Programador de Videojuegos

CLASE 3

Índice de temas de la clase

- Metodologías ágiles: Principios y valores.
- Scrum
- Programación Extrema – XP
- Lean Inception
- Kanban

Metodologías ágiles

Metodologías ágiles

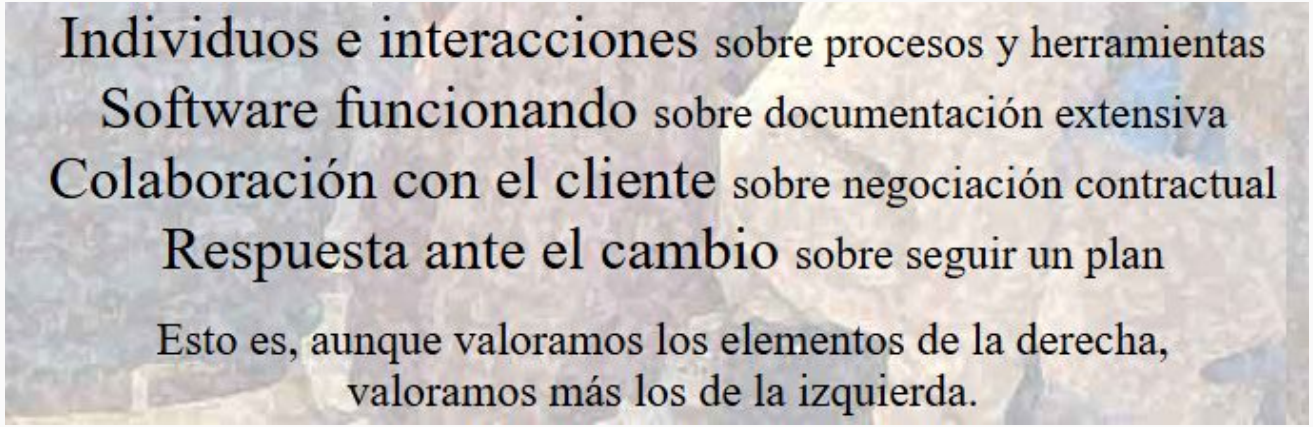
- Las metodologías ágiles son enfoques de desarrollo de software (y gestión de proyectos) que promueven la flexibilidad, la colaboración constante, la entrega rápida de valor y la adaptación al cambio. Surgieron como una respuesta a los modelos rígidos (como el modelo en cascada) que dificultaban la evolución del software en entornos cambiantes.
- Es un enfoque iterativo e incremental para el desarrollo de software, centrado en la colaboración con el cliente, la adaptación al cambio, y la entrega continua de valor. Se basa en los principios del Manifiesto Ágil y es ampliamente utilizado en entornos dinámicos y cambiantes.

Link manifiesto ágil: https://agilemanifesto.org/iso/es/manifesto.html

Metodologías ágiles: objetivos

- Entregar valor rápidamente al cliente.
- Adaptarse a los cambios de requisitos.
- Fomentar la comunicación continua entre todos los involucrados.
- Mejorar la calidad del software a través de iteraciones frecuentes.
- Crear equipos motivados y autoorganizados.

Metodologías ágiles: valores



Los 12 principios del Manifiesto Ágil

1. Nuestra mayor prioridad es satisfacer al cliente mediante la entrega temprana y continua de software con valor.
2. Aceptamos que los requisitos cambien, incluso en etapas tardías del desarrollo. Los procesos Ágiles aprovechan el cambio para proporcionar ventaja competitiva al cliente.
3. Entregamos software funcional frecuentemente, entre dos semanas y dos meses, con preferencia al periodo de tiempo más corto posible.
4. Los responsables de negocio y los desarrolladores trabajamos juntos de forma cotidiana durante todo el proyecto.
5. Los proyectos se desarrollan en torno a individuos motivados. Hay que darles el entorno y el apoyo que necesitan, y confiarles la ejecución del trabajo.
6. El método más eficiente y efectivo de comunicar información al equipo de desarrollo y entre sus miembros es la conversación cara a cara.

Los 12 principios del Manifiesto Ágil

7. El software funcionando es la medida principal de progreso.
8. Los procesos Ágiles promueven el desarrollo sostenible. Los promotores, desarrolladores y usuarios debemos ser capaces de mantener un ritmo constante de forma indefinida.
9. La atención continua a la excelencia técnica y al buen diseño mejora la Agilidad.
10. La simplicidad, o el arte de maximizar la cantidad de trabajo no realizado, es esencial.
11. Las mejores arquitecturas, requisitos y diseños emergen de equipos auto-organizados.
12. A intervalos regulares el equipo reflexiona sobre cómo ser más efectivo para a continuación ajustar y perfeccionar su comportamiento en consecuencia.

