Taller Práctico: Introducción a Metodologías Ágiles y Scrum

Integrante: Victor Ramírez García

Rut: 16.767.619-7

1. Descripción de los Fundamentos de las Metodologías Ágiles (2 puntos):

-- Explicar los orígenes de las metodologías ágiles y las diferencias clave con los modelos tradicionales de desarrollo como Waterfall y RUP.

Las metodologías ágiles aparecieron hace más de 30 años, a partir de la necesidad de detectar y corregir errores, o realizar cambios de la manera más rápida posible. Para abordar situaciones imprevistas o cambios de contexto, los enfoques tradicionales no lograban dar señas temprana de éxito o fracaso, ya que se debía esperar al final del desarrollo para hacer el balance: tomaban demasiado tiempo en entregar valor, no era adaptable a cambios, se centraba en hacer mucha documentación antes del desarrollo y los equipos tenían poca interacción con el cliente lo que llevaba a muchos errores, estimaciones de tiempo incorrectas o requerimientos incorrectos.

Luego de algunas iniciativas por cambiar estas metodologías rígidas (como el Modelo Waterfall), en 2001 se creó el Manifiesto Ágil, que oficializó esta metodología de desarrollo. El manifiesto Ágil fue creado por un grupo de desarrolladores en Utah, con el objetivo de mejorar el proceso de desarrollo de software, enfocándose en la colaboración, **flexibilidad** y entrega continua de valor.

Las metodologías ágiles se basan en este Manifiesto Ágil y difieren de los enfoques tradicionales en lo siguiente:

- Waterfall: Modelo secuencial donde cada fase debe completarse antes de avanzar.

Los cambios son difíciles de manejar una vez que se pasa de fase. Agile propone separar y subdividir tareas que son revisadas periódicamente con el objetivo de reaccionar con tiempo y readaptar el trabajo en caso de existir imprevistos.

- RUP (Rational Unified Process): Aunque es iterativo, RUP sigue un enfoque más

estructurado, puede ser costoso en proyectos pequeños, requiere un análisis y documentación extensos al inicio, y los riesgos se gestionan desde el inicio mediante identificación y mitigación. Mientras que Agile es más flexible y

adaptativo a los cambios constantes con planificación progresiva y los riesgos se manejan con entregas frecuentes y retroalimentación.

- PMBOK: Un enfoque que prioriza la planificación detallada y el control estricto del

proyecto mediante un conjunto de buenas prácticas, procesos y estándares. Pese a que es útil en proyectos grandes que requieren una planificación y documentación detallada, se presenta bastante rígida y protocolar para proyectos que tienen requisitos o situaciones en constante cambio. Agile, en cambio, promueve una mayor flexibilidad y adaptación continua a los requisitos.

Aspecto	Waterfall	RUP	PMBOK	Agile
Enfoque	Secuencial y lineal. El proyecto sigue fases rígidas en orden.	Basado en procesos y planificación. Se centra en definir y seguir un marco estructurado para el desarrollo.	Basado en procesos y planificación detallada. Define estándares y mejores prácticas para la gestión de proyectos en cualquier industria.	Iterativo y adaptable. Prioriza la entrega rápida de valor al usuario y la flexibilidad ante cambios.
Planificación	Se realiza al inicio, con un plan detallado de todo el proyecto en fases (1.Requerimientos, 2.Diseño, 3.Implementación, 4.Pruebas, 5.Despliegue, 6.Mantenimiento)	Requiere una planificación detallada al inicio, con fases bien definidas.	Se elabora un plan maestro con objetivos, cronograma, costos y riesgos desde el inicio.	Se planifica progresivamente, adaptándose a nuevas necesidades en cada iteración.
Flexibilidad	Poco flexible, los cambios requieren volver a fases anteriores.	Más rígido ante cambios (necesita aprobación formal).	Poco flexible, los cambios deben pasar por procesos formales de gestión del cambio.	Altamente adaptable a cambios en requisitos.
Entrega de Valor	Se entrega el producto solo al final del proyecto.	Se entregan versiones funcionales en cada iteración, pero el producto completo se lanza en la fase final.	Se entrega el producto al final del proyecto o en hitos predefinidos.	Se entregan versiones funcionales en cada iteración.
Documentación	Extensa y detallada, con documentos para cada fase.	Extensa, detallada y obligatoria en cada fase.	Extensa y detallada, con informes formales en cada fase.	Mínima, solo lo esencial para el desarrollo.
Gestión del riesgo	Los riesgos se analizan al inicio, pero pueden ser difíciles de manejar si surgen tarde.	Se identifican y gestionan los riesgos desde el inicio del proyecto.	Se identifican y mitigan riesgos desde el inicio del proyecto.	Los riesgos se abordan de manera iterativa con entregas continuas y retroalimentación.
Tamaño del	Equipos grandes con	Equipos grandes	Equipos grandes, con	Equipos pequeños y

