

Inginerie software

Inginerie software - definitii

- o ramură a informaticii care se ocupă cu proiectarea, implementarea și întreținerea programelor complexe
- un proces de dezvoltare a unui produs software într-o abordare sistematică, bine definită

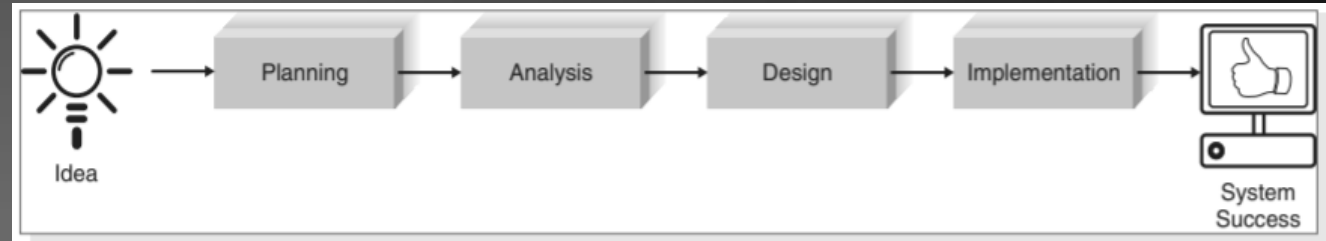
Scopul ingineriei software

- de a rezolva criza de software în care acesta este livrat cu întârziere, cu defecte și peste buget

Etapele dezvoltării unei aplicații software

- La crearea unui produs software există mai multe faze importante, de exemplu:
 - analiza (analiza cerințelor clientului);
 - proiectarea (descompunerea în module cât mai mici a produsului pentru ușurința implementării);
 - implementarea (scrierea efectivă a codului);
 - testarea;
 - întreținerea.

Etapele dezvoltării unei aplicații software



Etapele dezvoltării unei aplicații software

1

Planning

The purpose of this first phase is to find out the scope of the problem and determine solutions. Resources, costs, time, benefits and other items should be considered here.



2

Systems Analysis & Requirements

The second phase is where teams consider the functional requirements of the project or solution. It's also where system analysis takes place—or analyzing the needs of the end users to ensure the new system can meet their expectations.



3

Systems Design

The third phase describes, in detail, the necessary specifications, features and operations that will satisfy the functional requirements of the proposed system which will be in place.



4

Development

Now the real work begins! The development phase marks the end of the initial section of the process. Additionally, this phase signifies the start of production. The development stage is also characterized by instillation & change.



5

Integration & Testing

This phase involves systems integration and system testing (of programs and procedures)—normally carried out by a Quality Assurance (QA) professional—to determine if the proposed design meets the initial set of business goals.



6

Implementation

The sixth phase is when the majority of the code for the program is written, and when the project is put into production by moving the data and components from the old system and placing them in the new system via a direct cutover.



7

Operations & Maintenance

The last phase is when end users can fine-tune the system, if they wish, to boost performance, add new capabilities or meet additional user requirements.



Etapele dezvoltării unei aplicații software

- În cadrul lucrului la un proiect software sunt implicate mai multe persoane, fiecare are atribuții complementare celorlalți:
 - **utilizatori** - exploataatorii softului;
 - **șefi de proiect** - întotdeauna trebuie să existe;
 - **analiști de sistem** - execută modelul informatic al aplicației;
 - **proiectanți de sistem** - execută arhitectura programului aplicației;
 - **programatori** - execută modulele;
 - **ingineri de test** - verifică ceea ce s-a creat.

Unified Modeling Language

(UML)

Unified Modeling Language (UML)

- **UML** acronim pentru **Unified Modeling Language**
- este un limbaj vizual de modelare utilizat pentru specificarea, construirea și documentarea sistemelor de aplicații orientate obiect și nu numai.
- surprinde întreg ciclul de viață al dezvoltării software-ului
- datorită eficienței și clarității în reprezentarea unor elemente abstracte, UML este utilizat și dincolo de domeniul IT
 - există aplicații ale UML-ului pentru management de proiecte, pentru *business Process Design* etc.

Unified Modeling Language (UML)

- Limbajul UML este un limbaj grafic creat pentru descrierea, specificarea și documentarea funcțiilor unei aplicații în timpul fazei de proiectare (design).
- ajută la prezentarea posibilelor erori din structurile aplicațiilor, comportamentele sistemelor și alte procese de afaceri.

Unified Modeling Language (UML)

