Unidad 2: Gestión Lean Agile de Productos de Software

Frameworks de SCRUM a nivel de equipo y escala

SCRUM

Scrum es un marco ligero que ayuda a las personas, equipos y organizaciones a generar valor a través de soluciones adaptables para problemas complejos.
En pocas palabras, Scrum requiere un Scrum Master para fomentar un entorno donde:

1. Un propietario del producto (Product Owner) ordena el trabajo de un problema complejo en un Product Backlog.
2. El equipo de Scrum convierte una selección del trabajo en un incremento de valor durante un Sprint.
3. El equipo de Scrum y sus partes interesadas (stakeholders) inspeccionan los resultados y realizan los ajustes necesarios para el próximo Sprint.
4. Repetir

Scrum es simple. Pruébalo tal cual y determine si su filosofía, teoría y estructura ayudan a alcanzar metas y crear valor. El marco de Scrum es deliberadamente incompleto, solo define las partes necesarias para implementar la teoría de Scrum. Scrum se basa en la inteligencia colectiva de las personas que lo utilizan. En lugar de proporcionar a las personas instrucciones detalladas, las reglas de Scrum guían sus relaciones e interacciones.
En el marco se pueden emplear diversos procesos, técnicas y métodos. Scrum envuelve las prácticas existentes o las hace innecesarias. Scrum hace visible la eficacia relativa de la gestión actual, el entorno y las técnicas de trabajo, de modo que se pueden realizar mejoras.

Scrum se basa en el empirismo y el pensamiento Lean. El empirismo afirma que el conocimiento proviene de la experiencia y la toma de decisiones basadas en lo que se observa. El pensamiento Lean reduce los desperdicios y se centra en lo esencial.
Scrum emplea un enfoque iterativo e incremental para optimizar la previsibilidad y controlar el riesgo.
Scrum involucra a grupos de personas que colectivamente tienen todas las habilidades y experiencia para hacer el trabajo y compartir o adquirir tales habilidades según sea necesario.
Scrum combina cuatro eventos formales para la inspección y adaptación dentro de un evento contenedor, el Sprint. Estos eventos funcionan porque implementan los pilares empíricos de Scrum: transparencia, inspección y adaptación.

Pilares de Scrum

Transparencia: El proceso y el trabajo emergentes deben ser visibles para aquellos que realizan el trabajo, así como para los que reciben el trabajo. Con Scrum, las decisiones importantes se basan en el estado percibido de sus tres artefactos formales. Los artefactos que tienen poca transparencia pueden conducir a decisiones que disminuyen el valor y aumentan el riesgo.
La transparencia permite la inspección. La inspección sin transparencia genera engaños y desperdicios.

Inspección: Los artefactos de Scrum y el progreso hacia objetivos acordados deben ser inspeccionados con frecuencia y diligentemente para detectar varianzas o problemas potencialmente indeseables. Para ayudar con la inspección, Scrum proporciona cadencia en forma de cinco eventos.
La inspección permite la adaptación. La inspección sin adaptación se considera inútil. Los eventos de Scrum están diseñados para provocar cambios.

Adaptación: Si algún aspecto de un proceso se desvía fuera de los límites aceptables o si el producto resultante es inaceptable, el proceso que se está aplicando o los materiales que se producen deben ajustarse. El ajuste debe realizarse lo antes posible para minimizar la desviación adicional.
La adaptación se vuelve más difícil cuando las personas involucradas no están empoderadas o no poseen capacidad para autogestionarse. Se espera que un equipo de Scrum se adapte en el momento en que aprenda algo nuevo por medio de la inspección.

Valores de Scrum

El uso exitoso de Scrum depende de que las personas sean más competentes en vivir cinco valores:

Compromiso, Enfoque, Apertura, Respeto y Coraje
El equipo de Scrum se compromete a lograr sus objetivos y apoyarse mutuamente. Su enfoque principal es el trabajo del Sprint para hacer el mejor progreso posible hacia estos objetivos. El equipo de Scrum y sus partes interesadas están abiertos sobre el trabajo y los desafíos. Los miembros del equipo de Scrum se respetan mutuamente para ser personas capaces e independientes, y son respetados como tales por las personas con las que trabajan. Los miembros del equipo de Scrum tienen el valor de hacer lo correcto y de trabajar en problemas complejos.

Estos valores dan dirección al equipo de Scrum con respecto a su trabajo, acciones y comportamiento.

Las decisiones que se toman, las medidas tomadas y la forma en que se utiliza Scrum deben reforzar estos valores, no disminuirlos o socavarlos. Los miembros del equipo de Scrum aprenden y exploran los valores mientras trabajan con los eventos y artefactos de Scrum. Cuando estos valores son asimilados por el equipo de Scrum y las personas con las que trabajan, los pilares empíricos de Scrum de transparencia, inspección y adaptación cobran vida construyendo confianza.

El equipo Scrum (Scrum Team)

La unidad fundamental de Scrum es un pequeño equipo de personas, un equipo Scrum. El equipo Scrum consta de un Scrum Master, un propietario de producto (Product Owner) y desarrolladores. Dentro de un equipo de Scrum, no hay subequipos ni jerarquías. Es una unidad cohesionada de profesionales enfocada en un objetivo a la vez, el objetivo del Producto.
Los equipos de Scrum son multifuncionales, lo que significa que los miembros tienen todas las habilidades necesarias para crear valor en cada Sprint. También son autogestionados, lo que significa que internamente deciden quién hace qué, cuándo y cómo.
El equipo de Scrum es lo suficientemente pequeño como para permanecer ágil y lo suficientemente grande como para completar un trabajo significativo dentro de un Sprint, por lo general 10 o menos personas. En general, hemos descubierto que los equipos más pequeños se comunican mejor y son más productivos. Si los equipos de Scrum se vuelven demasiado grandes, se debe considerar la posibilidad de reorganizarse en varios equipos Scrum cohesionados, cada uno centrado en el mismo producto. Por lo tanto, deben compartir el mismo objetivo de producto, trabajo pendiente del producto (Product Backlog) y propietario del producto (Product Owner).
El equipo Scrum es responsable de todas las actividades relacionadas con los productos, desde la colaboración, verificación, mantenimiento, operación, experimentación, investigación y desarrollo, y cualquier otra cosa que pueda ser necesaria. Están estructurados y empoderados por la organización para gestionar su propio trabajo. Trabajar en Sprints a un ritmo sostenible mejora el enfoque y la consistencia del equipo de Scrum.
Todo el equipo de Scrum es responsable de crear un incremento valioso y útil en cada Sprint. Scrum define tres responsabilidades específicas dentro del equipo de Scrum: los desarrolladores, el propietario del producto (Product Owner) y el Scrum Master.

Desarrolladores

Los desarrolladores son las personas del equipo Scrum que se comprometen a crear cualquier aspecto de un incremento útil (funcional) en cada Sprint.
Las habilidades específicas que necesitan los desarrolladores son a menudo amplias y variarán con el dominio del trabajo. Sin embargo, los desarrolladores siempre son responsables de:

- Crear un plan para el Sprint, el Sprint Backlog;
- Inculcar la calidad adhiriéndose a una definición de Hecho;
- Adaptar su plan cada día hacia el Objetivo Sprint;
- Responsabilizarse mutuamente como profesionales.

Propietario del producto (Product Owner)

El Propietario del Producto es responsable de maximizar el valor del producto resultante del trabajo del equipo de Scrum. La forma en que esto se hace esto puede variar ampliamente entre organizaciones, equipos Scrum e individuos.

El Propietario del Producto también es responsable de la gestión eficaz de la pila del producto (Product Backlog), que incluye:

- Desarrollar y comunicar explícitamente el Objetivo del Producto;
- Creación y comunicación clara de elementos de trabajo pendiente del producto;
- Pedido de artículos de trabajo pendiente del producto;
- Asegurarse de que el trabajo pendiente del producto sea transparente, visible y comprendido.

El Propietario del Producto puede hacer el trabajo anterior o puede delegar la responsabilidad a otros. En cualquier caso, el propietario del producto sigue siendo responsable.
Para que los Propietarios de Productos tengan éxito, toda la organización debe respetar sus decisiones. Estas decisiones son visibles en el contenido y el orden del trabajo pendiente del producto, y a través del Incremento inspeccionable en la revisión de Sprint.
El Propietario del Producto es una persona, no un comité. El Propietario del Producto puede representar las necesidades de muchas partes interesadas en el trabajo pendiente del producto. Aquellos que deseen cambiar el trabajo pendiente del producto pueden hacerlo tratando de negociar con criterio con el Product Owner.

