# Projet dirigé

Introduction, méthodes de développement et étude des besoins

# Introduction

**Plan de la séance**

- Présentation du professeur
- Présentation des étudiants
- Place du cours dans le programme
- Présentation du plan de cours
- Déroulement de la session.
- Introduction, quelques notions théoriques

# Présentation

- Présentation du professeur
- Présentation des étudiants
- Les notes de cours, les laboratoires et le projet dirigé sont sur le site du cours: http://www.salihayacoub.com/420Kbe/420kbe.html

# Place du cours dans le programme

**Objectif du cours**:

- Utiliser une méthode de gestion agile de projet afin de concevoir, en équipe, une application.

**Présentation du cours**

- Ce cours porte essentiellement sur la gestion agile de projets de développement de logiciels, l'analyse des besoins, la modélisation et la conception d'application et le développement de stratégies de tests.

**Contribution du cours au programme**

- D'une part, ce cours va permettre à l'étudiant d'intégrer les compétences et les connaissances acquises durant les sessions précédentes et particulièrement **les cours de bases de données**.
- D'autre part, les compétences et les connaissances acquises dans ce cours seront utilisées dans le cours de projet d'intégration de  la cinquième session.

# Présentation du plan de cours

- Présentation du plan de cours
- Déroulement de la session

# Introduction, quelques notions théoriques

Processus de développement logiciel, c'est quoi et pourquoi ?

- Un processus digne de ce nom doit fournir des directives garantissant le développement efficace de logiciels de qualités.

Qu'est-ce qu'un système logiciel ?

- Un système logiciel se compose de TOUS les artefacts nécessaires à sa représentation sous forme lisible aussi bien pour les machines que pour les humains impliqués dans le développement logiciel.
- Un artefact désigne toute sorte d'information créé, modifiée ou produite ou utilisée par les intervenants dans le processus de développement logiciel (les modèles de BD, les interfaces utilisateur, les cas d'utilisation, les plannings de projet etc …..

# Introduction, activités de développement

Quelles sont les activités de développement d'un produit logiciel?

**Étude des besoins**, dont les objectifs principaux sont:

- De clarifier la demande du client
- D'évaluer la faisabilité du projet
- De fournir à la direction de l'organisation ou au comité directeur les données pertinentes pour prendre une décision au sujet de l'opportunité, de la faisabilité et de la rentabilité d'un projet de développement de logiciel.

# Introduction, activités de développement

- **Analyse**

Cette étape vient après qu'une décision positive ait été prise à l'étape précédente. L'analyse répond à la question **QUOI ?** en d'autres mots, elle répond à la question « Qu'est-ce que le produit logiciel doit faire ?» on ne s'intéresse pas à la question *Comment* ?

Les objectifs principaux sont:

- Évaluer la performance du processus actuel
- Comprendre les problèmes du système à l'étude afin de déterminer les véritables causes de ces problèmes
- Pointer les exigences et les contraintes imposées au système.
- Formaliser les besoins sous formes de cas d'utilisation ou d'user-stories.
- Prioriser et estimer les besoins.
- Proposer un plan de réalisation ou un échéancier

# Introduction, activités de développement

**Conception**

Cette activité répond à la question **COMMENT ?** Elle a pour objectifs de proposer un nouveau produit logiciel qui saura atteindre les objectifs établis au cours de l'analyse.

Elle consiste à déterminer toutes les composantes d'un produit logiciel qui permettrait d'éliminer les problèmes du système actuel et d'atteindre les objectifs établis lors du diagnostic

Les objectifs principaux sont:

- Acquérir une compréhension approfondie des questions concernant les exigences non fonctionnelles et les contraintes liées aux langages de programmation, à la réutilisation des composants, aux systèmes d'exploitation, etc.
- Constitue un point d'entrée pour l'implémentation, au sens ou l'implémentation est un raffinement direct de la conception.
- Déterminer les principales interfaces des sous-systèmes.

Il existe deux niveaux de conception: Global et détaillé

# Introduction, activités de développement

**Implémentation et test (réalisation technique)**

Le plus important produit de la réalisation technique est la portion informatisée du système d'information, c'est-à-dire le logiciel. Les responsables de cette activité devront aussi fournir de documents tels que des manuels d'utilisation et de la documentation sur le système.

Les principales tâches de la réalisation technique sont :

- Planification de la réalisation technique
- Conception physique
- Programmation
- Tests : unitaire, d'intégration
- Préparation et présentation de la documentation

# Introduction, activités de développement

**Le déploiement**

Cette activité est celle qui assure le passage entre l'ancien et nouveau. Afin que ce passage s'effectue avec le minimum de heurts, il est important qu'il ait été planifié avec soin. Les principales tâches de la mise en place sont :

- Planification de la mise en place
- Conversion
- Exploitation et entretien
- Évaluation

CONCLUSION

QUESTIONS ??

