Primer Parcial – Metodologías Ágiles ERNESTO MORETTI

1. **Identifique tres nuevas problemáticas de las enunciadas anteriormente**

   Tres nuevas problemáticas de las enunciadas anteriormente, los cuales demuestran porqué la modificabilidad y robustez son fundamentales, serán la siguientes:

   - El software puede llegar a ser muy complejo y amplio debido a los diversos sistemas y soluciones consolidadas que maneja la misma y utilizan la cartera de 30 clientes, dificultando como entender al sistema y mantenerlo durante su ciclo de vida. Si no se cuenta con un marco de trabajo óptimo y estructurado, las consecuencias negativas tendrán un mayor impacto.

   - Pueden aparecer nuevas regulaciones, normativa y/o legislaciones que cambien la forma en la cual se maneja el negocio. Ante esto, la empresa de software deberá enfrentar estos "desafíos" al tener que adaptar sus sistemas para así cumplir las nuevas reglas establecidas. Pero esto solo será posible si se implementa una metodología de trabajo, ya que de lo contrario la empresa puede verse metida en varios problemas de todo tipo (por ejemplo, legales). Una situación puede ser la pandemia del COVID-19, la cual obligó a modificar las formas de interactuar del negocio con los clientes.

   - La comprensión del sistema también puede ser afectado negativamente por el hecho de que el software es inherentemente invisible y difícil de visualizar. Si no se cuenta con las metodologías adecuadas, entonces representar y comunicar la estructura del software será difícil, causando que los equipos de desarrollo dentro de la empresa no puedan razonar sobre su diseño y tampoco puedan trabajar en conjunto.

2. **Para cada problemática debe indicar con cuales principios y prácticas de las metodologías se podría abordar. Se sugiere hacer una tabla con las siguientes columnas:**
   a. Problema b. Principios
   c. Prácticas d. Metodología

Problema	Principios	Prácticas	Metodología
Falta de planificación y organización	Mejora continua; Diversidad; Economía.	40 hour week; Planning game; On-site customer.	Extreme Programming (XP)
Dificultad para adaptarse a los cambios	Redundancia; Auto-similitud; Pequeños pasos.	Metáforas; Simple design; Entregas pequeñas.	Extreme Programming (XP), Proceso Unficado de Rational (RUP)
Baja calidad del software	Calidad; Pequeños pasos; Diversidad.	Programación en pares; Testing; Refactoring	Extreme Programming (XP), Test Driven Development (TDD)
Dificultad para la colaboración y la comunicación	Colaboración, Comunicación abierta	Programación en pares; Collaborative Workspaces	XP
Baja productividad	Eliminación de desperdicios; Eficiencia; Optimizar el conjunto.	Kanban	Lean
Falta de conocimientos en herramientas, técnicas de desarrollo e inteligencia artificial	Humanidad; Fallo (Aprendizaje); Mejora continua; Beneficio mutuo.	Integración contínua, Planning game.	XP
Complejidad inherente del software	Reflexión; Oportunidad; Redundancia.	Programación en pares; Refactorización; Iteraciones (pequeñas entregas)	XP, TDD
Conformidad con normativas y regulaciones	Desarrollo iterativo; Gestión de riesgos; Uso de casos de uso.	Modelado de casos de uso; Sistema de control de versiones; QA (Quality Assurance).	RUP
Invisibilidad y dificultad para visualizar el software	Enfoque minimalista; Trazabilidad alta; Iterativo e incremental.	Modelado de casos de uso; Modelo de dominio; Diagrama de clases; Diagrama de secuencia; Análisis de robustez.	ICONIX

3. **Diseñe un tablero Kanban para que la empresa lo pueda comenzar a usar. Por cada sección del tablero (o grupos de secciones) del tablero describa qué aspecto y/o problemáticas está abordando.**

Cuando implementamos Kanban, nosotros veremos las siguientes secciones:

- **Backlog:** se recopilan todas las tareas y funcionalidades que se necesitan realizar. Se muestra todo lo que está pendiente. Los problemas que se abordan aquí son:
    - Falta de planificación y organización.
    - Falta de conocimientos en herramientas y técnicas.

- **To Do:** se decide en que tareas del Backlog trabajar, las cuales se realizarán en el próximo sprint (períodos cortos de trabajo definidos). Similar a la lista del Backlog, pero más enfocada en una implementación a corto plazo. Los problemas que se abordan aquí son:
    - Baja productividad.
    - Dificultas para adaptarse a los cambios.

- **In Progress:** se comienza a trabajar en las tareas seleccionadas anteriormente. De desarrollarán y se codificarán a un lenguaje determinado, tomando las ideas y los requisitos en un código funcional. Los problemas que se abordan aquí son:
    - Baja calidad del software.
    - Falta de conocimiento en herramientas y ténicas.

- **Code Review**: cuando la tarea se ha desarrollado completamente, lo que se procede a hacer ahora es revisar el código para asegurar que cumple con los estándares de calidad y los requisitos especificados. Esto es obligatorio no solo para garantizar la calidad, sino también para identificar y corregir errores que puedan haber ocurrido durante el desarrollo. Los problemas que se abordan aquí son:
    - Baja calidad del software.
    - Dificultad para la colaboración y la comunicación.

- **Testing:** las tareas son puestas a prueba para verificar su correcto funcionamiento y asegurar que cumplen con los requisitos establecidos. Este paso es fundamental poder obtener un producto final robusto y libre de errores. Los problemas que se abordan aquí son:
    - Baja calidad del software.
    - Dificultad para la colaboración y la comunicación.

- **Done:** aquí encontraremos todas aquellas tareas completas que pasaron exitosamente las fases de desarrollo, revisión y pruebas. Se pueden hacer retrospectivas sobre estas para así aprender de esta “experiencia” y tomar la misma en consideración para el siguiente sprint. Los problemas que se abordan aquí son:
    - Baja productividad.
    - Resistencia al cambio organizacional.