

# Atelier Génie Logiciel

1

## CONSIGNES



## **Objectif**

Donner une vision opérationnelle de la gestion de projets informatiques en autonomie

## **Réalisation d'un projet de A à Z**

- Partie Cours qui rappelle les concepts vus les années précédentes
  - Partie accompagnement tout au long du projet
    - Enseignants : Valérie Guimard
  - Projet société Viveris et intervenants Viveris

# AGL

4

Le contenu des rendus décrits ci-dessous est un contenu à minima, vous y ajouterez tous les éléments de gestion de projet que vous avez mis en place même s'ils sont à vocation interne à l'équipe, comme par exemple des compte-rendu de réunion.

## **Rendu 0 : Point questions**

### **Rendu 1 : début 5 novembre 2024**

1- Première version de l'expression de besoin intégrant les précisions éventuelles suite à vos échanges avec votre client sous forme d'une liste d'exigences structurées avec les cas d'usages à tester.

2- Choix technologiques (même si non finalisés)

3- PAQ (Plan d'Assurance Qualité) finalisé avec planning prévisionnel, matrice des risques et matrice RACI

**Le PAQ permet de définir la manière dont on va travailler, collaborer. Il doit être minimaliste mais décrire simplement et efficacement votre organisation et les rôles et responsabilités.**

Il intègre le planning prévisionnel.

**Pour rappel : il doit être conçu pour que tout nouveau collaborateur intégrant l'équipe ait de quoi travailler correctement**

4- Présentation lors d'une réunion de lancement selon support fourni

## Rendu 2 : 16 Décembre 2024

- 1- Version finale de la matrice des exigences/cahier des charges intégrant les précisions éventuelles suite à vos échanges avec votre client. Il valide votre engagement vis à vis du client. Il inclut la matrice des exigences
- 2- Maquettes des IHM
- 3- **Mini prototype sur une seule fonctionnalité montrant la maîtrise des technologies choisies et notamment l'intégration des parties front et back** – A définir avec la maîtrise d'ouvrage
- 4- Analyse UML
  1. Diagrammes de classes (Vous n'êtes pas contraints de représenter les classes liées aux interfaces graphiques, mais à minima les interfaces (au sens objet) principales des classes de l'IHM)
  2. Diagrammes de 2 cas d'utilisation et/ou userflow
- 5- Planning mis à jour,  
Intégrant les premiers temps passés de chaque membre de l'équipe sur chaque tâche
- 6- Réunion selon support fourni

**Pré soutenance : 15 janvier 2025**

1- Démonstration selon état d'avancement

2- Cahier de recette, décrivant précisément le jeu d'essai que vous présenterez lors de la soutenance finale

**Rendu et soutenance finale : Semaine précédent les vacances**

1 - Une matrice de respect des exigences du cahier des charges indiquant pour chaque exigence si elle a été respectée, non respectée ou avec des réserves,

2- Dossiers de tests - Votre réalisé final, avec votre charge consommée et un rapide retour d'expérience sur les points qui vous ont posés problème,

3- La documentation utilisateur des fonctionnalités présentées

4- Ensemble des sources et fichiers de tests.

## Soutenance

Le découpage des soutenances sera le suivant :

- 5 minutes : Présentation "commerciale" des points forts du produit,
- 5 minutes : Présentation des points importants du projet côté technique/ développement / exploitabilité (orienté présentation du produit à une MOE),
- 5 minutes : Organisation des développements et répartition des tâches projet (bien clarifier qui a fait quoi),
- 10 minutes : démonstration de l'outil,
- 5 minutes : bilan du projet - retour d'expérience du chef de projet et de chaque membre de l'équipe (quels ont été les éléments qui se sont bien passés ou mal passés dans le projet, quelle expérience en tirez-vous pour des projets suivants),
- 20 minutes : retour enseignants/clients + questions/réponses.

