Trabajo practico N°4- Proceso de desarrollo del software

1. Un proceso del software es un conjunto de actividades y resultados asociados que producen un produelo de software. Estas actividades son llevadas a cabo por los ingenieros de software. Existen cuatro actividades fundamentales de procesos que son comunes para todos los procesos del software. Estas actividades son:

* Especificación del software donde los clientes e ingenieros definen el software a producir y las restricciones sobre su operación.
* Desarrollo del software donde el software se diseña y programa.
* Validación del software donde el software se válida para asegurar que es lo que el cliente requiere.
* Evolución del software donde el software se modifica para adaptarlo a los cambios requeridos por el cliente y el mercado.

1. Un modelo de procesos del software es una descripción simplificada de un proceso del software que presenta una visión de ese proceso. Estos modelos pueden incluir actividades que son parte de los procesos y productos de software y el papel de las personas involucradas en la ingeniería del software. Algunos ejemplos de estos tipos de modelos que se pueden producir son:

* Un modelo de flujo de trabajo. Muestra la secuencia de actividades en el proceso junto con sus entradas, salidas y dependencias. Las actividades en este modelo representan acciones humanas.
* Un modelo de flujo de datos o de actividad. Representa el proceso como un conjunto de actividades, cada una de las cuales realiza alguna transformación en los datos. Muestra cómo la entrada en el proceso, tal como una especificación, se transforma en una salida, tal como un diseño. Pueden representar transformaciones llevadas a cabo por las personas o por las computadoras.
* Un modelo de rol/acción. Representa los roles de las personas involucrada en el proceso del software y las actividades de las que son responsables.

1. El modelo cascada, es uno de los modelos de desarrollo de software más antiguos y ampliamente utilizados. Se caracteriza por tener una secuencia lineal y estructurada de etapas, donde cada etapa se realiza antes de pasar a la siguiente. Estas etapas son:

* Requisitos: Se recopilan y documentan los requisitos del software, identificando las necesidades del cliente y los usuarios finales.
* Diseño: Se crea un diseño detallado del software, que incluye la arquitectura, los componentes y las interacciones entre ellos.
* Implementación: Se realiza la codificación del software utilizando el diseño creado anteriormente.
* Pruebas: Se llevan a cabo pruebas exhaustivas para verificar el correcto funcionamiento del software.
* Despliegue: Se instala y configura el software en el entorno de producción y se proporciona la documentación y capacitación necesarias.
* Mantenimiento: Se realizan actividades de corrección de errores, actualizaciones y mejoras a lo largo del tiempo.

1. Ventajas del modelo cascada:

* Estructura clara: El modelo cascada ofrece una estructura clara y secuencial que facilita la comprensión y seguimiento del progreso del proyecto.
* Planificación temprana: Las etapas del modelo cascada permiten una planificación temprana y una estimación más precisa de los recursos y el - tiempo necesarios.
* Enfoque en la documentación: La documentación completa en cada etapa facilita la comprensión del software y su mantenimiento a largo plazo.
* Adecuado para proyectos pequeños y bien definidos: El modelo cascada se adapta bien a proyectos pequeños con requisitos claros y estables desde el principio.

1. El modelo cascada es más adecuado en situaciones donde los requisitos del software son estables y bien definidos desde el principio, y no se esperan cambios significativos en el transcurso del proyecto. Es útil cuando se dispone de suficiente información y experiencia previa para realizar una planificación detallada y precisa. Además, el modelo cascada se puede utilizar en proyectos más pequeños y menos complejos, donde la flexibilidad y adaptabilidad pueden no ser tan críticas como la planificación y la documentación rigurosa. Sin embargo, en proyectos más grandes o en entornos cambiantes, pueden ser más apropiados otros modelos de desarrollo de software, como los enfoques ágiles o iterativos.