Tecnológico Nacional de México
Instituto Tecnológico De Tijuana

Subdirección Académica
Departamento De Sistemas Y Computación

Semestre:
Enero - Junio 2024

Carrera:
Ing. Sistemas Computacionales

Materia: Temas Avanzados de Desarrollo de Software

Título: Unidad 4 Actividad 1

Unidad a evaluar:
Unidad 4

Nombre Y Número De Control Del Alumno:
Vargas Reyes Edwin Alejandro 19290961

Nombre Del Maestro (A):
Humberto Raiz Orozco

26 de Abril de 2024

Paradigma de agilidad en el desarrollo de software.

El enfoque ágil para el desarrollo de software busca distribuir de forma permanente sistemas de software en funcionamiento diseñados con iteraciones rápidas.

Las metodologías ágiles de desarrollo de software buscan proporcionar en poco tiempo pequeñas piezas de software en funcionamiento para aumentar la satisfacción del cliente. Estas metodologías utilizan enfoques flexibles y el trabajo en equipo para ofrecer mejoras constantes. Por lo general, el desarrollo ágil de software implica que pequeños equipos autoorganizados de desarrolladores y representantes empresariales se reúnan regularmente en persona durante el ciclo de vida del desarrollo de software. La metodología ágil favorece un enfoque sencillo de la documentación de software y acepta los cambios que puedan surgir en las diferentes etapas del ciclo de vida, en lugar de resistirse a ellos.

Principios Clave del Desarrollo Ágil:

1. **Iteración e Incremento**: Los proyectos ágiles se dividen en sprints o iteraciones cortas, que suelen durar entre una y cuatro semanas. Cada iteración produce una versión del software que es funcional y que puede ser probada y utilizada, permitiendo que se realicen ajustes antes de la siguiente iteración.
2. **Colaboración Continua**: El desarrollo ágil enfatiza la comunicación y colaboración constante entre todos los involucrados en el proyecto, incluyendo desarrolladores, gestores de proyecto, y clientes. Esto ayuda a asegurar que el producto final cumpla con las necesidades del usuario y que todos los miembros del equipo estén alineados con los objetivos del proyecto.
3. **Flexibilidad y Adaptabilidad**: A diferencia de los métodos tradicionales, el desarrollo ágil permite y fomenta los cambios en los requisitos del software incluso en etapas tardías del desarrollo. Esto es crucial en un entorno empresarial cambiante y competitivo.
4. **Enfoque en el Usuario y en la Entrega de Valor**: Los métodos ágiles priorizan la creación de software que realmente entregue valor al usuario final. Esto se logra a través de la implementación de características basadas en su prioridad y su impacto en el usuario final.
5. **Mejora Continua**: Después de cada iteración, los equipos se reúnen para discutir qué funcionó bien y qué puede mejorarse en el futuro. Este proceso de reflexión y adaptación es fundamental para mejorar la eficiencia y la calidad del desarrollo de software.

Metodologías Ágiles Populares:

- **Scrum**: Quizás el marco ágil más conocido, donde un Scrum Master facilita el proceso a un equipo autoorganizado y multifuncional. Las tareas se dividen en sprints con revisiones y ajustes regulares.
- **Kanban**: Enfocado en la visualización del flujo de trabajo, limitación del trabajo en progreso y la mejora continua del proceso.
- **Extreme Programming (XP)**: Enfoca en la excelencia técnica y la capacidad de adaptarse a los cambios en los requisitos del cliente a través de prácticas como la programación en pareja, desarrollo dirigido por pruebas, y despliegues frecuentes.

Tabla Comparativa

Aspecto	SCRUM	Extreme Programming (XP)	DevOps
Objetivo principal	Facilitar la gestión del proyecto y maximizar la transparencia en el equipo de desarrollo.	Mejorar la calidad del software y la capacidad de respuesta a los requisitos cambiantes del cliente.	Unificar el desarrollo de software y las operaciones para mejorar la colaboración y acelerar la entrega de software.
Enfoque	Gestión de proyectos y proceso de desarrollo iterativo e incremental.	Técnicas de desarrollo de software, incluyendo pruebas y programación en pareja.	Integración y entrega continuas, automatización de infraestructura, y operaciones.
Prácticas clave	Sprints, Scrum diario, revisiones de sprint, retrospectivas.	Programación en pareja, desarrollo dirigido por pruebas (TDD), integración continua, diseño simple.	Integración continua (CI), entrega continua (CD), monitoreo, automatización y uso de herramientas de gestión de infraestructuras como código.
Roles específicos	Scrum Master, Product Owner, Equipo de Desarrollo.	No define roles específicos de manera rígida; el enfoque está más	DevOps no define roles específicos pero promueve roles como Ingenieros de DevOps,

Aspecto	SCRUM	Extreme Programming (XP)	DevOps
		en las prácticas.	que combinan habilidades de desarrollo y operaciones.
Ciclo de vida	Dividido en sprints de duración fija (generalmente 2-4 semanas).	Continuo, con lanzamientos frecuentes.	Continuo, con énfasis en la automatización del despliegue y las operaciones.
Cultural	Colaboración y autoorganización son claves; el Scrum Master ayuda a remover obstáculos.	Comunicación abierta, retroalimentación continua, y bienestar del equipo.	Cultura de colaboración entre desarrolladores y operadores, fomentando una mentalidad de "fallar rápido" y aprendizaje continuo.
Beneficios	Mejora la visibilidad del progreso, adaptabilidad a cambios, y gestión del tiempo y recursos.	Mejora la calidad del software y la satisfacción del cliente mediante la adaptación rápida.	Mejora la eficiencia operativa, reduce el tiempo de comercialización, y aumenta la estabilidad de los sistemas en producción.
Desafíos	Requiere compromiso con los eventos de Scrum y adaptación a los ciclos cortos de feedback.	Requiere alta disciplina en las prácticas de desarrollo y puede ser intensivo en recursos.	Requiere un cambio cultural significativo y la integración de herramientas y procesos a lo largo de toda la organización.

Referencias:

¿Qué es la metodología ágil? (s.f.). https://www.redhat.com/es/topics/devops/what-is-agile-methodology

Pressman, R. S. (2010). *INGENIERIA DE SOFTWARE Un enfoque practico*.