Metodologías ágiles para el desarrollo de software

Scrum

Metodología ágil basada en sprints (iteraciones cortas de 1 a 4 semanas) que permite desarrollar software de manera iterativa e incremental con un equipo auto organizado.

Características clave:

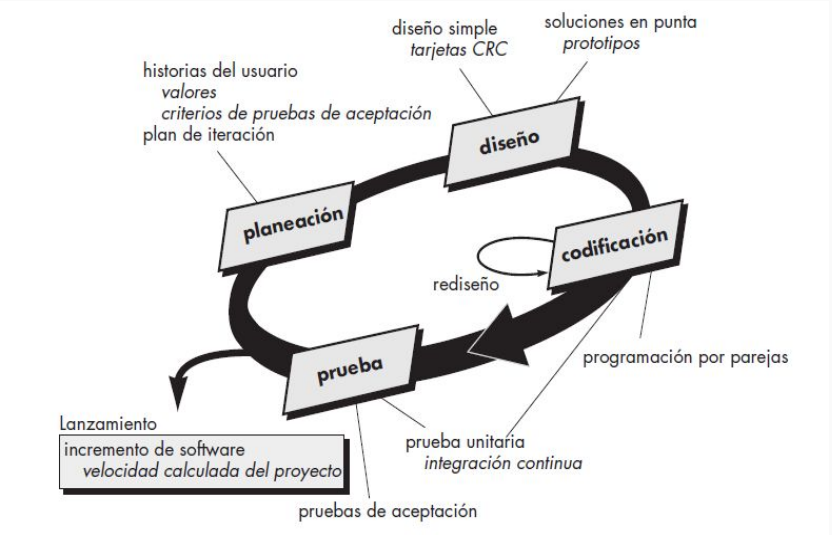
- Ciclos cortos llamados sprints.
- Roles definidos: Product Owner, Scrum Master y Equipo de desarrollo.
- Reuniones clave: Sprint Planning, Daily Scrum, Sprint Review y Sprint Retrospective.
- Gestión basada en un Product Backlog (lista priorizada de tareas o funcionalidades)

Programación Extrema (XP - Extreme Programming)

Metodología ágil enfocada en la excelencia técnica, con prácticas específicas de programación que promueven calidad, simplicidad y colaboración constante.

<u>Características clave:</u>

- Pruebas automatizadas desde el principio (TDD).
- Programación en pareja (Pair Programming).
- Integración continua.
- Iteraciones cortas con feedback rápido.
- Refactorización continua del código.



Lean Inception

Metodología previa al desarrollo ágil que ayuda a definir el Producto Mínimo Viable (MVP) de forma colaborativa en muy poco tiempo (usualmente en una semana).

Lean = "Magro" **Inception = Inicio**

Técnica para alinear a los equipos en torno a qué producto construir, por qué construirlo, para quién, y cuál será el MVP (Producto Mínimo Viable). Se realiza en una semana o menos, con sesiones de trabajo intensas.

Características clave:

- Talleres intensivos con stakeholders y el equipo.
- Se definen objetivos, funcionalidades, usuarios y el MVP.
- Enfoque en alineación de visión y valor antes de escribir código.
- Utilizado para reducir el riesgo de construir productos equivocados.

Resultado de una Lean Inception bien hecha:

- Visión clara del producto
- Alineación de negocio, usuarios y tecnología
- Definición del MVP
- Plan inicial de desarrollo

Lean Inception

Puede incluir:

