Activitats UD1-ED

Introducció

1. Descriu breument la relació que hi ha entre els components de maquinari principals d'un ordinador i l'emmagatzematge i l'execució del programari.

   Los principales Componentes de un ordebador son el disco duro, la memòria ram, el CPU, y algunos que no son tan importantes pero necesarios para su funcionamiento son los E/S (entrada y salida).

   El disco duro almacena de forma permanente los archivos y los ficheros de datos.

   La memoria RAM almacena de manara temporal el Código binario de los ficheros ejecutables y los datos necesarios para que se ejecute un programa o una aplicación.

   EL CPU es el procesador del ordenador y es el que lee y ejecuta las instrucciones almacenadas en la memoria RAM.

   Los E/S recogen información de entra para el ordenador. Ejemplo el teclado, el ratón… O muestran información como puede ser el monitor, los altavoces.

2. Defineix els conceptes següents:

   - Codi Font

   También es conocido simplemente como “código”, es un conjunto de instrucciones escritas en lenguaje de programación que un programador crea para desarrollar un software o una aplicación.

   - Codi objecte

   Es un forma intermedia de código de un programa que está escrito en código de un programa que se genera durante la compilación y debe ser analizado con otras partes del programa antes de que pueda ser ejecutado.

   - Codi executable

   El código ejecutable es el resultado del proceso de compilación (en lenguajes compilados) o interpretación (en lenguajes interpretados) que toma el código fuente escrito por los programadores y lo convierte en un formato que la computadora puede entender y ejecutar. Este código contiene todas las instrucciones y datos necesarios para que el software funcione según lo previsto.

Cicle de vida del programari

1. Defineix "Cicle de vida del programari".

El ciclo de vida de un programa, también conocido como ciclo de desarrollo de software, se refiere a las etapas y procesos que un programa de software atraviesa desde suconcepción hasta su retiro o descontinuación. Estas etapas están diseñadas para planificar, desarrollar, probar, implementar, mantener y en última instancia, retirar el software de manera efectiva.

2. Anomena les fases principals del desenvolupament de programari i explica breument que es fa a cadascuna.
    1. Requisitos: En esta etapa, se identifican y documentan los requisisots del software. Las necesidades del cliente.
    2. Diseño: Se crea un diseño detallado del software en ducnción de los requisitos. La arquitectura del sistema, la estructura de datos, la interfaz de usuario y otros aspectos técnicos.
    3. Implementación (Codificación): En esta etapa, se escribe el código fuente del programa basado en el diseño.
    4. Pruebas: Se realizan pruebas exhaustivas para verificar que el software funcione correctamente y cumpl con los requisitos definidos. Esto incluye pruebas de unidad, pruebas dde integraciñon y pruebas de aceptación.
    5. Mantenimiento: Una vez que el software está en uso, se requiere mantenimiento continuo para solucionar errores, agregar nuevas características o realizar mejoras.
    6. Actualización:A medida que cambian las necesidades y los requisitos, el software puede requerir actualizaciones periódicas para mantenerse relevante y seguro.

3. Avantatges i inconvenients del model en cascada.

    Ventajas: Estructura clara (bien definida y fácil de entender), documentación completa (cada fase del proceso tiene una documentación completa), control y gestión de proyectos (al ser secuencial y que ninguna fase va hasta terminar una), Adecuado para proyectos pequeños y simples (pues no se epsran cambios signifiactivos).

    Desventajas: Rigidez (no se adapta a proyectos que requieren flexibilidad o que requieran cambios a largo tiempo), dificultad en la detección temprana de problemas(no se pueden detetctar los problemas hasta que se alcance la fase de pruebas, lo que puede aumentar los costos), clientes insatisfechos (los resultados no son tangibles, y no se pueden tener hasta que las fases finales del proyecto y que no se queden insatisfechos con los requisitos), dificultad en la gestión de cambios (hacer cambios puede ser costoso ya que se tendría que volver a etapas anteriores), no adecuado para proyectos complejos o innovadores.

4. Explica com funciona el model de desenvolupament mitjançant la creació de prototips.

    El modelo de desarrollo mediante la creación de prototipos es un enfoque que se utiliza para desarrollar software de manera iterativa y colaborativa, centrándose en la construcción de prototipos o versiones tempranas del software para comprender y refinar los requisitos del cliente.

