Actividad 1

Entornos de Desarrollo
1º DAW Semipresencial

Alberto Requena Sáez
14 de Noviembre de 2023

1.

Define "Ciclo de vida del software"

El "Proceso para el Desarrollo de Software" o "Ciclo de Vida Software" es una metodología que se aplica para la creación de productos de software. Éste abarca desde que se consigue la información para determinar los requisitos del proyecto, hasta el despliegue de la solución final con su mantenimiento.

No importa si estamos trabajando con un Modelo Clásico, de Construcción de Prototipos o Evolutivo, siempre tendremos que utilizar las fases que define el "Ciclo de Vida del Software".

Finalmente, un plan o metodología para la creación de software resulta imprescindible para garantizar que se cumplen los requisitos de creación del producto. Sin él, los proyectos fácilmente fallan en cuanto a funcionalidades implementadas, tiempos de entrega y/o presupuesto.

2.

Nombra las fases principales del desarrollo de software y explica brevemente que se hace en cada una de ellas.

Si bien pueden diferenciarse diferentes fases en el "Ciclo de Vida del Software", dependiendo del autor, y también del proyecto, se le dará prioridad a unas sobre otras, fusionándolas o manteniéndolas independientes. En este caso, las definiré todas para una mejor comprensión:

A. Planteamiento

En el "Planteamiento del Problema" o "Requerimientos", tendremos muy claro el problema y seremos muy rigurosos al definirlo. Además, lo haremos en un lenguaje que sea entendible por parte del cliente puesto que en esta fase el participará de manera fundamental y necesitaremos validar aquello que nos transmita.

B. Planificación

En la planificación se evaluará el contexto del proyecto y la vialidad del mismo. Así como los requisitos de aseguramiento de calidad de las distintas fases.

C. Análisis

En el Análisis, estableceremos qué funcionalidades deberá tener el software y cuáles serán las características específicas de las mismas. Aquí empezaremos a tener

información de primera real sobre qué coste puede alcanzar el proyecto y sus diferentes partes.

D. Diseño

En esta fase se decide la estructura general del proyecto. Normalmente se realizarán sucesivas modificaciones del diseño para ir puliéndolo, hasta encontrar la solución más adecuada. Se realizarán:

- La estructura de la base de datos.
- Diagramas de flujo. Para ir teniendo claro cuál va a ser la lógica del programa o programas, y el flujo de datos.
- Interfaz de usuario. Que es el elemento final con el que va a interactuar el usuario final del producto. El tenerlo creado, aunque sea sin código, ayuda mucho a entender lo que se está creando y lo que se necesita para hacerlo realidad.

E. Codificación

Aquí, se deciden las herramientas a utilizar así como el lenguaje o lenguajes con los que se llevará a cabo el proyecto. Además, los programadores irán haciendo realidad el proyecto y se obtendrán los primeros prototipos, al mismo tiempo que se irá creando la documentación para su posterior uso.

F. Pruebas

El producto de software va a tener inevitablente errores o "bugs" que han de ser detectados antes de que el usuario final lo haga. El programa o programas será, por tanto, sometido a diferentes tests de estrés que ayudarán a encontrar los problemas ocultos en el código.

Una vez finalizada esta fase, el producto está prácticamente para entregar. En esta etapa también hay que concretar cómo se va a desplegar el producto para que esté en producción.

G. Mantenimiento

Una vez que el producto está oficialmente en funcionamiento, surgirán nuevas necesidades, que no han sido tenidas en cuenta en la fase de Planteamiento, y errores, que no han sido capturados en la fase de Pruebas. Un producto de software debe ser mantenido durante su ciclo de vida para no volverse, probablemente de manera rápida, obsoleto.

3.

Explica brevemente en qué consiste el modelo en cascada cuando hablamos de desarrollo de software.

En 1970 Winston W. Royce publicó la versión original del Modelo en Cascada, la cual sería revisada en los años 80.

En el Modelo en Cascada, las Fases del Ciclo de Vida del Software siguen un orden secuencial riguroso y definido, de tal forma que una etapa no comienza hasta que ha acabado la anterior.

Es un modelo antiguo y muchas veces criticado, sin embargo, está muy extendido.

4.

Ventajas e inconvenientes del modelo en cascada.

Si en la fase Planteamiento del "Ciclo de Vida del Software" se confirma o se descubre que los requisitos del producto son fijos y no van a sufrir modificaciones a lo largo de la creación del producto de software o en su posterior puesta en funcionamiento, entonces el Modelo en Cascada puede ser muy adecuado porque el proyecto se puede volver muy predecible a nivel de tiempos de entrega y costes.

Por otro lado, proyectos en los que es más difícil definir en detalle el problema o simplemente no se puede. El Modelo en Cascada puede convertirse en un lastre adicional en la creación del proyecto, puesto que dificulta enormemente los cambios rápidos.

Además, el Modelo en Cascada sigue normalmente tiempos de entrega altos en comparación con el resto de modelos de desarrollo, lo que reduce significativamente la comunicación con el cliente. Por lo que las posibles iteraciones de mejora del producto se alargan o retrasan mucho.