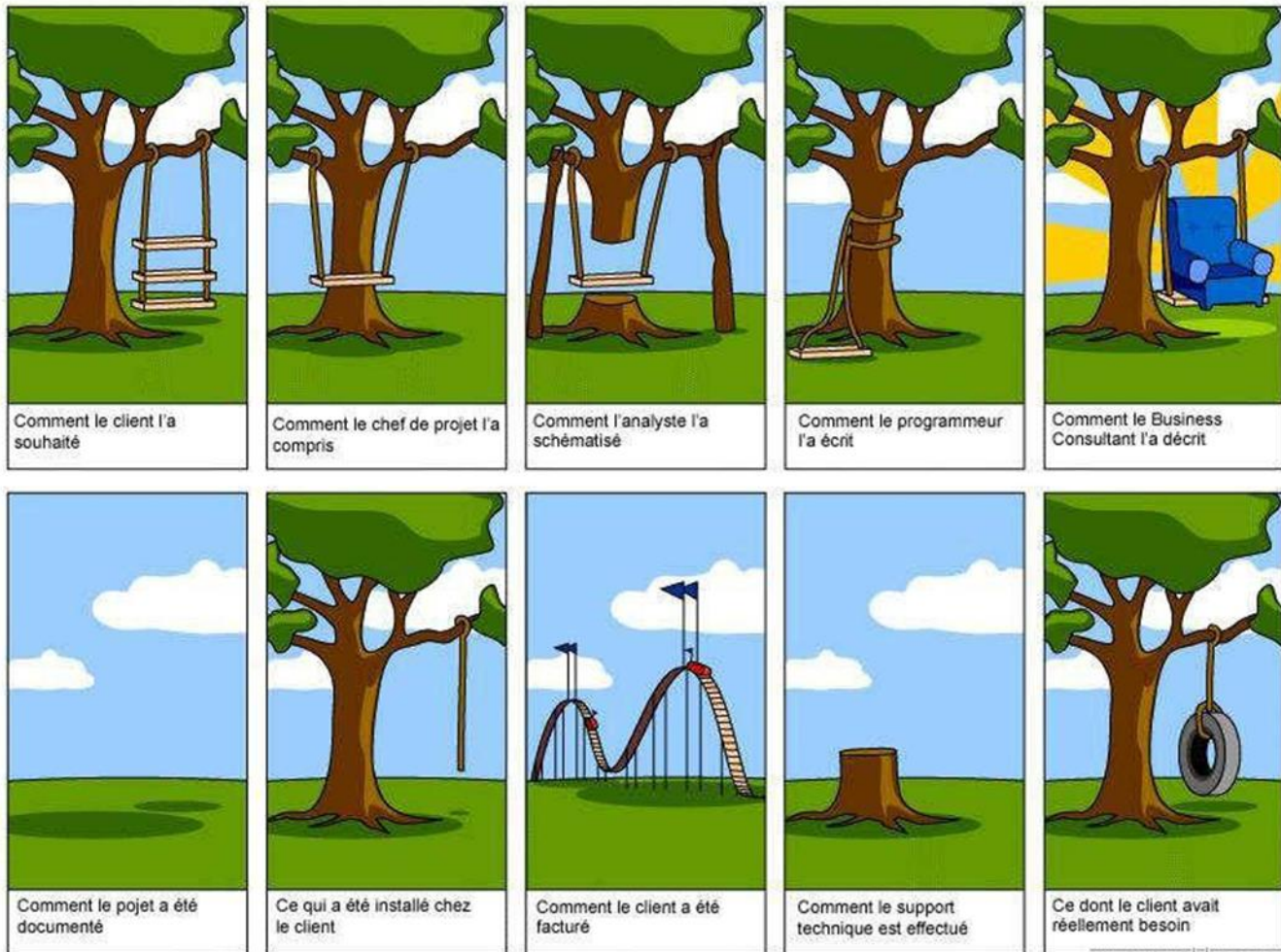
## **Modalités de rendu :**

- Les documents liés au projet seront à envoyer par mail à travers un lien
- la bonne réalisation du produit interviendra dans la notation finale mais ne constituera qu'un des éléments, au même titre que la qualité des documents fournis, la maîtrise du processus de développement et la manière de gérer les relations avec le client.
- la note peut être individualisée au vu du rôle de chacun et de sa contribution effective au projet
- Vous avez le choix des technos et des méthodes de développement du moment que les livrables requis sont fournis. Les méthodes qui demanderaient une forte réactivité du client sont à éviter ou des délais dans les réponses doivent être anticipés.
- les documents doivent être livrées sous une forme directement imprimable, en version pdf.



# Attention ...

9



# Ex Plan type d'un PQP

10



# Matrice des risques

11

<i>Risques</i>	Gravité (G : 1-4)	Fréquence (F : 1-4)	Criticité (GxF)	Prévention	Réparation	Responsable

# Matrice RACI

12

- Elaboration du Diagramme des responsabilités  
RACI : **R**éaliser **A**pprouver **C**onsulter (Avant) **I**nformer (Après)

N°	Libellé Tâche	Pierre	Paul	Jacques	Alex	Georges
G1	Tâche 1	<b>R</b>		<b>A</b>	<b>C</b>	<b>I</b>
D2	Tâche 2		<b>RA</b>			
S4	Tâche 3	<b>R</b>	<b>R</b>	<b>C</b>	<b>A</b>	
V4	Tâche 4	<b>RA</b>				<b>C</b>

Pour rappel : tâche x peut être un lot, un livrable ou encore un document  
La matrice sert à la gouvernance du projet et à la gestion des livrables

