen durante el sprint. Durante la reunión del sprint planning, el equipo selecciona un número de elementos del backlog (por norma general, historias de usuario) e identifica las tareas necesarias para completar cada historia de usuario. La mayoría de los equipos suelen estimar también el número de horas que cada tarea requerirá a un miembro del equipo para completarla.

Es fundamental que el equipo seleccione los elementos y estime un buen sprint backlog, ya que ellos son los que se comprometen a completar las tareas y demostrar esas funcionalidades al product owner al final del sprint.

Todo el trabajo que el equipo de desarrollo haya seleccionado para hacer durante el siguiente sprint pasa a esta lista de pendientes. Este artefacto es un elemento para visualizar el trabajo que se ha de realizar y está gestionado por los desarrolladores.

La lista de pendientes del sprint es un plan con un nivel de detalle suficiente como para que los cambios en el progreso se puedan entender en el Scrum diario.

Entre las características que destacan del sprint backlog tenemos:

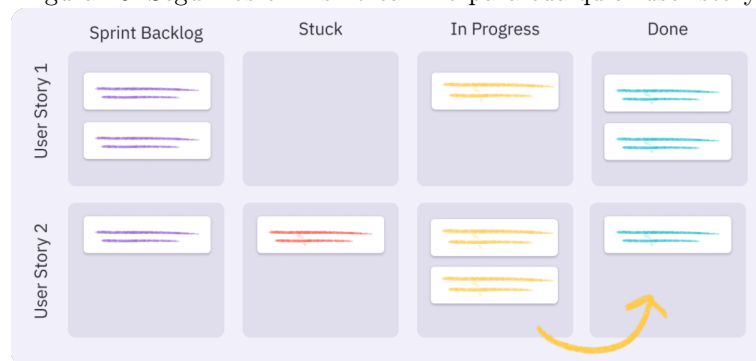
- Se podría denominar como el catálogo de elementos que se van a realizar en un sprint.
- La responsabilidad sobre él es completamente del equipo de desarrollo y ni el Scrum master ni el product owner pueden intervenir en su desarrollo (ni quitando, ni agregando elementos).
- Es el artefacto que permite visualizar las tareas o trabajos pendientes para conseguir alcanzar el objetivo del sprint.

- El equipo de desarrollo puede agregar tareas segundo considere necesario, también puede eliminar elementos que ya no considere que contribuyan al objetivo del sprint.
- En ocasiones los elementos que conforman el sprint backlog son desglosados en subactividades (sobre todo si es un elemento muy grande) para poder visualizar las variaciones del trabajo en las daily meetings.

Es común en la práctica que durante el desarrollo del sprint se deban adicionar nuevos elementos al sprint backlog. Para esto, el product owner suele hacer una petición la cual es negociada con el equipo de desarrollo, quien tendrá la última palabra sobre cómo se adicionará el trabajo. En ocasiones, para poder cubrir las metas del sprint se suele sustraer la misma carga de trabajo estimada de algún elemento que no sea de alta prioridad con el fin de introducir los nuevos elementos a la iteración.

Siempre se debe tener en cuenta el tiempo que genera analizar nuevos elementos que se están introduciendo en la iteración para poder ser desarrollados óptimamente.

Figura 10: Seguimos el mismo camino para cualquier user story



Tarea: investiga un poco acerca de qué herramientas o programas podríamos utilizar para visualizar el avance en las tareas del sprint. Es decir, ¿cómo podríamos crear tablero de *To-do*, *Doing* y *Done*?

4.1.4. Impediment backlog

Cuando se desarrolla un producto siempre existirá un flujo constante de retos que tienen el potencial de impedir el rendimiento correcto del equipo y la entrega temprana de software.

Estos desafíos deberían registrarse en el impediment backlog y el scrum master

debe asumir responsabilidades y trabajar para eliminar el impedimento de más alta prioridad tan pronto como sea posible, de forma que el equipo scrum sea capaz de trabajar centrado en finalizar el sprint.