equipo	roles especializados.	y estructurados con roles específicos.	roles bien definidos.	multidisciplinarios, autogestionados.
Jerarquía	Estructura jerárquica, con decisiones centralizadas.	Jerárquico, con roles bien definidos (analistas, arquitectos, desarrolladores, testers, etc.).	Estructura jerárquica, con niveles de liderazgo y supervisión.	Horizontal, con comunicación constante entre los miembros.
Roles principales	- **Gerente de Proyecto** (dirige el proceso). - **Analistas** (definen requisitos). - **Arquitectos** (diseñan la solución). - **Desarrolladores** (implementan el código). - **Testers** (prueban el software).	- **Gerente de Proyecto** (liderazgo y gestión). - **Analista de Negocios** (requisitos y especificaciones). - **Arquitecto de Software** (diseño de la arquitectura). - **Desarrolladores** (implementación). - **Testers** (pruebas y control de calidad).	- **Gerente de Proyecto (Project Manager)** (supervisa todo el proyecto). - **Patrocinador (Sponsor)** (apoya financieramente el proyecto). - **Equipo del Proyecto** (ejecuta tareas específicas).	- **Scrum Master** (facilitador del equipo). - **Product Owner** (gestiona el backlog y prioridades). - **Equipo de Desarrollo** (multidisciplinario, sin jerarquías fijas).
Interacción con el cliente	Se define al inicio y al final. Poco contacto durante el desarrollo.	Se define al inicio y se revisa en ciertas fases clave.	Se define al inicio y se mantiene a través de informes y reuniones planificadas.	Continua y frecuente, con iteraciones basadas en retroalimentación.
Comunicación	Formal, con reuniones planificadas y documentación detallada.	Formal, con reuniones planificadas y documentación extensa.	Formal, con informes y reuniones de seguimiento.	Informal y constante, con reuniones diarias (Daily Scrum en Scrum).

Según lo descrito en esta tabla, podemos destacar

1. Enfoque:
    - Waterfall: Secuencial y rígido, con fases definidas.
    - RUP: Basado en procesos estructurados y planificación detallada.
    - PMBOK: Enfatiza estándares y mejores prácticas en gestión de proyectos.
    - Agile: Iterativo, flexible y enfocado en la entrega rápida de valor.
2. Planificación:
    - Waterfall, RUP y PMBOK: Planificación detallada desde el inicio.
    - Agile: Planificación progresiva, ajustándose en cada iteración.
3. Flexibilidad:
    - Waterfall y PMBOK: Poco flexibles; los cambios requieren procesos formales.
    - RUP: Moderadamente rígido, con aprobación formal para cambios.
    - Agile: Altamente adaptable a cambios.
4. Entrega de Valor:
    - Waterfall y PMBOK: Producto entregado al final o en hitos.
    - RUP: Versiones funcionales en cada iteración, pero entrega final estructurada.
    - Agile: Entregas funcionales continuas.
5. Documentación:
    - Waterfall, RUP y PMBOK: Extensa y detallada.
    - Agile: Mínima, solo lo esencial.
6. Gestión del Riesgo:
    - Waterfall: Difícil de manejar si surgen tarde.
    - RUP y PMBOK: Identificación y mitigación desde el inicio.
    - Agile: Manejo iterativo con retroalimentación constante.
7. Tamaño del Equipo:
    - Waterfall, RUP y PMBOK: Equipos grandes y estructurados.
    - Agile: Equipos pequeños, multidisciplinarios y autogestionados.
8. Jerarquía:
    - Waterfall, RUP y PMBOK: Estructura jerárquica.
    - Agile: Organización horizontal y colaborativa.
9. Roles Principales:
    - Waterfall y RUP: Roles bien definidos con gestión centralizada.
    - PMBOK: Incluye gerente de proyecto y patrocinador.
    - Agile: Scrum Master, Product Owner y equipo autogestionado.
10. Interacción con el Cliente:

- Waterfall: Contacto mínimo.
- RUP y PMBOK: Interacción en fases clave o mediante informes.
- Agile: Comunicación continua y basada en retroalimentación.