Scrum Master

El Scrum Master es responsable de establecer Scrum tal como se define en la Guía de Scrum. Lo consigue ayudando a todos a comprender la teoría y la práctica de Scrum, tanto dentro del Equipo como en toda la organización.
El Scrum Master es responsable de la efectividad del Scrum Team. Lo logra al permitir que el equipo Scrum mejore sus prácticas, dentro del marco de Scrum.
Los Scrum Masters son verdaderos líderes que sirven al equipo Scrum y a toda la organización. El Scrum Master sirve al equipo de Scrum de varias maneras, incluyendo:

- Capacitar a los miembros del equipo en autogestión y multifuncionalidad;
- Ayudar al equipo de Scrum a centrarse en la creación de incrementos de alto valor que cumplan con la definición de hecho;
- Promover la eliminación de los impedimentos para el progreso del equipo Scrum;
- Asegurar que todos los eventos de Scrum se lleven a cabo, sean positivos, productivos y que se respete el tiempo establecido (time-box) para cada uno de ellos.

El Scrum Master sirve al Propietario del Producto (Product Owner) de varias maneras, incluyendo:

- Ayudar a encontrar técnicas para una definición eficaz de los objetivos del producto y la gestión de los retrasos en el producto;
- Ayudar al equipo de Scrum a comprender la necesidad de elementos de trabajo pendiente de productos claros y concisos;
- Ayudar a establecer la planificación empírica de productos para un entorno complejo;
- Facilitar la colaboración de las partes interesadas según sea solicitado o necesario.

El Scrum Master sirve a la organización de varias maneras, incluyendo:

- Liderar, capacitar y mentorizar a la organización en su adopción de Scrum;
- Planificar y asesorar sobre la implementación de Scrum dentro de la organización;
- Ayudar a las personas y a las partes interesadas a comprender y promulgar un enfoque empírico para el trabajo complejo;
- Eliminar las barreras entre las partes interesadas y los equipos de Scrum.

Eventos de Scrum

El Sprint es un contenedor para todos los eventos. Cada evento en Scrum es una oportunidad formal para inspeccionar y adaptar los artefactos de Scrum. Estos eventos están diseñados específicamente para permitir la transparencia necesaria. Si no se realizan los eventos según lo prescrito, se pierden oportunidades para inspeccionar y adaptarse. Los eventos se utilizan en Scrum para crear regularidad y minimizar la necesidad de reuniones no

definidas en Scrum. De manera óptima, todos los eventos se llevan a cabo al mismo tiempo y lugar para reducir la complejidad.

El Sprint

Los sprints son el latido del corazón de Scrum, donde las ideas se convierten en valor.
Son eventos de longitud fija de un mes o menos para crear consistencia. Un nuevo Sprint comienza inmediatamente después de la conclusión del Sprint anterior.
Todo el trabajo necesario para alcanzar el objetivo del producto, incluyendo la Planificación (Sprint Planning), Daily Scrums, Revisión del Sprint (Sprint Review) y la Retrospectiva (Sprint Retrospective), ocurren dentro del Sprints.

Durante el Sprint:

- No se hacen cambios que pongan en peligro el Objetivo Sprint;
- La calidad no disminuye;
- El trabajo pendiente del producto se refina según sea necesario;
- El alcance se puede clarificar y renegociar con el Propietario del Producto a medida que se aprende más.

Los Sprints permiten la previsibilidad al garantizar la inspección y adaptación del progreso hacia un objetivo del Producto, como mínimo, una vez al mes en el calendario. Cuando el horizonte de un Sprint es demasiado largo, el Objetivo de Sprint puede volverse obsoleto, la complejidad puede aumentar y el riesgo puede aumentar. Los Sprints más cortos se pueden emplear para generar más ciclos de aprendizaje y limitar el riesgo de coste y esfuerzo a un período de tiempo más pequeño. Cada Sprint puede considerarse un proyecto corto.
Existen varias prácticas para pronosticar el progreso, como gráficos de burn-downs, burn-ups, o flujos acumulativos. Si bien han demostrado ser útiles, estos no sustituyen la importancia del empirismo. En entornos complejos, se desconoce lo que sucederá. Solo lo que ya ha sucedido se puede utilizar para la toma de decisiones con vistas a futuro.

Un Sprint podría ser cancelado si el Objetivo del Sprint se vuelve obsoleto. Solo el Propietario del Producto tiene la autoridad para cancelar el Sprint.

1. Planificación de Sprint (Sprint Planning)

El Sprint Planning inicia el Sprint estableciendo el trabajo que se realizará para el mismo. Este plan resultante es creado por el trabajo colaborativo de todo el equipo de Scrum.
El propietario del producto (Product Owner) se asegura de que los asistentes estén preparados para discutir los elementos de trabajo pendiente de producto más importantes y cómo se asignan al objetivo del producto. El equipo de Scrum también puede invitar a otras personas a asistir a la planificación del Sprint para proporcionar asesoramiento. La planificación del Sprint aborda los siguientes temas:

Tema Uno: ¿Por qué este Sprint es valioso?

El Propietario del Producto (Product Owner) propone cómo el producto podría aumentar su valor y utilidad en el Sprint actual. A continuación, todo el equipo de Scrum colabora para definir un objetivo de Sprint que comunique por qué el Sprint es valioso para las partes interesadas. El Objetivo Sprint debe finalizar antes del final de la Planificación de Sprint.

Tema dos: ¿Qué se puede hacer este Sprint?

A través del debate con el propietario del producto (Product Owner), los desarrolladores seleccionan los elementos del Product Backlog para incluir en el Sprint actual. El equipo de Scrum puede refinar estos elementos durante este proceso, lo que aumenta la comprensión y confianza.
Seleccionar cuánto se puede completar dentro de un Sprint puede ser un desafío. Sin embargo, cuanto más sepan los desarrolladores sobre su rendimiento pasado, su capacidad futura y su definición de hecho, más seguro estarán en sus pronósticos de Sprint.

Tema Tres: ¿Cómo se realizará el trabajo elegido?

Para cada elemento de trabajo pendiente de producto (Product Backlog item) seleccionado, los desarrolladores planifican el trabajo necesario para crear un incremento que cumpla con la definición de hecho. Esto se hace normalmente mediante la descomposición de elementos de trabajo pendiente (Product Backlog item) del producto en elementos de trabajo más pequeños que se puedan realizar en un día o menos. La forma de hacerlo es según la discreción de los propios desarrolladores. Nadie más les dice cómo convertir los elementos de trabajo pendiente del producto en incrementos de valor.

El objetivo de Sprint (Sprint Goal), los elementos de trabajo pendiente de producto seleccionados para el Sprint, más el plan para entregarlos se conocen conjuntamente como el trabajo pendiente de Sprint (Sprint Backlog).
El Sprint Planning tiene una duración máxima de ocho horas para un Sprint de un mes. Para sprints más cortos, el evento suele ser más corto.

2. Scrum Diario (Daily Scrum)

El propósito del Daily Scrum es inspeccionar el progreso hacia el Objetivo Sprint y adaptar el Sprint Backlog según sea necesario, ajustando el próximo trabajo planeado.
El Daily Scrum es un evento de 15 minutos (máximo) para los desarrolladores del equipo de Scrum. Para reducir la complejidad, se lleva a cabo al mismo tiempo y lugar todos los días laborables del Sprint. Si el propietario del producto o el Scrum Master están trabajando activamente en los elementos del Trabajo pendiente de Sprint, participan como desarrolladores.
Los desarrolladores pueden seleccionar cualquier estructura y técnicas que deseen, siempre y cuando su Scrum diario se centre en el progreso hacia el objetivo de Sprint y produzca un plan accionable para el día siguiente de trabajo. Esto crea enfoque y mejora la autogestión.
Los Scrums diarios (Daily Scrum) mejoran la comunicación, identifican impedimentos, promueven una rápida para la toma de decisiones, y en consecuencia, eliminan la necesidad de otras reuniones.
El Daily Scrum no es la única vez que los desarrolladores pueden ajustar su plan. Frecuentemente se reúnen durante todo el día para debatir de forma más detalladas sobre la adaptación o replanificación del resto del trabajo del Sprint

3. Revisión del Sprint (Sprint Review)

El propósito de la revisión del Sprint es inspeccionar el resultado del Sprint y determinar futuras adaptaciones. El equipo de Scrum presenta los resultados de su trabajo a las partes interesadas clave y se discute el progreso hacia el Objetivo de Producto.
Durante el evento, el equipo de Scrum y las partes interesadas revisan lo que se logró en el Sprint y lo que ha cambiado en su entorno. En base a esta información, los asistentes colaboran en qué hacer a continuación. El trabajo pendiente del producto también se puede ajustar para satisfacer nuevas oportunidades. Sprint Review es una sesión de trabajo y el equipo de Scrum debe evitar limitarla a que se convierta en una simple presentación.
La revisión de Sprint es el penúltimo evento del Sprint y se utiliza en un plazo máximo de cuatro horas para un Sprint de un mes. Para sprints más cortos, el evento suele ser más corto.