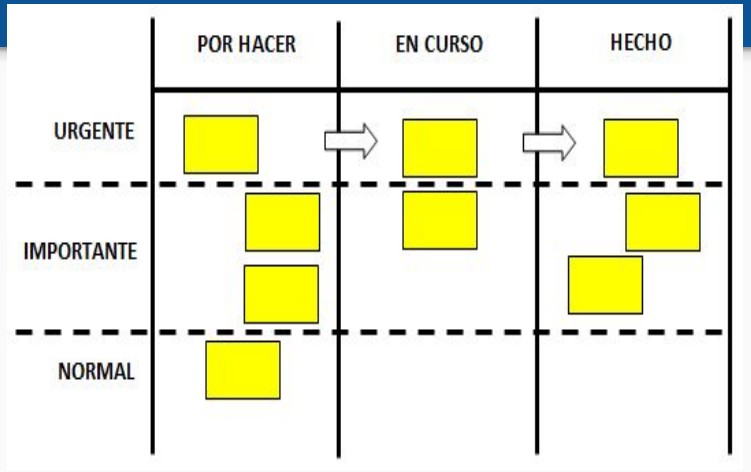
- **Product Vision:** define el propósito del producto, qué problema resuelve, a quién sirve y qué valor entrega.
- **Is, Is not | Does, Does not:** Una matriz que delimita claramente qué será y qué no será el producto, así como qué hará y qué no hará.
- **Describe the personas/Describir las personas:** Definir perfiles ficticios de usuarios típicos o claves que usarán el producto.
- **Discover the features/Descubrir las funcionalidades:** Identificar funcionalidades clave que el producto podría o debería tener.
- **Technical and Business review/Revisión técnica y de negocio:** Análisis de cada funcionalidad desde dos puntos de vista: Negocio: ¿Aporta valor? ¿Es prioritaria? Técnico: ¿Es factible? ¿Tiene complejidad alta o baja?
- **Show the User Journeys/Mostrar los recorridos del usuario:** Mapear los pasos que un usuario sigue para lograr un objetivo específico usando el producto.
- **Display features in journeys/Ubicar funcionalidades en los recorridos:** Relacionar las funcionalidades descubiertas con los pasos de los recorridos del usuario.
- **Sequence the features/Secuenciar las funcionalidades:** Ordenar las funcionalidades según: Dependencias técnicas. Valor de negocio. Impacto al usuario
- **Build MVP Canvas/Construir el lienzo del MVP:**Un lienzo visual para definir de manera colaborativa el Producto Mínimo Viable. Componentes típicos del lienzo MVP: Problema Personas Solución propuesta. Funcionalidades esenciales. Métricas de éxito. Supuestos a validar. Riesgos
- **Demo:** Una presentación final a los interesados (stakeholders) sobre lo trabajado durante la Lean Inception. Objetivo: Validar la alineación del equipo. Compartir la visión del MVP. Obtener retroalimentación. Aprobar el siguiente paso (desarrollo).

Kanban

Kanban es un marco de trabajo muy popular a la hora de implementar un desarrollo de software ágil. Requiere una comunicación en tiempo real sobre la capacidad y una total transparencia del trabajo. Los elementos de trabajo se representan visualmente en un tablero de kanban, lo que permite a los miembros del equipo ver el estado de cada uno en cualquier momento.

¿Cómo estructurar el flujo de kanban?

- **Visualiza el flujo de trabajo:** empieza por visualizar el flujo de trabajo de tu equipo en un tablero de Kanban. Ya sea físico o virtual, el tablero debe representar cada fase del proceso de desarrollo, desde el inicio hasta la finalización de la tarea.
- **Estandariza el flujo de trabajo:** define y estandariza las etapas del flujo de trabajo según los procesos y requisitos de tu equipo. Las etapas más comunes son "Por hacer", "En progreso" y "Hecho", pero personaliza según sea necesario para reflejar tu flujo de trabajo único.
- **Identifica los impedimentos y las dependencias:** kanban permite identificar inmediatamente los impedimentos y las dependencias. Esta transparencia permite una resolución rápida y evita las interrupciones del flujo de trabajo.
- **Fija límites de trabajo en progreso (WIP):** implementa límites de WIP en cada fase del flujo de trabajo para evitar la sobrecarga y mantener un flujo de trabajo estable. Los límites de WIP ayudan a optimizar la asignación de recursos y a reducir la multitarea, lo que fomenta una mayor productividad.
- **Fomenta la colaboración:** favorece una cultura de colaboración en tu equipo, en la que los miembros aborden colectivamente los cuellos de botella y trabajen juntos para garantizar una progresión fluida del flujo de trabajo. Este enfoque colaborativo promueve la eficiencia y acelera la finalización de las tareas.
- **Utiliza tarjetas de kanban:** representa cada tarea como una tarjeta de kanban en el tablero, con detalles esenciales como la descripción de la tarea, la persona asignada y el tiempo estimado de finalización. Las tarjetas de kanban facilitan el seguimiento visual del progreso de las tareas y promueven la transparencia dentro del equipo.