Desafortunadamente, el impediment backlog se pasa por alto muy a menudo y no se le da la atención que necesita para que el equipo tenga éxito durante el sprint. Un buen scrum master debería hacer todo lo posible para asegurar que el equipo trabaja para conseguir el objetivo del sprint (sprint goal).

Una vez que el sprint se ha empezado, no existe nada más importante para el scrum master que eliminar los impedimentos del impediment backlog.

Como curiosidad, antes de que existieran las herramientas de gestión de proyectos de forma ágil (scrum) como Jira, todo se hacía de forma más manual: tarjetas o notas adhesivas, paredes llenas de post-it, etc.

4.1.5. Incremento

El incremento en pocas palabras es aquello desarrollado que se puede poner a disposición del cliente; es decir, aquello que le aporta valor al producto y que va aumentando de forma progresiva con la sucesión de los sprint.

El incremento es lo que se pone a la disposición de los clientes, este debe ser funcional (se puede utilizar) aun si el cliente decide no liberarlo.

Entre los aspectos más relevantes podemos enunciar:

- Las funcionalidades que son puestas a disposición del cliente deben ser inspeccionadas, ya que mediante esta inspección y de forma colaborativa se pueden tomar decisiones para los siguientes sprint.
- El product owner es quien tiene la última palabra sobre si se pone una funcionalidad a disposición del cliente, sin importar si la funcionalidad está done o no; es decir, el incremento debe estar en condiciones de utilizarse sin importar si el dueño de producto decide liberarlo o si decide lo contrario.

La definición de hecho (*done*)

La definición de hecho (done) es un concepto muy propio de cada equipo de Scrum, ya que son ellos quienes definirán las condiciones, parámetros y reglas explícitas para decidir cuándo algo está realmente terminado. Es a estos acuerdos a lo que se determina como el definition of done.

Estos acuerdos son clave para que cada miembro del equipo entienda que, una vez un elemento se declara done, no queda nada por hacer. Algunas de las

características clave para hacer una correcta definition of done son:

- Es correcto detallar en justa medida las reglas o acuerdos para el definition of done. Si estas se detallan en gran medida se podrán detectar más errores; sin embargo, habrá que tener en cuenta que requerirán más esfuerzo para completarse.
- Usualmente las organizaciones tienen listas de verificación para determinar si una funcionalidad está done. El equipo Scrum como mínimo deberá cubrir estos puntos, pero se encuentra en plena libertad de agregar alguno más.
- Si existen varios equipos de Scrum colaborando, se debe establecer una definition of done que sea común e integrada con todos los equipos.
- La definition of done evoluciona con la madurez de los equipos Scrum, se va ajustando y adquiriendo características más rigurosas para elevar los criterios de calidad.

4.2. Herramientas para la gestión

Como potencialmente miembros del equipo de desarrollo que seremos de aquí a unos meses (siempre que nos encontremos en un marco ágil), es necesario que tengamos conocimientos de software de gestión de este tipo de proyectos.

La mayoría de las herramientas de gestión de proyectos se centran en tres puntos comunes:

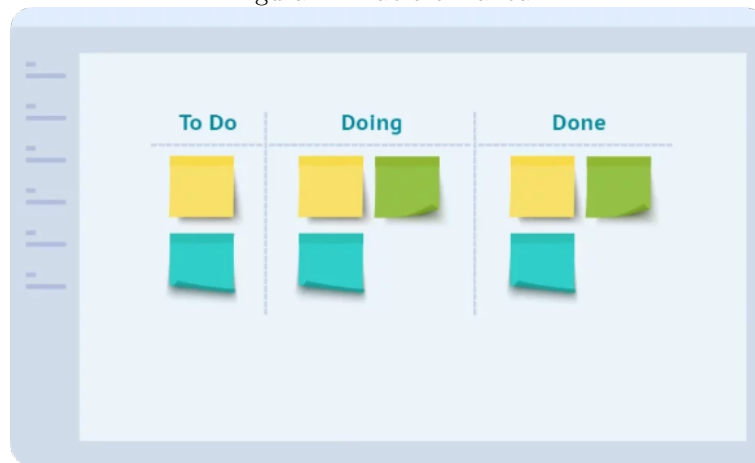
- **Métricas e informes de agilidad:** seguimiento y proyección del tiempo, informes de progreso fáciles de entender para los stakeholders, aseguramiento de la calidad y porcentaje de cumplimiento de las funcionalidades.
- **Comunicación:** comunica actualizaciones con los equipos locales y distribuidos, y comparte lista de tareas, retroalimentación y asignaciones de las tareas.
- **Evaluación del proyecto:** identificar y resolver los obstáculos del proyecto, evaluar el desempeño y valorar el tema financiero.

