GUÍA Nº 2 – Metodologías de Desarrollo de Software

FACULTAD	CURSO	AMBIENTE
INGENIERÍA	DISEÑO Y ARQUITECTURA DE SOFTWARE	LABORATORIO DE DISEÑO Y ARQUITECTURA DE SOFTWARE 77C0206

ELABORADO POR	GONZALO GARCIA MARTINEZ	**APROBADO POR**	
VERSIÓN	001	**FECHA DE APROBACIÓN**	

1. LOGRO GENERAL DE UNIDAD DE APRENDIZAJE

- Comprender los principios y enfoques clave de diferentes metodologías de desarrollo de software.
- Identificar las ventajas y desventajas de cada metodología en diferentes contextos de desarrollo de proyectos.
- Demostrar habilidades para evaluar y seleccionar la metodología adecuada según los requisitos del proyecto y las necesidades del equipo.
- Aplicar conceptos y prácticas de metodologías de desarrollo de software en proyectos reales para mejorar la calidad y la eficiencia del desarrollo.

2. OBJETIVOS ESPECÍFICOS DE LA PRÁCTICA

- Metodologías en Diseño y Arquitectura de Software.

3. MATERIALES Y EQUIPOS

- Computadoras Personales.
- Sistema Operativo Windows.
- Pizarra
- Plumón
- Mota

4. PAUTAS DE SEGURIDAD

Las computadoras y laptops deben de estar prendidas mientras se usan. Pero al terminar el laboratorio estas deben dejarse apagadas.

- En el laboratorio debe estar prendido el aire acondicionado para evitar sobrecalentamientos y

averías, especialmente en épocas de altas temperaturas.
- Los estudiantes no pueden llevar alimentos que puedan derramar sobre los computadores.
- Computadoras, router, switch, puntos de acceso (caídas).
- Eléctricos, por contacto directo o indirecto, electricidad estática y por fenómeno térmico. Puede producir: electrocuciones y quemaduras.
- Procedimiento ante Corte de Energía Eléctrica
- No tocar el equipo eléctrico en el que se encuentra trabajando, puede que retorne la energía.
- Comunicarse con el Asistente de Operaciones de turno quien se comunicará con el Técnico.

5. FUNDAMENTO

La asignatura de Diseño y Arquitectura de Software es de carácter teórico-práctico y tiene el propósito de potenciar en el estudiante sus habilidades para analizar y diseñar una arquitectura de software. Se desarrolla los siguientes contenidos: Introducción a la arquitectura de software, vistas y estilos de la arquitectura, requisitos de calidad de un software, diagramación UML orientada al diseño arquitectónico de software, patrones de arquitectura, arquitectura orientada a servicios (SOA), Arquitecturas en Cloud Computing, Arquitecturas para software en dispositivos móviles y documentación de una arquitectura de software.

6. INTRODUCCIÓN (MARCO TEÓRICO)

Bienvenidos a la sección dedicada a las metodologías en Diseño y Arquitectura de Software. En este espacio, exploraremos una variedad de enfoques y metodologías utilizadas en la industria del desarrollo de software para gestionar proyectos, organizar equipos y optimizar el proceso de desarrollo.

Desde los métodos tradicionales hasta las prácticas ágiles más innovadoras, abordaremos cómo cada metodología influye en el diseño y la arquitectura de software, y cómo pueden adaptarse a diferentes contextos y necesidades de proyectos.

Nos sumergiremos en las siguientes metodologías ampliamente utilizadas:

Modelo en Cascada: Este enfoque secuencial sigue una estructura lineal, donde cada etapa del proceso de desarrollo (requisitos, diseño, implementación, pruebas y mantenimiento) se completa antes de pasar a la siguiente. Exploraremos cómo este modelo influye en el diseño y la arquitectura de software, así como sus ventajas y limitaciones.

Scrum: Una de las metodologías ágiles más populares, Scrum se centra en equipos autoorganizados que trabajan en iteraciones cortas y regulares llamadas sprints. Analizaremos cómo Scrum aborda los desafíos de diseño y arquitectura de software, fomentando la flexibilidad y la colaboración continua.