# Tableaux divers

13

## Matrice des exigences

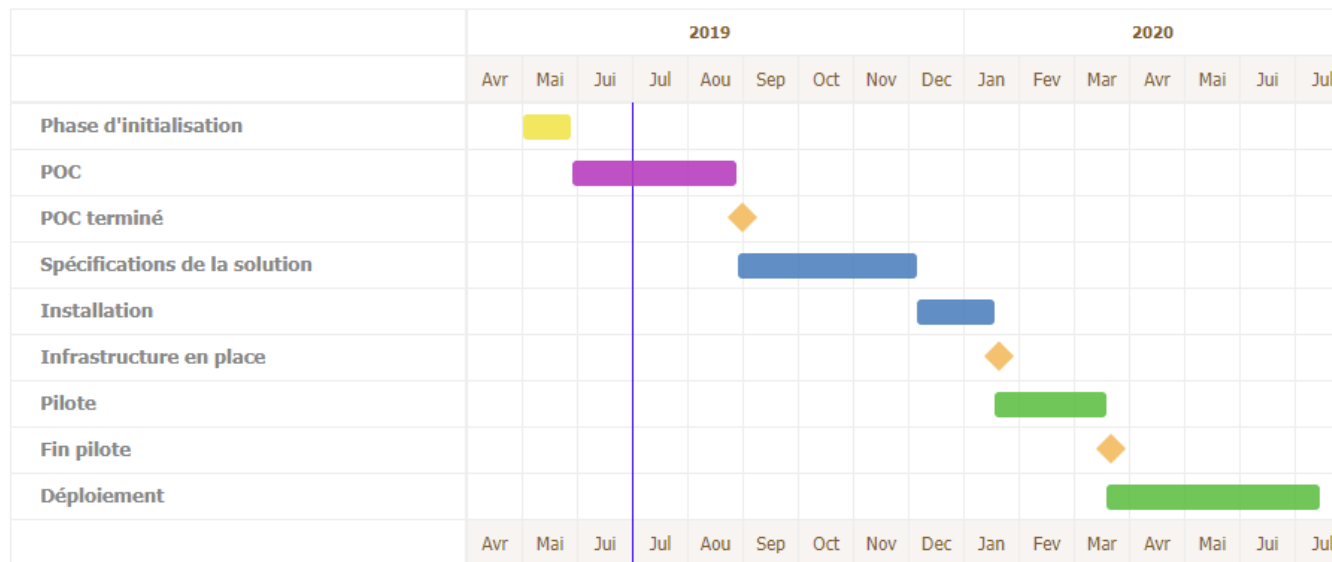
Id	Exigence	Description	Priorité	Statut
REQ1	Authentification des utilisateurs	Les utilisateurs doivent pouvoir se connecter au système à l'aide d'un identifiant et d'un mot de passe valides.	1	En cours

## Cahier de Scénarios des grandes fonctionnalités

Id	Scénario	Description	Etapes	Résultats attendus
s1	Réservation d'un billet	Un utilisateur souhaite réserver un billet en ligne.	1- sélectionner l'évènement 2- Choisir le nombre de billets 3- sélectionner les places 4- Effectuer le paiement	Le billet est réservé avec succès et un numéro de confirmation est affiché