    Utiliza:1) Recopilación de requisitos iniciales (requisitos del cliente). 2)Desarrollo del Prototipo (se pasa rápido a la fase de diseño y codificación, un prototipo rápido de software, se centra en los aspectos clave de los requisitos).3) evaluación del prototipo (se presenta el prototipo al cliente y usuarios para que lo evalúen). 4)Refinamiento de requisitos (Basándose en la retroalimentación del prototipo, se refinan y ajusta los requisitos). 5)Desarrollo Iterativo (se repiten los pasos interiores en ciclos hasta que los requisitos estén bien definidos y el cliente esté satisfecho). 6)Desarrollo completo del software (una ves se alcanza un conceso con el cliente, se procede con el desarrollo completo del software). 7) (se llevan a cabo pruebas exhaustivas en el software para garantizar que cumple con los requisitos y funciona correctamente). 8) Implementación y entrega. 9)Mantenimiento continuo.

5. Quins principis regeixen el desenvolupament àgil expressats al Manifest Àgil.
    1. La satisfacción del cliente a través de la entrega temprana y continua del software valioso.

2. Aceptar cambios en los requisitos, incluso en etapas tardías del desarrollo. Los cambios son bienvenidos como una fuente de ventaja competitiva.

3. Entregar software funcional con frecuencia, con preferencia a intervalos cortos. Esto ayuda a obtener retroalimentación temprana y a mantener el software siempre listo para su implementación.

4. Colaboración constante entre los miembros del equipo de desarrollo y los interesados. La comunicación efectiva es clave para el éxito.

5. Construir proyectos en torno a individuos motivados. Darles el entorno y el apoyo que necesitan y confiar en que harán el trabajo.

6. Utilizar conversaciones cara a cara como la forma más eficiente y efectiva de comunicación. La comunicación directa minimiza malentendidos.

7. El software funcionando es la medida principal del progreso. La documentación y los informes son importantes, pero el software en funcionamiento es esencial.

8. Mantener un ritmo de desarrollo sostenible. Los equipos deben ser capaces de mantener el ritmo a largo plazo para mantener la calidad.

9. Atención continua a la excelencia técnica y al buen diseño. La calidad del software es fundamental.

10. Simplicidad: maximizar la cantidad de trabajo no realizado es esencial. Evitar la complejidad innecesaria.

11. Los mejores diseños, requisitos y arquitecturas emergen de equipos autoorganizados.

12. A intervalos regulares, el equipo reflexiona sobre su eficacia y ajusta su comportamiento en consecuencia. El aprendizaje continuo y la mejora son esenciales.

https://youtu.be/tGEYClsYyuw?feature=shared

6. Què és una història d'usuari? Consulta el següent enllaç i posa un exemple propi.

Les podem formular de la següent manera: “Com a [perfil], vull [objectiu del software], per a poder [resultat]”.

Las historias de usuario son una herramienta esencial en metodologías ágiles como Scrum o Kanban, ya que permiten que el equipo de desarrollo y el cliente tengan una comprensión común de lo que se debe construir y facilitan la priorización de las tareas. Además, al centrarse en las necesidades del usuario final, las historias de usuario ayudan a garantizar que el software entregado satisfaga las expectativas y sea útil para quienes lo utilizarán.

Ej.: Yo (ivan12DAM) quieor que [mi perfil lo pueda abrir en la televisión], para que se vean en [grande la serie que estoy viendo].

https://es.wikipedia.org/wiki/Historias_de_usuari

7. KANBAN. Estudia els avantatges i els inconvenients de tenir una pissarra web digital per a la metodologia Kanban. Pots consultar els següents enllaços:

https://leankit.com/learn/kanban/kanban-board/

https://trello.com/es
https://taiga.io/
https://kanbantool.com/es/

Ventajas:

·Visulalización del flujo de trabajo
·Flexibilidad (Se puede personalizar según las necesidades específicas de cada equipo)
·Enfoque en el trabajo en curso WIP(evita la sobrecarga de trabajo)
·Priorización continua (el cambio hace que proyectos se puedan cambiar)
·Facilita la colaboración (los elementos del tablero se pueden cambiar segúnlas necesidades del proyecto)
·Identificación rápida de cuellos de botella (identifica las áreas donde el flujo de trabajo se ralentiza)

