

# Projet AGL - UML

Sujet proposé par  
Pierre Gérard

pierre.gerard@iutv.univ-paris13.fr

*DUT Informatique S2  
IUT de Villetaneuse  
Université de Paris 13*



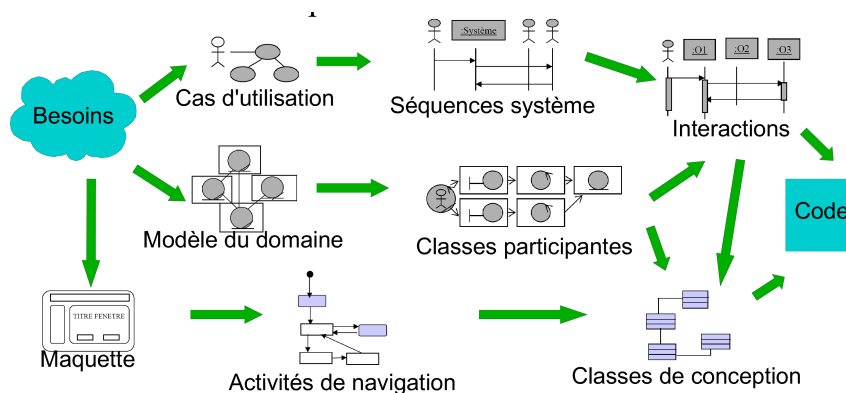
## Résumé

Ce projet a pour objectif l'apprentissage de l'utilisation d'un « Atelier Génie Logiciel » pour mener de bout en bout la conception d'un logiciel avec une démarche cohérente.

## 1 Organisation du projet

### 1.1 Méthode de conception

Pour la conception, vous devrez utiliser la méthode « minimale » décrite en cours et résumée par le diagramme suivant :



Les besoins précis sont ceux que vous jugerez opportuns. Ils devront être exprimés d'abord par un cahier des charges sommaire que vous écrirez, puis par un diagramme de cas d'utilisation. Vous ne ferez pas de maquette et ne finaliserez pas le projet en écrivant le code mais toutes les autres étapes devront être prises en compte. Une manière de procéder pourrait donc être :

1. Spécification des besoins à l'aide d'un diagramme de cas d'utilisation
2. Définition d'un modèle du domaine à l'aide d'un diagramme de classes
3. Spécification de l'interface graphique à l'aide d'un diagramme d'activités de navigation entre les différentes options présentées.
4. Pour chaque cas d'utilisation, on produira un diagramme de séquence système illustrant les interactions entre le logiciel et les acteurs. Ceci nous aidera à définir les opérations système.
5. La première version du diagramme de classes sera reprise pour y adjoindre des classes de contrôle et d'interface. Les opérations système seront réparties entre les différentes classes d'interface.
6. Chaque diagramme de séquence système sera repris pour être détaillé en prenant en compte les informations apportées par le diagramme de classes participantes. En particulier, on éclatera à chaque fois le système en plusieurs classes participantes de manière à montrer comment les différentes classes interagissent pour réaliser chacun des cas d'utilisation.
7. Les messages échangés entre les classes participantes permettront de définir les opérations de chacune des classes participantes. Ces opérations seront spécifiées dans la nouvelle version du diagramme de classes : le diagramme de classes de conception. On veillera à ce que toutes les opérations soient réalisables, et que tout ce qui a été défini dans les activités de navigation soit présent.

Cette méthode est minimale. Si vous trouvez pertinent de spécifier certains aspects du logiciel en utilisant d'autres diagrammes, vous êtes naturellement libres de le faire. On pourrait – par exemple – utiliser des diagrammes d'activité pour décrire certains algorithmes parmi les plus complexes, ou encore des diagrammes d'états pour modéliser la dynamique de certains objet.