# Macro Planning

14



# Sommaire Réunion de lancement

15

## **Présentation du projet : Contexte**

Bénéfices attendus du projet / Objectifs et KPI

L'équipe Projet / Les parties prenantes du projet

Les phases du projet / Macro planning

1ère analyse des risques du projet

Gouvernance

Communication

Prochaines réunions

# Sommaire Copil

16

## **Point planning**

Détail avancement du projet

Sujets divers

Décisions à prendre

Communication

Synthèse

Prochaine réunion



17

Planning : 😊 ➡ 😞

Couts : 😊 ➡

Qualité : 😊 ➡

Organisation : 😊 ➡

Cadrage

Conception

Réalisation

Recette

Déploiement

Initial

01/12/2020

Révisé

15/01/2021

## 1. Principales actions menées

XXX

## 2. Décisions prises

## 3. Points à soulever & Risques

XXX

## 4. Actions à mener

## 5. Décisions à prendre

XXX

## 6. Réunions prévues et échéances

# Pourquoi Modéliser ?

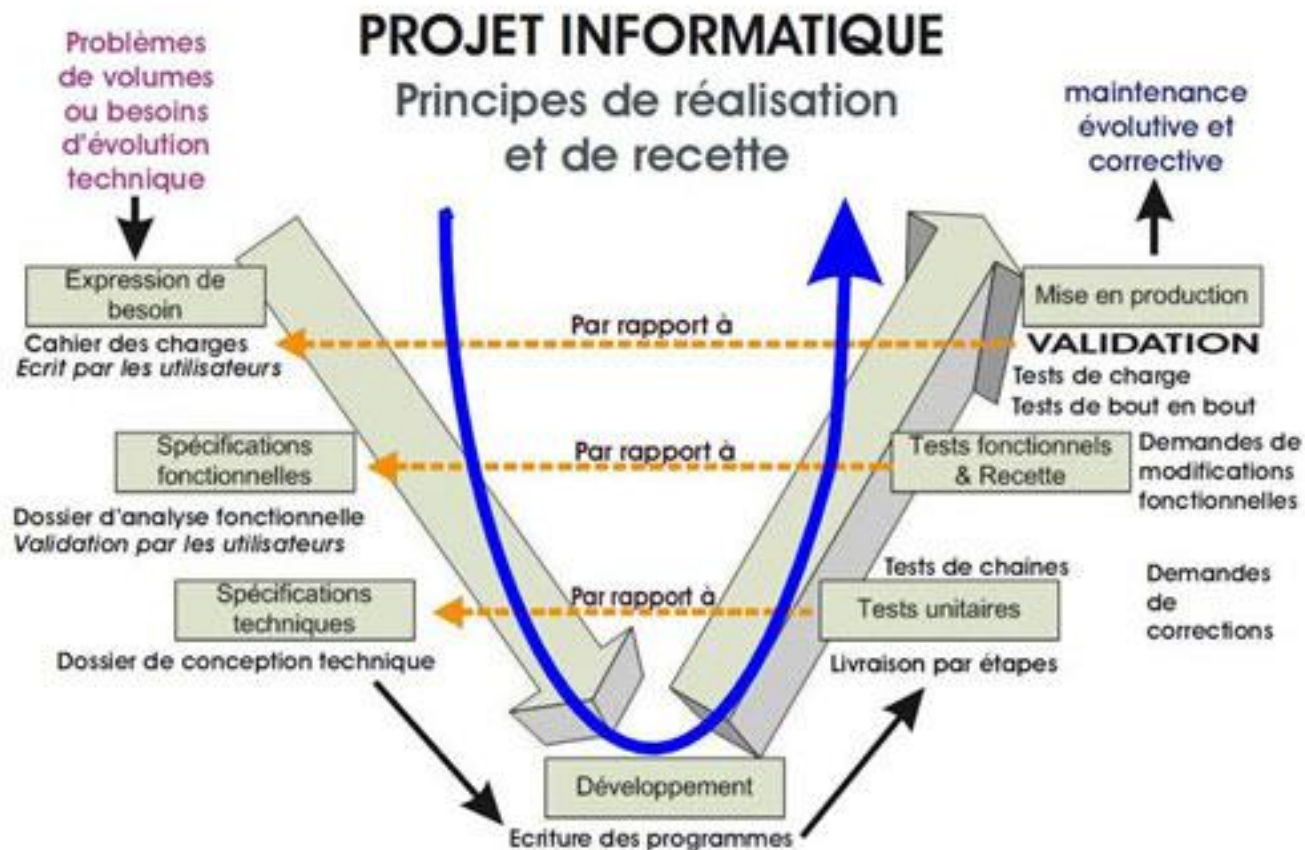
18

- Pour **mieux comprendre** le fonctionnement du système.
- Pour **maîtriser** sa complexité
- Pour **assurer sa cohérence**.
- Pour définir un **langage commun, précis**, qui est connu par tous les membres de l'équipe
- Pour **communiquer**.

→ Cette communication est essentielle pour aboutir à une compréhension commune aux différentes parties prenantes (notamment entre la maîtrise d'ouvrage et la maîtrise d'œuvre informatique) et précise d'un problème donné.

# Cycle en V

(19)



# Principe de la Méthode Agile

20

- **Les méthodes agiles** partent du principe que spécifier et planifier **dans les détails** l'intégralité d'un produit avant de le développer (approche prédictive) est contreproductif.

- **Explications**

Cela revient à planifier dans les détails un trajet "Paris - Barcelone" en voiture par les petites routes. Spécifiant chaque villes et villages traversés, l'heure de passage associée, chaque rue empruntée dans les agglomérations, litres d'essence consommés, kilomètres parcourus, etc.

Les imprévus ne manqueront pas d'arriver : embouteillages, déviations, travaux, sens de circulation inversés, voire la panne, etc. Rendant votre planification et vos spécifications très vite obsolètes.

Le temps passé à planifier cet itinéraire semble perdu et on ressent une grande frustration de ne pas pouvoir appliquer notre plan à la lettre. Si l'on revient sur le projet informatique, il y a une incertitude inévitable, et le besoin ne peut pas être complètement connu tant que les utilisateurs ne l'ont pas utilisé

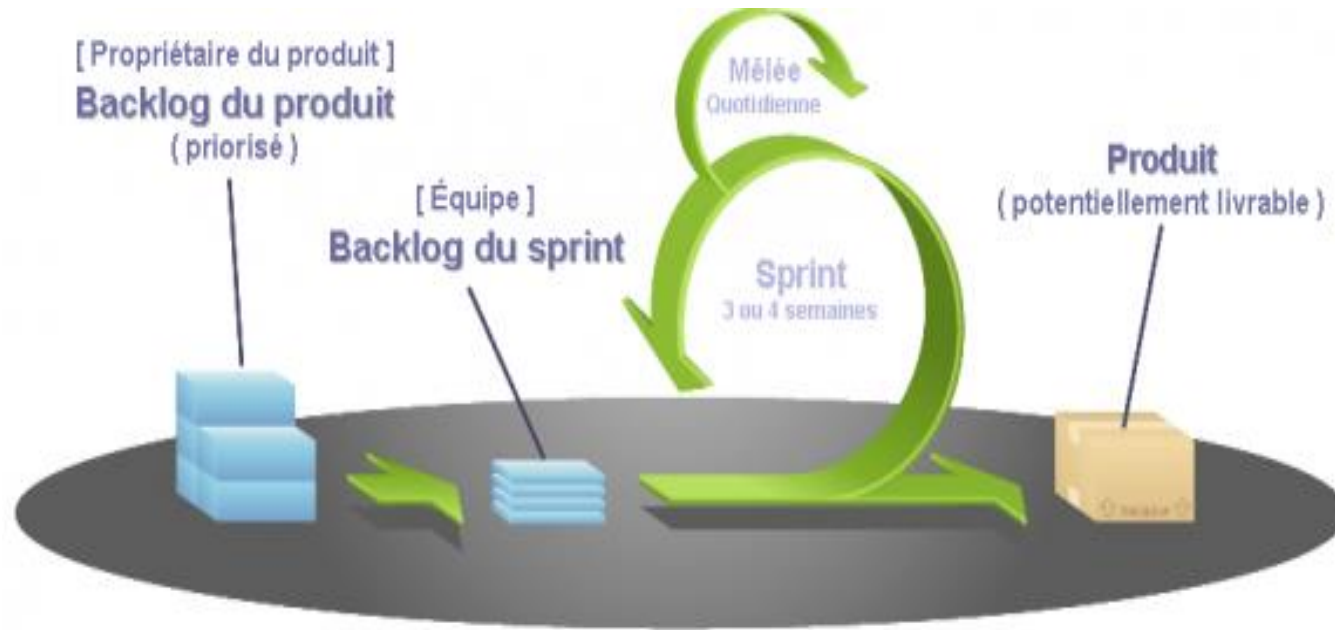
L'idée consiste donc, à se fixer un premier objectif à court terme (une grande ville par exemple) et se lancer sur la route sans tarder.

