TEMA1 Inginerie Software

1)Ce este UML și pentru ce este utilizat?

UML permite prezentarea aspectelor variate ale unui sistem software (cerințele funcționale și nefuncționale, structurile de date, fluxurile de date și fluxurile de informații) într-un cadru unic, utilizând concepte orientate pe obiecte.

UML este o unealtă puternică pentru dezvoltatorii de software, ajutându-i să conceapă, să documenteze și să implementeze sisteme software complexe într-un mod structurat și eficient.

2) Ce sunt modelele și care este utilitatea lor?

-Modelele sunt reprezentări abstracte sau simplificate ale unui sistem sau fenomen din realitate.

-Ele sunt folosite pentru a înțelege, comunica și gestiona complexitatea sistemelor sau proceselor.

-Modelele sunt bazate pe abstractizare, adică pe identificarea și evidențierea aspectelor esențiale ale unui sistem, în timp ce se omit detalii irelevante.

Utilizare

Înțelegere și comunicare: Modelele permit sintetizarea și prezentarea informațiilor complexe într-un format accesibil și ușor de înțeles. Ele facilitează comunicarea între diferite părți interesate și promovează o mai bună înțelegere a sistemelor și proceselor.

Proiectare și planificare: Utilizarea modelelor în fazele inițiale ale dezvoltării permite proiectarea și planificarea detaliată a sistemelor, ajutând la identificarea cerințelor și a riscurilor potențiale.

Simulare și analiză: Modelele pot fi folosite pentru a simula comportamentul sistemelor în diferite scenarii și pentru a evalua impactul posibilelor modificări sau decizii. Ele facilitează analiza și optimizarea sistemelor într-un mediu controlat și fără riscuri.

Documentare: Modelele pot servi ca documente formale sau informale care capturează arhitectura, funcționalitățile și interacțiunile sistemelor. Ele sunt utile pentru a crea documentație clară și cuprinzătoare pentru dezvoltatori și utilizatori.

Ghidare în implementare: Modelele pot servi drept ghiduri pentru implementarea practică a sistemelor. Ele furnizează un cadru clar și structurat pentru dezvoltarea codului și a altor artefacte software.

4)Caracterizați succint fiecare diagrama prezentată în [1] – Capitolul 2 (cele

5 menționate mai sus)

1. **Diagrama de clasă:**

- Reprezintă structura statică a sistemului.
- Arată clasele din sistem, împreună cu atributele, metodele și relațiile între acestea.
- Utilizată pentru a modela structura conceptuală a unui sistem software și relațiile între diferitele sale componente.

2. **Diagrama de cazuri de utilizare (Use case diagram):**
    - Identifică actorii care interacționează cu sistemul și acțiunile pe care aceștia le pot efectua.
    - Folosită pentru a reprezenta cerințele funcționale ale sistemului și interacțiunile sale cu actorii externi.
3. **Diagrama de activitate (Activity diagram):**
    - Ilustrează fluxurile de control și activitățile din cadrul unui proces sau a unei funcționalități a sistemului.
    - Utilizată pentru a modela logica de control și secvența de acțiuni din cadrul unei activități sau a unui proces.
4. **Diagrama de secvență (Sequence diagram):**
    - Arată interacțiunile între obiecte într-o secvență de timp, evidențiind mesajele trimise între acestea.
    - Utilizată pentru a modela comportamentul dinamic al sistemului, evidențiind ordinea în care sunt efectuate acțiunile între diferitele obiecte.
5. **Diagrama de stare (State machine diagram):**
    - Descrie stările diferite pe care un obiect le poate avea și tranziițiile între aceste stări.
    - Utilizată pentru a modela comportamentul unui obiect sau al unui sistem care trece prin diferite stări în funcție de evenimentele și acțiunile care au loc.