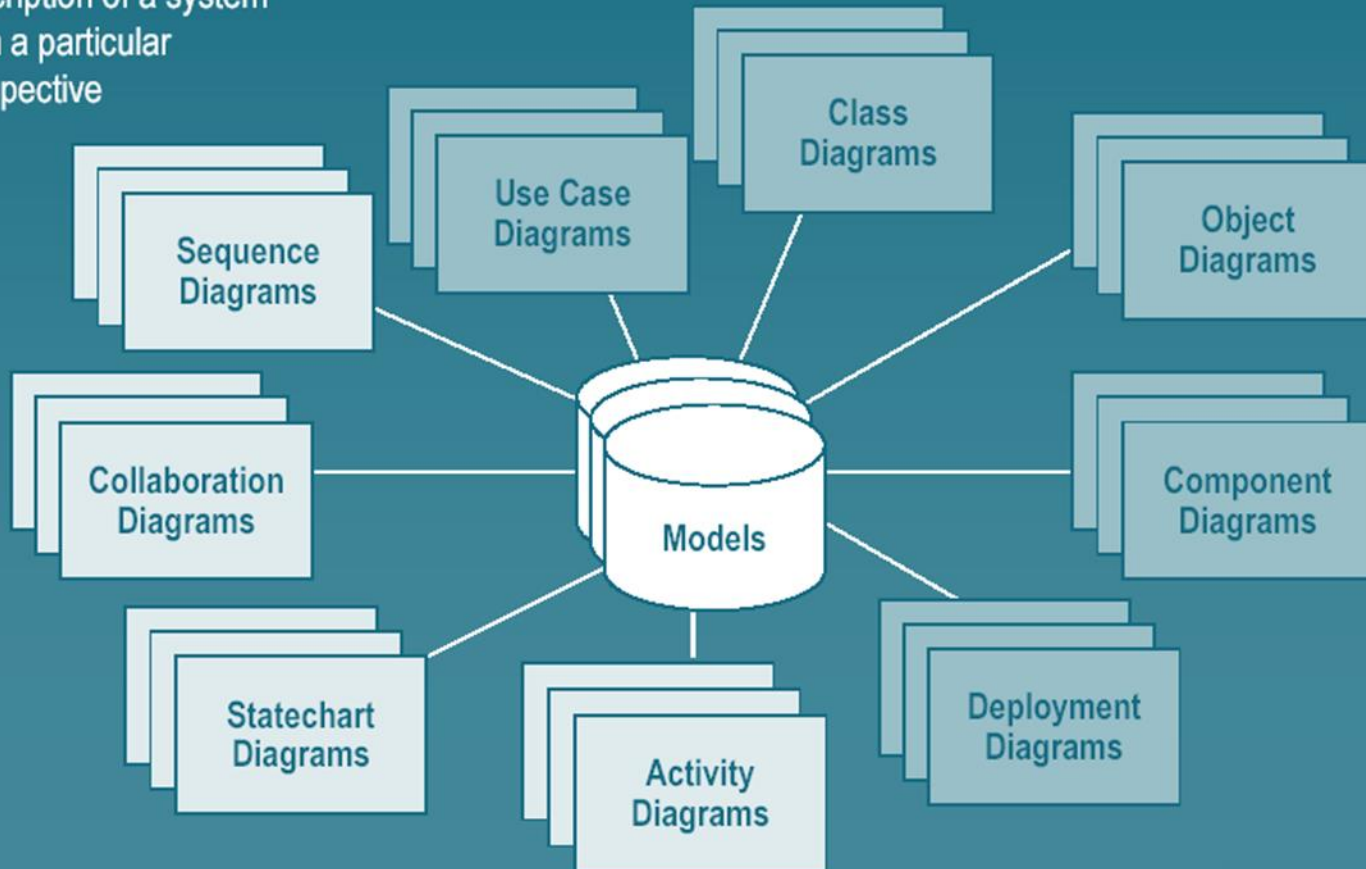
- **Model:** descrierea sau specificarea unui sistem.
 - Un model este de obicei prezentat ca o combinație de desene și text.
- **Utilizarea modelului:** cei responsabili pentru succesul sau eșecul unui proiect se pot asigura încă dinainte de începerea scrierii codului că:
 - funcționalitatea proiectului va fi corectă și completă
 - toate necesitățile utilizatorului final vor fi satisfăcute
 - designul programului va satisface cerințele de scalabilitate, robustețe, extendabilitate etc.

Beneficiile UML

- Simplifică complexitățile
- Menține deschise canalele de comunicare
- Automatizează producția de software și procesele
- Contribuie la rezolvarea problemelor arhitecturale persistente
- Sporește calitatea muncii
- Reduce costurile și timpul de punere în vânzare

Diagramele UML

A **model** is a complete description of a system from a particular perspective



Tipuri de diagrame UML

- **Analiză**

- Diagrama cazurilor de utilizare (Use Case Diagram)
- Diagrama de activități (Activity Diagram)

- **Proiectare**

- Structura: Diagrama de clase (Class Diagram)
- Comportamentul:
 - Diagrama de stări (Statechart Diagram)
 - Diagrama de obiecte (Object Diagram)
 - Diagrama de secvență (Sequence Diagram)
 - Diagrama de colaborare (Collaboration Diagram)

- **Implementare**

- Diagrama de componente (Component Diagram)
- Diagrama de desfășurare (Deployment Diagram)

Diagramele UML

- **Diagrama cazurilor de utilizare:** un caz de utilizare este o descriere a unei funcționalități (o utilizare specifică a sistemului) pe care o oferă sistemul. Diagrama prezintă actorii externi și cazurile de utilizare identificate, precum și conexiunile identificate între actori și cazurile de utilizare.
- **Diagrama de activități:** prezintă algoritmi sau fluxuri de date.
- **Diagrama de clase:** descrie clasele, pachetele, proprietățile lor și diferitele relații care pot apărea între acestea.
- **Diagrama de stări:** prezintă stările pe care le pot avea obiectele în cadrul unui sistem.
- **Diagrama de obiecte:** descrie un anumit scenariu;

Diagramele UML

- **Diagrama de secvență și colaborare:** descrie secvența de mesaje schimbate de obiecte în timpul execuției.
- **Diagrama de componente:** descrie componentele implementate (Ex: fișierele sursă și fișierele obiect).
- **Diagrama de desfășurare:** descrie felul cum componentele sunt repartizate pe diferitele computere.