Une fois ce premier objectif atteint, on marque une courte pause et on adapte son itinéraire en fonction de la situation du moment.

**A coupler impérativement avec une approche globale car sinon on risque de se retrouver à Rome ....**

# SCRUM

21



COPYRIGHT © 2005. MOUNTAIN GOAT SOFTWARE

# Kanban

22



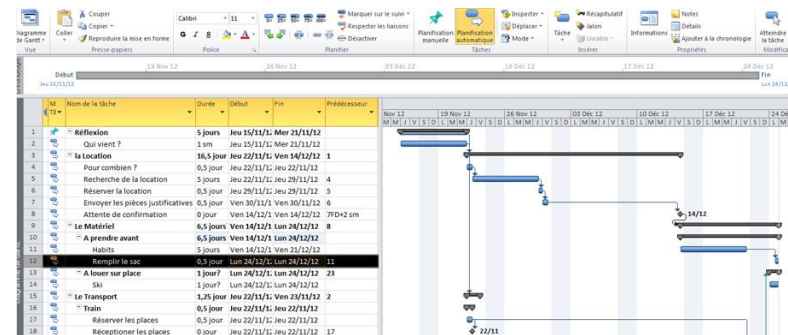
**Qualifiée de méthode agile au même titre que Scrum, Kanban prône la visualisation des flux de travail par le biais d'un tableau (dit tableau Kanban), permettant de prioriser et suivre l'état d'avancement des tâches à accomplir.**

# Planification

23

- **Identifier les différents acteurs du projet pour chaque tâche élémentaire** et les personnes affectées au pilotage des différents lots
- **Ordonnancer et évaluer les actions** : décrire les liens de dépendance chronologique entre les actions et estimer le temps nécessaire à leur réalisation (durée) ainsi que les dépenses à engager (charges).
- **Intégrer la notion de risque dans le planning prévisionnel** :
  - ✧ Les risques techniques qui ont un impact sur la durée et donc le coût de l'action (ex : le contenu des tâches n'est pas totalement figé car il dépend de décisions à prendre pendant la phase précédente ou des résultats de cette phase, la technicité du produit est complexe)
  - ✧ Les risques liés aux moyens ou à la planification : incertitude des données ou de disponibilités de moyens (personnel performant et adaptés). La simultanéité de certaines actions peut entraîner des conflits de ressources.

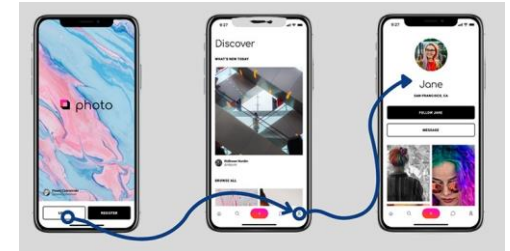
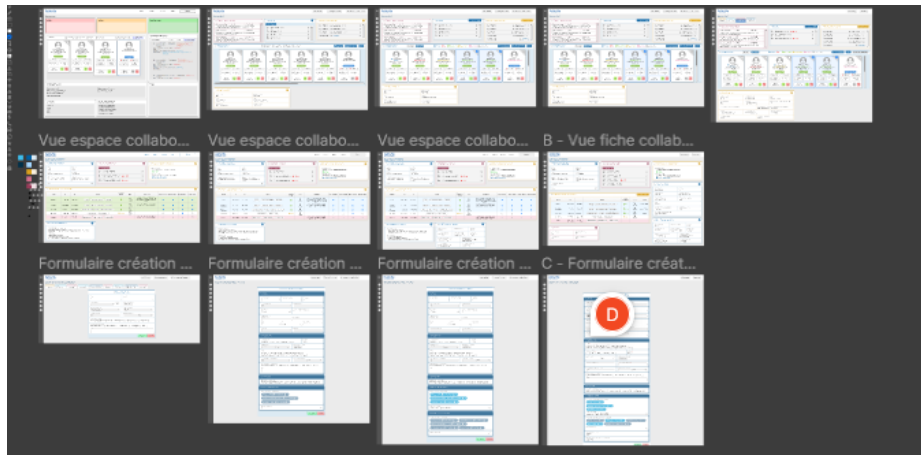
Lot	Tâche (N°)	Tps j/h	Prédécesseur



**Ne jamais oublier de suivre le chemin critique (PERT Réseau de Tâches)**

# Maquettage et Userflow

24



**Qualifiée de méthode agile au même titre que Scrum, Kanban prône la visualisation des flux de travail par le biais d'un tableau (dit tableau Kanban), permettant de prioriser et suivre l'état d'avancement des tâches à accomplir.**



# Rappels UML

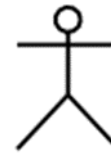
25

## Éléments des diagrammes d'utilisateurs

- L'acteur représente

- Une personne
- Un processus
- Une chose

qui interagit avec le système



Client

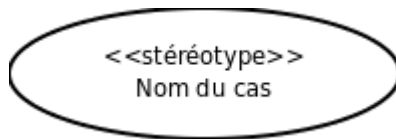
ou



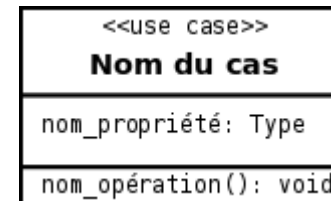
- Le cas d'utilisation représente

- une fonctionnalité visible de l'extérieur qui réalise un service de bout en bout, avec un déclenchement, un déroulement et une fin, pour l'acteur qui l'initie.

