

PRINCIPIOS DE LEAN SOFTWARE DEVELOPMENT

Juan Diego Solís Martínez - 23720
Nils Muralles Morales - 23727
Víctor Manuel Pérez Chávez - 23731
Diego Oswaldo Flores Rivas - 23714
Isabella Recinos Rodríguez- 23003

TAREA INVESTIGATIVA 3

CONCEPTOS PRINCIPALES





¿QUÉ ES LEAN SOFTWARE DEVELOPMENT (LSD)?

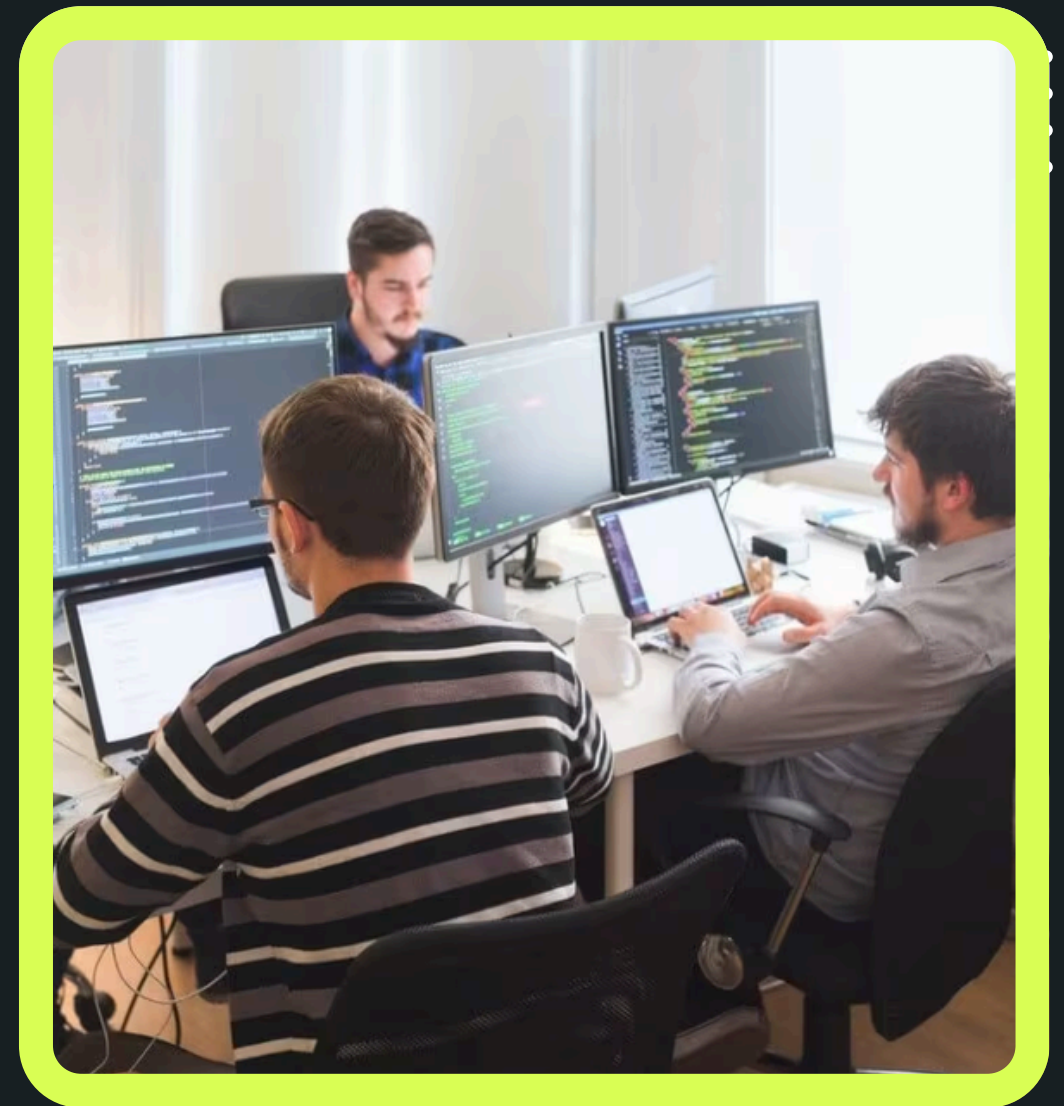


- Metodología ágil basada en Lean Manufacturing (Toyota).
- Objetivo: optimizar el flujo de trabajo, minimizar desperdicios y entregar software de manera continua y eficiente.




