Actividad en clase número 1

Nelmer Daniel Roa Cárdenas 1232390107

Samuel Sebastián Jaimes Arenas 1091354646

1. ¿Es apropiado considerar que el software es más que programas que son ejecutados por los usuarios finales de un sistema?

R: Si es apropiado considerar al software como un producto mayor que solo los programas que son ejecutados por usuarios finales. El software se compone por un conjunto amplio de elementos para su desarrollo que también incluye componentes como la documentación, el software de sistema, las librerías, bases de datos y datos asociados al sistema; como muchos otros elementos, por lo que negar que el software es más que solo programas ejecutables es tener una visión limitada y errónea del ecosistema que rodea a la ejecución de este.

1. ¿Cuáles son las diferencias entre el desarrollo de un producto de software genérico y el desarrollo de un software personalizado?

R: La principal diferencia que determina un producto de software genérico y uno personalizado, es el mercado al que apela. Un software genérico se desarrolla para un mercado amplio, con funciones que satisfacen necesidades comunes de muchos usuarios, mientras que un software personalizado se desarrolla a medida para un cliente específico con necesidades particulares. Otras diferencias notables son el control del proceso, la flexibilidad, la personalización, el tiempo de desarrollo, la escalabilidad y el costo.

1. ¿Cuáles son los cuatro atributos importantes que todos los productos de software deben tener? Sugiera otros cuatro atributos que pueden ser significativos.

R: Gracias al aporte de autores como Sommerville, se conoce que los productos de software deben poseer cuatro atributos esenciales, los cuáles son:

* Mantenibilidad: se refiere a la facilidad con la que el software puede ser modificado para corregir errores, añadir nuevas funcionalidades o mejorar el rendimiento de los recursos que está utilizando. Un software mantenible puede adaptarse a cambios futuros sin grandes complicaciones.
* Confiabilidad: la capacidad del software para funcionar de manera correcta y consistente, sin errores ni fallos, durante un tiempo determinado. Esto implica que el sistema no debería presentar problemas y ninguna interrupción inesperada.
* Usabilidad: este atributo evalúa qué tan fácil es para el usuario interactuar con el software y lograr sus objetivos. Incluye la facilidad de aprendizaje, la eficiencia y la satisfacción general del usuario.
* Seguridad: es la capacidad del software para protegerse contra accesos no autorizados, uso indebido de la información, ataques, o interrupción de sus servicios. Un software seguro salvaguarda los datos del usuario y garantiza la integridad del sistema.

Añadiendo a los cuatros atributos recién enunciados, otros atributos que pueden tener relevancia para definir a los productos de software pueden ser:

* Eficiencia: todo software debe hacer un uso adecuado de los recursos del sistema (memoria, tiempo de CPU, red, etc.) evitando fallas en los procesos al momento de ejecutar cualquier tarea.
* Escalabilidad: es la capacidad del software de crecer a lo largo de su vida, aumentando funcionalidades, procesos y capacidad de usuarios sin perder el rendimiento.
* Portabilidad: esto hacer referencia a que el sistema pueda ejecutarse en distintos entornos (sistemas operativos, dispositivos, navegadores).
* Accesibilidad: este atributo indica que el sistema deba integrar elementos para que pueda ser utilizado por personas con diferentes capacidades físicas, sensoriales o cognitivas. Un ejemplo de esto sería incluir lectores de pantalla, subtítulos, navegación por teclado o contraste adecuado en la interfaz.

1. ¿Cuál es la diferencia entre un modelo del proceso del software y un proceso del software? Sugiera dos formas en las que un modelo del proceso del software ayuda en la identificación de posibles mejoras del proceso.

R: Cualquier proceso de software es una actividad y/o tarea que se realiza para crear un sistema de software. Se puede definir como el conjunto real de actividades, métodos, prácticas y transformaciones que se llevan a cabo para desarrollar y mantener software en una organización. Mientras que los modelos del proceso del software son las representación visual y abstracta de estos procesos, que proporcionan un marco o enfoque genérico para organizar esas actividades, en lugar de la ejecución concreta de las mismas.