Si vous êtes loin de vous demander quoi faire en plus et craignez plutôt ne pas pouvoir faire la totalité des diagrammes demandés, il est alors préférable de se concentrer sur la conception des fonctionnalités les plus centrales et pour celles-ci, d'aller jusqu'au diagramme de classes de conception, quitte à négliger certaines fonctionnalités secondaires. Ce sera préférable à une exploitation de la totalité des cas d'utilisation mais en restant toujours au plus haut niveau d'abstraction. En d'autres termes, un travail en profondeur d'abord sera préféré à un travail en largeur d'abord.

## 1.2 Rendu

Le projet doit être réalisé individuellement. Vous devrez rendre tous les diagrammes que vous aurez développés. Le rendu se fera par mail :

- A : pierre.gerard@univ-paris13.fr
- Sujet : [Projet AGL] VotreNom
- Pièces jointes : VotreNomV?.zip, VotreNom.txt, VotreNom.pdf

Nous vous prions de bien vouloir respecter scrupuleusement la syntaxe des différents éléments du mail. **VotreNom** est bien sûr à remplacer par votre vrai nom. Si par exemple votre nom de famille était Dupont, l'objet du mail serait « [Projet AGL] Dupont ».

Une partie de l'évaluation concernera l'utilisation de l'AGL PowerAMC. Aussi nous vous demandons de bien vouloir noter dans un fichier annexe (**VotreNom.txt**) les manipulations que vous aurez effectuées mais qui ne seraient pas immédiatement visibles.

Nous vous demandons également de bien vouloir commencer le projet par la rédaction d'un cahier des charges sommaire que vous nommerez **VotreNom.pdf**. Si vous avez des difficultés à produire un fichier pdf, vous pouvez aussi rendre le fichier source (**VotreNom.doc**, **VotreNom.odp**...).

Dans la méthode proposée, certains diagrammes sont affinés au fil de la conception. Par exemple,

- Le premier diagramme des classes est le « modèle du domaine » qui est ensuite enrichi de classes d'interfaces et de contrôle pour devenir « diagramme des classes participantes » puis enfin « diagramme des classes de conception »

- A chaque cas d'utilisation est associé un « diagramme de séquence système » généralement assez simple mais à qui on associe ensuite un diagramme de séquences plus complexe qui rend compte des « interactions » à l'intérieur du système. Souvent, la production de ces nouveaux diagrammes amène à reconsidérer certains choix opérés en amont dans les « diagramme de séquence système », et même parfois dans les « diagrammes de cas d'utilisation »

Il est indispensable que ce que vous rendrez permette au correcteur de mesurer l'évolution de la conception<sup>1</sup>. Nous vous demandons donc de rendre trois versions de la conception que vous aurez produite :

- V1** avec le diagramme de cas, le modèle du domaine, le diagramme de séquences système et les activités de navigation
- V2** avec les classes participantes, les diagrammes d'interactions et les diagrammes précédents éventuellement modifiés à la lumière de vos dernières réflexions
- V3** avec les classes de conception et les diagrammes précédents éventuellement modifiés à la lumière de vos toutes dernières réflexions

Chacune de ces versions pourra être sauvegardée dans un dossier différent `VotreNomV?` puis archivée dans un fichier `VotreNomV?.zip` ou `VotreNomV?.tgz`. Ici, « ? » vaut pour les trois numéros de version. Par exemple, un étudiant nommé Dupont joindra à son mail un fichier `DupontV1.zip`, un autre `DupontV2.zip` et encore un autre `DupontV3.zip`.

## 2 Présentation du projet

### 2.1 Objectif général

Il vous est demandé de développer un jeu de type « MMORPG » au tour par tour. Parmi les exemples célèbres de jeux de ce type, on peut citer « Mountyhall » (<http://www.mountyhall.com>) ou encore « Les Souterrains de Delain » (<http://www.jdr-delain.net>). MMORPG signifie « Massively Multi-Player Online Role Playing Game » :

- « Massively Multi-Player » parce que le jeu doit être multi-joueurs et doit permettre à un grand nombre de personnes de jouer simultanément
- « Role Playing Game » parce que chaque joueur incarne un personnage livrant bataille dans un monde peuplé de monstres et d'autres personnages.
- « Online » parce que le jeu doit se dérouler en ligne au moyen d'une connexion IP. L'état du monde est en permanence stocké sur un serveur et les clients (joueurs) doivent s'y connecter pour pouvoir jouer.