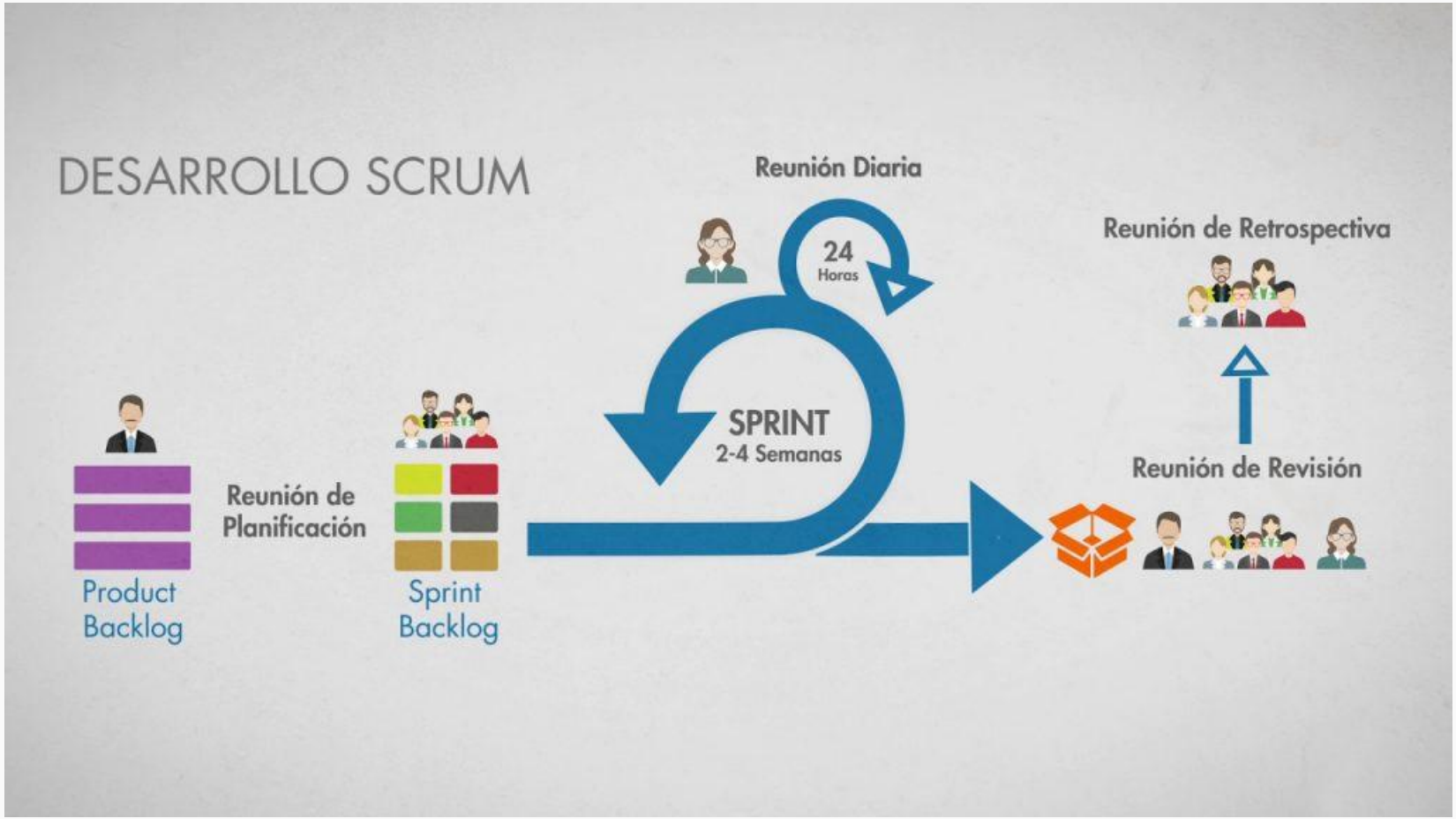
Metodologías ágiles

Tema	Metodología ágil			
	SCRUM	**XP**	**LEAN INCEPTION**	**KANBAN**
Iteraciones	Sprints	Iteraciones (1-2 semanas)	No aplica (fase previa)	Flujo continuo
Enfoque	Gestión del proyecto	Calidad del código	Definir MVP	Flujo de trabajo
Roles definidos	Sí	No obligatorio	Facilitador / equipo	No obligatorio
Herramientas clave	Backlog, Scrum board	TDD, Pair programming	Talleres colaborativos	Tablero Kanban
Ideal para	Proyectos medianos a grandes	Proyectos críticos donde la calidad técnica es esencial	Fase de inicio para alinear visión y valor	Equipos con tareas constantes o soporte
Cambio de requerimientos	Moderado	Muy alto	Antes del desarrollo	Se adapta fácilmente

SCRUM

SCRUM

El scrum es un marco ágil que ayuda a los equipos a estructurar su trabajo en ciclos de desarrollo cortos denominados "sprints". Los equipos de scrum se comprometen a entregar el trabajo al final de cada sprint y adoptan prácticas y una estructura de equipo que les ayuden a alcanzar este ritmo. El scrum lleva los principios de la metodología ágil un paso más allá y crea una estructura que ayuda a los equipos a aplicarlos en su trabajo diario. El scrum es un marco ágil bien documentado que muchos equipos pueden adoptar sin mucha interrupción.