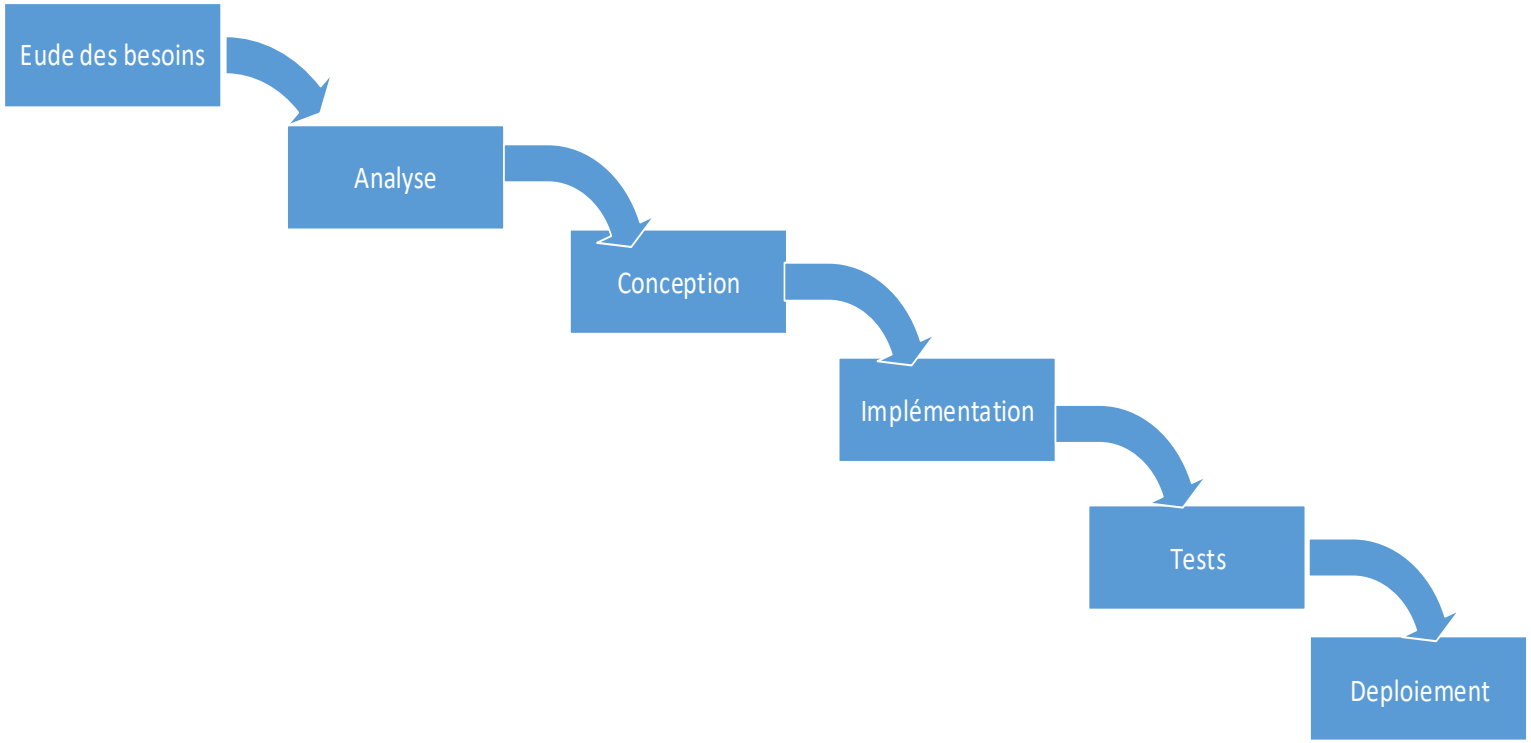
# Méthodes de développement

**Plan de la séance**

- Retour sur la dernière séance
  - Point de vue de l'enseignant
  - Point de vue de l'étudiant
- Rappels
- L'approche en cascade
- Le PU
- L'approche agile
- Manifeste agile
- Conclusion

# L’approche en cascade (**Waterfalls**)

Méthodes en cascade



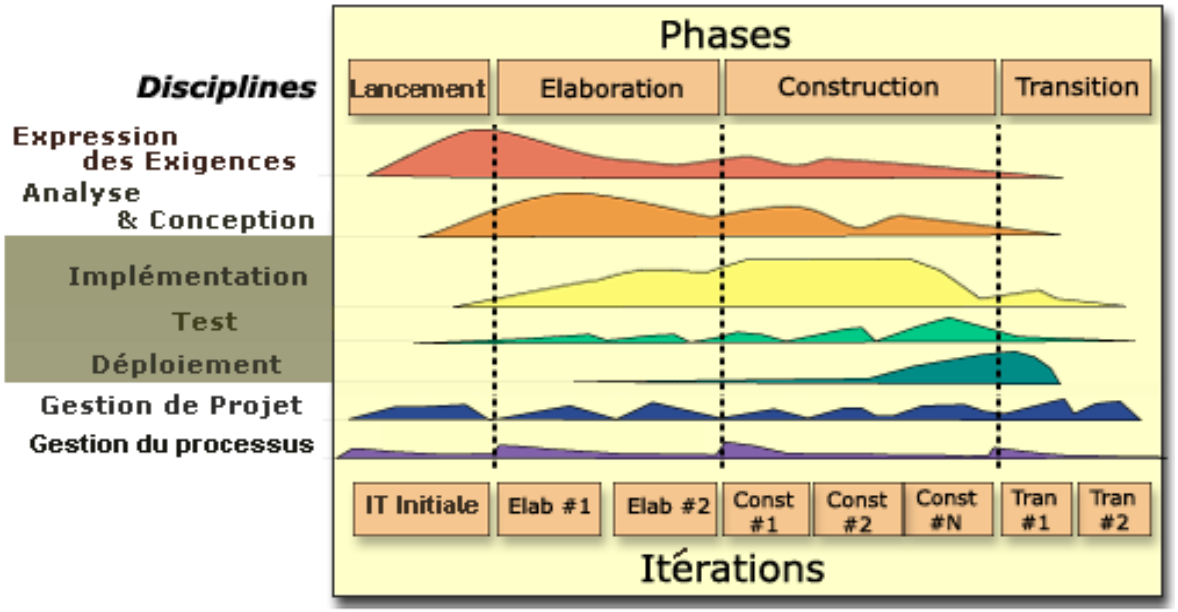
# L’approche en cascade

Problèmes de la méthode en cascade:

- Implication du client, juste au début
- La propagation des erreurs
- La difficulté à corriger les erreurs
- Satisfaction du client: uniquement vers la fin
- Les besoins évoluent.
- Ce n’est pas vrai que l’on puissent comprendre TOUS les besoins au début. Un peu comme vous pour un travail de session, il faut tout le temps valider avec le prof surtout lorsque l’énoncé n’est pas claire.

# Le processus unifié

Le processus unifié




*Source* : https://www.ibm.com/developerworks/rational/library/may05/brown/brown.html

# Le processus unifié

Avantages du PU

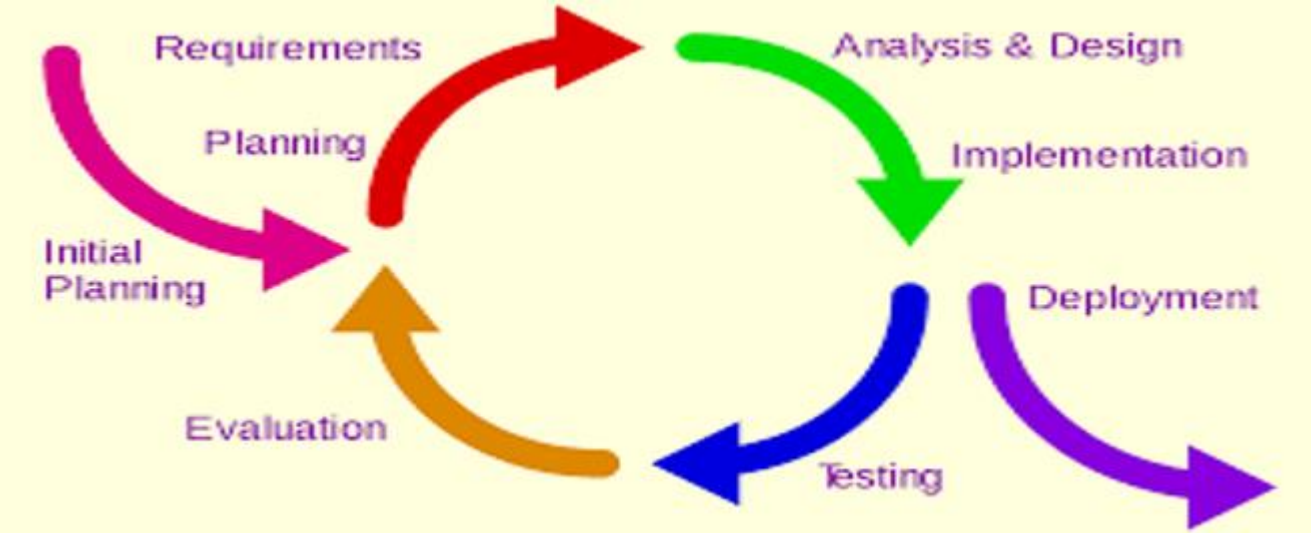
- Itératif et incrémentale
- Repose sur UML pour la modélisation (modélisation objet et unifiée)
- Formalisation des besoins: les uses-case.
- Fusion : Analyse et conception pourrait être vu comme un avantage.

Inconvénients du PU

- Processus trop lourd, surtout pour les petits projets
- Le principe du PU est bon, mais malheureusement sa pratique tend vers le cascade
- La description des uses-case peut être fastidieuse. Documentation lourde

# Approches agiles

L’approche agile




Source de l’image : https://en.wikipedia.org/wiki/Iterative_and_incremental_development

# Approches agiles

Avantages:

- Itératif et incrémental pour vrai pas uniquement dans le principe.
- Livraison tôt et souvent
- Implication du client: les besoins évoluent, feedback tôt.
- Les itérations (sprint) sont courtes.
- La documentation n’est pas lourde.
- Le rythme de travail est constant

# Manifeste agile

**Manifeste agile: 4 valeurs et 12 principes**

**Valeurs agiles**

1. Les personnes et leurs interactions sont plus importantes que le processus et les outils
2. Un logiciel qui fonctionne prime sur la documentation
3. La collaboration avec les clients est préférable à la négociation contractuelle
4. La réponse au changement passe avant le suivi d’un plan.

# Les principes agiles

Le Manifeste annonce douze (12) principes, qui ne définissent pas une méthode agile.

