

# Module G1: Génie Logiciel.

## Crise de logiciel.

- le délai non respecté.
  - coût non respecté (difficulté de prévoir le coût est très élevé)
  - complexité du logiciel (difficile à mesurer)
  - le non satisfaction du cahier de charge (Windows 3 est abandonné à cause du non satisfaction du cahier de charge)
  - logiciel non fiable
- fiabilité** = capable de fonctionner même il ya des problèmes physique : (des conditions anormales)
- non performant** : performance = tant d'exécution de logiciel (tout qui a une relation avec l'exécution du logiciel)
- non efficaces :
- correctif** = corriger les erreurs après livraison
  - adaptative** = les modification du logiciel selon l'environnements et les besoins des utilisateur (mise-à-jour)
  - perfective** = améliorer le logiciel taille, execution, ratissage des erreurs ...)
- maintenance**



qualité du logiciel.

la fiabilité.

la réutilisabilité une partie ou tout le logiciel.

l'extensibilité modifier ou ajouter

la compatibilité il peut être combiné avec plusieurs

programmes (les noyaux).

la portabilité = capable d'être transféré

portabilité = (matériel  $\rightarrow$  matériel)

compatibilité (avec les autres logiciels).

l'intégrité = la qualité de la sécurité (les codes).

facilité d'utilisation

la Validité qui satisfait tous les fonctions du cahier  
cahier de charge

Cycle de vie de logiciel:

- ① Analyse des besoins et étude de faisabilité.
- ② Spécification.
- ③ conception  $\left\{ \begin{array}{l} \text{architecture} \\ \text{Détailé.} \end{array} \right.$
- ④ programmation.
- ⑤ tests unitaire
- ⑥ intégration
- ⑦ tests d'acceptation
- ⑧ exploitation.
- ⑨ Maintenance



① dégager et établir le cahier de charge pour cela.  
il faut étudier le domaine d'application  
(l'état actuel et les ressources disponibles, ...)  
⇒ 1<sup>er</sup> version du cahier de charge (n'est pas final)

mactage = est faire un interface sans programme.

② Décrire le futur Système : Décrire *qu'est-ce qu'il faut faire* sans *comment* il le faire.

③ Comment réaliser (comment le fait).

\* architectural : donner l'architecture de la solution  
(les composants et les informations).

\* Détaillé : donner les structures de données et donner  
les détails de réalisation étape par étape  
pour chaque composant (ex: algorithme.)

④ Programmation : écrire chaque composante au langage  
de programmation (ou plus langage) ensemble de Prog.

⑤ Vérifier que les prog sont ~~satisfaisable~~ satisfaisable.

⑥ les tests d'intégration

⑦ Vérifier si le logiciel satisfait (les conditions du  
cahier de charge est répond aux objectifs ou non.

⑧ exploitation : le logiciel est utilisé.

\* étape C plusieurs activités.

\* les étapes nomme avec l'activité la plus importante.



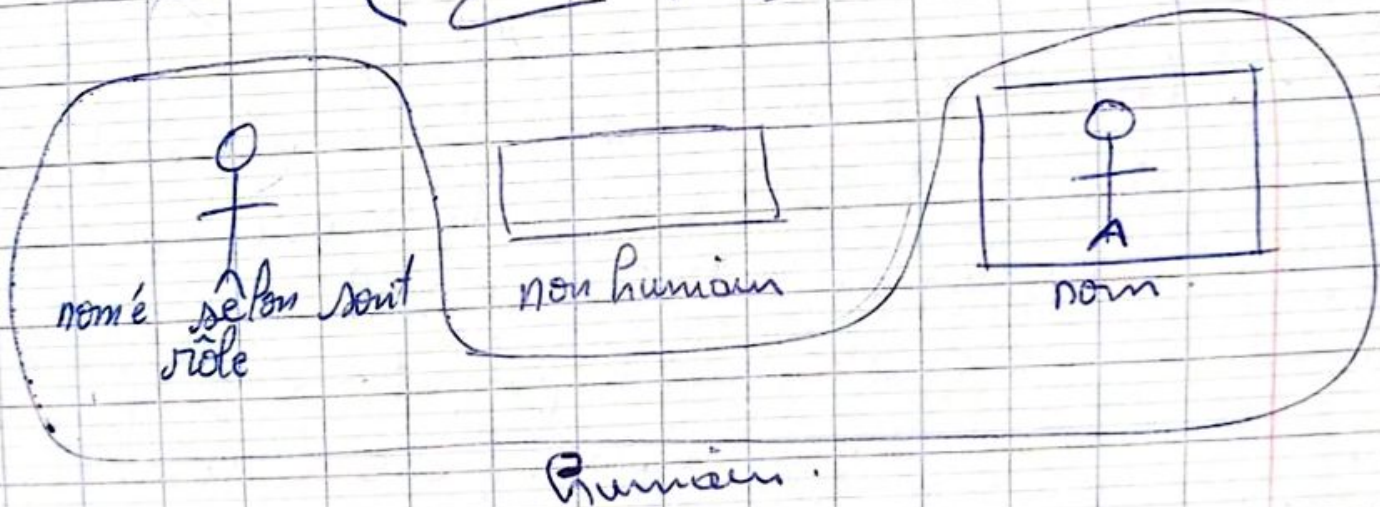
Approche fonctionnel  
un ensemble de fonction  
+ des structures