4. La retrospectiva del Sprint (Sprint Retrospective)

El propósito de la retrospectiva Sprint es planificar formas de aumentar la calidad y la eficacia.
El equipo de Scrum inspecciona cómo fue el último Sprint con respecto a individuos, interacciones, procesos, herramientas y su definición de Hecho. Los elementos inspeccionados a menudo varían según
el dominio del trabajo. Las suposiciones que los desviaron se identifican y se exploran sus orígenes. El equipo de Scrum analiza qué fue bien durante el Sprint, qué problemas encontró y cómo esos problemas fueron (o no fueron) resueltos.
El equipo de Scrum identifica los cambios más útiles para mejorar su eficacia. Las mejoras más impactantes se abordan lo antes posible. Incluso se pueden agregar al Sprint Backlog para el próximo Sprint.
La retrospectiva Sprint concluye el Sprint. Se utiliza un intervalo de tiempo de hasta un máximo de tres horas para un Sprint de un mes. Para sprints más cortos, el evento suele ser más corto.

5. Refinamiento del Product Backlog

Artefactos de Scrum

Los artefactos de Scrum representan trabajo o valor. Están diseñados para maximizar la transparencia de la información clave. Por lo tanto, cada uno de los que los inspecciona tienen la misma base para la adaptación. Cada artefacto contiene un compromiso para garantizar que proporciona información que mejora la transparencia y el enfoque con el que se puede medir el progreso:

- Para el trabajo pendiente del producto es el objetivo del producto.
- Para el Sprint Backlog es el Sprint Goal.
- Para el Incremento es la Definición de Hecho.

Estos compromisos existen para reforzar el empirismo y los valores de Scrum para el equipo de Scrum y sus partes interesadas.

1. Pila del producto (Product Backlog)

El trabajo pendiente del producto es una lista emergente y ordenada de lo que se necesita para mejorar el producto. Es la única fuente de trabajo emprendida por el equipo Scrum.
Los elementos de trabajo pendiente de producto que puede ser hecho por el equipo de Scrum dentro de un Sprint se consideran listos para su selección en un evento de planificación de Sprint. Por lo general adquieren este grado de transparencia después de las actividades de refinación. El refinamiento de Backlog del producto es el acto de descomponer y definir aún más los elementos de trabajo pendiente del producto en artículos más pequeños y precisos. Esta es una actividad en curso para agregar detalles, como una descripción, un pedido y un tamaño. Los atributos a menudo varían con el dominio del trabajo.
Los desarrolladores que realizarán el trabajo son responsables del tamaño. El Propietario del Producto (Product Owner) puede influir en los desarrolladores ayudándoles a entender y seleccionar mejores alternativas.

Compromiso: Objetivo del producto (Product Goal)

El objetivo del producto (Product Goal) describe un estado futuro del producto que puede servir como objetivo para el equipo Scrum contra el cual planificar. El objetivo del producto se encuentra en el trabajo pendiente del producto (Product Backlog). El resto del trabajo pendiente del producto surge para definir "qué" cumplirá el objetivo del producto.

Un producto es un vehículo para entregar valor. Tiene un límite claro, partes interesadas conocidas, usuarios o clientes bien definidos. Un producto podría ser un servicio, un producto físico o algo más abstracto.

El objetivo del producto es el objetivo a largo plazo para el equipo Scrum. Deben cumplir (o abandonar) un objetivo antes de asumir el siguiente.

2. La pila del Sprint (Sprint Backlog)

El Trabajo pendiente de Sprint se compone del objetivo sprint (por qué), el conjunto de elementos de trabajo pendiente de producto seleccionados para el Sprint (qué), así como un plan accionable para entregar el incremento (cómo).
El Trabajo pendiente de Sprint es un plan por y para los desarrolladores. Es una imagen muy visible y en tiempo real del trabajo que los desarrolladores planean realizar durante el Sprint para lograr el Objetivo
Sprint. Por lo tanto, el Sprint Backlog se actualiza a lo largo del Sprint a medida que se aprende más. Debe tener suficientes detalles para que puedan inspeccionar su progreso en el Scrum Diario.

Compromiso: Sprint Goal

El Sprint Goal es el único objetivo para el Sprint. Aunque el objetivo de Sprint es un compromiso de los desarrolladores, proporciona flexibilidad en términos del trabajo exacto necesario para lograrlo. El Objetivo Sprint también crea coherencia y enfoque, animando al equipo de Scrum a trabajar juntos en lugar de en iniciativas separadas.
El objetivo de Sprint se crea durante el evento Sprint Planning y, a continuación, se agrega al Trabajo pendiente de Sprint. A medida que los desarrolladores trabajan durante el Sprint, tienen en cuenta el objetivo de Sprint. Si el trabajo resulta ser diferente de lo que esperaban, colaboran con el propietario del producto para negociar el alcance del Trabajo pendiente de Sprint dentro del Sprint sin afectar al objetivo de Sprint.

3. Incremento (Increment)

Un Incremento es un paso de hormigón hacia el Objetivo del Producto. Cada Incremento es aditivo a todos los Incrementos anteriores y verificado a fondo, asegurando que todos los Incrementos funcionen juntos. Para proporcionar el valor, el incremento debe ser utilizable.
Se pueden crear varios incrementos dentro de un Sprint. La suma de los Incrementos se presenta en la Revisión Sprint apoyando así el empirismo. Sin embargo, un Incremento puede ser entregado a las partes interesadas antes del final del Sprint. La revisión de Sprint nunca debe considerarse una puerta para liberar valor.
El trabajo no se puede considerar parte de un Incremento a menos que cumpla con la Definición de Hecho.

Compromiso: Definición de Hecho (Definition of Done)

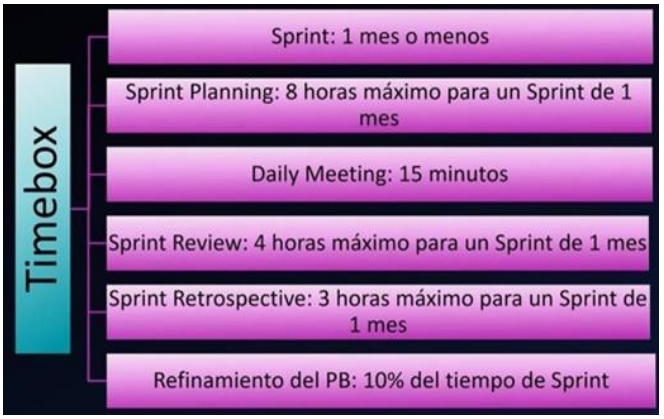
La Definición de Hecho es una descripción formal del estado del Incremento cuando cumple con las medidas de calidad requeridas para el producto.
En el momento en que un elemento de trabajo pendiente de producto cumple con la definición de hecho, se crea un incremento.
La definición de Hecho crea transparencia al proporcionar a todos una comprensión compartida de qué trabajo se completó como parte del Incremento. Si un elemento de trabajo pendiente de producto no cumple con la definición de hecho, no se puede liberar, ni siquiera presentar en la revisión de Sprint. En su lugar, vuelve al Trabajo pendiente del producto para su consideración futura.
Si la definición de hecho para un incremento forma parte de los estándares de la organización, todos los equipos de Scrum deben seguirla como mínimo. Si no es un estándar organizativo, el equipo de Scrum debe crear una definición de hecho adecuada para el producto.
Los desarrolladores deben ajustarse a la definición de Hecho. Si hay varios equipos de Scrum trabajando juntos en un producto, deben definir y cumplir mutuamente con la misma definición de hecho.