DESARROLLO SCRUM
Reunión Diaria
24
Horas
Reunión de Retrospectiva
SPRINT
2-4 Semanas
Reunión de Revisión
Reunión de
Planificación
Product
Backlog
Sprint
Backlog

SCRUM: Pilares empíricos

- **Transparencia** Los aspectos significativos del proceso deben ser visibles para aquellos que son responsables del resultado. La transparencia requiere que dichos aspectos sean definidos por un estándar común, de tal modo que los observadores compartan un entendimiento común de lo que se está viendo.
- **Inspección:** Los usuarios de Scrum deben inspeccionar frecuentemente los artefactos de Scrum y el progreso hacia un objetivo, para detectar variaciones. Su inspección no debe ser tan frecuente como para que interfiera en el trabajo. Las inspecciones son más beneficiosas cuando se realizan de forma diligente por inspectores expertos, en el mismo lugar de trabajo.
- **Adaptación:** Si uno o más aspectos de un proceso se desvían de límites aceptables o si el producto resultante no será aceptable, el proceso o el material que está siendo procesado deben ser ajustados. Dicho ajuste debe realizarse cuanto antes para minimizar desviaciones mayores.

SCRUM: Roles

- **Product Owner o responsable del producto**

Esta es la persona a cargo de la lista de trabajo pendiente del producto (product backlog). Está conectado a las necesidades del usuario y se centra en transmitir el punto de vista del usuario a su equipo y a otros ejecutivos involucrados. Los buenos encargados de productos aportan claridad sobre qué es lo siguiente que se debe entregar debido a su importancia. En última instancia, deberían ser ellos quienes decidan cuándo algo está listo para ser entregado (con una tendencia a realizar entregas con frecuencia).

- **Scrum Master**

El Scrum Master es la persona que dirige los distintos eventos de Scrum. Considéralo el gerente del proyecto y facilitador de Scrum. El Scrum Master debe promover las reuniones diarias de actualización y organizar las reuniones de planificación, revisión y análisis retrospectivo del sprint.

- **Equipo Scrum**

El equipo de Scrum son todos los que están trabajando en el sprint. Los miembros del equipo deben auto-organizarse y ser colaborativos para lograr el objetivo de Scrum de mejora continua.

SCRUM: Iteraciones

Los sprints y las iteraciones en Scrum son ciclos limitados en el tiempo que suelen oscilar entre una y cuatro semanas, durante los cuales los equipos de Scrum trabajan para garantizar que se produzca un valor específico.

- Ciclo de trabajo fijo (usualmente de 1 a 4 semanas).
- Comienza con una planificación y termina con una revisión y retrospectiva.
- Durante el Sprint, no se deben hacer cambios significativos que afecten el objetivo del Sprint.
- Entregable: un Incremento de producto funcional y potencialmente entregable.

SCRUM: Iteraciones

Aunque los sprints y las iteraciones son componentes de metodologías ágiles, tienen diferencias clave. Los sprints son solo para Scrum, mientras que las iteraciones se pueden utilizar en el enfoque ágil general.
Los sprints son intervalos de trabajo limitados a un período establecido de entre 2 y 4 semanas, mientras que las iteraciones pueden durar más. Los sprints son un conjunto de actividades definidas como la planificación del sprint y la revisión del sprint, pero las iteraciones no lo son. Las iteraciones suelen estar predeterminadas para la duración de un proyecto, mientras que los sprints se pueden cambiar según los requisitos del proyecto.