approche orienté objet  
un ensemble d'objet

UML = langage de Modélisation.

Notion de base.

Acteur = un utilisateur peut être un humain ou non.  
( ~~exemple~~ exemple )



entité externe qui communiquer avec le logiciel  
cas d'utilisation : un cas spécifique (manière SP)  
d'utilisation.

Présentation nom  
ou nom



TD 03 =

Diagramme de cas d'utilisation principal.

Acteur

Acteur Secondaire.

Relation entre Acteur, cas d'utilisation

Relation entre Acteurs.

Relation entre cas d'utilisation

inclusion

extension

generalisation

specification

Description textuelle du cas d'utilisation

Identification du cas d'utilisation.

On répondant à deux questions complémentaires

- Quelle sont les services fournis par le système.

- chaque utilisateur comment utilise le système (ce qu'il attend du système).

Le cas d'utilisation est nommé par rapport un acteur (au point de vue de l'acteur)

l'acteur est nommé selon son rôle.

✓ Identification du cas d'utilisation :

- Nom:

- Résumé objectif

- Responsable

- Dates (création, mise à jour)

- Acteur principaux

- Acteur secondaire.

- Version



Description fonctionnel :

1) Precondition = واش لازم يكون باه بيكون شي سيستم

2) Sequences d'interaction

- Nominale (با لتفصيل اعمل) تسر حو وايت قاعد يبراق

- Alternative (الحالة الباردة) حالة معينة (خلافه لا)

- d'exception . tous est possible en cas où il ya un prb.

- d'exception :

Sinargou Nominal est un Sinargou entre cas d'utilisation et l'acteur parfait.

3) - La poste condition l'etat du system apres execution

3) non fonctionnel (Diagramme cas d'utilisation)

Description fonctionnel :

1-



# Diagramme de classe: (La vie interne de Système)

un objet:



tout élément **individuel**  
**identifiable**

un classe:

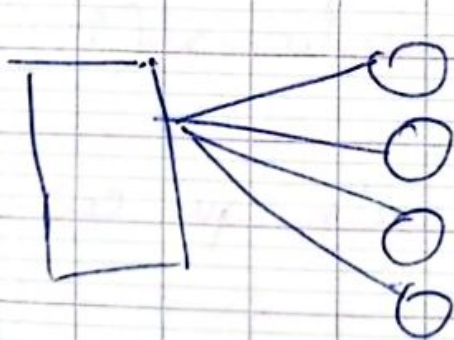
Nom.



La boîte qui de  
caractéristique Statique  
+ comportement, méthode  
fonction.

Les relations entre les classe:

- 1 - Association.
- 2 - Agrégation. Symbole composé 2 
- 3 - Composition. Agrégation + contrainte 
- 4 - généralisation specification.

1/ tout relation peut exister entre deux classes  
(les coordonnées ou set).



هذا امثلة، من يكون عند علاقات  
بخلاف هذا   
و يكون عند علاقات براف مع هذا  
فيكون هذا  يروح هذا



peut être **unidirectionnel** :

اسم واحد ونظيره

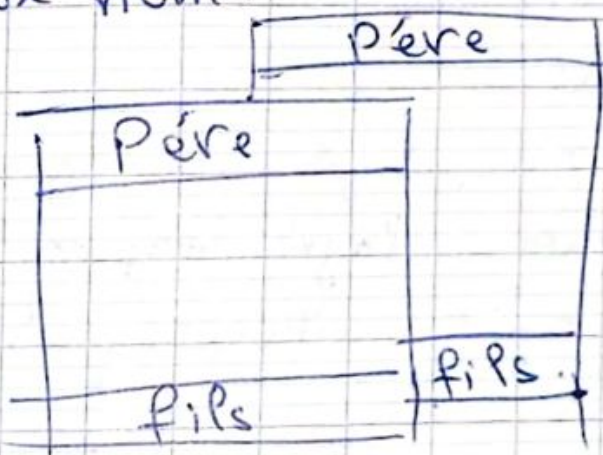
Nom → اسم

أما نفقهو بلي، لعلاقة في اتجاه واحد و في الاتجاه الآخر صندجاش معنى

**bidirectionnel** (العلاقة موجودة بين الطرفين)

a. **Symétrique** : elle a le même sens dans les deux directions نفس معنى الاسم ومكانه في كلا الاتجاهين

b. **Non Symétrique** : la relation n'est pas le même sens dans les deux directions : elle a deux nom.



