Etapas y Herramientas de SDLC

A - Conceptos: Responde de forma conceptual y sintética el siguiente cuestionario

1. ¿Qué es el Ciclo de Vida del Software y por qué es importante conocerlo?

El Ciclo de Vida del Software es el proceso que guía el desarrollo de un sistema desde la idea inicial hasta su retiro, y conocerlo es fundamental porque permite organizar el trabajo, mejorar la calidad del producto, facilitar la colaboración entre equipos y prevenir errores durante el proyecto.

2. ¿Cuál es la diferencia fundamental entre la fase de Análisis y la fase de Diseño en el SDLC.

La diferencia principal entre la fase de análisis y la de diseño es que en el análisis se busca entender bien el problema y lo que necesita el cliente, mientras que en el diseño se piensa cómo se va a construir esa solución. Primero se escucha y se define qué se quiere lograr, y después se planifica cómo hacerlo, mostrando ideas, bocetos y ejemplos para que el cliente vea cómo va a funcionar el sistema.

3. Según el material, ¿dónde radica el principal problema de las aplicaciones que fallan o son ineficientes

Las aplicaciones fallan o son ineficientes principalmente por una mala planificación: no entender bien qué se necesita, diseñar mal la solución o no considerar la seguridad desde el inicio. El problema no suele ser la tecnología, sino cómo se usa.

4. Menciona y describe brevemente dos tipos de pruebas realizadas en la fase de Pruebas del SDLC.

Pruebas Unitarias: Se encargan de verificar que cada parte pequeña del código (como una función) funcione correctamente por sí sola.

Pruebas de Usabilidad: Se enfocan en que el usuario pruebe el sistema. Sirven para ajustar el diseño y la experiencia, asegurando que el software sea fácil de usar y cumpla con los requisitos del usuario.

5. ¿Qué implica la fase de Despliegue y por qué se considera un proceso complejo?

La fase de despliegue es cuando el software ya está listo y se pone en marcha para que los usuarios lo usen. Es un proceso complicado porque hay que instalarlo, configurarlo bien y asegurarse de que funcione en el lugar donde se va a alojar.

6. ¿Cuál es el propósito principal de la fase de Mantenimiento?

El propósito principal de la fase de mantenimiento es detectar y corregir errores que no fueron identificados durante las pruebas, o que aparecen una vez que el sistema está en uso real. Además, esta fase permite realizar mejoras o ajustes solicitados por el cliente, asegurando que el software siga funcionando correctamente y se adapte a nuevas necesidades o cambios en el entorno.

7. Compara brevemente las "historias de usuario" y los "casos de uso" como herramientas de análisis.

Los casos de uso se emplean más en metodologías clásicas y son diagramas formales en UML que muestran a los usuarios y las acciones que realizan en el sistema. Y las historias de usuario son más narrativas y simples, permite al usuario expresar de forma clara y práctica sus requerimientos sin entrar en tanta formalidad técnica.

8.¿Qué son los mockups y cuál es su utilidad en la fase de Diseño?

Los mockups son prototipos visuales que muestran cómo se verá la interfaz del sistema. En la fase de Diseño se usan para que el cliente y el equipo comprendan la apariencia y funcionamiento básico de la aplicación antes de desarrollarla.

9.Menciona al menos tres herramientas o conceptos clave en la fase de Desarrollo del software.

En la fase de Desarrollo del software se utilizan herramientas como:

- Lenguajes de programación (Java,Javascript, C#, etc.) para construir el sistema.
- Entornos de desarrollo (IDE) como Eclipse, Visual Studio Code o IntelliJ, se utilizan para escribir código.
- Sistemas de control de versiones (Git), que permiten organizar y coordinar el trabajo en equipo.

10. Explica por qué la seguridad es un tema importante a considerar en el Ciclo de Vida del Software y en qué etapas puede abordarse.

La seguridad es importante en el Ciclo de Vida del Software porque protege los datos y la información del sistema, evitando riesgos como accesos no autorizados, fraudes o pérdidas críticas.

Se puede abordar en varias etapas: en el análisis, al definir requisitos de seguridad; en el diseño, al planificar arquitecturas seguras; en el desarrollo y pruebas, al aplicar controles y realizar pruebas de seguridad; y en el mantenimiento, corrigiendo fallas nuevas y reforzando la protección.

B- Análisis y reflexión: desarrollar los siguientes temas propuestos basados en el texto e investigación sobre las etapas y herramientas del SDLC.