Timebox

Esto existe para que no se pierda más tiempo del necesario en las distintas tareas y para aprender a negociar en términos del alcance del producto y no del tiempo o los costos (relacionado con la triple restricción).

Si no se llega al final de un sprint, en vez de extender el tiempo, se entrega menos.

De esta manera, nos volvemos más confiables, con un ritmo de trabajo sostenible. Todas las ceremonias son Timebox. El refinamiento del Product Backlog al ser una actividad continua, no tiene definido un tiempo exacto, sino que el tiempo se expresa en términos porcentuales sobre la definición del Sprint.



Definición de Listo (DoR)

El DoR es un criterio que define cuando una US está lo suficientemente bien formulada como para poderse incluir en un Sprint. Por ejemplo, que cierta US debe tener un prototipo, entonces cuando el prototipo esté realizado, el DoR dirá “se ha definido el prototipo para la US”.

El DoD es un criterio que dice cuando una historia está lo suficientemente bien implementada como para entrar al Sprint Review y ser mostrada al PO, entonces él será el encargado de aprobar o no la US y en caso de que esté aprobada decidirá si desea ponerla en producción.

En el DoD se arma un checklist con todo lo que debe cumplir una US para entrar en producción.

Definición de Hecho (DONE)
❑ Diseño revisado
❑ Código Completo
❑ Código refactorizado
❑ Código con formato estándar
❑ Código Comentado
❑ Código en el repositorio
❑ Código Inspeccionado
❑ Documentación de Usuario actualizada
❑ Probado
❑ Prueba de unidad hecha
❑ Prueba de integración hecha
❑ Prueba de sistema hecha
❑ Cero defectos conocidos
❑ Prueba de Aceptación realizada
❑ En los servidores de producción

Capacidad del equipo en un Sprint

La capacidad es una métrica que utiliza SCRUM para determinar cuánta cantidad de trabajo puede comprometer un equipo para un determinado Sprint.

La capacidad se estima y se mide.

Una de las cosas que se hace en el Sprint Planning es determinar la capacidad del equipo, y entonces, teniendo en cuenta la capacidad, se contrasta en cuántas US del Product Backlog se pueden incorporar en el Sprint Backlog.

Para equipos más maduros, la capacidad puede ser estimada en Story Points y para equipos menos maduros se puede estimar en horas ideales.

Niveles de Planificación

Nivel	Horizonte	Quién	Foco	Entregable
Portfolio	1 año o más	Stakeholders y Product Owners	Administración de un Portfolio de Producto	Backlog de Portfolio
Producto	Arriba de varios meses o más	Product Owner y Stakeholder	Visión y evolución del producto a través del tiempo	Visión de Producto, roadmap y características
Release	3 (o menos) a 9 meses	Equipo Scrum entero y Stakeholders	Balancear continuamente el valor del cliente y la caldad global con las restricciones de alcance, cronograma y presupuesto	Plan de reléase
Iteración	Cada iteración (de 1 semana a 1 mes)	Equipo Scrum entero	Que aspectosentregar en el siguiente Sprint	Objetivo del Sprint y Sprint Backlog
Día	Diaria	Scrum Master y Equipo de desarrollo	Cómo completar lo comprometido	Inspección del progreso

Herramientas de SCRUM

Tarjeta de tarea: Es una tarjeta que cuenta con 5 partes:

1. Código del Ítem Backlog: es un identificador del ítem en el Product Backlog.
2. Nombre de la actividad: suele ser una frase verbal que define qué es lo que hay que hacer.
3. Iniciales del responsable de realizar la tarea.
4. Número de día del sprint.
5. Estimación del tiempo para finalizar la tarea.

Tablero: Tablero utilizado por el Equipo de Desarrollo en el cual se colocan las tarjetas de tareas a realizar durante el sprint. Se encuentra dividido en 5 secciones:

1. Story: el nombre de la historia.
2. To Do: aquí se encuentran las tareas por hacer en el Sprint.
3. Work In Process: aquí se encuentran las tareas que se están realizando.
4. To Verify: aquí se encuentran las tareas finalizadas que deben verificarse.
5. Done: aquí se encuentran las tareas terminadas: finalizadas y verificadas.

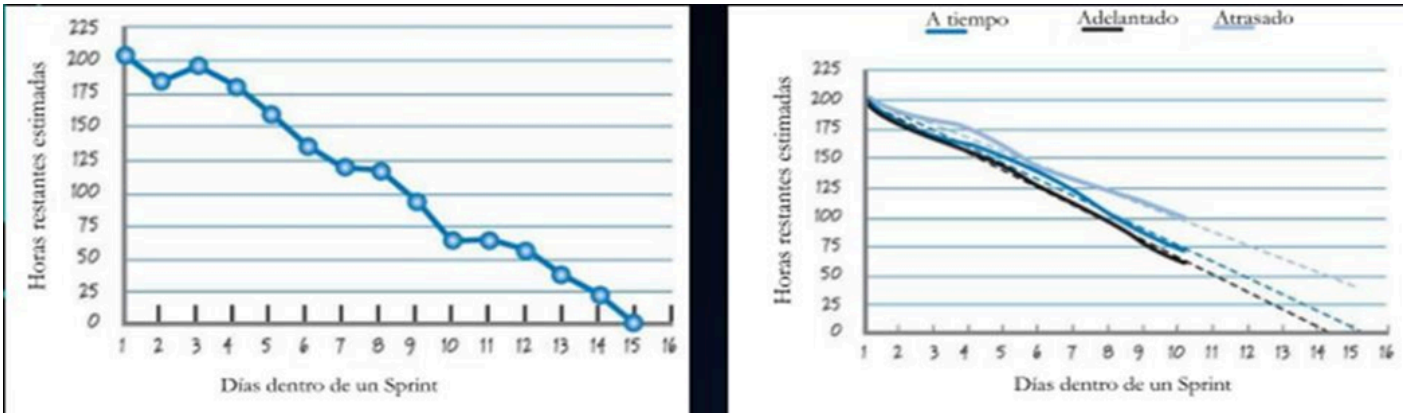
Lo ideal es que el nivel de granularidad de las tareas sea lo más fino posible, de tal forma que se detalle profundamente el avance del sprint, teniendo varias tareas en cada sección del tablero.

Gráficos del Backlog

Gráficos que brindan información acerca del progreso de un Sprint, de una release o del producto. El backlog de trabajo es la cantidad de trabajo que queda por ser realizado (en horas), mientras que la tendencia del backlog compara esta cantidad con el tiempo (medido en días).

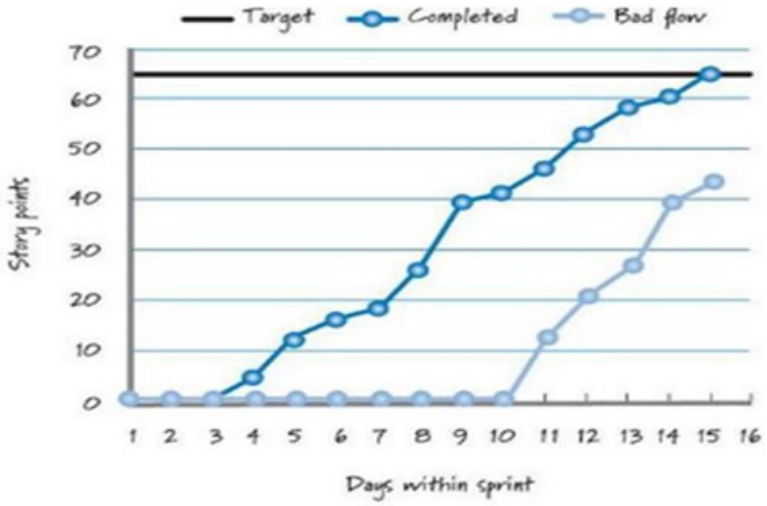
Sprint Burn-Down Chart

Visualiza con una curva descendente cuánto trabajo queda para terminar. En el eje de las abscisas se coloca el tiempo que va desde el inicio del sprint. En el eje de las ordenadas se coloca los puntos de historia a realizar en el sprint.