5.

¿Qué se entiende por verificación? ¿Y por validación?

La verificación y la validación referidas al software, son conceptos que se utilizan en los procesos de comprobación y análisis. Y, que aseguran que el producto creado se corresponde con las necesidades del cliente o de las personas interesadas.

Por un lado, la verificación hace referencia a las funcionalidades de la aplicación. En concreto, si éstas funcionan tal y como se espera.

Por otro, la validación tiene un componente de experiencia de usuario o experiencia del cliente. Es decir, ¿la aplicación creada cumple con mis expectativas y cubre las necesidades que esperaba?

6.

Explica cómo funciona el modelo de desarrollo mediante creación de prototipos.

El Modelo de Prototipos pertenece a la familia de modelos de Desarrollo Evolutivo. Con esta metodología se pretende crear muy rápidamente un producto para conseguir lo antes posible las impresiones del cliente, las cuales van a ser fundamentales para continuar con el trabajo hasta crear el producto final.

Este modelo se utiliza cuando se dispone de poca información previa o, por lo menos, no se dispone del detalle de la información y, sí principalmente generalidades.

Trabajando así, también permite ver rápidamente resultados al cliente. Sin embargo, éste puede quedar algo confuso durante el proceso. Pues, para la creación del prototipo se suele desatender la calidad para acortar plazos, y el cliente puede sentirse decepcionado con el resultado porque considera que el producto final va a ser similar al prototipo.

Las fases de este modelo de desarrollo son:

1. Requisitos
2. Modelaje y desarrollo del código
3. Evaluación
4. Modificaciones
5. Documentación
6. Pruebas
7. Entrega final

Los prototipos pueden ser de 2 tipos:

- Prototipos Rápidos. Donde el prototipo finalmente se deshecha.
- Prototipos Evolutivos. Donde el prototipo creado sirve finalmente de base para la creación del producto final.

7.

Explica cómo funciona el modelo espiral cuando se aplica al desarrollo orientado a objetos.

El Modelo de Desarrollo en Espiral fue publicado por primera vez en 1986 por Barry Boehm.

En este modelo se van añadiendo funcionalidades al producto por iteraciones. Y se podría interpretar que cada iteración sigue el modelo de desarrollo en cascada, pues tan pronto acaba una fase comienza la siguiente. Por este motivo, al Modelo de Desarrollo en Espiral se le considera un híbrido de los modelos de iteraciones y del modelo en cascada.

Cada iteración se divide principalmente en 4 etapas:

1. Determinar objetivos
2. Análisis de riesgos. En esta fase, también se evalún alternativas y se desarrollan prototipos.
3. Desarrollar y probar.
4. Planificación de las siguientes fases

Es un modelo de desarrollo recomendado para grandes sistemas de software en los que hay que tener un cierto control sobre los riesgos y los costes, al mismo tiempo que hay que tener cierta flexibilidad para poder adaptarse a los cambios.

8.

¿Qué cuatro principios rigen el desarrollo ágil expresados en el Manifiesto Ágil?

El Manifiesto Ágil fue publicado en el año 2001 en Estados Unidos como una herramienta que sería base de nuevos modelos de desarrollo que primarían la velocidad y la flexibilidad en la creación de nuevos productos, principalmente de software. Sus 4 principios son:

1. Valorar más a los individuos y sus interacciones que a los procesos y las herramientas. Este es el principio más importante de los 4, y enfatiza en la idea de que son los trabajadores los verdaderos motores de la creación de los productos. Y, por tanto, el resto de variables como la organización, y las herramientas deben girar entorno a las personas y no al revés.

2. Valorar más el software funcionando que la documentación exhaustiva. En el Manifiesto no se desprecia el valor de la documentación, pero se pone en valor al propio producto y a la comunicación directa entre las personas. La documentación no puede ser el principal medio de comunicación entre los diferentes actores que crean o hacen uso del producto de software, pues de ser así el proceso se relentiza considerablemente y se pierde mucha información relevante.

3. Valorar más la colaboración con el cliente que la negociación contractual. Los modelos de desarrollo que tienen como base el Manifiesto Ágil crean productos

que inicialmente son muy difíciles de definir. Este proceso de ir concretando los requerimientos iniciales, los cuales suelen ser muy generales, se consigue según se va creando el producto y se va consiguiendo retroalimentación por parte del cliente. Por tanto, el cliente no debe ser un agente externo al proyecto, debe involucrarse en el proceso y debe haber cierta flexibilidad por parte de todos los actores para llegar a un buen fin.

4. Valorar más la respuesta ante el cambio a seguir un plan. El cambio y la inestabilidad de las especificaciones son 2 de los aspectos más relevantes de los proyectos de software que son creados con los modelos de desarrollo que siguen el Manifiesto Ágil. Por tanto, se deben abrazar conceptos como la flexibilidad y la evolución rápida y continua, y dejar un poco de lado los planes preestablecidos.