Extreme Programming (XP): XP es una metodología ágil que se centra en la mejora continua y la entrega de software de alta calidad. Exploraremos cómo las prácticas de XP, como la programación en parejas y las pruebas unitarias, influyen en el diseño y la arquitectura de software, promoviendo la simplicidad y la adaptabilidad.

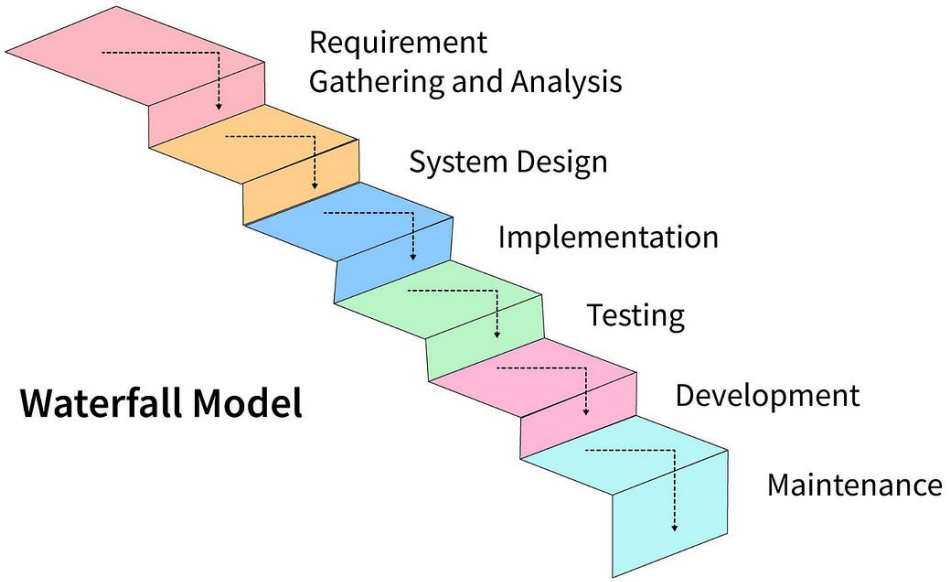
Kanban: Originario del sistema de producción de Toyota, Kanban se centra en la visualización del flujo de trabajo y la limitación del trabajo en curso. Analizaremos cómo Kanban puede aplicarse al diseño y la arquitectura de software, permitiendo una gestión eficiente de tareas y una respuesta ágil a los cambios.

Lean: Inspirado en los principios de Lean Manufacturing, Lean se centra en la eliminación de desperdicios y la maximización del valor para el cliente. Exploraremos cómo los principios lean pueden aplicarse al diseño y la arquitectura de software, optimizando el proceso de desarrollo y reduciendo el tiempo de entrega.

A lo largo de esta sección, examinaremos los principios, prácticas y herramientas asociados con cada metodología, proporcionando una comprensión integral de cómo influyen en el diseño y la arquitectura de software y cómo pueden utilizarse para mejorar la eficiencia y la calidad en proyectos de desarrollo.

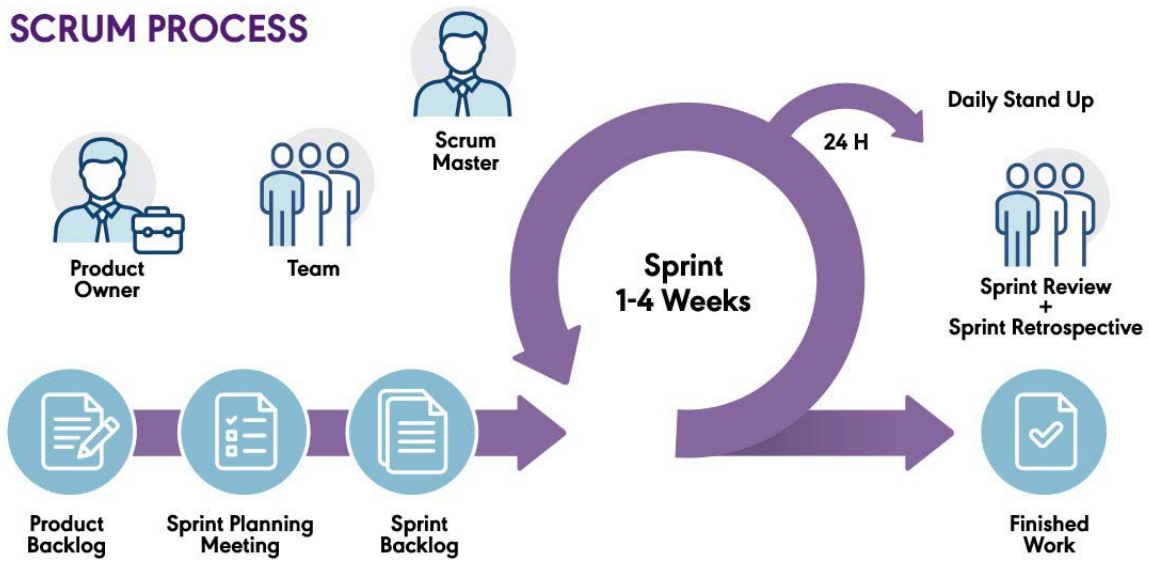
7. PROCEDIMIENTO (DESARROLLO DE LA PRÁCTICA)