Estas herramientas, adicionalmente a las funcionalidades mencionadas de elaboración de métricas, comunicación y evaluación de la actividad, también suelen incluir recursos y métricas específicas adaptadas a la metodología como los tableros o los gráficos de burn-down charts.

4.2.1. Los tableros

Son recursos que permiten tener los artefactos al alcance de todos los miembros del equipo Scrum; se les denomina «radiadores de información», ya que permiten visualizar en un mismo sitio todos los trabajos que se están realizando en el sprint

Figura 11: Tablero Kanban

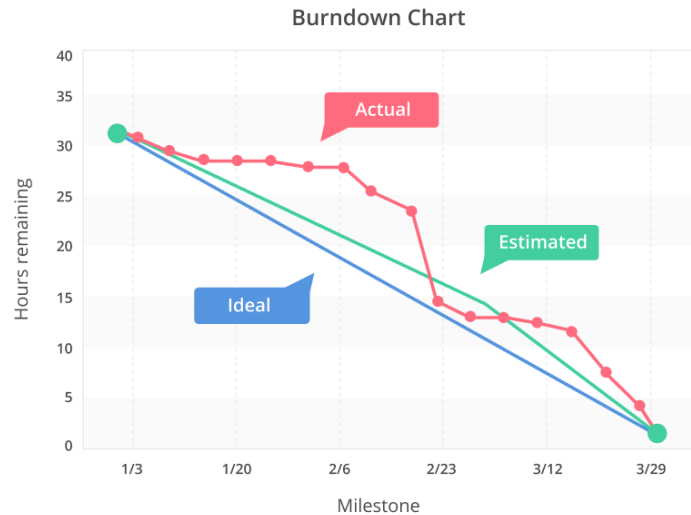


4.2.2. Burn-down chart

Existen varias métricas para medir el rendimiento de los equipos a lo largo de las iteraciones y determinar posibles fallas en el desarrollo o en la estimación. Particularmente en esta sección resaltaremos una de ellas.

Por un lado, el burn-down chart es una herramienta en la cual se puede visualizar la cantidad de trabajo pendiente y, además, se puede comparar con el rendimiento de trabajo ideal estimado en la planificación. Además, los asociados directamente a los objetivos del sprint suelen ser los primeros que se abordan y estos se estima que representan el 30-70 % del trabajo total del sprint.

Figura 12: Ejemplo de diagrama burn-down



4.2.3. Jira

Una de las herramientas más extendidas en el mundo de metodología ágil scrum es Jira. Atlassian Jira es una de las marcas de confianza en el mundo de software ágil. Los equipos scrum pueden usar este producto tanto como una solución alojada en la nube o en los propios servidores de la empresa.

Atlassian Jira + Agile ofrece scrum, kanban y se integra con Jira, Confluence y otros productos Atlassian. Los project managers pueden establecer flujos de trabajo personalizados, visualizar problemas de control de calidad y mantener una comunicación constante a través de HipChat.

Además, ofrece un software llamado «Release Hub» que asegura que tu producto está realmente completo y preparado cuando se envía al cliente final.