1. Satisfaire le client en livrant tôt et régulièrement des logiciels utiles qui offre une véritable valeur ajoutée.
2. Accepter les changements même tard dans le développement.
3. Livrer fréquemment une application qui fonctionne.
4. Collaborer quotidiennement entre clients et développeurs.
5. Bâtir le projet autour de personnes motivées en leur fournissant environnement et support, et en leur faisant confiance.
6. Communiquer par des conversations face à face.
7. Mesurer la progression avec le logiciel qui fonctionne.
8. Garder un rythme de travail durable.
9. Rechercher l'excellence technique et la qualité de conception
10. Laisser l'équipe s'auto-organiser.
11. Rechercher la simplicité.
12. À intervalle régulier, réfléchir aux moyens de devenir plus efficace.

# Les principes agiles

- Si les principes et les valeurs sont universels, la façon de les mettre en œuvre sur des projets varie. Cette application se fait par l'intermédiaire de ce qu'on appelle une pratique.
- Une pratique est une approche concrète et éprouvée qui permet de résoudre un ou plusieurs problèmes courants ou d'améliorer la façon de travailler lors d'un développement.

Parmi les approches nous avons: XP, **SCRUM**, KANBAN etc….

# Méthodes de développement

CONCLUSION

QUESTIONS ??

# Étude des besoins

**Plan de la séance**

- Retour sur la séance précédente
  - Point de vue de l'enseignant
  - Point de vue de l'étudiant
- Objectifs de l'étude des besoins
- Recueillir les besoins
- Formaliser les besoins
- Les cas d'utilisation: formalisme et description
- Description des cas d'utilisation
- Laboratoire 1.

# Étude des besoins: Objectifs

1

COMPRENDRE CE QUE LE CLIENT VEUT

2

ÉTABLIR LA FAISABILITÉ DU PROJET

3

FAIRE DES PROPOSITIONS DE SOLUTIONS

# Étude des besoins, problèmes

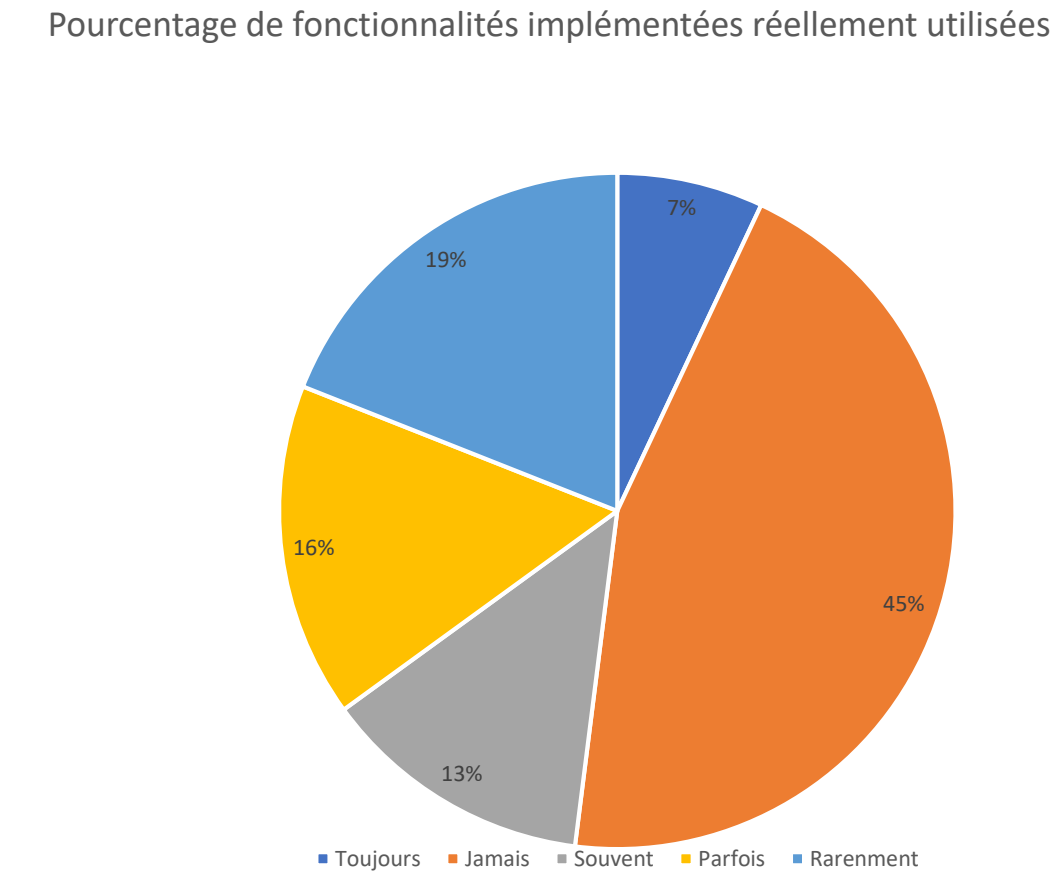



Figure 1, (Source : Gestion de projet agile (3e Edition, Véronique Messager Rota)