7.1. Metodologías de Desarrollo de Software



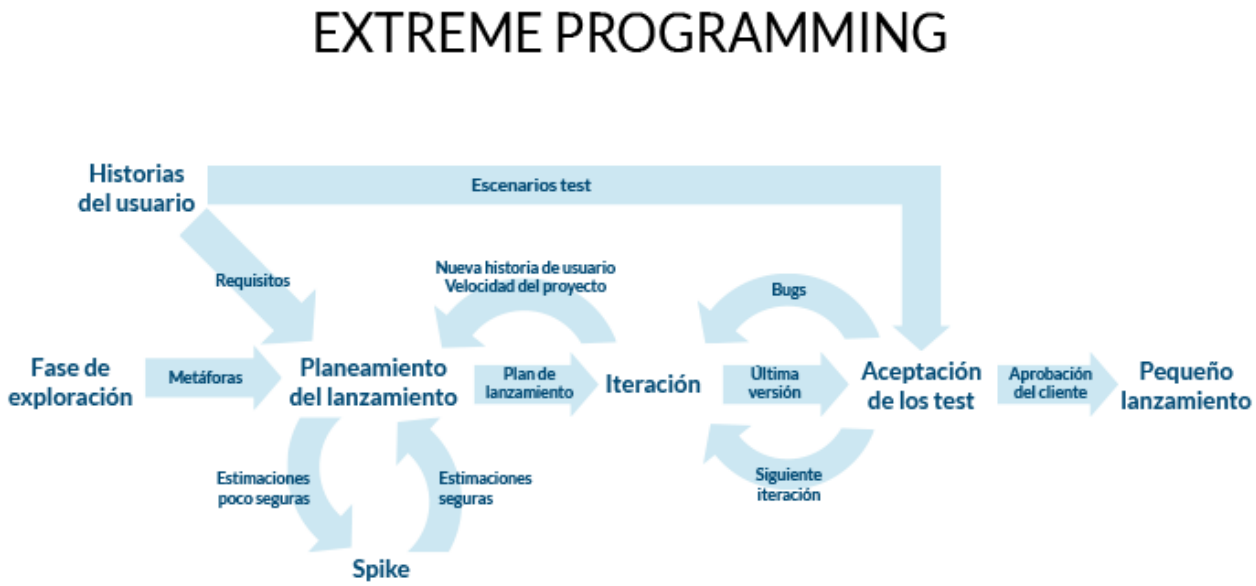
Cascada:

- ¿Cuáles son las principales fases del modelo en cascada y en qué orden se llevan a cabo?
- ¿Cuáles son las ventajas y desventajas del modelo en cascada en comparación con las metodologías ágiles?
- ¿Cómo influye el modelo en cascada en la planificación y el diseño de la arquitectura de software?



Scrum:

- ¿Qué roles clave existen en Scrum y cuáles son sus responsabilidades?
- ¿Cómo se estructura un sprint en Scrum y cuáles son sus principales actividades?
- ¿Cómo fomenta Scrum la colaboración entre equipos y la adaptación a los cambios en los requisitos del proyecto?