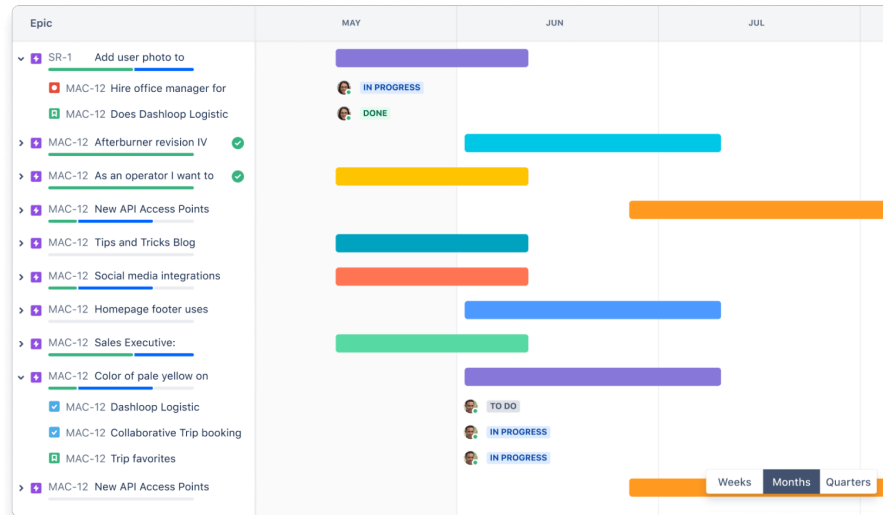
Obviamente, este tipo de productos no suelen ser gratuitos y se pagan por licencias, de forma que, si una empresa tiene 1000 trabajadores trabajando en diferentes proyectos, todos pueden usar esta herramienta mientras tengan 1000 licencias compradas.

Además, esta herramienta está cada vez mejor pensada para la integración continua, por lo que tiene extensiones y plugins que hacen que la integración con esta herramienta sea mucho más sencilla, como, por ejemplo, Jenkins, HipTest, Confluence, Bamboo...

Una de las extensiones más populares es GreenHopper, extensión que se utiliza

para establecer un tablero Kanban tipo scrum, con el que poner las tareas que tiene una historia de usuario en diferentes estados.

Figura 13: Interfaz de Jira



4.2.4. Asana

Asana es una plataforma en línea que facilita la gestión de proyectos y tareas, diseñada para que los equipos trabajen de manera más organizada y eficiente. Fundada en 2008 por Dustin Moskovitz y Justin Rosenstein, ex empleados de Facebook, Asana se ha convertido en una herramienta popular entre empresas de diversos tamaños.

La principal función de Asana es la gestión de proyectos, permitiendo planificar proyectos de manera detallada, establecer plazos y asignar tareas específicas a los miembros del equipo. Además, facilita el seguimiento del progreso a través de diversas visualizaciones como tableros Kanban, listas y cronogramas, lo que permite a los equipos elegir el formato que mejor se adapte a sus necesidades.

En cuanto a la colaboración, Asana mejora significativamente la comunicación dentro del equipo. Los usuarios pueden dejar comentarios en las tareas, actualizar el estado del trabajo y mencionar a otros miembros del equipo para asegurar que todos estén al tanto de los avances y cambios. También es posible adjuntar archivos directamente a las tareas, centralizando así la información y documentos relevantes en un solo lugar.

Una característica destacada de Asana es su capacidad de automatización. Los usuarios pueden crear reglas automáticas que agilizan los procesos, como mover