# Étude des besoins, problèmes

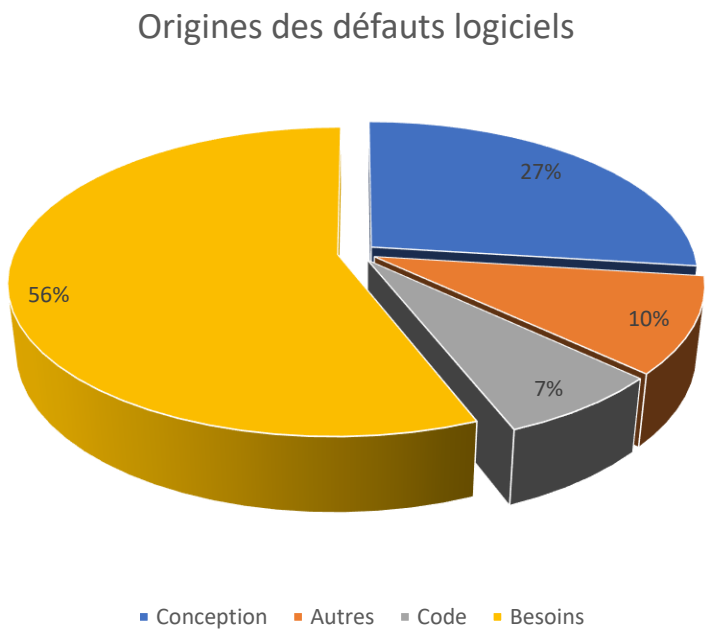



Figure 2, Origines des défauts logiciels (Source : Gestion de projet agile (3e Edition, Véronique Messager Rota)

# Étude des besoins, problèmes

- Comment est-il possible de développer des fonctionnalités qui ne seront jamais utilisées ?
- Est-il possible que nous ayons mal communiqué avec le client ?
- Est-il possible que nous ayons mal compris les besoins du client?
- Le client a-t-il bien exprimé ses besoins ?
- Les attentes du client sont-elles clairement exprimées

# Recueillir les besoins

**Partager une vision**

La vision d'un produit ou d'un projet est l'orientation générale donnée à l'équipe, l'objectif global à atteindre. Sans objectif, sans vision l'équipe de projet ne va nulle part. *« il n'est pas de vent favorable à celui qui ne sait où il va »*, Séneque

**Faire émerger les besoins de manière itérative et incrémentale car:**

- Entre la représentation qu'a le client de son futur produit, la difficulté parfois à le décrire, l'interprétation possible de cette description par le l'équipe de réalisation et le produit qui est livré au final, il y a de nombreux risques de perdre le besoin initial.
- Une idée initiale peut s'avérer inutile, trop coûteuse, ou trop « secondaire » après analyse.
- Tous les besoins n'ont pas la même priorité.
- Le client pourrait introduire une autre demande, renoncer à une demande.

# Recueillir les besoins

**Recueillir les besoins : comment ?**

- Brainstorming
- L'interview
- L'observation
- Le questionnaire
- L'analyse de l'existant

Faire impliquer le client le plus souvent possible car les besoins changent, les besoins évaluent.(exemple: ce qui nous semblait important au début, est finalement peu important ou trop couteux à réaliser)

Ne pas oublier des acteurs dans le système (le puzzle pour lequel il manquerait une pièce)

# Formaliser les besoins

**Formaliser les besoins** : **Pourquoi ?**

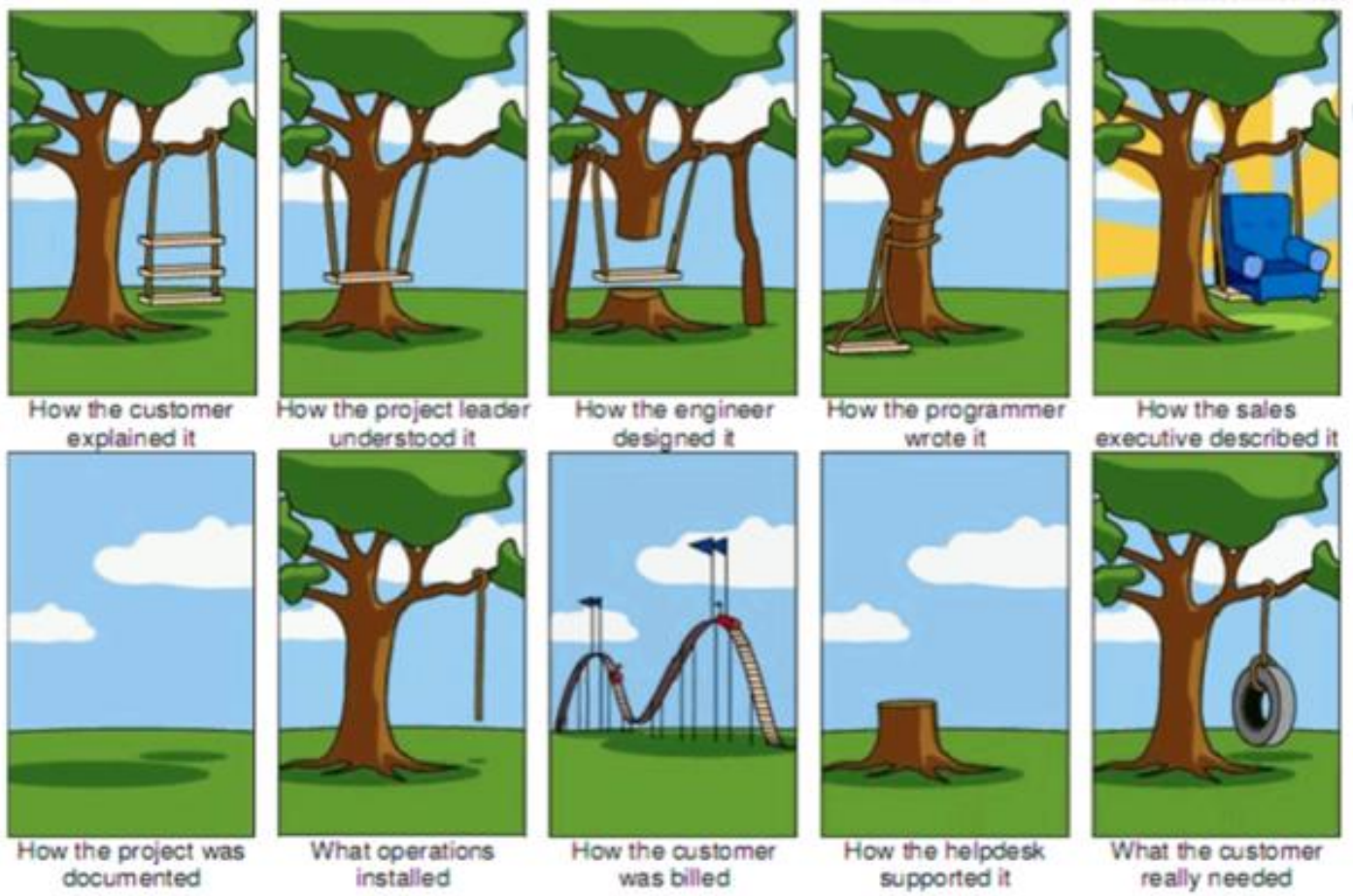
Car on veut retrouver la trace des besoins que nous venons de recueillir. On voudra les consulter, les mettre à jour etc..

