1. Desenvolupament de programari

El desenvolupament de programari és el procés de dissenyar, crear, provar i mantenir programes informàtics que resolguin problemes o compleixin tasques específiques. A continuació, es resumeixen els aspectes clau d'aquest procés.

1.1 Concepte de programa informàtic

Un **programa informàtic** és un conjunt d'instruccions escrites en un llenguatge de programació que un ordinador pot executar per dur a terme una tasca específica. Els programes poden ser molt variats, des de simples aplicacions de càlcul fins a sistemes operatius o aplicacions web complexes. Els programes operen sobre dades, les processen i produeixen resultats segons les especificacions del disseny.

1.2 Codi font, codi objecte i codi executable; tecnologies de virtualització

- **Codi font**: és el conjunt d'instruccions escrites pel programador en un llenguatge de programació de més alt nivell (per exemple, Python, Java, C++). Aquest codi no es pot executar directament per l'ordinador. Per tal d'executar-lo, cal que passi per un procés de compilació o interpretació.

- **Codi objecte**: Un cop compilat, el codi font es converteix en codi objecte, un conjunt d'instruccions en format binari que la màquina pot comprendre. En sistemes **semi-compilats** com Java, el codi objecte es converteix en ***bytecode***, que és un format intermedi. Aquest *bytecode* no és executat directament pel processador de l'ordinador, sinó per una màquina virtual.

- **Codi executable**: El codi objecte, un cop processat i enllaçat amb altres biblioteques, es converteix en un fitxer executable que pot ser llançat directament pel sistema operatiu. Aquests fitxers poden ser de tipus `.exe` o altres formats específics del sistema.

- **Tecnologies de virtualització i màquines virtuals**:
  Les tecnologies de virtualització permeten executar programes en un entorn de virtualització que simula una màquina física, el que ajuda a la **portabilitat** i a la **seguretat**. En el context del desenvolupament de programari, la **virtualització** permet que els programes, com els escrits en llenguatges semi-compilats, s'executin sobre diferents plataformes sense necessitat de recompilar-los.

  - **JVM (Java Virtual Machine)**: En el cas de **Java**, el codi font es compila primer a ***bytecode***, un llenguatge intermedi. La **JVM** és una màquina virtual que interpreta o compila aquest *bytecode* en temps d'execució, cosa que permet que les aplicacions Java siguin portables i puguin ser executades en qualsevol plataforma que tingui una JVM compatible. Això proporciona el gran avantatge de la **portabilitat**: un mateix programa Java pot executar-se en un sistema Windows, Linux o macOS sense modificar el codi font.

Les màquines virtuals de sistemes semi-compilats (com la JVM, el CLR de .NET o el V8 de Google) ofereixen una gran flexibilitat en el desenvolupament, cosa que permet que el codi escrit una sola vegada sigui executat en múltiples entorns sense necessitat

de canviar o recompilar-lo. A més, proporcionen mecanismes de seguretat, gestió de memòria i gestió d'excepcions.

1.3 Tipus de llenguatges de programació. Paradigmes

Els llenguatges de programació es poden classificar en diferents tipus i paradigmes:

- **Llenguatges de programació**: poden ser de baix nivell (com l'assemblador o el C) o de alt nivell (com Python, Java o JavaScript). Els llenguatges de baix nivell interactuen més directament amb el maquinari, mentre que els de més alt nivell són més fàcils d'entendre i d'utilitzar. ** Cal matissar el tema de C com a llenguatge de baix nivell.

- **Paradigmes de programació**:

  - **Imperatiu**: El programador especifica les instruccions que l'ordinador ha de seguir en una seqüència (com C o Java).
  - **Declaratiu**: El programador descriu el problema a resoldre i no com resoldre'l (com SQL, Prolog, XSLT o, en certs aspectes, Haskell).
  - **Orientat a objectes (OOP)**: Organitza el codi en objectes que contenen dades i mètodes (com Java o Python).
  - **Funcional**: Enfocament basat en funcions matemàtiques i evitant estats i dades mutables (com Haskell o Scala).
  - **Lògic**: Utilitza regles de lògica per descriure relacions entre dades i resoldre problemes (com Prolog).

1.4 Característiques dels llenguatges més difosos

Alguns dels llenguatges de programació més populars tenen característiques pròpies que els fan útils en diferents contextos:

- **Python**: És un llenguatge de programació altament llegible i versàtil, utilitzat per a desenvolupament web, ciència de dades, intel·ligència artificial i automatització. La seva sintaxi és simple i clara, el que facilita el seu aprenentatge i ús.