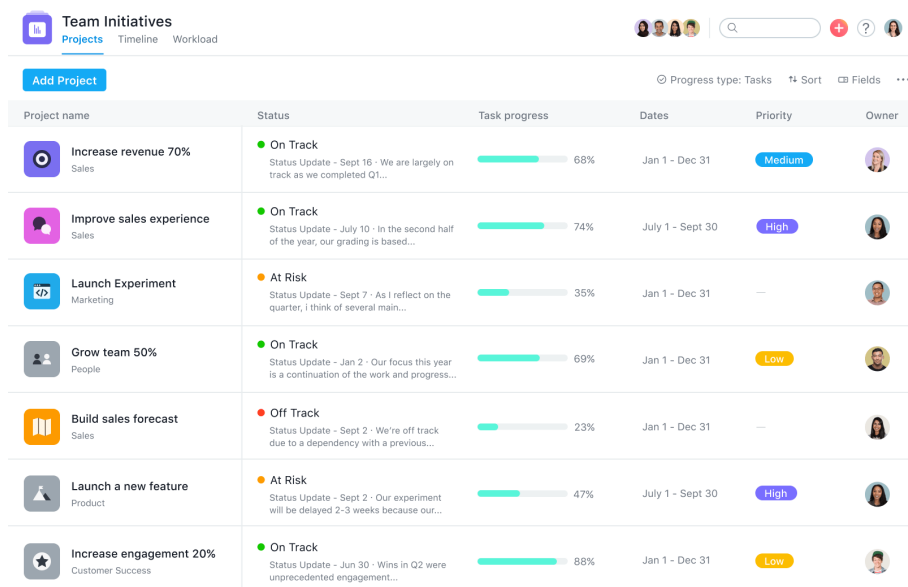
tareas entre secciones o asignar tareas repetitivas sin intervención manual. Esto reduce el tiempo dedicado a tareas administrativas y permite a los equipos enfocarse en el trabajo más importante.

Asana se integra con una amplia variedad de aplicaciones como Slack, Google Drive y Microsoft Teams, entre otras, lo que facilita un flujo de trabajo más coherente y unificado. Esta conectividad permite a los equipos aprovechar al máximo las herramientas que ya utilizan, integrándolas sin problemas en sus procesos de trabajo diarios.

Además, Asana ofrece herramientas de informes y análisis que ayudan a los equipos a evaluar su rendimiento, identificar cuellos de botella y mejorar la productividad. Los informes proporcionan una visión clara del progreso y el desempeño del equipo, lo que facilita la toma de decisiones informadas.

La plataforma es adecuada para equipos de todos los tamaños, desde pequeñas empresas hasta grandes corporaciones, gracias a su capacidad de escalabilidad y personalización. En resumen, Asana es una herramienta versátil que ayuda a los equipos a organizar sus tareas y proyectos, mejorar la comunicación y colaboración, automatizar procesos y obtener una visión clara del progreso del trabajo.

Figura 14: Interfaz de Asana



Team Initiatives					
<div>Projects</div> <div>Timeline</div> <div>Workload</div>		<div> <div> <div></div> <div></div> <div></div> </div> <div> <div></div> <div></div> <div></div> </div> </div> <div> <div></div> <div></div> <div></div> </div>			
Add Project		<div>Progress type: Tasks</div> <div>Sort</div> <div>Fields</div> <div></div>			
Project name	Status	Task progress	Dates	Priority	Owner
<div></div> <div>Increase revenue 70%</div> <div>Sales</div>	<div>On Track</div> <div>Status Update - Sept 16 - We are largely on track as we completed Q1...</div>	<div>68%</div>	Jan 1 - Dec 31	Medium	
<div></div> <div>Improve sales experience</div> <div>Sales</div>	<div>On Track</div> <div>Status Update - July 10 - In the second half of the year, our grading is based...</div>	<div>74%</div>	July 1 - Sept 30	High	
<div></div> <div>Launch Experiment</div> <div>Marketing</div>	<div>At Risk</div> <div>Status Update - Sept 7 - As I reflect on the quarter, I think of several main...</div>	<div>35%</div>	Jan 1 - Dec 31		
<div></div> <div>Grow team 50%</div> <div>People</div>	<div>On Track</div> <div>Status Update - Jan 2 - Our focus this year is a continuation of the work and progress...</div>	<div>69%</div>	Jan 1 - Dec 31	Low	
<div></div> <div>Build sales forecast</div> <div>Sales</div>	<div>Off Track</div> <div>Status Update - Sept 2 - We're off track due to a dependency with a previous...</div>	<div>23%</div>	Jan 1 - Dec 31		
<div></div> <div>Launch a new feature</div> <div>Product</div>	<div>At Risk</div> <div>Status Update - Sept 2 - Our experiment will be delayed 2-3 weeks because our...</div>	<div>47%</div>	July 1 - Sept 30	High	
<div></div> <div>Increase engagement 20%</div> <div>Customer Success</div>	<div>On Track</div> <div>Status Update - Jun 30 - Wins in Q2 were unprecedented engagement...</div>	<div>88%</div>	Jan 1 - Dec 31	Low	

