Santiago Manuel Tamayo Arozamena - Curso 23/24 DAM1

Asignatura: Entornos de Desarrollo **Fecha: 06-10-23**

Programa Informático: Conjunto de instrucciones ejecutadas de forma secuencial encargadas de realizar una tarea o resolver un problema.

Tipos de Software

De Sistema

Administra el hardware e interactua con el usuario. Sistema Operativo. Facilitar interaccion y gestionar recursos.

De Programación

Herramientas que nos permiten desarrollar programas informaticos. IDEs.

De Aplicación

Programas con finalidad mas concreta. Aplicaciones en general...

Lenguajes de Programación

Conjunto de simbolos y reglas que combinados se usan para expresar algoritmos. Un **algoritmo** es un procedimiento preciso y no ambiguo que resuelve un problema. Un **procedimiento** es una secuencia de operaciones bien definidas, que requieren una cantidad finita de memoria y se realiza en un tiempo finito.

Identificadores Constantes Operadores Instrucciones Comentarios

Tipos de lenguaje de programacion

Bajo Nivel

Lenguaje que entiende la maquina. 1s y 0s. Ensamblador.

Medio Nivel

Permite interactuar con el hardware. Sintaxis mas entendible y permite crear estructuras complejas. Para SO. C

Alto Nivel

Palabras basadas en lenguaje natural. Facilmente entendible, necesitan interprete o un compilador. C++, Java, Python...

Estructurados

Tres estructuras de control
- Sentencias secuenciales
- Sentencias selectivas (condicionales)
- Sentencias repetitivas (iteraciones)

Ventajas: faciles de leer, mantenimiento sencillo, estructura sencilla y clara.
Desventajas: Un unico bloque, no permite reutilizacion eficaz.

Orientados a Objetos

Trata los programas como un conjunto de objetos que colaboran entre ellos.

Ventajas: Codigo reutilizable, errores localizables con mas facilidad.

Codigo Fuente: conjunto de instrucciones escritas en un determinado lenguaje.

Codigo Objeto: codigo resultante de compilar el codigo fuente. Codigo maquina o bytecode.

Codigo Ejecutable: codigo resultante de enlazar nuestro codigo objeto con librerías.

Compiladores

Traducen codigo fuente a codigo objeto. Compilación.

Interpretes

Interpretan linea a linea el codigo, no guardan el resultado.

Lenguajes compilados

Codigo fuente>Codigo Objeto

Enlazador une CO con DLLs

Se ejecuta

Fortran, C, Ada, Pascal...

Lenguajes interpretados

Instrucciones ejecutadas linea a linea

Perl, PHP, Bash, Cobol...

Lenguajes virtuales

Similar a compilados pero genera bytecode.

Bytecode puede ser interpretado por MV

Maquinas virtuales de sistema

Emulan completamente un sistema informatico. Permiten varias maquinas completas.

Maquinas virtuales de proceso.

Programas que se ejecutan para interpretar instrucciones. Cuando acaban se cierran.

Desarrollo de software

Ciclo de vida del software

Analisis

Definir que debe hacer el software.
Requisitos:
Funcionales: cosas que debe hacer la aplicacion
No funionales: Caracteristicas del sistema, tiempos de respuesta...
Todo queda reflejado en ERS (Especificacion de requisitos de software)

Diseño

Se divide el sistema en partes y se establecen relaciones. Se recoge en el SDD (Documento de diseño de software)

Codificacion

Se realiza la programación.
El programador debe cumplir los requisitos de las fases de analisis y diseño.

Pruebas

Se prueba el comportamiento de los programas para depurarlos.

Documentacion

Se elaboran:
Guia tecnica
Guia de uso
Guia de instalación

Explotacion

Pruebas en entorno semi-real.
BETA testing.

Mantenimiento

Proceso de control, mejora y optimización del software.

Modelos de ciclo de vida del software

Modelo cascada

Primer modelo en crearse. Cada etapa es independiente de la siguiente y ha de completarse antes de seguir. Permite no arrastrar fallos. No se adapta bien a proceso dinámico. Puede ser retroalimentada, cuando termina vuelve a un paso anterior.

Modelo incremental

Modelo iterativo, completa las secuencias una detras de otra tantas veces como sea necesario. Entrega algo de valor periodicamente.

Modelo espiral

Modelo iterativo incremental. Se eligen las actividades en funcion del analisis de riesgo.

Metodología ágil
SCRUM

Caracteristicas y beneficios de Scrum

Adaptativo y flexible.

Reduccion de riesgos y minimiza errores.

Trabajo en equipo.

Permite teletrabajo.

Mayor productividad

Resultados anticipados para el cliente.

Los proyextos se ejecutan en periodos temporales cortos de 1 a 4 semanas. Se produce un producto potencialmente usable en cada iteración. Se comienza con la idea, y con una reunion entre las prtes que exponen sus requerimientos. Este conjunto de requerimientos ordenados por valor es el Product Backlog.
A continuacion comienza el primero de los Sprints con la planificacion del Scrum Planning, se realizan reuniones diarias, o Daily Scrums, la revision final del sprint o Sprint Review y el Sprint termina con la reunion de retrospectiva, Sprint Retrospective. En este punto comienza un nuevo Sprint.

Roles de Scrum

Product Owner: Responsable de la vision del producto.
Scrum Master: Responsable de aplicar Scrum.
Development Team: Son los encargados de construir el producto.
Stakeholders: Son los usuarios, el cliente, el patrocinador...

Artefactos de Scrum

Product Backlog: Conjunto de requisitos y funcionalidades.
Sprint Backlog: Elementos del Product Backlog a atajar en el Sprint.
Incremento: Resultado de cada una de las iteraciones o Sprints.

Eventos de Scrum

Sprint: Dentro de un Sprint se construye un producto con valor para el cliente.
Sprint Planning: Reunion al comienzo de un Sprint donde Product Owner, Scrum Master y el equipo de desarrollo generan el Sprint Backlog.
Daily Scrum: Reunion corta diaria donde se examina los elementos del trabajo.
Sprint Review: Reunion final del Sprint para inspeccionar el incremento. Se muestra el producto desarrollado.
Sprint Retrospective: Despues del Review se analiza que ha fallado y lo que se puede mejorar.