SCRUM: Tipos de reuniones

Reunión	Frecuencia	Duración sugerida (para un Sprint de 2 semanas)	Objetivos
Sprint Planning	Al inicio de cada Sprint	2-4 horas	Planificar qué se va a hacer en el Sprint y cómo.
Daily Scrum (Scrum Diario)	Todos los días	15 minutos	Sincronizar al equipo y detectar impedimentos
Sprint Review	Al final del Sprint	1-2 horas	Mostrar lo realizado, recoger feedback de stakeholders
Sprint Retrospective	Después de la Review, al final del Sprint	1-1.5 horas	Evaluar y mejorar el proceso del equipo
Backlog Refinement (no oficial pero recomendado)	1-2 veces por Sprint	Variable	Refinar, dividir y estimar ítems del backlog

SCRUM: Artefactos

Los artefactos Scrum son herramientas imprescindibles de lo que es Scrum. En Scrum, un artefacto es algo que creas, como una herramienta para resolver un problema. Existen tres artefactos que definen qué es Scrum: la pila de producto o product backlog, la pila de sprint o sprint backlog y el incremento del producto.

SCRUM: Artefactos - Pila de producto o Product backlog

El Product Backlog es el artefacto Scrum que recoge la lista del trabajo que debe realizarse.
Es una lista priorizada de todo lo que se necesita para construir o mejorar el producto. Representa el único origen de requisitos para cualquier cambio a realizar en el producto.
El encargado del Product Backlog o pila de producto es el product owner que deberá clasificar los elementos de esta lista.

Características:

- Es dinámico y evoluciona continuamente.
- Cada ítem del backlog se denomina PBI (Product Backlog Item).
- Contiene funcionalidades, mejoras, corrección de errores, tareas técnicas, etc.
- Está ordenado por prioridad y valor para el cliente.
- Solo el Product Owner puede modificar su contenido o prioridad.

Contenido típico de un PBI:

- Título y descripción.
- Criterios de aceptación.
- Estimación de esfuerzo.
- Prioridad o valor de negocio.

SCRUM: Artefactos - Sprint Backlog

Es un subconjunto del Product Backlog que contiene los elementos seleccionados para ser desarrollados durante un Sprint específico, junto con un plan para entregarlos.

- Se crea durante la Sprint Planning.
- Contiene:
  - Los PBIs seleccionados para el Sprint.
  - Tareas técnicas necesarias para implementar cada ítem.
  - Objetivo del Sprint (Sprint Goal).
- Es propiedad del equipo de desarrollo.
- Es actualizado diariamente durante el Sprint (por ejemplo, en el Daily Scrum).
- El equipo de desarrollo se autoorganiza y gestiona este artefacto.

SCRUM: Artefactos - Incremento

- El incremento del producto es lo que se entrega al final de cada sprint. Este artefacto Scrum puede consistir en un nuevo producto o función, una mejora o corrección de errores, o cualquier otra cosa dependiendo de tu equipo.
- Es el conjunto de todos los elementos del Product Backlog que fueron completados durante un Sprint, más los incrementos de Sprints anteriores. Debe ser un producto funcional y potencialmente entregable.
- Características:
  - Debe cumplir con la Definición de "Terminado" (Definition of Done).
  - Es verificable y usable.
  - Se presenta en la Sprint Review.
  - Cada Sprint debe producir al menos un Incremento.
- ¿Quién lo entrega? El equipo de desarrollo, con apoyo del Scrum Master y supervisión del Product Owner.

- Definición de "Terminado" (Definition of Done - DoD)

Un acuerdo común del equipo sobre qué significa que un trabajo esté completo. Incluye criterios técnicos, funcionales y de calidad. Es fundamental para asegurar la calidad de los entregables.
Ejemplo de elementos en la DoD: Código desarrollado y revisado. Pruebas unitarias y de integración ejecutadas y pasadas. Documentación actualizada. Desplegado en un entorno de pruebas. Aprobación por el Product Owner.

SCRUM: Implementación

Para implementar Scrum en una organización o equipo, se suelen seguir estos pasos:

1. Formación del equipo Scrum: Se definen los roles (Scrum Master, Product Owner y Developers).
2. Creación del Product Backlog: Lista priorizada de características, requerimientos o tareas del producto.
3. Planificación del Sprint (Sprint Planning): Se seleccionan los elementos del Product Backlog que se trabajarán en ese Sprint.
4. Ejecución del Sprint: El equipo trabaja durante un tiempo fijo (generalmente 2-4 semanas).
5. Reuniones diarias (Daily Scrum): Para sincronizar esfuerzos y resolver impedimentos.
6. Revisión del Sprint (Sprint Review): Presentación del incremento funcional terminado al Product Owner y stakeholders.
7. Retrospectiva del Sprint (Sprint Retrospective): Análisis del proceso para mejorar continuamente.
8. Ciclo repetitivo: Se repite el proceso con los siguientes elementos del Product Backlog.