4.2.5. Pivotal Tracker

Pivotal Tracker es una herramienta específicamente desarrollada para desarrolladores web y móviles. Además de que permite soportar diferentes tipos de proyectos, burndown charts, mensajes entre usuarios, etc. Es una herramienta realmente fácil y sencilla de usar.

Suele ser gratuita para instituciones de enseñanza, de forma que los alumnos puedan comprobar cómo gestionar un proyecto usando scrum.

4.2.6. Active Collab

Active Collab es una solución asequible para pequeñas empresas porque es muy fácil de usar. Los scrum master de los proyectos ni siquiera necesitan gastar tiempo en enseñar a su equipo a usar esta herramienta, ya que es muy sencilla e intuitiva.

Tiene una gestión de documentación fantástica, funcionalidad de comunicación basada en e-mail, control y priorización de tareas... Tampoco es una herramienta gratuita y suelen cobrar por mes e integrantes del equipo.

5. La agilidad organizacional

En última instancia, siempre lo que se trata es de tomar ventaja de las oportunidades en relación con la competencia. Así, el beneficio real es lograr desarrollar la estrategia a través de una ventaja competitiva y ganar más lealtad de los clientes y, en definitiva, más ingresos y beneficios, de manera que se pueda seguir alimentando el desarrollo de la organización.

Esta competitividad se logra mediante revisiones del negocio más rápidas y ciclos de decisión más cortos, lo que provoca una mayor adaptabilidad del negocio. Pensemos en una organización que tiene departamentos estancos, con planes a 10-12 años vista. Pueden haber incrementado su competitividad mediante mejora continua, mejora de procesos, etc., pero en la edad de la transformación digital y con una intensa competencia, este sistema de organización no puede ser válido. Esta organización, como todas en este momento, necesita cambiar de estrategia de una forma mucho más ágil y llevar estas decisiones al plano operativo. En este proceso de «agilización», el instrumento más importante es el proyecto, ya sea desde una perspectiva tradicional en cascada o una aproximación Agile. El proyecto es la herramienta de aceleración de la estrategia.

La toma de decisiones en una organización tiene que estar orientada hacia abajo,

a un nivel que esté más cercano del cliente, más cercano a las personas que pueden tomar esas decisiones con eficacia en la «línea del frente».

La agilidad organizacional está también utilizando la gestión de cambios de una forma más efectiva, más cercana a las necesidades del cliente. Tradicionalmente los proyectos han luchado para incorporar la voz del cliente en su trabajo de desarrollo. Las organizaciones que pueden identificar la voz del cliente de forma más efectiva incorporan este elemento como palanca de cambio a toda la estrategia.

5.1. Evolucionar hacia la agilidad técnica y organizacional

Lo que las organizaciones en la actualidad buscan es cómo lograr esa «transformación ágil» dentro de sus organizaciones de una forma que sea eficiente y entregue los mejores resultados posibles. Particularmente hay dos aspectos que se deben consolidar dentro de una organización para lograr tener una «transformación ágil» que genere resultados positivos.

El enfoque está desarrollado en función de dos ramas que articuladas permiten empoderar a los equipos y dar resultados a la empresa.

5.1.1. Agilidad técnica

Cuando se habla de la agilidad técnica se hace referencia a aquellos rasgos prácticos, la metodología para hacer, en los equipos, en las actividades del día a día. El objetivo de esto es proporcionar las herramientas que habiliten a las personas y a los equipos de la organización a conseguir las actividades que deben realizar. Comúnmente, cuando se comenta sobre la agilidad técnica, se suele asociar con las mejoras tecnológicas dentro de la organización; sin embargo, la agilidad técnica comprende un espectro un poco más amplio que se puede resumir en tres vertientes.