Un cas d'utilisation modélise donc un service rendu par le système, sans imposer le mode de réalisation de ce service.




Nom du cas : Verbe à l'infinitif



# Rappels UML

26

## Comment identifier les acteurs ?

- Chaque acteur doit être nommé. Ce nom doit refléter son rôle car un acteur représente un ensemble cohérent de rôles joués vis-à-vis du système.
- Pour trouver les acteurs d'un système, il faut identifier
  - quels sont les différents rôles que vont devoir jouer ses utilisateurs (ex : responsable clientèle, responsable d'agence, administrateur, approubateur, ...).
  - Il faut également s'intéresser aux autres systèmes avec lesquels le système va devoir communiquer comme :
    - ✖ les périphériques manipulés par le système (imprimantes, hardware d'un distributeur de billet, ...);
    - ✖ des logiciels déjà disponibles à intégrer dans le projet ;
    - ✖ des systèmes informatiques externes au système mais qui interagissent avec lui, etc.
- Pour faciliter la recherche des acteurs, on peut imaginer les frontières du système.  
 Tout ce qui est **à l'extérieur** et qui interagit avec le système est un **acteur**,  
Tout ce qui est **à l'intérieur** est une **fonctionnalité** à réaliser.
- Vérifiez que les acteurs communiquent bien *directement* avec le système par émission ou réception de messages. Une erreur fréquente consiste à répertorier en tant qu'acteur des entités externes qui n'interagissent pas directement avec le système, mais uniquement par le biais d'un des véritables acteurs.
  - Par exemple, l'hôtesse de caisse d'un magasin de grande distribution est un acteur pour la caisse enregistreuse, par contre, les clients du magasins ne correspondent pas à un acteur car ils n'interagissent pas directement avec la caisse.

# Rappels UML

27

## Comment recenser les cas d'utilisation

Chaque cas d'utilisation correspond à une fonction métier du système, selon le point de vue d'un de ses acteurs.

Il faut se placer du point de vue de chaque acteur et déterminer comment et surtout pourquoi il se sert du système.

Il faut éviter les redondances et limiter le nombre de cas en se situant à un bon niveau d'abstraction. Trouver le bon niveau de détail pour les cas d'utilisation est un problème difficile qui nécessite de l'expérience.

les cas d'utilisation avec un verbe à l'infinitif suivi d'un complément en vous plaçant du point de vue de l'acteur et non pas de celui du système.

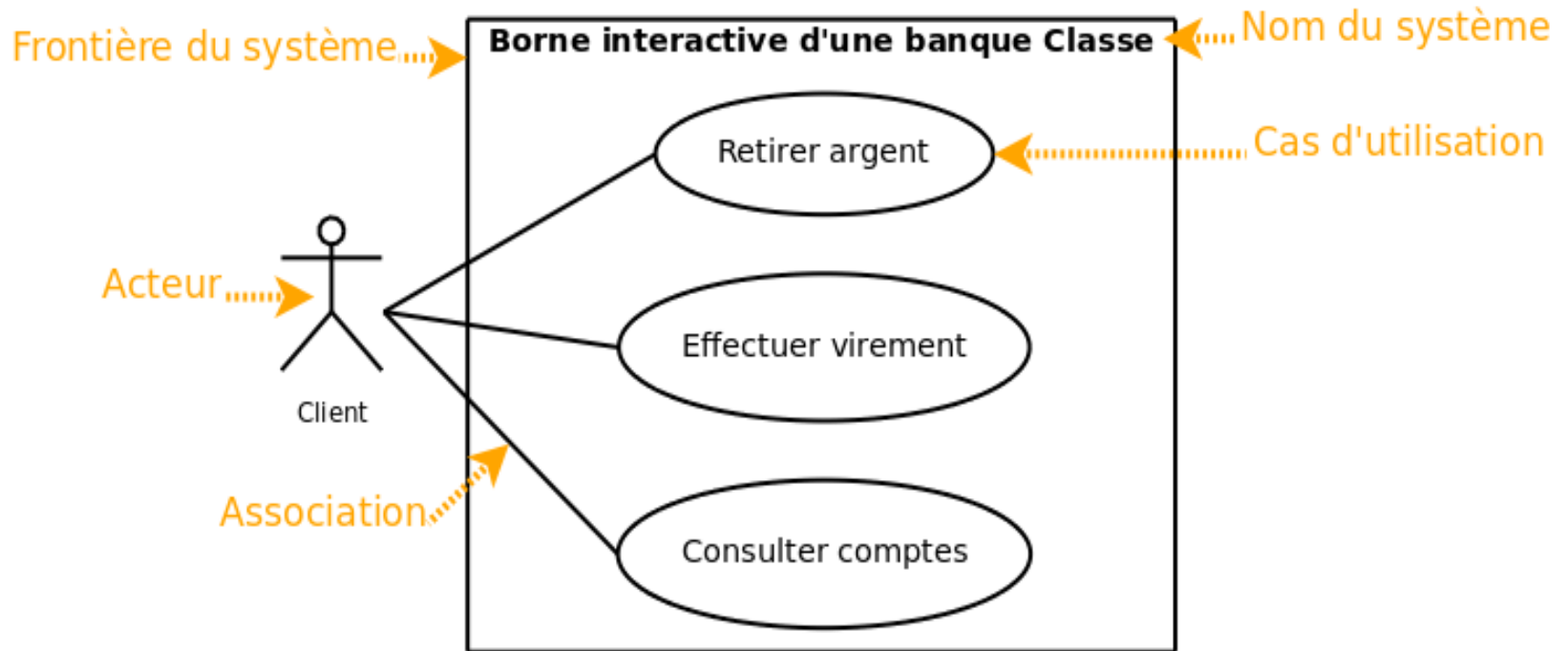
Par exemple, un distributeur de billets aura probablement un cas d'utilisation *Retirer de l'argent* et non pas *Distribuer de l'argent*.