Sprint Burn-Up Chart

Visualiza con una curva ascendente cuánto trabajo se ha completado a lo largo de la release, es decir que visualiza los puntos de historia terminados en cada sprint.



Combined Burn Chart

Visualiza cuánto trabajo queda para terminar y cuanto trabajo se ha realizado.

Frameworks para escalar SCRUM

Dado que Scrum impone un límite en la cantidad máxima de integrantes del Equipo de Desarrollo, surge la necesidad de escalar este Framework para poder gestionar productos de mayor envergadura, ya que las practicas agiles se utilizan en ambientes complejos para reducir tal complejidad.

Existen diferentes frameworks para escalar como Nexus, scale scrum, less, SAFe.

Framework Nexus: Nexus es un Framework que consiste en roles, eventos, artefactos y técnicas que vinculan el trabajo de aproximadamente tres a nueve equipos Scrum que trabaja sobre un único Product Backlog común para la construcción de un incremento integrado de producto "Terminado". Con un solo Product Owner para entregar un único producto.

Nexus surge para lidiar con la complejidad que supone varios Equipos Scrum trabajando sobre un mismo Product Backlog. Esta complejidad está dada por las siguientes dependencias entre equipos:

1. Requerimientos: el alcance de estos puede superponerse. La forma en que se implementan también puede afectar a los demás.

2. Conocimiento del dominio: el conocimiento del sistema de negocio debería mapearse a los Equipos Scrum para minimizar las interrupciones entre los mismos durante el Sprint.

Responsabilidades

· **Nexus Integration Team:** Responsable de asegurar que se produzca un incremento integrado al menos una vez en cada sprint. La integración incluye abordar las restricciones técnicas y no técnicas del equipo multifuncional que pueden impedir la capacidad de un Nexus para entregar constantemente un incremento integrado.

· **Product Owner:** Es responsable de maximizar el valor del producto y el trabajo realizado e integrado por los Scrum Teams en un Nexus, así mismo también es responsable de la gestión eficaz del Product Backlog.

· **Scrum Master:** responsable de asegurar que el marco de trabajo Nexus se entienda y se promulgue como se describe en la Guía de Nexus. Puede ser un Scrum Master en uno o más de los scrums teams en el Nexus.

· **Uno o más miembros del Nexus Integration Team:** a menudo está formado por miembros de los Scrum Teams que ayudan a los Scrum Teams a adoptar herramientas y prácticas que contribuyen a mejorar la capacidad de los Scrum Teams para entregar un Integrated Increment útil y de valor que cumple con la Definición de Terminado.

Eventos: Nexus agrega o extiende los eventos definidos por Scrum. La duración de los eventos Nexus se guía por la duración de los eventos correspondientes en la Guía de Scrum. Tienen definido un bloque de tiempo adicional a sus correspondientes eventos de Scrum.

· **Nexus Sprint:** Los scrums teams producen un solo Integrated Increment (Incremento de integración), es lo mismo que scrum.

· **Refinamiento entre equipos:** El Refinamiento Entre Equipos del trabajo del Product Backlog reduce o elimina las dependencias entre equipos dentro de un Nexus. El Product Backlog debe descomponerse para que las dependencias sean transparentes, se identifiquen entre equipos y se eliminen o se minimicen.

El Refinamiento Entre Equipos del Product Backlog a escala sirve a un propósito dual: Ayuda a los Scrum Teams a prever qué equipo entregará qué elementos del Product Backlog e Identifica dependencias entre estos equipos.

El Refinamiento Entre Equipos es continuo. La frecuencia, duración y participación del Refinamiento Entre Equipos varía para optimizar estos dos propósitos.

· **Nexus Sprint Planning:** El propósito de la Nexus Sprint Planning es coordinar las actividades de todos los Scrum Teams dentro de un Nexus para un solo Sprint. Los representantes apropiados de cada Scrum Team y el Product Owner se reúnen para planificar el Sprint.

Resultados:

- Objetivo de Sprint para cada Scrum team.
- Objetivo de Nexus.
- Un Nexus sprint backlog.
- Un Sprint backlog.

· **Nexus Daily Scrum:**

- Se discuten problemas de integración y se inspecciona el progreso hacia el objetivo de sprint del nexus.
- Solo asisten representantes de cada Scrum team.
- La Daily Scrum de cada Scrum Team complementa la Nexus Daily Scrum creando planes para el día, centrados principalmente en abordar los problemas de integración planteados durante la Nexus Daily Scrum.
- Los Scrum teams no solo se reúnen o coordinan en el Nexus daily scrum, sino que durante el día, puede existir comunicación entre equipos con mayor detalle sobre adaptar o replanificar el resto del trabajo del sprint.

· **Nexus Sprint Review:** La Nexus Sprint Review se lleva a cabo al final del Sprint para brindar retroalimentación al Integrated Increment terminado que el Nexus ha construido durante el Sprint y determinar adaptaciones futuras. La nexus sprint review reemplaza a las sprint reviews de cada scrum team.

· **Nexus Sprint Retrospective:** El propósito de la Nexus Sprint Retrospective es planificar formas de mejorar la calidad y la eficacia en todo el Nexus. El Nexus inspecciona cómo fue el último Sprint con respecto a individuos, equipos, interacciones, procesos, herramientas y su Definición de Terminado. Además de las mejoras de cada equipo, las Retrospectivas de los Scrum Teams complementan la Nexus Sprint Retrospective mediante el uso de inteligencia de abajo hacia arriba para centrarse en los problemas que afectan al Nexus en su conjunto.

Artefactos

Representan trabajo o valor y están diseñados para maximizar la transparencia. El Nexus Integration Team trabaja con los Scrum Teams dentro de un Nexus para garantizar que se logre la transparencia en todos los artefactos y que el estado del Integrated Increment se entienda.

Cada artefacto contiene un compromiso, y estos existen para reforzar el empirismo y el valor de Scrum para el Nexus y sus interesados.

· **Product Backlog:** Hay uno solo que contiene la lista de todo lo que el Nexus y sus Scrum Teams necesitan para mejorar el producto. El Product Backlog debe entenderse con tal nivel que permita detectar y minimizar las dependencias. Es responsabilidad del Product Owner.

El compromiso es el Objetivo del Producto, que describe el estado futuro del producto y sirve como un objetivo a largo plazo para el Nexus.

· **Nexus Sprint Backlog:** está compuesto del Objetivo del Sprint de Nexus y elementos del Product Backlog de cada Scrum Team del Nexus. Se usa para resaltar las dependencias y el flujo de trabajo durante el Sprint, y se actualiza durante el mismo a medida que se obtiene más conocimiento. Este debe tener un nivel de detalle tal que permita que Nexus pueda inspeccionar su progreso en la Nexus Daily Scrum.

Su compromiso es el Objetivo de Sprint de Nexus, que es un único objetivo para Nexus. Es la suma de todo el trabajo y los Objetivos de Sprint de los Scrum Teams dentro del Nexus. Se crea en la Nexus Sprint Planning y se agrega al Sprint Backlog de Nexus. Los Scrum Teams lo tienen en cuenta a medida que trabajan. En la Nexus Sprint Review se debe de mostrar la funcionalidad de valor y útil que está terminada para alcanzar el Objetivo del Sprint.

· **Integrated Increment:** es la suma actual de todo el trabajo integrado completado por un Nexus en relación con el Objetivo del Producto. Se inspecciona en la Nexus Sprint Review pero puede entregarse antes de la misma. Debe cumplir con la definición de terminado.

El compromiso de este artefacto es la Definición de Terminado, que define el estado del trabajo integrado cuando cumple con la calidad y las medidas requeridas para el producto. El incremento se termina sólo cuando está

integrado, es de valor y utilizable. Es responsabilidad del Nexus Integration team una definición de terminado que se pueda aplicar en cada sprint, y todos los Scrum Team deben definir y adherirse a esta definición. Cada Scrum Team se autogestiona para llegar a este estado, pudiendo aplicar una definición de terminado más estricta pero nunca usar criterios menos rigurosos de lo acordado para el Integrated Increment.

Framework LeSS: LeSS es un marco de trabajo que permite escalar Scrum y se aplica a productos de entre dos a ochos equipos. LeSS propone un solo Product Owner, un Scrum Master que puede servir de uno a tres equipos, gestionando un único Product Backlog. El Sprint es común para todos los equipos. Si se trabaja con más de ocho equipos, se utiliza LeSS Huge.