Autonomía

La autonomía es uno de los valores clave en el marco de la agilidad técnica. Se refiere a la capacidad de los equipos de desarrollo para tomar decisiones y ser responsables de su trabajo sin la necesidad de una supervisión constante.

La autonomía permite a los equipos de desarrollo ser más eficientes y efectivos, ya que les brinda la libertad para experimentar, innovar y resolver problemas de manera autónoma. Además, permite a los equipos ser más responsables y comprometidos con su trabajo, lo que a su vez conduce a una mayor motivación y satisfacción en el trabajo.

En el marco de la agilidad técnica, los equipos de desarrollo tienen un papel activo en la toma de decisiones y en la gestión del proyecto. Esto implica que los equipos deben ser capaces de trabajar juntos, colaborar y comunicarse efectivamente para lograr los objetivos del proyecto.

La autonomía es una parte integral de la cultura ágil y de la forma en que se trabaja en la agilidad técnica. Se fomenta la toma de decisiones descentralizada y la responsabilidad individual y colectiva en la gestión del proyecto. Además, la autonomía también implica la capacidad de los equipos de desarrollo para autogestionarse y autoorganizarse, lo que permite una mayor eficiencia y efectividad en la entrega de proyectos y productos.

Conocimiento

El conocimiento es otro valor clave en el marco de la agilidad técnica. El conocimiento se refiere a la comprensión profunda y el dominio de los procesos, herramientas, tecnologías y metodologías utilizadas para el desarrollo del proyecto.

En la agilidad técnica, el conocimiento es esencial para que los equipos de desarrollo puedan tomar decisiones informadas y resolver problemas de manera efectiva. Además, el conocimiento permite a los equipos ser más ágiles y responder rápidamente a los cambios y desafíos del proyecto.

El conocimiento también es importante para el aprendizaje continuo y la mejora continua. Además, el conocimiento es un valor clave en la colaboración y la comunicación efectiva.

Se logra a través de la capacitación, consiste en la mejora continua, adaptación y aplicación de nuevas técnicas de trabajo (Lean, Agile, design thinking, etc.).

Tecnología

Referido a las herramientas, arquitectura de sistemas escalables, automatización de procesos, lo que permite no solo colaborar y comunicarse de manera efectiva sino también, lograr una mayor eficiencia y efectividad en el desarrollo del proyecto.

5.1.2. Agilidad organizacional

La agilidad organizacional es la otra componente necesaria para transformar a una organización, en este caso ya no se habla de personas, sino de los aspectos más internos de la organización. Nos referimos a su estructura, a sus procesos y a los enfoques estratégicos propios de la empresa que, en algunos casos, pueden

llevar muchos años establecidos, lo cual hace que sea más difícil cambiarlos o mejorarlos.

Cultura y actitud

La cultura de la organización en cuanto a la tolerancia a los errores, su tolerancia al riesgo, la transparencia en los procesos y la confianza entre los participantes.

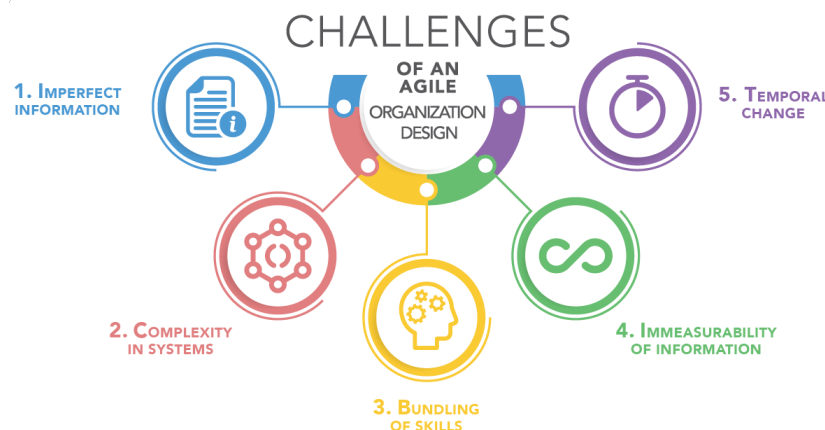
Mejora continua

La organización debe permanecer en constante evolución, donde exista retroalimentación de los procesos a todo nivel, se enfoque en general valor al producto final, se midan los resultados y se tenga un proceso iterativo.