11. Comunicación:

- Waterfall, RUP y PMBOK: Formal, con reuniones planificadas.
- Agile: Informal y constante, con reuniones diarias.

Identificar las diferencias entre Agile, Agilidad y Agilismo, describiendo cómo se aplican en el contexto actual del desarrollo de software.

Tip: Explica con claridad cómo las metodologías ágiles promueven la flexibilidad y la entrega de valor continuo en comparación con modelos más rígidos.

- **Agile:** Es un conjunto de metodologías basadas en los principios del Manifiesto Ágil. Algunas de las más conocidas son Scrum (entregas rápidas en ciclos cortos), XP (prioriza la calidad del código), Kanban (flujo de trabajo continuo), Lean Software Development (eficiencia en equipos grandes), Crystal (flexible para equipos pequeños o medianos), Feature-Driven Development – FDD (estructura para proyectos grandes) y Dynamic Systems Development Method – DSDM (estructura en proyectos de gran escala).
- **Agilidad**: Representa la capacidad de un equipo u organización para adaptarse de manera rápida y efectiva a los cambios y nuevas necesidades. Un equipo ágil, por ejemplo, puede modificar su estrategia en respuesta a imprevistos sin comprometer sus objetivos.
- **Agilismo:** Es la mentalidad o cultura organizacional que adopta los principios de Agile en diferentes áreas, más allá del desarrollo de software, como negocios, educación o marketing. Implementarlo implica flexibilidad en la planificación, iteración en los procesos y aprendizaje continuo basado en pruebas y resultados. Sin embargo, muchas empresas lo malinterpretan como simplemente "trabajar más rápido".

2. Valores y Principios del Manifiesto Ágil (1.5 puntos):

-- Identificar los 4 valores y los 12 principios del Manifiesto Ágil. Explicar cómo estos principios guían la toma de decisiones en equipos de desarrollo ágiles. Tip: Relaciona los principios ágiles con ejemplos concretos en proyectos reales o experiencias vividas.

En 2001, un grupo de 17 desarrolladores de software creó el **Manifiesto Ágil** con la intención de optimizar el proceso de desarrollo, priorizando la colaboración, la flexibilidad y la entrega continua de valor.

Para poner en práctica esta filosofía, el Manifiesto Ágil establece **4 valores fundamentales** y **12 principios esenciales**:

Valores:

1. Priorizar a las personas y su interacción por encima de procesos y herramientas.
2. Dar más importancia al software en funcionamiento que a la documentación extensa.
3. Favorecer la colaboración con el cliente en lugar de centrarse solo en contratos.
4. Adaptarse a los cambios en lugar de seguir un plan rígido.

Principios:

1. La satisfacción del cliente es lo más importante, lograda a través de entregas rápidas y continuas.
2. Se deben aceptar cambios en los requisitos en cualquier etapa del proyecto.
3. El software funcional debe entregarse de manera frecuente.
4. Equipos de desarrollo y de negocio deben trabajar en conjunto.
5. Es clave contar con equipos motivados, apoyados y de confianza.
6. La mejor comunicación es la directa y cara a cara.
7. El éxito se mide principalmente por el software en funcionamiento.
8. Se debe promover un ritmo de trabajo sostenible para evitar el agotamiento del equipo.
9. La excelencia técnica y el buen diseño contribuyen a una mayor agilidad.
10. Mantener la simplicidad y la eficiencia en las tareas.
11. Equipos autoorganizados producen mejores soluciones.
12. Reflexionar periódicamente y mejorar continuamente el proceso.

La aplicación de estos principios permite a los equipos evaluar qué elementos del proyecto pueden ajustarse o postergarse en función de las prioridades del momento. Por ejemplo, si se requiere entregar una versión en una fecha específica, pero surgen cambios en los requisitos, se podrían tomar decisiones como aplazar la documentación técnica para el final, enfocarse en garantizar la funcionalidad principal, deshabilitar temporalmente ciertas características o integrar el diseño de manera progresiva.