Principios

- Scrum a gran escala es Scrum;
- Más con LeSS;
- Pensamiento sistémico;
- pensamiento Lean;
- Control empírico de procesos;
- Transparencia;
- Mejora continua hacia la perfección;
- Centrado en el consumidor;
- Enfoque de producto completo;
- Teoría de colas.

Miembros del Equipo

- **Scrum Master:** enseña Scrum y LeSS a la organización y los entrena en su adopción. Domina Scrum y LeSS y usa este conocimiento para guiar a todos a descubrir cómo pueden contribuir mejor a crear el producto más valioso, crea el ambiente para que las personas tengan éxito. El Scrum Master dedica todo su tiempo a cumplir ese rol y un Scrum Master sirve entre uno a tres equipos. Su enfoque es hacia los equipos, el Product Owner, la organización y las prácticas de desarrollo; se enfoca en todo el sistema organizacional no en un solo equipo.

- **Product Owner:** enseña Scrum y LeSS a la organización y los entrena en su adopción. Domina Scrum y LeSS y usa este conocimiento para guiar a todos a descubrir cómo pueden contribuir mejor a crear el producto más valioso, crea el ambiente para que las personas tengan éxito. El Scrum Master dedica todo su tiempo a cumplir ese rol y un Scrum Master sirve entre uno a tres equipos. Su enfoque es hacia los equipos, el Product Owner, la organización y las prácticas de desarrollo; se enfoca en todo el sistema organizacional no en un solo equipo.

- **Equipos:** un equipo en LeSS es el mismo que en Scrum de un solo equipo. El objetivo del equipo en LeSS es agregar ítems del Product Backlog al producto durante el Sprint. Los equipos trabajan en estrecha colaboración con los clientes/usuarios para aclarar la definición de los elementos y con el Product Owner en la priorización. Coordinan e integran su trabajo con el de los demás equipos, para final del Sprint haber producido un solo incremento del producto. Cada equipo tiene la responsabilidad de manejar las relaciones con los demás equipos. Cada equipo es autogestionado, multifuncional.

Eventos

- **Sprint Planning One:** es atendida por el Product Owner y los equipos o representantes de estos. Juntos seleccionan tentativamente a los ítems que cada equipo trabajará en ese Sprint. Los equipos identifican oportunidades para trabajar juntos y se aclaran las preguntas finales.

- **Sprint Planning Two:** es para que decidan cómo harán los ítems seleccionados. Esto generalmente involucra el diseño y la creación de su Sprint Backlog.

- **Daily Scrum:** cada equipo tiene su propia Daily Scrum y no hay diferencia con las Daily Scrum para un solo equipo. Tiene una duración de 15 minutos. Y los miembros del equipo deben responder a tres preguntas.

o ¿Qué hice ayer?

- ¿En qué voy a trabajar hoy?
- ¿Qué queda por hacer?

· **Sprint Review:** hay un review para el producto y es común para todos los equipos. Es el punto de inspección – adaptación y se realiza al final del Sprint; es la ocasión para todos los equipos de ver el incremento (el enfoque se pone en todo el producto). Durante esta ceremonia, clientes y Stakeholders examinan lo que los equipos construyeron durante el Sprint y se discute sobre cambios y se aportan nuevas ideas. Los equipos deben estar juntos y el PO, clientes7Stakeholders definen la dirección del producto. Es importante que los Stakeholders formen parte y aporten la información necesaria para una inspección y adaptación efectiva.

· **Retrospective**: Al final del Sprint, cada equipo individualmente realiza su propia Sprint Retrospective.

· **Overall Retrospective:** nueva reunión el LeSS. Su propósito es discutir problemas entre equipos organizacionales y sistémicos dentro de la organización. Algunas preguntas que se hacen son: ¿Qué tan bien trabajan los equipos juntos?; ¿Hay algo que los equipos hagan que puedan compartir?; ¿Están los equipos aprendiendo juntos?

· **Product Backlog Refinement:** es necesario dentro de cada Sprint para refinar los elementos y estar listos para futuros Sprints. Las actividades claves son: dividir elementos grandes; aclarar elementos hasta que estén listos para la implementación y estimar tamaño, valor, riesgos. A esto lo realiza todo el equipo junto y no el PO por separado.

· **LeSS Sprint:** hay un sprint a nivel producto, no un Sprint diferente para cada equipo. Cada equipo comienza y termina el Sprint al mismo tiempo.

Artefactos

· **Incremento de producto potencialmente entregable:** la salida de cada Sprint es un incremento de producto potencialmente entregable. Es la forma de integrar el trabajo de todos los equipos al finalizar el Sprint. Potencialmente enviable es una declaración sobre la calidad del software y no sobre el valor o comerciabilidad del este. Si el producto se puede enviar realmente dependerá del Definition of Done.

Hay un DoD común para todos los equipos.
- Cada equipo puede tener su propia DoD más detallada, pero siempre expandiendo de la común.
- El objetivo es mejorar la DoD para que resulte en su envío de producto en cada Sprint (o incluso con más frecuencia).

· **Product Backlog:** Todos los equipos construyen un solo producto en base a los ítems de un solo Product Backlog. Estos ítems no están preasignados a los equipos. La definición de producto debe ser tan amplia y centrada en el usuario final/cliente como sea práctico. Con el tiempo, la definición de producto podrá expandirse. El Product Backlog de LeSS es el mismo que en un entorno Scrum de un solo equipo.

· **Sprint Backlog:** cada equipo tiene su propio Sprint Backlog. Es la lista de trabajo que el equipo debe realizar para poder completar los ítems del Product Backlog. Por lo tanto, el Sprint Backlog es por equipo y no hay diferencia entre un LeSS Sprint Backlog y un Sprint Backlog.

LeSS Huge: Para más de 8 equipos.

· ¿Qué es igual a LeSS?

- Un Product Backlog para todos.
- Un DoD.
- Un DoR.
- Un Product Manager.
- El Sprint es común para todos los equipos.
- Un incremento potencialmente entregable.

· ¿Qué es diferente a LeSS?

- Ahora hay un Area Product Manager.
- Hay un Area Product Owners.
- Hay un Area Product Backlogs.
- Hay un Area Product Backlogs.
- Hay un Area Prodcut Vision.
- Hay más de 8 equipos.

Métricas ágiles y en otros enfoques

¿Qué es una métrica?
Una métrica es un número que representa el grado o presencia de un determinado conjunto de atributos respecto de lo que se quiere medir (proceso, producto o proyecto), expresadas de tal forma que permitan una medición objetiva de la realidad. Por ejemplo, la escala: NS, S, B, MB, E no sirve. Lo que sí sirve es cuántos E hay, cuántos S hay, etc. Debe ser posible medir en términos de la definición de esta y del esfuerzo que se debe aplicar para medirla. Es decir que la métrica no es la escala sino la cantidad de cierta escala.

Todas las métricas poseen un costo asociado, debido a que se deben destinar recursos para su planificación, ejecución y análisis, y no siempre es factible disponer de esos recursos en un proyecto. Esto implica un análisis de costo-beneficio para evitar definir métricas que no aporten un valor a la organización/equipo/proyecto/etc. En ese análisis costo- beneficio, también influye la relación precisión-medición, ya que muchas veces no es necesaria tanta precisión. Esto también quiere decir que las métricas no sólo hay que calcularlas y dejarlas, hay que utilizarlas para que justifiquen ese costo. (El paso de “no medir” a “comenzar a medir” no tiene que ser excesivo, porque ese es otro de los factores que termina generando resistencia, por costo excesivo, ser una pérdida de tiempo en algo que después no se mira, etc.)

Las métricas no son para evaluar a la gente. No deben ser utilizadas para castigar o beneficiar a la gente, ya que es posible que las personas comiencen a fingir en el valor de las métricas para lograr objetivos finales, lo cual resulta perjudicial para el análisis del fenómeno que se desea medir.

En el software es complicado tener métricas de algunas características del producto, como por ejemplo la usabilidad del producto, por lo que no se mide directamente al producto, sino que se utilizan conceptos asociados (epifenómenos) a la característica que se desea medir. Por ejemplo, si se quiere medir el uso de una funcionalidad, se puede medir a través de la cantidad de clicks en el botón que permite ejecutar esa funcionalidad.

Por ejemplo, una métrica que dice cantidad de requerimientos tomado en función de la cantidad de requerimientos que deberíamos tomar, no es una métrica válida.