Une interface très basique en mode texte pourrait ressembler à cela :

---

<sup>1</sup>A ce titre, il est tout à fait normal que le premier diagramme de cas - par exemple - soit imparfait et qu'il soit corrigé par la suite

```

#####
#           C           #
#           L           #
# 1 #####L           #
#           #L         #
# a         #         #
#           C         #
#           #         #
# L         #         #
# #####L           #
# C         p         L #
#           #         #
# L         #         #
# p         C#         2 #
#           #         #
# C         #         L #
#           a #####   #
#           #         C #
# L         #         p #
#           #         #
#####

```

JOUEUR 1 : Vous n'avez subi aucune attaque  
 Vos objets équipés :  
 - épée (dégâts +3 degrés)  
 - vêtements de cuir (défense +1 degré)  
 Votre sac :  
 - potion d'explosion  
 - potion de soins  
 Vos caractéristiques :  
 - force : 3D  
 - adresse : 3D  
 - endurance : 3D  
 + initiative : 2D+2  
 + attaque : 3D  
 + défense : 3D+1  
 + dégâts : 4D  
 Votre niveau de blessures : en forme  
 Vos points d'action : 5  
 Vous pouvez :  
 1 - vous déplacer (2PA)  
 2 - attaquer (3PA)  
 3 - utiliser un objet (Variable)  
 4 - ramasser/déposer un objet (2PA)  
 5 - finir et garder les PA restants  
 Votre choix : \_

Dans cette représentation du monde, les # représentent des murs, les 1 et 2 représentent les joueurs; p et a valent pour une potion et une arme à terre et C/L sont des monstres comme des chatons tueurs ou des lapins garous<sup>2</sup>. Deux personnages ne peuvent pas occuper la même case.

## 2.2 Règles du jeu

Les règles du jeu suivantes sont données à titre indicatif pour vous permettre de mieux démarrer. Elles sont incomplètes et vous pouvez les adapter à vos besoins et vos envies.

### 2.2.1 Caractéristiques des PJ

On distingue deux types de personnages : les PJ (personnages joueurs) et les PNJ (personnages non joueurs, souvent des monstres). Les PJ sont définis par un certain nombre de caractéristiques :

**Force** pour la puissance physique

**Adresse** qui représente la capacité du personnage à mouvoir son corps comme il l'entend

**Résistance** qui représente le point jusqu'auquel le corps du personnages peut résister aux agressions extérieures

<sup>2</sup>Vous aurez toute liberté pour choisir votre univers et, aux lapins garous et chatons tueurs, vous pourrez préférer des trolls et des orques ou encore des Nexus et des Rancors.

Chacune de ces caractéristiques se voit attribuer un niveau mesuré en degrés. Ces niveaux sont comparés pour opposer certains caractéristiques à d'autres. Par exemple, pour déterminer si un PJ peut en toucher un autre lorsqu'il attaque, on comparera leur niveau d'adresse.

Les création de PJ par les joueurs doit être personnalisée. Chaque personnage se voit attribuer 18 degrés à répartir entre les trois caractéristiques.

### 2.2.2 Tirages aléatoires

Pour mettre un peu d'incertitude dans la résolution des actions, on introduit un facteur aléatoire en transformant les degrés dans les caractéristiques niveaux exprimés en dés (D). Chaque dé donne lieu à un tirage aléatoire entre 1 et 6. Par exemple, un tirage de  $2D+2$  est en fait la somme de deux nombres aléatoires indépendants et tirés chacun entre 1 et 6, auquel on ajoute 2. On a 1 dé pour 3 degrés et les degrés restants sont des unités additionnelles. Par exemple, une caractéristique avec 7 degrés donne lieu à un tirage de  $2D+1$  et une caractéristique avec 14 degrés donne lieu à un tirage de  $4D+2$ .