Dos formas en las que el modelo del proceso puede ayudar para la identificación de mejoras en el proceso serían:

* Utilizando dos modelos, uno como referencia y el otro sería el de la empresa. Con esto, se compara cómo trabaja realmente la organización frente a distintos modelos, ayudando a detectar ineficiencias, pasos redundantes o carencias.
* Se pueden identificar fases críticas en los procesos de una empresa, representando visualmente el flujo de actividades de la misma, esto permite proponer mejoras en planificación, recursos o comunicación.

1. Explique por qué los costos de pruebas de software son particularmente altos para productos de software genéricos que se venden a un mercado amplio.

R: Existen varias causas de estos costos elevados, pero su razón principal es el hecho que deben cubrir un amplio abanico de escenarios y usuarios potenciales, aumentando la complejidad de las pruebas, el tiempo de ejecución y la cantidad de recursos necesarios. Esto se le suma una alta expectativa en la calidad del producto y que sea funcional durante un largo periodo de tiempo. Los atributos que requiere un software de tipo genérico (ser robusto, escalable, flexible y confiable), generan mayor cantidad de pruebas exhaustivas y rigurosas para asegurar la calidad y la confiabilidad en entornos diversos para millones de usuarios.

1. ¿Qué es una herramienta CASE?

R: Las herramientas CASE son un conjunto de aplicaciones informáticas, usadas para automatizar actividades del ciclo de vida de desarrollo de sistemas. Es un software que apoya y automatiza actividades del proceso de desarrollo de software, facilitando tareas de análisis, diseño, programación, pruebas y mantenimiento. Existen varios tipos de herramientas CASE, cada una encargada de una fase distinta del ciclo de vida de desarrollo de software.

* Herramientas de análisis: enfocadas en la fase de análisis de requisitos y diseño, como Rational Rose para modelado UML.
* Herramientas de diseño: se centran en el diseño de la arquitectura, bases de datos e interfaces.
* Herramientas de generación de código: permiten la generación automática de código fuente.
* Herramientas de pruebas y depuración: utilizadas para automatizar pruebas y detectar errores.
* Herramientas de gestión de la configuración: ayudan a gestionar versiones del código y el control de cambios.

También se puede clasificar en otros tres tipos:

* Upper CASE → Apoyan fases tempranas (análisis y diseño). Ejemplo: Rational Rose, Enterprise Architect.
* Lower CASE → Apoyan fases tardías (codificación, pruebas, mantenimiento). Ejemplo: JUnit, Selenium.
* Integrated CASE (I-CASE) → Cubren casi todo el ciclo de vida. Ejemplo: Visual Paradigm, IBM Rational Software Architect.

1. Comente si los ingenieros profesionales deben atestiguar de la misma forma que los doctores o los abogados.

R: Totalmente de acuerdo en que los ingenieros profesionales deban atestiguar. Profesionalmente, manejan una cantidad amplia de información, incluida información privada de usuarios que fue conferida por las empresas que solicitaron un sistema, por esto, en el ejercicio de sus funciones al llevarse a cabo un proyecto que incluya este tipo de información, los ingenieros deben atestiguar ante un juez con la misma seriedad y responsabilidad ética que médicos y abogados, porque sus decisiones también afectan vidas y sociedades.

1. Para contrarrestar al terrorismo, muchos países están desarrollando sistemas informáticos que siguen la pista de un gran número de sus ciudadanos y de sus acciones. Desde luego, esto tiene implicaciones sobre la privacidad. Comente la ética de desarrollar este tipo de sistema.