1. Analice críticamente la afirmación sobre que la tecnología no es la culpable de los problemas en el desarrollo de software, sino el diseño. ¿Está de acuerdo? Justifique su respuesta con ejemplos y argumentos basados en el texto.

La afirmación de que la tecnología no es la culpable de los problemas en el desarrollo de software, sino el diseño, es correcta. El SDLC muestra que los errores más frecuentes surgen en las fases iniciales, como el análisis y diseño, cuando los requerimientos no se comprenden bien o se plantean inadecuadamente. En estos casos, aunque se utilice la mejor tecnología (nube, microservicios, lenguajes modernos), el resultado será no satisfactorio.

Por ejemplo:

Un sistema bancario mal diseñado en sus reglas de negocio seguirá fallando aunque

se programe en Java o se despliegue en AWS. El texto lo confirma al señalar que “los problemas rara vez son solo tecnológicos; suelen tener raíces en una deficiente planificación o diseño”.

La tecnología es una herramienta, el verdadero éxito depende de un diseño sólido que traduzca correctamente las necesidades del cliente en soluciones viables.

2. Compare y contraste las herramientas de análisis para metodologías clásicas (casos de uso, UML) y ágiles (historias de usuario, scrum), destacando sus enfoques y beneficios para el levantamiento de requerimientos.

Las metodologías clásicas se basan en Casos de Uso. Utilizan UML (Lenguaje de Modelado Unificado) para diagramar las interacciones de los actores con el sistema. Estos diagramas identifican las acciones que el usuario realizará dentro del sistema.

Por otro lado, las metodologías ágiles usan Historias de Usuario. Son similares a los Casos de Uso, pero tienen un enfoque más narrativo. Permiten que el usuario o actor exprese sus requerimientos de manera concreta y eficiente. Se utilizan en modelos como los de Scrum.

3. Explique la importancia de las pruebas de usabilidad, seguridad y calidad dentro del Ciclo de Vida del Software, y cómo una omisión o deficiencia en cualquiera de estas puede afectar el éxito de un proyecto.

Las pruebas de usabilidad, seguridad y calidad son totalmente fundamentales en el Ciclo de Vida del Software.

- La usabilidad garantiza que el sistema sea intuitivo; si falla, los usuarios lo rechazan.
- La seguridad protege los datos; una deficiencia expone vulnerabilidades y daña la confianza.
- La calidad asegura que el software cumpla los requerimientos; si se omite, se entregan productos defectuosos.

En conjunto, una falla en cualquiera de estas pruebas puede comprometer el éxito del proyecto y su aceptación final.

4. Describa los principales desafíos y consideraciones al seleccionar herramientas y plataformas para la fase de despliegue de software, haciendo énfasis en la evolución de los servidores locales a la infraestructura en la nube y los servicios de despliegue móvil.

Los principales desafíos al seleccionar herramientas y plataformas para la fase de despliegue incluyen la amplia cantidad de tecnologías disponibles y la dificultad de estimar recursos como servidores, memoria RAM y discos duros. El sector ha pasado de los servidores locales, ahora en desuso, a la infraestructura en la nube, que ha ganado popularidad debido a su conveniencia. La nube ofrece opciones como alojamiento web, para sitios de alcance medio, y servidores virtuales (VPS), que permiten entornos

personalizados. La elección de la plataforma depende del conocimiento, la extensión y la complejidad del sistema. En el caso del despliegue de aplicaciones móviles, se usan tiendas como Apple App Store y Google Play Store, cada una con sus propias políticas y requisitos. En esta etapa, el rol de Arquitecto de Despliegue es muy importante, debido a que son profesionales expertos en la puesta en marcha de aplicaciones.

5. Reflexione sobre la relevancia de la capacitación y especialización profesional en las diferentes etapas del Ciclo de Vida del Software, considerando las oportunidades tanto para trabajar en una organización como de manera independiente.

La capacitación y especialización profesional en las distintas etapas del Ciclo de Vida del Software es totalmente necesaria porque cada fase, análisis, diseño, pruebas, despliegue y mantenimiento requiere habilidades y herramientas específicas.

Un profesional que domina estas etapas puede integrarse en equipos de organizaciones aportando valor en un rol muy concreto, o también trabajar de manera independiente gestionando proyectos completos por su cuenta.

Aunque uno se especialice en una fase, comprender el ciclo en su totalidad otorga una ventaja competitiva, ya que permite abordar los proyectos de forma integral y responder mejor a las necesidades del cliente. En conclusión, la formación continua asegura no solo calidad técnica, sino también mayores oportunidades laborales y de emprendimientos.