Attention à ne pas retomber dans une décomposition fonctionnelle descendante hiérarchique. Un trop grand nombre de cas d'utilisations est en général source d'erreurs  
Dans tous les cas, il faut bien garder à l'esprit qu'il n'y a pas de notion temporelle dans un diagramme de cas d'utilisation.

# Rappels UML

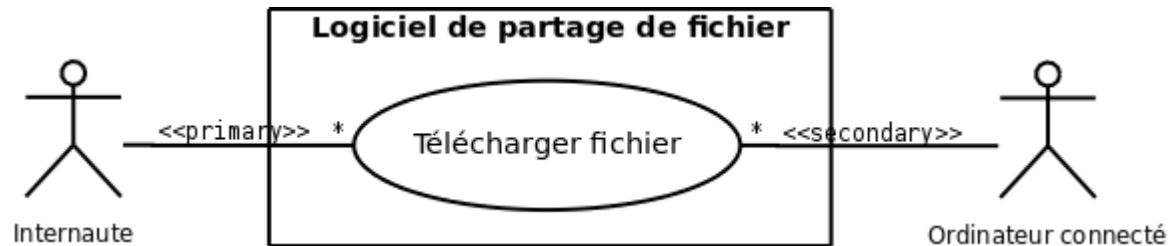
28



# Rappels UML

29

## Relation d'association



Une relation d'association est chemin de communication entre un acteur et un cas d'utilisation et est représenté **un trait continu**

### Multiplicité

- Lorsqu'un acteur peut interagir plusieurs fois avec un cas d'utilisation, il est possible d'ajouter une multiplicité sur l'association du côté du cas d'utilisation. Le symbole \* signifie *plusieurs*, exactement  $n$  s'écrit tout simplement  $n$ ,  $n..m$  signifie entre  $n$  et  $m$ , etc. Préciser une multiplicité sur une relation n'implique pas nécessairement que les cas sont utilisés en même temps.

### Acteurs principaux et secondaires

- **Principal** : Un acteur est qualifié de *principal* pour un cas d'utilisation lorsque ce cas rend service à cet acteur. L'acteur principal obtient un résultat observable du système. Un cas d'utilisation a au plus un acteur principal. En général, l'acteur principal initie le cas d'utilisation par ses sollicitations
- **Secondaire** : Les autres acteurs sont alors qualifiés de *secondaires*. Un acteur secondaire est sollicité pour des informations complémentaires.
- Le stéréotype << *primary* >> vient orner l'association reliant un cas d'utilisation à son acteur principal, le stéréotype << *secondary* >> est utilisé pour les acteurs secondaires

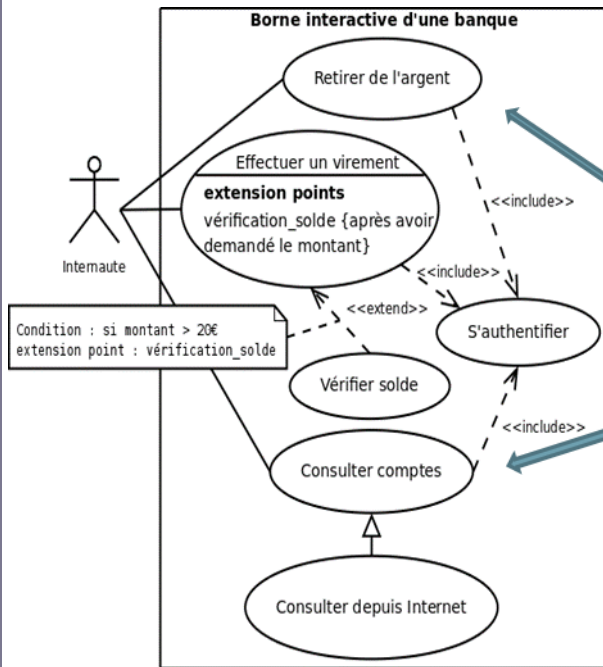
Quand un cas n'est pas directement relié à un acteur, il est qualifié de *cas d'utilisation interne*.