Estas decisiones, alineadas con los principios ágiles, garantizan un desarrollo flexible, adaptable y sostenible, promoviendo la mejora continua a lo largo del proyecto.

3. Explicación del Marco de Trabajo Scrum y Roles (2 puntos):

-- Explicar los principios y valores de Scrum y cómo este marco de trabajo ágil ayuda a organizar el desarrollo de proyectos.

Scrum es un marco de trabajo ágil diseñado para gestionar el desarrollo de software y productos digitales en entornos con requisitos cambiantes. Su estructura se basa en ciclos de trabajo cortos e iterativos llamados **sprints**, que tienen una duración definida por el equipo, generalmente entre 1 y 4 semanas.

Este enfoque se alinea con los principios del **Manifiesto Ágil**, promoviendo la colaboración, la transparencia y la mejora continua.

Principios Fundamentales de Scrum:

1. **Transparencia:** Todo el equipo tiene visibilidad sobre el progreso y los desafíos del proyecto.
2. **Inspección:** Se evalúa regularmente el avance y la calidad del trabajo.

3. **Adaptación:** Se realizan ajustes según sea necesario para mejorar los resultados.

Los 5 Valores Clave de Scrum:

1. **Compromiso:** Cada miembro del equipo asume la responsabilidad de alcanzar los objetivos del sprint.
2. **Coraje:** Se fomenta la toma de decisiones y el enfrentamiento de desafíos con confianza.
3. **Enfoque:** Se priorizan las tareas más importantes en cada sprint para maximizar el impacto.
4. **Respeto:** Se valora y respeta a cada integrante del equipo como profesional.
5. **Apertura:** Se promueve la retroalimentación y la disposición a mejorar continuamente.

Scrum facilita un ambiente de trabajo **horizontal**, donde cada integrante tiene la capacidad de proponer mejoras y ajustes, siempre con el objetivo de optimizar la calidad del producto. Al adoptar estos valores, los equipos pueden mantenerse enfocados, desarrollar pensamiento crítico y creativo, enfrentar obstáculos con determinación y apoyarse mutuamente en la búsqueda de soluciones.

Además, el enfoque iterativo de Scrum permite que el trabajo realizado no se considere simplemente como una tarea terminada, sino como un proceso en constante evolución, impulsando el compromiso con los objetivos del sprint y la mejora continua.

--Describir los roles clave en Scrum (ScrumMaster, ScrumDeveloper, ProductOwner) y sus responsabilidades dentro de un equipo ágil.

1 Tip: Utiliza ejemplos que demuestren cómo los roles trabajan en conjunto en un entorno ágil.

Roles en Scrum

Scrum Master: facilitador del equipo

El **Scrum Master** es el encargado de guiar al equipo en la implementación de Scrum, asegurándose de que se respeten sus principios y prácticas. Su rol es actuar como un coach, ayudando a eliminar obstáculos, fomentando la mejora continua y garantizando que el equipo funcione de manera eficiente dentro del marco ágil.

Scrum Developer (Team Members): responsables del desarrollo

Los **Scrum Developers** son los miembros del equipo encargados de llevar a cabo el trabajo técnico. Su labor incluye estimar tareas, desarrollar, probar e implementar incrementos funcionales del producto en cada sprint. Se organizan de manera autónoma, trabajan en colaboración y aportan distintas especialidades para asegurar la calidad del producto.

Product Owner: gestiona el backlog y define prioridades

El **Product Owner** se encarga de maximizar el valor del producto, asegurando que el equipo se enfoque en las tareas más importantes. Es responsable de gestionar el **Product Backlog**, organizando y priorizando funcionalidades en función de su impacto. Además, actúa como enlace directo con los stakeholders, comunicando necesidades y expectativas del producto.