Procesos y estructura

Es uno de los puntos más difíciles para cambiar y tiene que ver con el funcionamiento transversal entre los departamentos de la organización. Una de las formas de lograrlo es a través de las estructuras organizacionales orientadas a las cadenas de valor.

Figura 15: Algunos de los retos que enfrenta la agilidad organizacional



Las organizaciones en la actualidad están enfocadas en «transformar» a través de la implementación de tecnología punta. Ciertamente son herramientas que ayudarán a mejorar el rendimiento; sin embargo, si no existe la evolución integrada entre la agilidad técnica y la agilidad organizacional, muy difícilmente se pueden obtener grandes resultados.

5.1.3. La cadena de valor

La cadena de valor refleja los intervinientes principales del ciclo a través del cual pasa un determinado producto o servicio que se está creando. En la actualidad,

aunque hay muchos más participantes dentro del desarrollo de un producto o servicio, a grandes rasgos se puede definir esta cadena de valor con cuatro participantes:

Cliente

Solicita un producto o servicio con base en unas necesidades específicas.

Negocio

Establece el contacto inicial con el cliente y traduce esas necesidades a las especificaciones técnicas del producto o servicio para que pueda ser desarrollado.

Desarrollo de productos

Se encarga de interpretar las necesidades de un producto o servicio y ejecutarlas.

Operaciones

Una vez que el producto se encuentra completamente definido, se opera o construye en masa.

Figura 16: La cadena de valor de Porter



Es común que en la actualidad se presenten fallas de comunicación entre los distintos departamentos mencionados, debido a que cada uno se centra en sus tareas y no hay una intercomunicación en el proceso completo. A esto se le denomina «silos» y ocasionan la fragmentación de la cadena de valor tal y como la conocemos.

Tarea: investiga un poco más acerca de qué es la cadena de valor de Porter.

La importancia de vencer estos muros es una de las claves de la agilidad organizacional, ya que de esta forma la información comienza a fluir de forma transversal por las unidades de trabajo. En primer lugar (aunque no es la so-

lución completa), se deben establecer equipos multidisciplinarios que crucen las barreras departamentales.

Entre las soluciones para romper con estos muros (identificando cada solución con el número de muro respectivo) se encuentran las siguientes herramientas:

- Esta sección corresponde a la comunicación de la organización con el cliente y no se puede limitar «solo a escuchar al cliente», sino que debe ser un desarrollo colaborativo en donde se encuentren las mejores ideas y se identifiquen las necesidades que aportan valor. La metodología indicada para derribar este muro es el design thinking.
- Por otra parte, entre la unidad de negocio y el equipo de desarrollo, aunque se encuentran dentro de la misma organización, es común que haya problemas de entendimiento. Una forma de mejorar y eliminar esta barrera es a través de las metodologías ágiles y, como hemos mencionado, particularmente uno de los marcos de trabajo más utilizados, descrito en las pasadas unidades, es el Scrum.
- Entre las operaciones y el desarrollo debe existir una sinergia que permita obtener la agilidad de producción óptima. Para eliminar este muro ciertamente la mejor opción es a través de Lean.

Pese a que hemos mencionado estrategias y metodologías diferentes para cada uno de los muros, es importante destacar que estas metodologías y marcos de trabajo (design thinking, Scrum y Lean) son altamente integrables ya que guardan mucha relación entre sus orígenes. Por ello, la capacitación es fundamental para que los integrantes de los equipos sepan, a su vez, integrar los procesos y métodos de trabajo a fin de obtener una alta eficiencia.