PRINCIPIOS FUNDAMENTALES

- Se basa en siete principios clave para mejorar el desarrollo de software.
- Promueve entregas frecuentes de software funcional.
- Permite a los clientes evaluar el producto y dar retroalimentación temprana.




PRÁCTICAS PARA IMPLEMENTAR LSD



Kanban:
Visualizar el
flujo de trabajo
y reducir
desperdicios.

**Entrega
continua:**
Obtener
retroalimentación
rápida.

**Pruebas
automatizadas:**
Garantizar
calidad desde el
principio.



**Mejora
continua:**
Revisiones
frecuentes del
proceso.

**Involucrar al
cliente:**
Asegurar que
el software
entregue valor
real.

¿EN QUÉ TIPOS DE PROYECTOS ES ÚTIL?

Proyectos con
requerimientos
cambiantes

Equipos pequeños o
medianos que necesitan
eficiencia y agilidad.

Empresas emergentes que
validan rápidamente ideas
con entregas incrementales.

Organizaciones que buscan
reducir costos y mejorar el
flujo de desarrollo.



PRINCIPIOS



1 ELIMINAR DESPERDICIOS

Es identificar y eliminar todo aquello que no le aporta valor al cliente.

2 AMPLIFICAR EL APRENDIZAJE

Es fomentar el aprendizaje continuo entre los miembros del equipo de desarrollo.

3 TOMAR DECISIONES TARDÍAS

Es retrasar las decisiones hasta que se tenga la mayor cantidad de información posible para adaptarse a cambios en los requisitos o necesidades del cliente.



4 ENTREGAR LO ANTES POSIBLE

Es realizar entregas frecuentes de software que incluyan funcionalidades alineadas con las necesidades más importantes de los usuarios.

5 POTENCIAR EL EQUIPO

Es involucrar a los desarrolladores en la toma de decisiones, tales como la estimación del tiempo, priorización de las tareas y otros puntos clave.



6 CREAR LA INTEGRIDAD

Es garantizar que el software sea fácil de mantener, mejorar y reutilizar a través de prácticas de integración continuas y pruebas automatizadas.

7 VISUALIZAR TODO EL CONJUNTO

Es adoptar una visión general del proyecto para entender cómo es que cada componente se integra en el conjunto y contribuye al valor final para el cliente.



HERRAMIENTAS DE CADA PRINCIPIO



1 ELIMINAR DESPERDICIOS

1. Value Stream Mapping.

- a. Generar un mapa del proceso actual
- b. Buscar y eliminar los desperdicios
- c. Generar un mapa del proceso mejorado
- d. Implementar el proceso que se utilizará en el futuro

2. Diagramas de Ishikawa.

2 AMPLIFICAR EL APRENDIZAJE

1. Desarrollo basado en pruebas.

2. Integración continua.



3 TOMAR DECISIONES TARDÍAS

1. Desarrollo basado en conjuntos.

4 ENTREGAR LO ANTES POSIBLE

1. Scrum: conjunto de buenas prácticas que se hacen con el fin de mejorar el trabajo para obtener el mejor resultado posible.

5 POTENCIAR EL EQUIPO

1. Kanban: Todos los miembros se ven involucrados en el flujo de trabajo, siendo capaces de estar al tanto de cuáles son sus responsabilidades y que deben hacer.



6 CREAR LA INTEGRIDAD

1. Integración continua (CI)
2. Desarrollo basado en pruebas (TDD) continua.

7 VISUALIZAR TODO EL CONJUNTO

1. Mapas de procesos (Value Stream Mapping - VSM).
2. Gestión visual con Kanban.
3. Arquitecturas de software modulares.
4. Diagramas UML.



GRACIAS

