Laborator nr 1 IS

EX 2: Ce este UML și pentru ce este utilizat?

UML (Unified Modeling Language) este un limbaj de modelare vizual utilizat în ingineria software pentru specificarea, proiectarea, documentarea și vizualizarea sistemelor software. Este folosit pentru a reprezenta arhitectura, comportamentul și interacțiunile dintre componentele unui sistem.

Utilizări principale ale UML:

1. Analiză și proiectare software – Ajută la înțelegerea și definirea cerințelor unui sistem.

2. Documentare – Oferă o reprezentare clară și standardizată a arhitecturii software.

3. Comunicare între echipe – Facilitează colaborarea între dezvoltatori, analiști și părțile interesate.

4. Modelare arhitecturală – Reprezintă structura sistemului și modul în care componentele sale interacționează.

5. Testare și mentenanță – Permite identificarea rapidă a erorilor și modificarea sistemului într-un mod organizat.

Principalele tipuri de diagrame UML:

• Diagrame structurale: descriu structura statică a sistemului.

• Diagrama de clase

• Diagrama de obiecte

• Diagrama de componente

• Diagrama de distribuție

• Diagrame comportamentale: descriu dinamica și interacțiunile sistemului.

• Diagrama de cazuri de utilizare (Use Case)

• Diagrama de secvență

• Diagrama de activitate

• Diagrama de stare

UML este utilizat în dezvoltarea software, în special în metodologiile orientate pe obiecte (ex. utilizând Java, C++, Python etc.), dar și în ingineria sistemelor și modelarea proceselor de afaceri.

EX 3

Modelele ne permit să descriem sistemele eficient și elegant. Un sistem este un întreg integrat format din componente care sunt legate de una altul și se influențează reciproc în așa fel încât să poată fi primit ca o unitate unică, bazată pe sarcini sau cu scop. În acest sens, ei se separă de mediul înconjurător [52]. Exempleale sistemelor sunt lucruri materiale, cum ar fi mașini sau avioane, ecologice. medii, precum lacuri și păduri, dar și unități organizatorice precum ca universitate sau companie. În tehnologia informației, ne intereseaza cum este dezvoltat în special în sisteme software și deci în modele care descriu sisteme software.

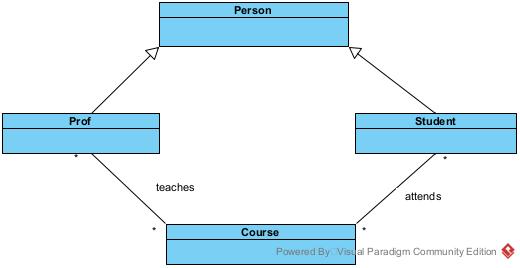
Pentru a face un rezumat exista 3 aplicatii ale modelelor

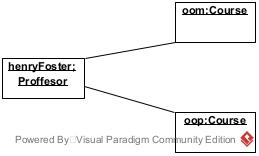
1Models as a sketch

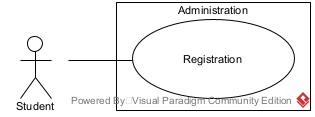
2Model as a blueprint

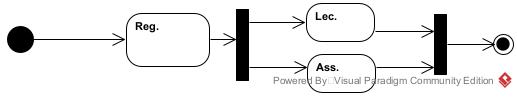
3Models as executable programs

Ex 4











### 

Ex 5

### **Asemănări: Reprezintă aspecte ale unui sistem software** – Toate aceste diagrame sunt utilizate pentru modelarea unui sistem informatic, fie structural, fie comportamental. **Standardizate în UML** – Toate fac parte din standardul UML și urmează reguli și simboluri bine definite.**Utilizate în analiza și proiectarea software** – Ele ajută la înțelegerea, proiectarea și comunicarea cerințelor unui sistem. **Legate între ele** – De exemplu, o diagramă de clase poate influența o diagramă de secvențe sau o diagramă de activitate.

**Deosebiri:**

| **Tip diagramă** | **Scop principal** | **Reprezintă** | **Tip** |
| --- | --- | --- | --- |
| **Diagrama de clasă** | Structura statică a sistemului | Clase, atribute, metode și relațiile dintre clase | Statică |
| **Diagrama de cazuri de utilizare** | Interacțiunea dintre utilizatori și sistem | Actori, cazuri de utilizare, relații | Statică |
| **Diagrama de activitate** | Fluxul logic al proceselor dintr-un sistem | Acțiuni, decizii, ramificări, bucle | Dinamică |
| **Diagrama de secvență** | Interacțiunea dintre obiecte în timp | Mesaje între obiecte, ordinea apelurilor | Dinamică |
| **Diagrama mașinii de stare** | Stările unui obiect și tranzițiile dintre ele | Stări, evenimente, tranziții | Dinamică |
|  |  |  |  |
|  |  |  |  |

**Ex 6 Caracterizarea diagramelor UML**

1. **Diagrama de clasă (Class Diagram)**
   * **Tip:** Statică
   * **Scop:** Reprezintă structura sistemului prin clase, atribute, metode și relațiile dintre ele.
   * **Elemente principale:** Clase, asocieri, moștenire, compunere, agregare.
   * **Utilitate:** Modelarea structurii interne a aplicației și a relațiilor dintre entități.
2. **Diagrama de cazuri de utilizare (Use Case Diagram)**
   * **Tip:** Statică
   * **Scop:** Descrie interacțiunea dintre utilizatori (actori) și sistem.
   * **Elemente principale:** Actori, cazuri de utilizare, relații de asociere, extindere și includere.
   * **Utilitate:** Definirea cerințelor funcționale și a scenariilor de utilizare.
3. **Diagrama de activitate (Activity Diagram)**
   * **Tip:** Dinamică
   * **Scop:** Modelează fluxul logic al execuției proceselor dintr-un sistem.
   * **Elemente principale:** Acțiuni, decizii, ramificări, bucle, puncte de început și sfârșit.
   * **Utilitate:** Descrierea fluxurilor de lucru și a proceselor secvențiale sau paralele.
4. **Diagrama de secvență (Sequence Diagram)**
   * **Tip:** Dinamică
   * **Scop:** Reprezintă interacțiunea dintre obiecte într-un anumit scenariu, incluzând ordinea mesajelor transmise.
   * **Elemente principale:** Actori, obiecte, mesaje sincrone/asyncrone, linii de viață.
   * **Utilitate:** Modelarea execuției unui caz de utilizare pas cu pas.
5. **Diagrama mașinii de stare (State Machine Diagram)**
   * **Tip:** Dinamică
   * **Scop:** Modelează comportamentul unui obiect în funcție de stările prin care trece.
   * **Elemente principale:** Stări, evenimente, tranziții, starea inițială și finală.
   * **Utilitate:** Descrierea ciclului de viață al unui obiect, cum ar fi un proces sau un dispozitiv.

Fiecare dintre aceste diagrame oferă o perspectivă diferită asupra sistemului și contribuie la o mai bună înțelegere și proiectare a acestuia.

