1. Desarrollo del software

Tags Entornos

▼ Software

Parte inmaterial que hace que un sistema informático entienda las instrucciones (programas) y procese datos. Se clasifica en:

- software de sistema (SO, controladores, herramientas del SO y sus archivos)
- Software de programación (IDE, editores, compiladores)
- Software de aplicación(Tareas específicas, ofimática, etc)

▼ Lenguajes de programación

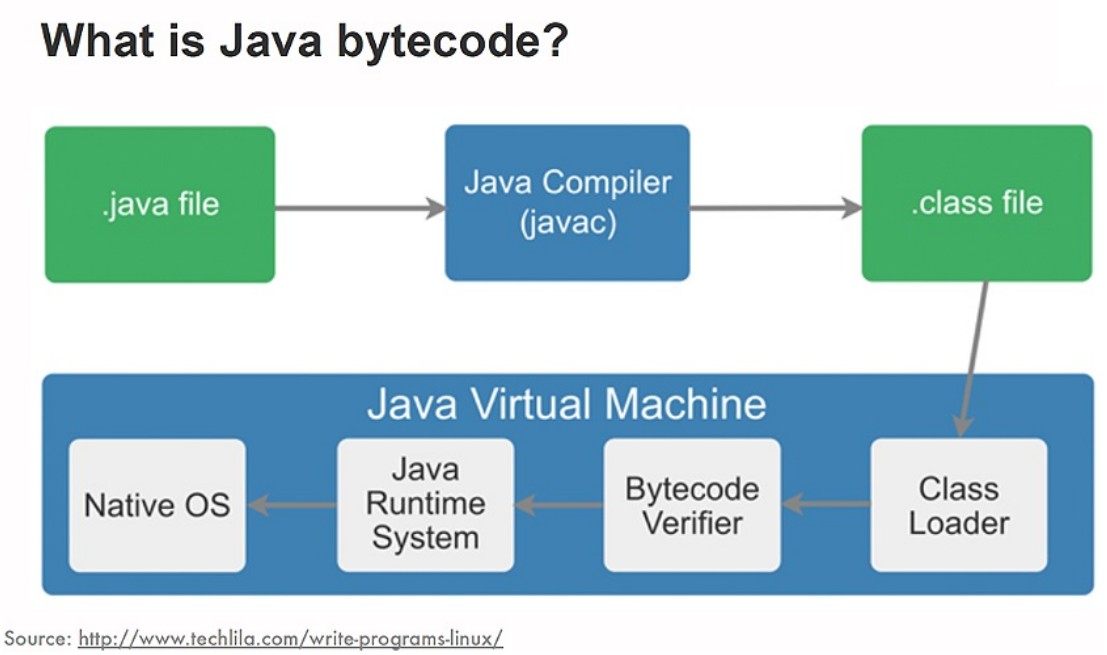
Son lenguajes artificiales que permiten comunicarse con el ordenador y crear programas (conjuntos de instrucciones) usando algoritmos (pasos que indican la secuencia de operaciones).

▼ Niveles de lenguaje

- bajo: maquina, ininteligible aunque no necesita ser traducido para que lo entienda la maquina, tiene 3 operaciones. ensamblador, utiliza mnemotécnicos, difícil de comprender
- medio: precisos, independientes de la arquitectura, necesitan un traductor y reducen el tamaño de los programas
- alto: intuitivo y sencillo, tienen librerias, funciones programadas, frameworks

▼ Formas de ejecución

- Compilados: necesitan compilador para que lo entienda el ordenador. C C++
- Interpretados: necesita un programa que interprete línea por línea el código y lo reproduzca
- Virtuales: se compila un archivo intermedio (bytecode) que es interpretado por una maquina virtual.



Source: http://www.techlila.com/write-programs-linux/

- Estilo
  - Imperativo
  - Orientado a objetos
  - funcional
  - logica

Traducción y compilación

Los traductores convierten el código fuente en algo a nivel de máquina

- interprete: traduce el código línea a línea
- Compilador: traduce a código maquina
  - Fases
    - Se extrae la información en tablas
    - se analiza el léxico
    - se analiza la sintaxis y semántica
    - se genera el código

Desarrollo de una aplicación

Fases

1. Fase inicial

   Se planifica el proyecto, se hacen presupuestos, la más compleja, se necesita mucha experiencia

2. Análisis

   Se analiza el problema, se recopilan, examinan y formulan los requisitos del cliente, documento de requisitos (funcionales, no funcionales, objetivos)

3. Diseño

   Requisitos generales de la arquitectura de la aplicación, como hacer el programa, elegir lenguaje, bases de datos, etc.