Ejemplo de interacción entre roles

Para entender cómo trabajan estos roles en conjunto, imaginemos un equipo que debe desarrollar la funcionalidad de gestión de cuentas de usuario y tareas en una aplicación:

1. **El Product Owner** define en el Product Backlog las características esenciales, como la creación de cuentas, asignación de tareas y notificaciones de estado ("pendiente", "hecho", "archivado").
2. **El Scrum Master** facilita la reunión de planificación, asegurando que el proceso fluya sin problemas.
3. **Los Scrum Developers** seleccionan las historias de usuario que abordarán en el sprint y crean el **Sprint Backlog**.
4. Durante el sprint, el equipo trabaja colaborativamente:
    - **El Product Owner** está disponible para resolver dudas y aclarar requisitos.
    - **El Scrum Master** supervisa que no haya bloqueos y que el equipo siga las prácticas ágiles.
    - **Los Developers** diseñan la interfaz, programan el backend y ejecutan pruebas.
5. Al finalizar el sprint, el **Product Owner** revisa si los objetivos se cumplieron, los clientes prueban la aplicación y el equipo realiza una retrospectiva para analizar mejoras para futuros sprints.

Este ciclo iterativo permite una evolución constante del producto, asegurando calidad, flexibilidad y alineación con las necesidades del usuario.

4. Identificación de las Prácticas y Artefactos en Scrum (1.5 puntos):

-- Describir las prácticas principales de Scrum: SprintPlanning, DailyScrum, SprintReview, SprintRetrospective. Explicar los artefactos de Scrum: ProductBacklog, SprintBacklog, Incremento, y cómo contribuyen al progreso iterativo del proyecto. Tip: Relaciona cada artefacto con su rol dentro del ciclo de vida de un proyecto Scrum.

Principales prácticas de Scrum: Ceremonias

Scrum incluye varias ceremonias clave que estructuran el flujo de trabajo del equipo:

1. **Sprint Planning:** Se define el trabajo a realizar en el sprint. El **Scrum Master** facilita la reunión, el **Product Owner** presenta las prioridades del **Product Backlog**, y el **Development Team** selecciona las tareas y planifica su ejecución.
2. **Daily Scrum:** Breve reunión diaria (máximo 15 minutos) en la que el equipo revisa el progreso y detecta obstáculos. Cada miembro responde tres preguntas clave: *¿Qué hice ayer?*, *¿Qué haré hoy?* y *¿Tengo algún impedimento?*
3. **Sprint Review:** Se presenta el trabajo finalizado al cliente con el objetivo de obtener retroalimentación y determinar si se requieren ajustes.
4. **Sprint Retrospective:** Espacio de reflexión donde el equipo analiza su desempeño en el sprint y propone mejoras para el siguiente.
5. **Sprint Refinement (Refinamiento del Backlog):** Aunque no es una ceremonia oficial, es común realizar una reunión para planificar con antelación el siguiente sprint y aplicar mejoras derivadas de la retrospectiva.

Beneficios de las ceremonias Scrum:

Estas reuniones garantizan que el equipo:

- Tenga una comprensión clara de las tareas a realizar.
- Mantenga el enfoque y la alineación.
- Reciba retroalimentación continua.
- Optimice su proceso de trabajo de forma iterativa.

Artefactos principales de Scrum

Scrum utiliza varios artefactos para organizar el trabajo y garantizar la entrega de valor:

- **Product Backlog:** Lista priorizada de características, mejoras y correcciones necesarias en el producto.
    - Incluye historias de usuario, requisitos técnicos y ajustes.
    - Evoluciona constantemente con base en las necesidades del negocio y el feedback del usuario.
    - El **Product Owner** es responsable de su gestión y priorización, mientras que el equipo lo refina de manera continua.
- **Sprint Backlog:** Conjunto de tareas seleccionadas del **Product Backlog** que el equipo trabajará durante el sprint.
    - Se define en el **Sprint Planning** e incluye únicamente tareas que puedan completarse dentro del sprint.
    - Puede ajustarse en función del progreso del trabajo.
    - Es responsabilidad del **Development Team**, que decide qué incluir y cómo ejecutarlo, con el apoyo del **Scrum Master** para eliminar impedimentos.
- **Incremento:** Versión funcional del producto con las nuevas características desarrolladas en el sprint.
    - Debe estar listo para su uso y cumplir con los estándares de calidad.

- Se presenta en la **Sprint Review** y se acumula con incrementos previos hasta construir el producto final.
- El **Development Team** garantiza que cumpla con la *Definition of Done (DoD)*, mientras que el **Product Owner** verifica que responda a las necesidades del negocio y gestiona la retroalimentación de los stakeholders o clientes.

Este marco de trabajo permite entregar valor de manera continua, fomentar la colaboración y mejorar el producto de forma iterativa.