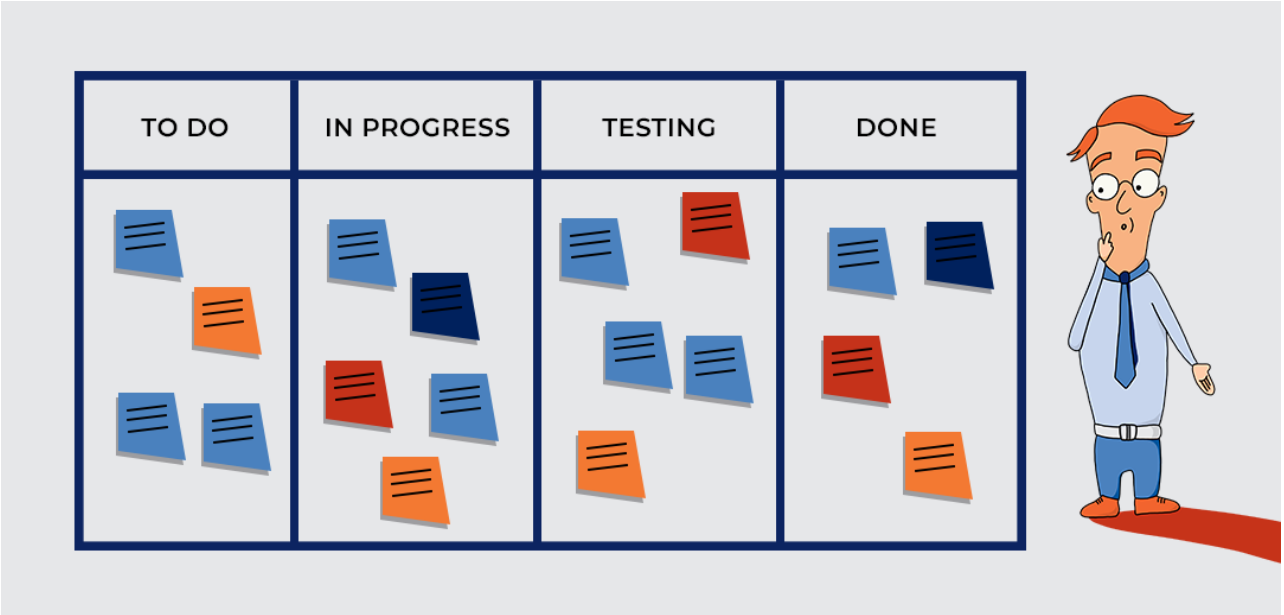


Extreme Programming (XP):

- ¿Cuáles son las prácticas principales de XP y cómo influyen en el diseño y la arquitectura

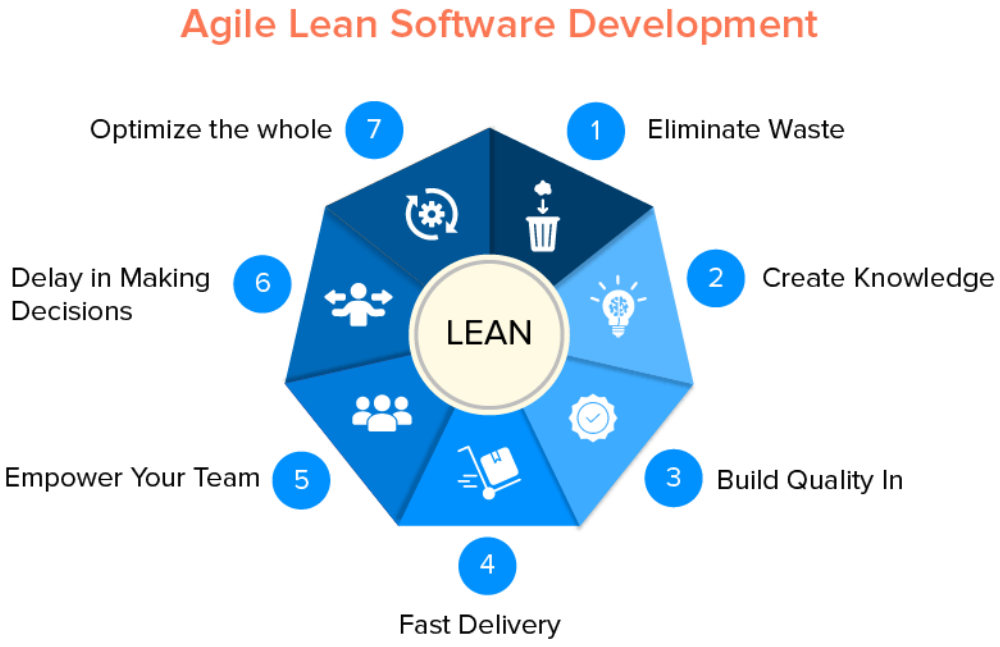
de software?