Desventajas:

·Puede ser menos estructurado. (No apto para trabajos que necesitan tener más rigor)
·Falta de planificación a largo plazo (no se centra en proyectos a largo plazo)
·Dependencia de la disciplina del equipo (la falta de disciplina puede causar problemas en el flujo de trabajo)
·No enfocado en sprints.
·Posible falta de enfoque en la calidad (Kanban se centra en la eficiencia y el flujo de trabajo pero no en la calidad del producto)

8. KANBAN. Fes un resum de la metodologia Kanban i indica les seves diferències davant de SCRUM. Pots consultar el següent enllaç:

https://ca.atlassian.com/agile/kanban

Kanban es una metodología de gestión visual que se utiliza en el desarrollo de software y en una variedad de otros contextos para administrar tareas y procesos de manera eficiente.

Kanban y Scrum son dos metodologías ágiles que se utilizan comúnmente en el desarrollo de software y la gestión de proyectos. Aunque comparten algunas similitudes en su enfoque en la flexibilidad y la colaboración, también tienen diferencias significativas en términos de estructura y prácticas.

Estructura de procesos

Kanban es una metodología de gestión visual que se centra en el flujo de trabajo continuo. No tiene roles predefinidos ni plazos fijos. Las tareas se mueven a través de un tablero Kanban con etapas que representan diferentes estados del proceso (por ejemplo, "Por hacer", "En progreso" y "Terminado"). Kanban permite una adaptación continua y no tiene un ciclo de tiempo fijo como los sprints en Scrum.

Scrum tiene una estructura más formal y prescriptiva. Define roles específicos, como el Scrum Master, el Product Owner y el Equipo de Desarrollo. Los proyectos Scrum se dividen en sprints de tiempo fijo (generalmente de 2 a 4 semanas) y tienen ceremonias específicas, como la reunión de planificación del sprint, la reunión diaria de Scrum y la revisión del sprint.

Plazos

Kamban no impone plazos fijos para completar entregas. No hay una estructura de tiempo específica.

Scrum trabaja concilcos de tiempo llamados sprints, que tienen una duración fija dente 2 y 4 semanas.

Priorización

Kanban prioriza el trabajo en función de la demanda actual y el flujo de trabajo.

Scrum prioriza el trabajo en base a la planificación de sprint. Las tareas se selecciñonan al comienzo de cada sprint y s ecompleyan durante ese período.

Roles

Kanban no define roles específicos

Scrum define roles claramente, incluyendo Scrum Master (responsable de facilitar el proceso), el Product Owner (responsable de gestionar el backlog del producto) y el Equipo de Desarrollo (responsable de la entrega del trabajo).

9. SCRUM. Explica com funciona Scrum. Consulta els enllaços següents:

   Es un método que se basa en ciclos de desarrollo llamados “sprints” que tienen una duración de 2 a 4 semanas. De los cuales un esquipo autoorganizado planifica, desarrolla y entrega un proyecto. Su objetivo principal es permitir la entrega de productos de alta calidad de manera más rápida y eficiente.

10. SCRUM. Defineix els termes següents:

    - Product backlog: El Product Backlog en Scrum es una lista de deseos priorizada para un proyecto. Es manejado por el Product Owner, quien representa al cliente y actualiza continuamente la lista con nuevas ideas y cambios. Los elementos más valiosos de la lista se trabajan en cada iteración de desarrollo llamada Sprint.
    - Sprint backlog: es una lista específica y detallada de tareas seleccionadas del Product Backlog que el equipo de desarrollo se compromete a completar durante un Sprint. Representa el trabajo concreto que se llevará a cabo en esa iteración.