¿Para qué medir?
· Para señalar, controlar y supervisar el desarrollo del proyecto.
· Para predecir al estimar proyectos de software.
· Para evaluar, es decir, para tener una noción de los costos, por ejemplo.
· Para mejorar el proceso, el proyecto o el producto.

Dominio de métricas: Las métricas se dividen en tres dominios, el cual, cada uno posee un foco distinto de medición:

· **Métricas de proceso:** son métricas estratégicas a nivel organizacional, orientadas a mejorar el proceso y tienen como objetivo generar indicadores que permitan mejorar los procesos de software a largo plazo. Son públicas y se despersonifican las mediciones de personas, áreas o proyectos particulares, para obtener un número aislado y tener una métrica sobre el proceso de la organización en general, como un todo.
Son responsabilidad del Ingeniero de Procesos, quien realiza las mediciones y luego publica los datos para que todos los empleados de la organización puedan acceder a ellos.

Por ejemplo, se toman métricas de cantidad de defectos en todos los productos que se realizaron, se despersonalizan, se promedian a cada una de ellas y se publican a toda la organización sin hablar particularmente de un proyecto o producto, sino en forma general al proceso de la organización.
En este enfoque, se tiene en cuenta que la experiencia de cada proyecto es extrapolable, por eso hacen uso de esta forma de tomar las métricas, desvinculándose de un producto o proyecto particular para hablar del comportamiento de la organización.

Los indicadores de proceso permiten tener una visión profunda de la eficacia de un proceso, determinando qué funciona de acuerdo a lo esperado y qué no.
Proporcionan beneficios significativos a medida que la organización trabaja para mejorar su nivel global de madurez del proceso.

Ejemplos:

- Desviación organizacional de estimaciones.
- Defectos por severidad en productos de la organización.
- Errores previos a releases por proyecto.
- Defectos detectados por usuarios.
- Esfuerzo realizado.
- Tiempos de planificación promedio por proyectos.
- Propagación de errores de fase a fase.

· **Métricas de proyecto:** están enfocadas a los recursos que se dedican al proyecto, como costos, esfuerzos, estimaciones y tiempo. Son responsabilidad del Líder de Proyecto y permiten al equipo adaptar el desarrollo de los proyectos y de las actividades técnicas. Estas métricas son privadas de ese proyecto y solo son visibles para los involucrados en el mismo.

Se utilizan para mejorar la planificación del desarrollo, generando ajustes que eviten retrasos, reduzcan riesgos potenciales y por lo tanto, problemas.
Además, se utilizan para evaluar la calidad de los productos en todo momento y en caso de ser necesario, modificar el enfoque para mejorar la calidad, minimizando defectos, retrabajo y por ende el costo total del proyecto.
Las métricas de proyecto se consolidan con el fin de crear métricas de procesos que sean públicas para la organización de software como un todo.

Ejemplos:

- Eficiencia de estimación del proyecto.
- Costos estimados versus costos reales.
- Esfuerzo/Tiempo por tarea del Ingeniero de Software.
- Errores no cubiertos por hora de revisión.
- Fechas de entregas reales versus programadas.
- Cantidad de cambios y sus características.

· **Métricas de Producto:** están enfocadas en lo que se construye, son responsabilidad del equipo de desarrollo y de Testing y son particulares de ese producto.

Se utilizan con propósitos técnicos y tienen como objetivo generar indicadores en tiempo real de la eficacia del análisis, el diseño, la estructura del código, la efectividad de los casos de prueba y calidad del software a construir.
Se deben controlar los artefactos resultantes del proceso de desarrollo (componentes y modelos) para garantizar:

- Que cumplan con los requerimientos del cliente;
- Que cumplan con los requerimientos de calidad;
- Que estén libre de errores;
- Que se realizaron bajo los procedimientos de calidad. Ejemplos:

- Cantidad de líneas de código del producto;
- Defectos por severidad de un producto;
- Promedio de métodos por clase;
- Cantidad de métodos de cada clase;
- Cantidad de casos de uso por complejidad.

Métricas en el enfoque tradicional: En el enfoque tradicional se hace énfasis en los 3 dominios arriba mencionados. Están basadas en la gestión de proyectos definidos, poseen un abanico de métricas mayor a los demás enfoques.
Son una disciplina transversal. Cuando se planifican los proyectos, allí se definen qué métricas se utilizarán, quiénes serán responsables de estas, cómo se calcularán, etc.
No todas las métricas le interesan a todo el mundo, dependiendo del momento y rol es lo que interesa medir. Los perfiles más técnicos por ejemplo apuntan a cuestiones relacionadas con el producto y esfuerzo (porque son ahí donde los desarrolladores asumen compromiso) y en ámbitos organizacionales interesa plata (costo) y tiempo. Por ejemplo:

- **Testers:** les interesa el esfuerzo y cantidad de defectos encontrados principalmente, porque es lo que más afecta a este rol.

- **Líder de proyecto:** El enfoque de las métricas estará más relacionado al calendario, los costos, plazos de tiempo a cumplir, etc. por el contrato que se tiene con el cliente, y la relación entre el esfuerzo y el tiempo.

Métricas básicas para un proyecto de software: Son las mínimas e imprescindibles que uno tiene que utilizar si tiene la intención de desarrollar una cultura de medición:

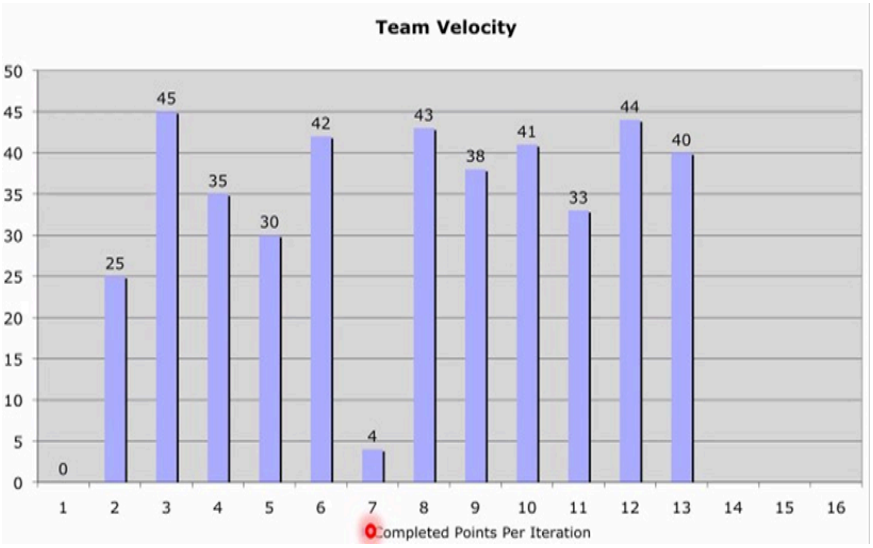
- Tamaño del producto: asociada a métrica de producto.
- Esfuerzo: asociada a métrica de proyecto.
- Calendario: asociada a métrica de proyecto.
- Defectos: asociada a métrica de producto.

Estas métricas están relacionadas con la triple restricción en un proyecto (esfuerzo, alcance y tiempo).

Métricas en ambientes ágiles: Las métricas sirven para un equipo y no son extrapolables por lo que dice el agilismo a otros proyectos o equipos. En ágil hay un principio que habla específicamente de métricas (la mejor métrica de progreso es el software funcionando). Este principio apunta a que vamos a medir producto (no proceso ni proyecto). Esto es una reacción a proyectos tradicionales que tenían todas las métricas posibles cubriendo los tres enfoques, pero no teniendo avances sobre el producto (se perdía mucho tiempo y no se progresaba). Solo se debe medir lo que sea necesario y nada más, es decir, lo que agregue valor para el cliente.