4. Codificación

   Se programa, el programa debe cumplir los requisitos anteriores y ser legible para que pueda ser mantenido

5. Pruebas

   Se prueba la funcionalidad del producto, unitarias y de integración (prueba de todo el conjunto del programa), no funcionales(seguridad, estabilidad)

6. Explotación

   Se prueba a conciencia y en casos reales el programa

7. Mantenimiento

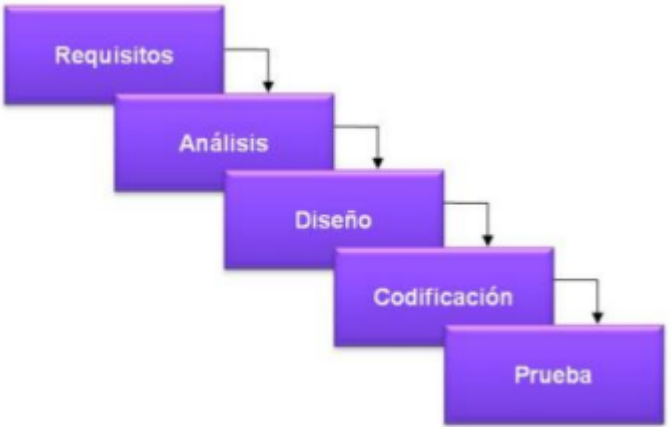
   Se corrigen errores que surjan a lo largo del tiempo

▼ Metodología

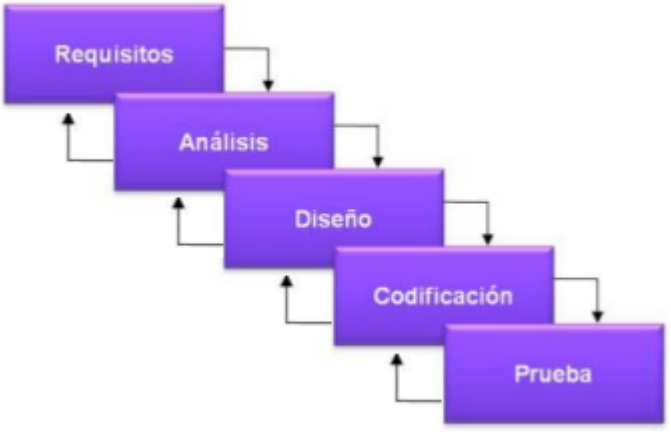
La metodología define el ciclo de vida del software (fases de producción), tenemos:

- ▼ Tradicional

  ciclo de vida en cascada, no admite cambios



Puede ser retroalimentado para producir cambios si son necesarios pero entorpece el proceso



▼ Evolutivos

- Iterativo incremental

  En cascada retroalimentado pero las fases se repiten

- En espiral

  Combinación del anterior y en cascada, muy complicado

▼ Agile

La metodología tradicional era ineficiente, la agile se centra menos en la documentación y mas en hacer frente a los problemas y cambios.

Busca satisfacer al cliente con muy buena comunicación y prometiendo una funcionalidad básica desde el inicio y teniendo un equipo muy productivo

▼ Scrum

Se basa en la colaboración de grupos autogestionados y multifuncionales

1. Product owner: representa al cliente y define las características
2. ScrumMaster: facilita el proceso y elimina obstáculos
3. Equipo de desarrollo: crea el producto bajo ciertos estándares

Primero se hace una lista de las funcionalidades que necesita la aplicacion (Product backlog) y luuego se asigna un sprint de maximo 1 mes en el que se va a terminar el producto. Se hace una breve reunión diaria en la que se comparte el estado, luego se presenta al owner en el sprint review

▼ Roles

- Jefe de proyecto: dirige, gestiona tiempos, habla con el cliente
- Arquitecto: decide como se va a realizar el proyecto, tiene conocimiento profundo en frameworks y librerías
- Analista: produce el diseño, relación fluida con stakeholder
- Analista programador: programador senior, ayuda con diseño y programación
- Programador: Debe conocer el lenguaje y codificar las tareas.

▼ Documentación final

- Manual de usuario: como usar el programa
- manual técnico: como funciona el programa
- manual de instalación