**Formaliser les besoins** : **Comment ?**

Comment garder un lien entre les besoins initiaux et les besoins des étapes intermédiaires ? Comment s'assurer que tous les besoins sont traités ? En d'autres mots, comment limiter la rupture dans le processus de développement ?

Voir image suivante.

# Formaliser les besoins




Source de l'image : https://www.supinfo.com/articles/single/7225--definition-besoin

# Formaliser les besoins: norme IEEE830

**Formaliser les besoins : Comment ?**

- **En utilisant la norme IEEE 830**.→on parle de SEL(spécification des exigences logiciel)
  - Doit être exprimée de manière claire, concise et cohérente
  - Doit être non ambiguë
  - Doit être valide
  - Doit avoir un bénéfice qui l'emporte sur le coût
  - Doit être vérifiable
  - Doit être identifiable de manière unique
  - Doit être modifiable (car elle évolue)
  - Doit être importante dans la résolution du problème
  - Doit être réaliste en considérant les ressources disponibles.

# Formaliser les besoins: les cas d’utilisation

- **En utilisant :Les use-case ou les cas d’utilisation d’UML (Unified Modeling Langage)**

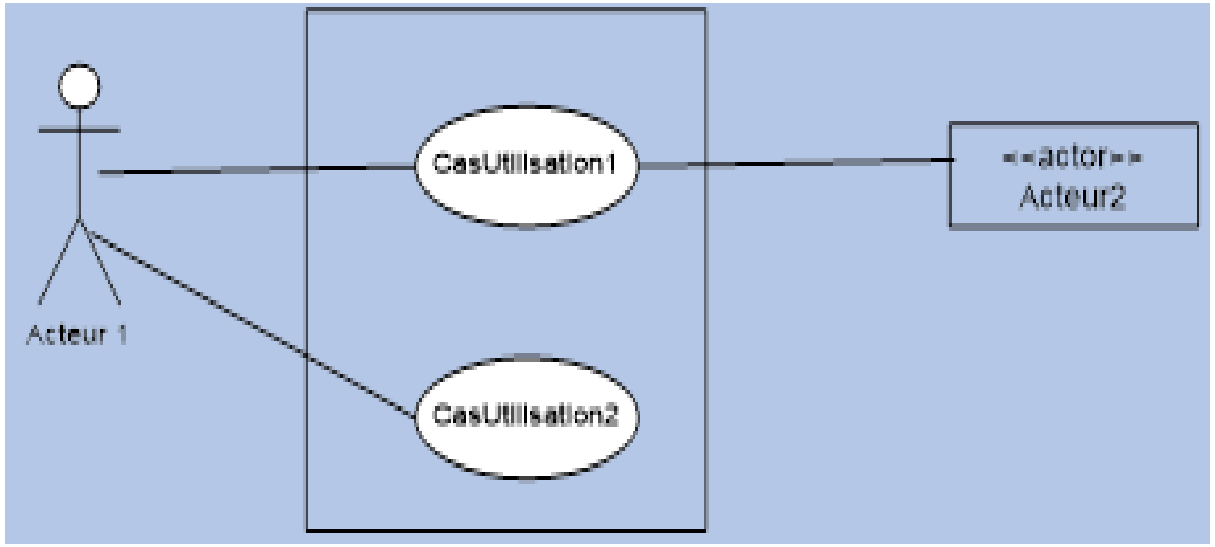
Les cas d’utilisation est une technique de formalisation des besoins utilisée dans le processus unifié. Le diagramme de cas d’utilisation d’UML est utilisé pour représenter les besoins du système ou ce que le système doit faire. Ils décrivent le comportement du système du point de vue de l’utilisateur.