- **Java**: Orientat a objectes, Java és conegut per la seva portabilitat gràcies al principi "*write once, run anywhere*" (escriure una vegada, executar en qualsevol lloc), utilitzant la màquina virtual Java (JVM). S'utilitza per a aplicacions empresarials, mòbils (Android) i en el desenvolupament web.

- **JavaScript**: És el llenguatge dominant en el desenvolupament web per a la creació d'interfícies d'usuari dinàmiques. JavaScript s'executa al navegador i permet la creació de pàgines interactives.

- **C++**: Basat en C, afegeix capacitats orientades a objectes. S'utilitza per a aplicacions de rendiment alt, com simulacions, videojocs i programari de sistemes operatius.

- **Ruby**: És conegut per la seva simplicitat i la seva sintaxi elegant i és utilitzat principalment per al desenvolupament web, especialment amb el framework Ruby on Rails.

1.5 Fases del desenvolupament d'una aplicació

El desenvolupament d'una aplicació sol passar per diverses fases interrelacionades, que inclouen:

1. **Anàlisi**: S'estudien els requisits del sistema, es defineixen les necessitats del client i es determina el que ha de fer l'aplicació. Aquest procés inclou la recopilació de dades i la identificació de funcionalitats.

2. **Disseny**: Es defineixen l'arquitectura del sistema, les bases de dades, les interfícies d'usuari i la forma en què els diferents components interactuaran entre si.

3. **Codificació**: Els programadors escriuen el codi font seguint les especificacions establertes en les fases anteriors. Aquesta és la fase més tècnica del desenvolupament.

4. **Proves**: Es fan proves unitàries, d'integració i de sistema per assegurar-se que l'aplicació funciona correctament. Es busquen errors o defectes (bugs) i es corregeixen.

5. **Documentació**: S'elabora la documentació tècnica i d'usuari per explicar com s'ha dissenyat el sistema i com utilitzar-lo.

6. **Explotació**: Un cop l'aplicació ha estat desplegada, s'inicia la seva utilització. Aquesta fase implica mantenir el sistema en funcionament, resolent incidències que puguin sorgir.

7. **Manteniment**: Inclou actualitzacions, millores i correccions d'errors que poden sorgir després de l'explotació. Aquesta fase pot durar anys en aplicacions a gran escala.

1.6 Procés d'obtenció de codi executable a partir del codi font; eines implicades

El procés d'obtenir el codi executable a partir del codi font implica diverses etapes:

1. **Compilació**: El compilador converteix el codi font escrit pel programador en codi objecte, un codi binari que pot ser entès per la màquina.

2. **Enllaç**: Un enllaçador (linker) agafa el codi objecte generat pel compilador i l'enllaça amb biblioteques externes o altres mòduls per crear el codi executable final.

3. **Depuració**: Abans de la compilació o després de la creació del codi executable, el depurador permet al programador detectar i solucionar errors (bugs) en el codi.

Eines implicades: Els entorns de desenvolupament integrats (IDE) com Visual Studio, Eclipse o PyCharm, ofereixen eines per escriure, compilar, depurar i provar el codi de manera eficient.

1.7 Metodologies àgils. Tècniques. Característiques

Les **metodologies àgils** són enfocaments de desenvolupament de programari que busquen flexibilitat, col·laboració i iteració ràpida per respondre als canvis i millorar contínuament el producte. Algunes tècniques i característiques clau inclouen:

- **Scrum**: Un dels enfocaments més utilitzats, Scrum divideix el treball en sprints curts, habitualment de dues a quatre setmanes, amb reunions diàries (scrums) per revisar l'estat i planificar properes tasques.

- **Kanban**: Es fa servir per gestionar el flux de treball a través de taulers visuals, on les tasques es mouen entre diferents estats (per fer, en procés, fet). Es centra en la millora contínua i l'optimització del flux de treball.

- **XP (Extreme Programming)**: Promou pràctiques com la programació en parelles, el desenvolupament iteratiu, la programació dirigida per les proves (TDD), i una comunicació constant entre el desenvolupador i el client.

- **Característiques**: Les metodologies àgils valoren la interacció amb el client, la resposta al canvi més que seguir un pla rígid, la simplicitat i la sostenibilitat del desenvolupament. La flexibilitat per ajustar-se als canvis en els requisits durant el desenvolupament és una de les seves característiques més importants.