R: Existen diferentes perspectivas sobre este tema. Por un lado, estos sistemas pueden contribuir a la seguridad nacional, ayudando a prevenir ataques y protegiendo la vida de miles de personas. Desde una perspectiva utilitarista, se podría justificar su implementación si el beneficio colectivo supera el costo de la vigilancia, siempre que exista un marco legal claro y mecanismos de supervisión estrictos que limiten su uso únicamente con fines de seguridad.

Por otro lado, la vigilancia masiva conlleva serias implicaciones sobre la privacidad y la libertad individual. El riesgo de que esta información se use de forma indebida, ya sea para control político, discriminación o represión; convierte al ciudadano en un sospechoso frecuente, lo que generaría desconfianza frente a las instituciones. Éticamente, un sistema de este tipo solo podría considerarse aceptable si garantiza transparencia, proporcionalidad, protección de datos y rendición de cuentas. De lo contrario, puede convertirse en una herramienta de violación de derechos fundamentales en lugar de una medida legítima de protección.

1. Se le ha nombrado gestor de proyecto dentro de una organización de sistemas de información. Su trabajo es construir una aplicación que es bastante similar a otras que ha construido su equipo, aunque ésta es mayor y más compleja. Los requisitos han sido detalladamente documentados por el cliente. ¿Qué estructura de equipo elegiría y por qué? ¿Qué modelo(s) de proceso de software elegiría y por qué?

R: Confiando en lo requerido, la estructura de equipo más adecuada sería la jerárquica o controlada, de manera que cada miembro trabaje en tareas específicas con claridad de responsabilidades debido a que este tipo de organización permite mantener un control centralizado bajo la dirección del gestor del proyecto. Al tratarse de un sistema extenso, la estructura jerárquica favorece la coordinación entre los distintos módulos y evita desviaciones en la interpretación de los requisitos, garantizando coherencia en los resultados y eficiencia en la comunicación interna.

En cuanto al modelo de proceso de software, la opción más conveniente sería utilizar un modelo en cascada complementado con un enfoque incremental. El modelo en cascada resulta apropiado porque los requisitos están claramente definidos desde el inicio, lo que facilita utilizar un desarrollo en fases. Y, al ser un sistema grande y complejo, aplicar un enfoque incremental permitiría dividir el proyecto en entregas parciales, reduciendo riesgos y posibilitando la validación temprana de componentes críticos.

1. Se le ha nombrado gestor de proyecto de una pequeña compañía de productos software. Su trabajo consiste en construir un producto innovador que combine hardware de realidad virtual con software innovador. Puesto que la competencia por el mercado de entretenimiento casero es intensa, hay cierta presión para terminar el trabajo rápidamente. ¿Qué estructura de equipo elegiría y por qué? ¿Qué modelo(s) de proceso de software elegiría y por qué?

R: Dado que el proyecto busca desarrollar un producto nunca antes visto que combine hardware de realidad virtual con software novedoso, la estructura de equipo más adecuada sería la de un equipo democrático o de tipo abierto. En este tipo de organización se fomenta la colaboración, la creatividad y la comunicación horizontal, lo cual resulta esencial en un entorno donde la innovación y la experimentación son claves para diferenciarse de la competencia. Además, al trabajar de manera flexible y participativa, los miembros pueden aportar ideas originales y soluciones rápidas a los problemas técnicos que surjan, lo que acelera la evolución del producto.

En cuanto al modelo de proceso de software, la mejor opción sería un modelo incremental o ágil (como Scrum o Kanban). Estos enfoques permiten entregar prototipos y versiones funcionales en ciclos cortos, lo que facilita validar rápidamente las innovaciones y hacer ajustes según vaya surgiendo retroalimentación. Al existir presión por lanzar el producto al mercado en el menor tiempo posible, los modelos ágiles reducen riesgos, permiten priorizar funcionalidades críticas y garantizan que el producto final se adapte a las tendencias cambiantes del sector de entretenimiento. De esta forma, se logra un equilibrio entre rapidez, innovación y calidad.