Métrica de velocidad

Es la métrica más importante del enfoque (métrica de producto), y mide la cantidad de puntos de historia que se realizaron en un Sprint y fueron aceptados por el Product Owner. Hay que recordar que no se cuentan las historias parcialmente terminadas, sólo las que están completadas y aceptadas por el PO.
Esta métrica se calcula (no se estima) luego de que el PO. acepte la implementación de una US.
Esta métrica suele representarse con un gráfico de barras para medir la estabilidad del equipo. Estos gráficos son permanentes durante todo el proyecto y son visibles para todo el mundo.
Esta métrica, y sus valores a lo largo de un proyecto ágil, permiten analizar si el equipo posee una estabilidad a lo largo del mismo, lo



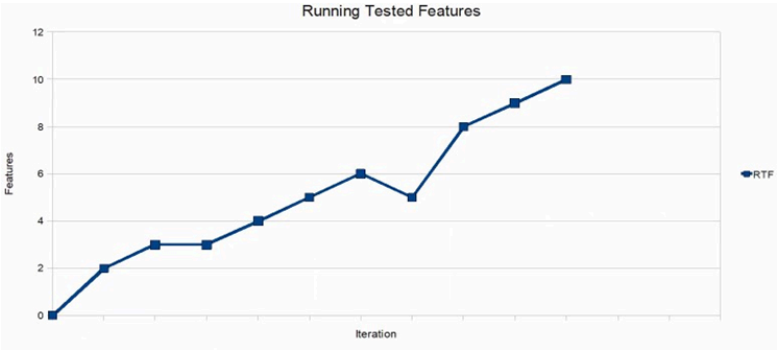
cual también se relaciona con un principio del manifiesto ágil (desarrollo sostenible).

Métrica de capacidad

Es una métrica de proyecto que mide el compromiso de un equipo para un determinado sprint, en horas de trabajo ideales. Es por esto que, se utiliza para planificar, ya que permite definir cuántas historias de usuario se van a tomar del Product Backlog para implementar en el próximo sprint.
Se estima al principio de un sprint, por lo que también se la llama velocidad estimada. Se puede estimar en horas ideales o en Story Points.

Running Tested Features (RTF)

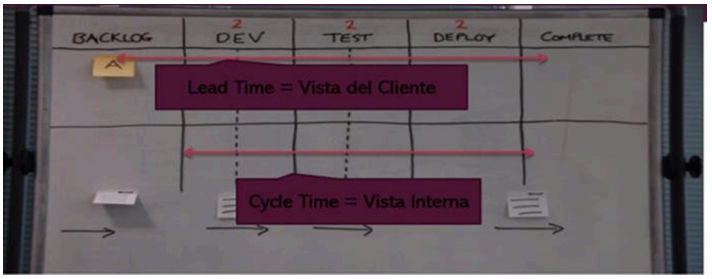
Mide la cantidad de features testeadas que están funcionando, es decir, cuantas piezas de producto (historias de usuarios, casos de usos, requerimientos) se terminaron y están en ejecución.
El problema de esta métrica es que no tiene en cuenta la complejidad de las piezas de producto que se implementan. Por ejemplo, si se toma como piezas de producto a historias de usuario, se mide cuántas historias de usuarios están funcionando, pero no se sabe de cuantos puntos de historias tiene cada una de ellas. Es una medición absoluta.



Si la curva es constante (llana) o tiene pendiente negativa, entonces indica la existencia de un problema, ya que las características no incrementan o disminuyen en el tiempo. Esta última situación se presenta cuando una funcionalidad del sistema deja de ser válida.
Las métricas de defectos también son necesarias en todos los ambientes.

Métricas de software en Lean (KANBAN)

El foco de medición de Kanban es el proceso, ya que el sistema de trabajo de Kanban es introducir mejoras en un flujo de trabajo continuo, es decir, no existe un proyecto con principio y fin. Es decir, se mide el comportamiento del proceso en función al tiempo que se demoran las características.



Lead Time o Elapsed Time: Métrica de vista o perspectiva del cliente. Es la más importante para el cliente. Mide desde el momento en que el cliente me pide algo (entra al Backlog) hasta que yo se lo entrego.

Cycle Time: Mide el tiempo desde que el equipo comenzó a trabajar sobre una funcionalidad pedida, hasta que se lo entrega, eliminando el tiempo de espera en el Backlog. Por esto se la considera como una vista interna, que sirve al equipo de trabajo y no es tan relevante para el cliente. Es igual al Lead Time menos el tiempo que estuvo en el Backlog.

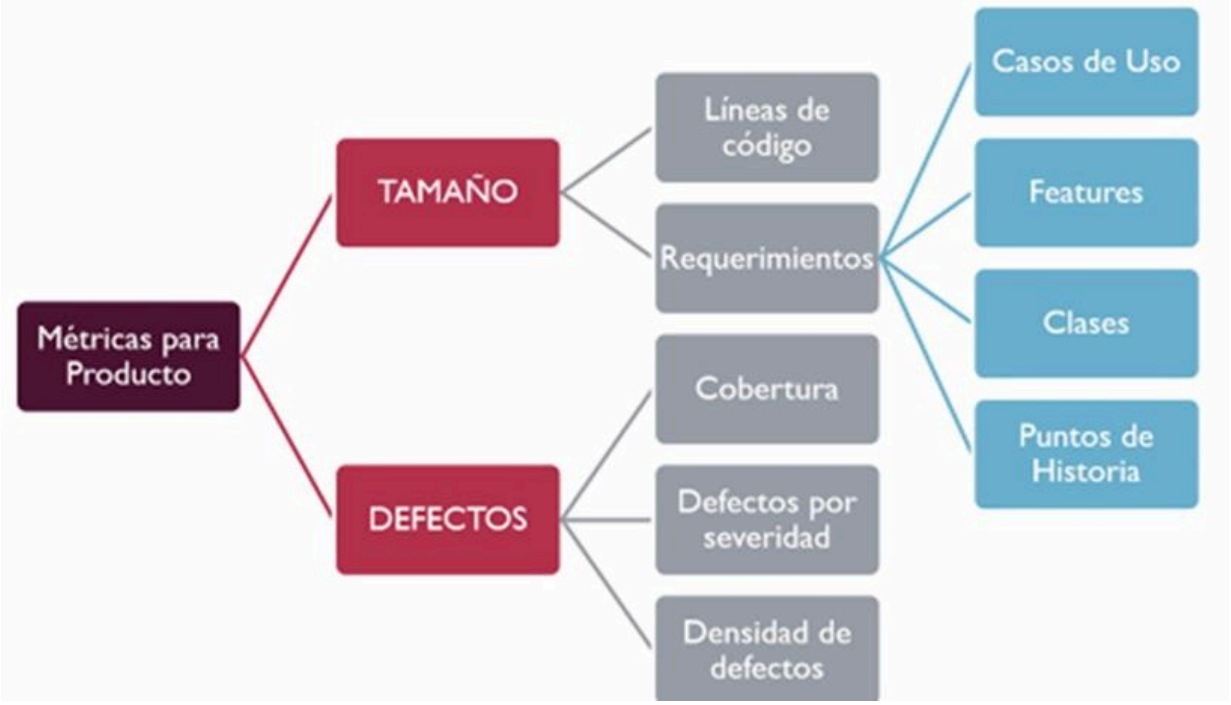
Touch Time: Mide los tiempos en los cuales las personas están efectivamente trabajando sobre una unidad de trabajo (columnas de producción), sin tener en cuenta aquellos tiempos de espera (columnas de acumulación).
En cada columna, por ejemplo, en la columna Test de la imagen, se tiene una columna de trabajo y otra de acumulación para poder realizar el concepto de sistema “pull” o de arrastre.

Touch Time ≤ Cycle Time ≤ Lead Time

Eficiencia del ciclo de proceso: Esta métrica mide la eficiencia del tiempo para entregar un trabajo, respecto al tiempo destinado a su ejecución. Se calcula como Touch time / Lead time.
Mientras más cercano a 1 sea su valor, más eficiente es el proceso, ya que todo el tiempo se estuvo trabajando sobre alguna funcionalidad, y no hubo muchos momentos de espera en columnas de acumulación.

Métricas de Producto de Software: Hoy en día no es recomendable medir en líneas de código. Defectos

- Cobertura: tratar de que sea la mayor posible para cubrir el 100% del análisis de los defectos del producto.
- Defectos por severidad: cantidad de defectos por escala de severidad de cada defecto. 5 defectos graves.
- Densidad de defectos: cuantos defectos se encuentran por historia de usuario.



Resumen métricas en cada enfoque

Tradicional	Ágil	Lean
Esfuerzo	✓ Velocidad	✓ Lead time – Elapsed time
Tiempo	✓ Capacidad	✓ Cycle time
Costos	✓ Running Tested Features	✓ Touch time
Riesgos		✓ Eficiencia de proceso