  - Permettent de structurer les besoins des utilisateurs et les objectifs correspondants d'un système.
  - Ils centrent l'expression des exigences du système sur ses utilisateurs ils se limitent aux préoccupations "réelles" des utilisateurs ; ils ne présentent pas de solutions d'implémentation et ne forment pas un inventaire fonctionnel du système.
  - Ils identifient les utilisateurs(acteurs) du système et leur interaction avec celui-ci

# Formaliser les besoins: les cas d’utilisation

**En utilisant :Les use-case ou les cas d’utilisation d’UML (Unified Modeling Langage)**

Le diagramme des use-case permet, d’un seul coup d’œil, de voir les utilisateurs du système et leur interaction avec lui (système



# Formaliser les besoins: les cas d’utilisation

- **En utilisant :Les use-case ou les cas d’utilisation d’UML (Unified Modeling Langage)**

Comment faire un diagramme des use-case ?

- Recenser **tous les acteurs** du système
- Comprendre ce que les acteurs font dans le système ou avec le système
- Un cas d’utilisation n’est pas une action à faire. Un cas d’utilisation est un ensemble de séquences d’actions qui sont réalisées par le système et qui produisent un résultat observable pour un acteur particulier du système
- Déterminer les cas principaux en premier. Toujours revenir au BUT initial.
- Déterminer les liens entre les cas d’utilisation.
- Si un diagramme de cas d’utilisation est trop gros…(trop de cas) il perd de son objectif.

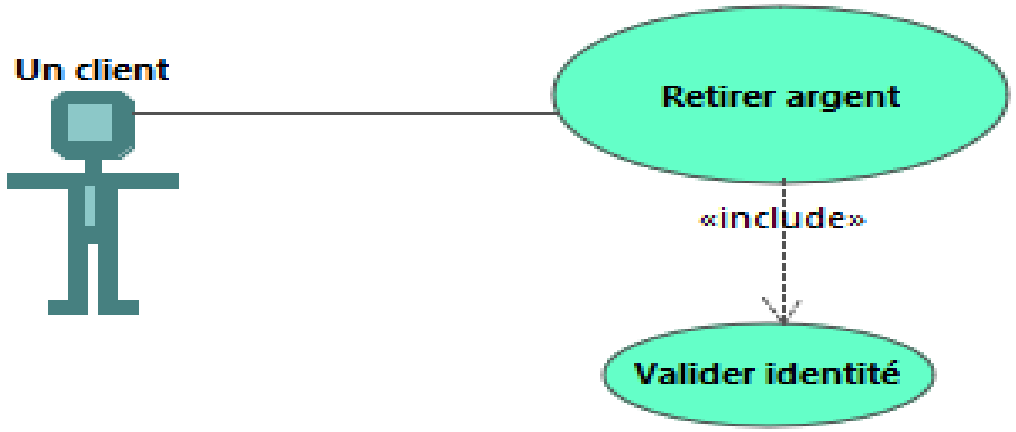
# Formaliser les besoins: les cas d'utilisation

Relation entre cas d'utilisation: La relation « include » ou « uses » (utilise)

Un cas A utilise un cas B si les actions de A ne peuvent s'exécuter

avant l'exécution des actions de B. En d'autres mots, pour exécuter A il faut d'abord exécuter B.

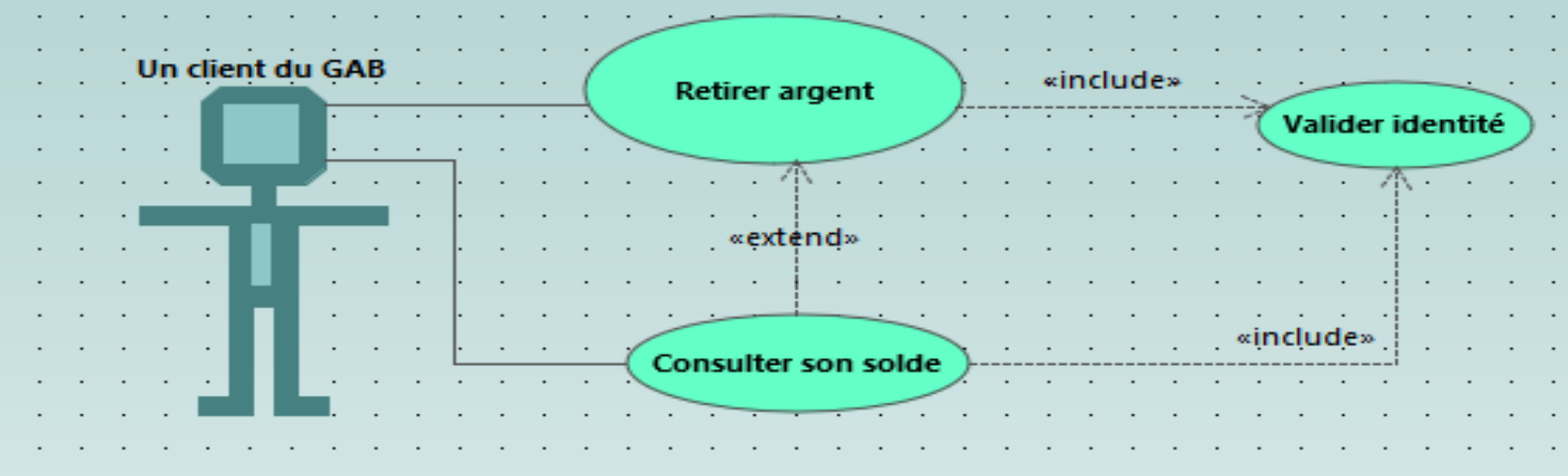
Exemple, un client ne peut retirer de l'argent sans s'identifier



# Formaliser les besoins: les cas d'utilisation

Relation entre cas d'utilisation: La relation « extend »

Un cas A étend son action sur un autre cas B si B est la suite logique de A.  De plus A et B pourrait s'exécuter de manière indépendante.