# Rappels UML

30

## Relation d'inclusion

- Un cas A inclut un cas B si le comportement décrit par le cas A inclut le comportement du cas B : le cas A dépend de B. Lorsque A est sollicité, B l'est obligatoirement
- Les inclusions permettent essentiellement de factoriser une partie de la description d'un cas d'utilisation qui serait commune à d'autres cas d'utilisation (1)



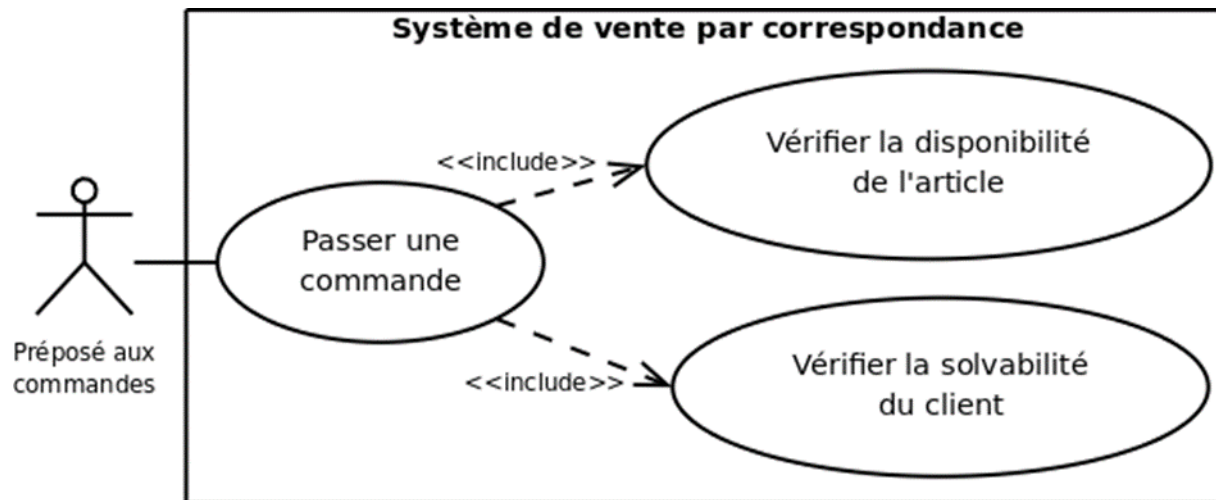
(1) Par exemple, l'accès aux informations d'un compte bancaire inclut nécessairement une phase d'authentification avec un identifiant et un mot de passe que ce soit pour consulter les comptes ou retirer de l'argent

# Rappels UML

31

## Relation d'inclusion

- Les inclusions permettent également de décomposer un cas complexe en sous-cas plus simples. Cependant, il ne faut surtout pas abuser de ce type de décomposition : il faut éviter de réaliser du découpage fonctionnel d'un cas d'utilisation en plusieurs *sous-cas d'utilisation* pour ne pas retomber dans le travers de la décomposition fonctionnelle.
- Attention également au fait que, les cas d'utilisation ne s'enchaînent pas, puisqu'il n'y a aucune représentation temporelle dans un diagramme de cas d'utilisation



# Rappels UML

32

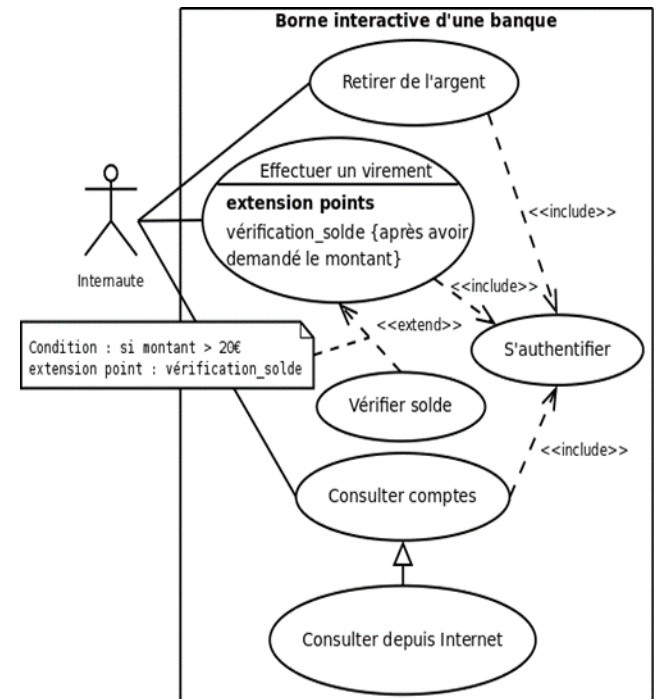
## Relation d'extension

- A étend un cas d'utilisation B lorsque le cas d'utilisation A **peut** être appelé au cours de l'exécution du cas d'utilisation B. Exécuter B peut **éventuellement** entraîner l'exécution de A : contrairement à l'inclusion, **l'extension est optionnelle**. L'extension peut intervenir à un point précis du cas étendu. Ce point s'appelle le point d'extension. Il porte un nom, qui figure dans un compartiment du cas étendu sous la rubrique *point d'extension*, et est éventuellement associé à une contrainte indiquant le moment où l'extension intervient. **Une extension est souvent soumise à condition.**

Graphiquement, la condition est exprimée sous la forme d'une note.

Exemple d'une banque où la vérification du solde du compte n'intervient que si la demande de retrait dépasse 20 euros.

La relation d'extension est probablement la plus utile car elle a une sémantique qui a un sens du point de vue métier au contraire des deux autres qui sont plus des artifices d'informaticiens.





# Rappels UML

33

## Relation de généralisation

- La seule relation possible entre deux acteurs est la généralisation
- un acteur A est une généralisation d'un acteur B si l'acteur A peut être substitué par l'acteur B. Dans ce cas, tous les cas d'utilisation accessibles à A le sont aussi à B, mais l'inverse n'est pas vrai.

Le préposé aux commandes peut être  
Remplacé par le directeur des ventes mais  
Pas l'inverse.

De plus le directeur des ventes  
Peut gérer le stock ce que ne peut pas  
Faire le préposé aux commandes