## Etablissement Diagramme de classe.

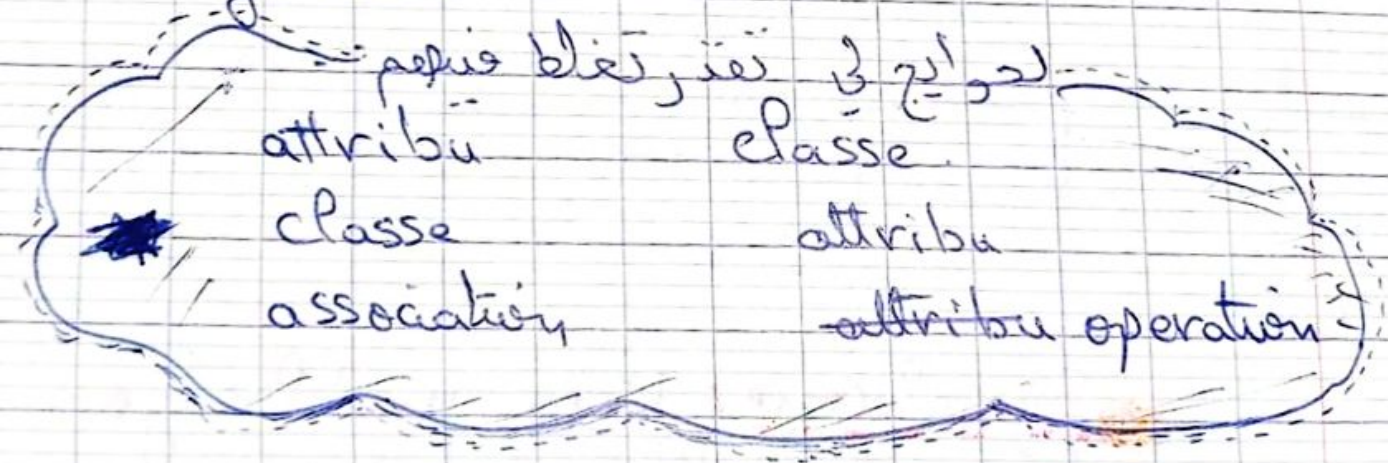
- 1) déterminer toute les classe.
- 2) trouver toutes les associations (relations) qui relient les classes
- 3) déterminer les attributs qui caractérisent les classe.



4) Trouver toutes les opérations.

5) Raffinement + factorisation.

~~des classe isolé ou des classes qui ont les m~~  
fonctionnalité pour mon Système  $\Rightarrow$  éliminer  
la 1<sup>er</sup> et représenté les dans une même  
classe pour 2<sup>em</sup> pour éviter la redondance  
deux classe ont plusieurs attribus  
communs ont les regrouper dans une classe  
générale.

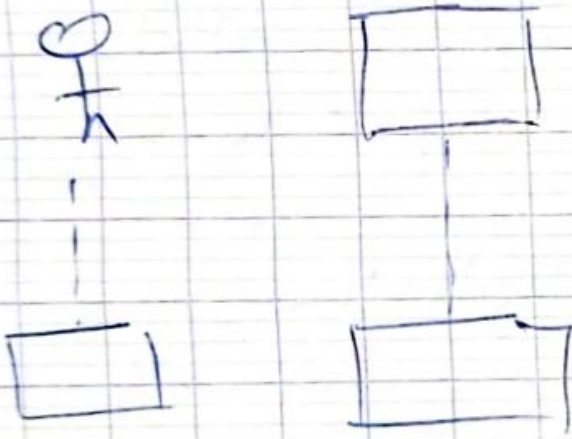




## Diagramme de séquence :

modéliser les communications entre le système et les acteurs ou d'un cas d'utilisation avec les différents acteurs.

représenté par une ligne discontinuée !  
appeler ligne de vie.



**message** → Synchrone : il y a un blocage  
jusqu'à l'envoi du retour.

→ Asynchrone : il n'y a pas de  
blocage →