11. SCRUM. A la terminologia Scrum quins termes s'utilitzen com a sinònim de:

    - Cap de projecte: Scrun Master
    - Client: Product Owner
    - Equip de desenvolupament:”Tram”

12. XP. Quines són les característiques distintives de XP davant d'altres metodologies àgils? Explica-les. Pots consultar el següent enllaç:

    Las características distintivas de Extreme Programming (XP) frente a otras metodologías ágiles incluyen:

    1. **Enfoque en la programación:** XP se centra en la mejora continua de la calidad del código y la programación en pareja, lo que resulta en un software más limpio y confiable.
    2. **Pruebas automáticas:** XP enfatiza las pruebas unitarias y de aceptación automatizadas para garantizar la calidad del software y la detección temprana de errores.
    3. **Iteraciones cortas y entregas frecuentes:** XP promueve ciclos de desarrollo cortos, con entregas frecuentes de software funcional al cliente.
    4. **Cliente involucrado:** El cliente trabaja de cerca con el equipo de desarrollo en XP y proporciona retroalimentación continua, lo que garantiza que el producto se alinee con las necesidades del cliente.

5. **Simplicidad:** XP prioriza la simplicidad en el diseño y desarrollo del software, eliminando características innecesarias.
6. **Comunicación y colaboración:** Fomenta una comunicación y colaboración efectiva entre todos los miembros del equipo.
7. **Cambio constante:** XP abraza el cambio de requisitos, adaptándose fácilmente a las necesidades cambiantes del cliente.
8. **Metáfora:** Utiliza una metáfora compartida para facilitar la comprensión del sistema por parte de todos los involucrados.

Llenguatges de programació

1. Quina diferència hi ha entre els llenguatges declaratius i els imperatius? Anomena almenys 2 de cada tipus.

Los declarativos indican el reslutado sin especificar los pasos mientras que los imperativos indican los pasos que se deben seguir para obtener un resultado

Declarativos ejm:SQL

Imperativos: C++

2. Explica què és compilar? Explica què és interpretar?

Compilar: Es el proceso de traducir el código fuente de un programa escrito en un lenguaje de programación a un lenguaje de bajo nivel (código máquina) que la computadora puede entender y ejecutar. El resultado es un archivo ejecutable que puede ejecutarse varias veces sin necesidad de volver a traducir el código fuente.

Interpretar: Es el proceso de leer y ejecutar el código fuente de un programa línea por línea, en tiempo real, sin crear un archivo ejecutable separado. Un intérprete ejecuta el código directamente, traduciéndolo a medida que avanza, lo que permite la ejecución inmediata pero a menudo es más lenta que la compilación.

3. Abatates dels llenguatges compilats.

Mayor velocidad de ejecución.

Mayor optimización del código.

Errores detectados antes de la ejecución.

Mayor seguridad en el código compilado.

4. Avantatges dels llenguatges interpretats

Mayor portabilidad.
Facilidad de depuración.
Menos tiempo de desarrollo.
Flexibilidad y capacidad de modificación en tiempo real.

5. Anomena 2 llenguatges compilats i altres 2 interpretats.

Compilados: C y C++

Interpretados: PHP y JavaScript

6. Es pot considerar codi objecte el **bytecode** generat a Java després de la compilació? Explica'n la resposta.

7. Posa un exemple de llenguatge dels tipus següents:

   - Baix nivel: Lenguaje ensamblador (assembly language)
   - Nivell mitjà: C++
   - Alt nivel: Phyton

8. Quin paradigma de programació segueixen els llenguatges següents?

   - C→Imperativa
   - C++→ Oriantada a objetos, genérica, imperativa…
   - SQL→ Declarativa
   - Java→ imperativa, estructurada, concurrente, genérica y funcional
   - Javascript→ Oriantada a objetos, funcional, basada en eventos, asicrona…
   - Lisp→ Funcional
   - Prolog→ Lógica

Pots consultar el següent enllaç:

https://ca.wikipedia.org/wiki/Paradigma_de_programaci%C3%B3n

9. Explica quins criteris es poden seguir a l'hora d'escollir un llenguatge de programació per al desenvolupament de programari.

Pues los comapos de aplicación, experiencia previa, herramientas d edesarrollo, documentación disponible, imposisición del cliente, entre otros…

Activitat final (no realitzar encara)

Elabora les respostes de les preguntes d'aquesta Unitat i guarda l’arxiu a GitHub en un repositori anomenat 'ActivitatsUD1_ED'.