# Formaliser les besoins: les cas d'utilisation

- Les cas d'utilisation ***Retirer argent*** et *Consulter son solde* sont deux cas indépendants, en ce sens qu'un client du GAB peut consulter son solde sans retirer de l'argent ou qu'il peut retirer de l'argent sans effectuer d'interrogation de solde.

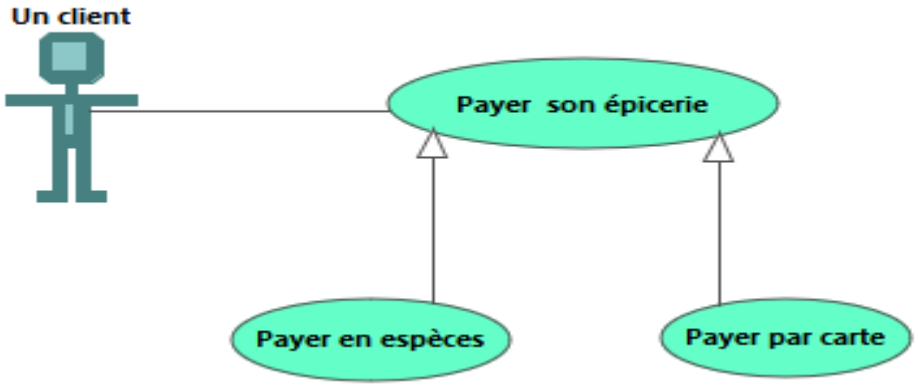
- Cependant le cas ***Consulter son solde*** peut etendre son action au cas *Retirer de l'argent*. Un client après avoir consulter son solde peut décider de retirer de l'argent.

- Quelque soit le cas d'utilisation à exécuter (*Retirer argent* ou *Consulter son solde*), ils doivent faire appel (utilise) au d'utilisation Identifier *le client* (carte débit et nip).

# Formaliser les besoins: les cas d'utilisation

Relation entre cas d'utilisation: La relation de généralisation

- Un cas A est une généralisation d'un cas B si B est un cas particulier de B
- Cette relation de généralisation/spécialisation est présente dans la plupart des diagrammes UML et se traduit par le concept d'héritage dans les langages orientés objet.



# Formaliser les besoins: les cas d’utilisation

- Les cas d’utilisation ont besoin d’être décrits soit textuellement, soit en utilisant un autre diagramme. Deux parties sont importantes lors de la description d’un cas d’utilisation.

1- Le sommaire d’identification

  - Le titre
  - Résumé
  - Acteurs
  - Date de création
  - Version
  - Date de mise à jour
  - Responsable

# Formaliser les besoins: les cas d’utilisation

2- Le scénario nominal.

- Préconditions
- Scenario nominal (enchaînement des opérations dans le cas où le cas d’utilisation se déroule normalement.)
- Postconditions
- Scénario alternatif
- Exceptions

Exemple

# Formaliser les besoins: les cas d’utilisation

Avantages de Use-case:

- Permet de voir les acteurs du système ainsi que leur interaction avec le système.
- D’un seul coup d’œil on peut voir les cas d’utilisation principaux du système.
- La description des cas d’utilisation est riche en information pertinente.

Inconvénient

- L’écriture des cas d’utilisation: c’est très long.

# Formaliser les besoins: les cas d'utilisation

Conclusion

- Les diagramme des cas d'utilisation présente le système du point de vue de ses **acteurs.** Pour déterminer les cas d'utilisation, il faut se placer du point de vue des acteurs et déterminer leur but face au système.
- Il est très important de recenser tous les acteurs du système. Certains acteurs son principaux (car ils interagissent avec le système) d'autres sont secondaires (car il ne font que déclencher le système)
  - Exemple lors d'un prêt de livre à la bibliothèque du CLG, le commis est l'acteur principal pour le prêt, l'étudiant est un acteur secondaire.
- Il est très important de ne pas confondre opération et cas d'utilisation. Un cas d'utilisation est un ensemble d'opérations ayant un but, une finalité pour l'utilisateur.
- Lors de l'élaboration d'un diagramme de cas d'utilisation, il est important de déterminer en premier le ou les cas principaux.
- Le diagramme de cas d'utilisation a pour rôle de voir ce que le système doit faire du premier coup d'œil . S'il y a trop de cas , probablement qu'il faut faire un découpage de votre système en sous-systèmes.
- La description des cas d'utilisation est importante. Elle vient enrichir la compréhension du diagramme des cas d'utilisation. La description doit rester au niveau métier c'est-à-dire doit utiliser le vocabulaire des utilisateurs

# Étude des besoins

CONCLUSION

QUESTIONS ??