- ¿Qué beneficios se derivan de la programación en parejas y las pruebas unitarias en XP?
- ¿Cómo se asegura XP de que el software entregado cumpla con los estándares de calidad requeridos?



Kanban:

- ¿Cuál es el objetivo principal de Kanban y cómo se implementa en el desarrollo de software?
- ¿Cuál es la diferencia entre el tablero Kanban y otros métodos de seguimiento de tareas?
- ¿Cómo puede Kanban ayudar a identificar cuellos de botella y mejorar la eficiencia en el proceso de desarrollo?



Lean:

- ¿Cuáles son los principios fundamentales del Lean y cómo se aplican al desarrollo de software?
- ¿Cómo se pueden identificar y eliminar desperdicios en el proceso de desarrollo utilizando principios lean?
- ¿Qué impacto tiene Lean en la arquitectura de software y la entrega de valor al cliente de manera más eficiente?

Ejercicio: Evaluación y Aplicación de Metodologías en Diseño y Arquitectura de Software

Descripción:

Los estudiantes trabajarán individualmente para seleccionar una metodología de desarrollo de software (por ejemplo, Cascada, Scrum, XP, Kanban o Lean) y realizarán un estudio detallado sobre sus principios, prácticas y aplicaciones en el diseño y la arquitectura de software. Luego, aplicarán la metodología seleccionada a un proyecto de desarrollo de software ficticio y elaborarán un informe detallado que describa cómo se implementaría esta metodología en diferentes etapas del proyecto.

Pasos del Ejercicio:

Selección de Metodología: Cada estudiante elegirá una metodología de desarrollo de software de la lista proporcionada (Cascada, Scrum, XP, Kanban o Lean) para estudiar en profundidad.

Estudio Detallado: Los estudiantes investigarán la metodología seleccionada, analizando sus principios, prácticas, roles involucrados y casos de uso típicos en el desarrollo de software. Deberán comprender cómo la metodología influye en el diseño y la arquitectura de software, así como en la gestión de proyectos.

Aplicación a un Proyecto Ficticio: Una vez que los estudiantes hayan comprendido la metodología seleccionada, deberán aplicarla a un proyecto de desarrollo de software ficticio. Deberán describir cómo se utilizaría la metodología en diferentes etapas del proyecto, desde la planificación hasta la implementación y las pruebas.

Elaboración del Informe: Los estudiantes elaborarán un informe detallado que incluya lo siguiente:

- Introducción a la metodología seleccionada y su relevancia en el desarrollo de software.
- Descripción de los principios y prácticas clave de la metodología.
- Aplicación de la metodología al proyecto ficticio, detallando su implementación en cada etapa del ciclo de vida del desarrollo de software.
- Análisis de los beneficios y desafíos de utilizar la metodología en el proyecto propuesto.
- Conclusiones y reflexiones sobre la idoneidad de la metodología para diferentes tipos de proyectos y contextos.

8. ENTREGABLES

- Cuestionario y ejercicio sobre metodologías en desarrollo de software.

9. FUENTES DE INFORMACIÓN COMPLEMENTARIA

- Cohn, Mike. (2005). "Agile Estimating and Planning". Prentice Hall.
- Pressman, Roger S. (2014). "Software Engineering: A Practitioner's Approach". McGraw-Hill Education.
- Sutherland, Jeff. (2014). "Scrum: The Art of Doing Twice the Work in Half the Time". Crown Business.
- Schwaber, Ken, & Beedle, Mike. (2001). "Agile Software Development with Scrum". Prentice Hall.
- Poppendieck, Mary, & Poppendieck, Tom. (2003). "Lean Software Development: An Agile Toolkit". Addison-Wesley Professional.
- Beck, Kent. (2004). "Extreme Programming Explained: Embrace Change". Addison-Wesley Professional.