Les objets portés par les PJ leur permettent d'accroître certaines de leurs capacités. Par exemple, une arme permettra de faire plus de dégâts et une armure permettra de mieux encaisser les coups. Ces objets donnent donc lieu à certains bonus lorsqu'ils sont portés.

### 2.2.3 Capacités

Les caractéristiques et les bonus apportés par les objets permettent de calculer des niveaux de capacités comme :

**L'initiative** (adresse - bonus vêtements) qui représente la capacité des personnages à attaquer avant les autres

**L'attaque** (adresse) qui représente l'habileté du personnage à en toucher un autre quand il l'attaque

**L'esquive** (adresse) qui représente l'habileté du personnage à éviter les coups dont il est la cible

**La défense** (résistance + bonus vêtements) qui représente la capacité du personne à supporter les dommages physiques

**Les dégâts** (force + bonus armes) qui représentent la puissance des coups portés par le personnage.

Les PJJ ont aussi ces capacités. Pour les monstres, elles sont définies directement et ne sont pas calculées (les monstres n'ont pas de caractéristiques et ne portent pas d'objets) mais pour d'éventuels autres types de PNJ plus « évolués » on pourra considérer des caractéristiques et des bonus liés aux objets.

### 2.2.4 Points d'action

Pour entreprendre une action, un joueur doit dépenser des points d'action (PA). Ces PA permettent aux joueurs d'agir : tant qu'il en a, le joueur peut dépenser des points d'action à tout moment en choisissant une action parmi :

**Attaquer** (coût 3PA)

**Se déplacer** d'une case (coût 2PA) : le monde est en fait un ensemble de cases côte à côte sur lesquelles se déplacent PJ et PNJs.

**Utiliser un objet** (coût variable selon l'objet)

Dès qu'une action est entreprise, son coût est déduit du nombre de PA restants au joueur. Une action ne peut pas être entreprise si le joueur ne dispose pas de suffisamment de PA.

Les PA se regagnent par période de 12 heures, selon un barème qui dépend de l'initiative du PJ : un personnage avec beaucoup d'initiative regagne plus de PA par tranche de 12 heures qu'un autre plus pataud. A la fin d'une période, la moitié des PA (arrondie à l'inférieur) restants sont conservés pour la période suivante.

L'heure de la fin d'une période de 12 heures est appelée DLA (date limite d'action). La première DLA correspond à la date de création d'un joueur à laquelle on ajoute 12 heures. La seconde intervient encore 12 heures plus tard. A tout moment, il est possible à un joueur de reculer sa DLA de manière à synchroniser autrement ses cycles de 12 heures.

### 2.2.5 Utilisation d'objets

Le résultat de l'utilisation d'un objet dépend de l'objet utilisé. Dans le cas d'une arme ou d'un vêtement, cette action prend la forme d'un équipement / déséquipement de l'objet. Dans le cas d'un autre objet comme une potion, par exemple, il s'agira d'appliquer les effets de l'objet et de le supprimer. Une potion de soins permettra par exemple au PJ de regagner un nombre défini de niveaux de blessures. Une potion molotov permettra d'infliger des dégâts à tous les personnages sur les cases alentours.

### 2.2.6 Combats

En cas d'attaque, il faut désigner un personnage cible qui doit se situer sur une case adjacente. La cible tire un score d'esquive et l'attaquant un score d'attaque. Si le score d'attaque est supérieur au score d'esquive, la cible est touchée et on procède à la détermination des blessures.

Pour ce faire, on procède de manière similaire : le personnage touché tire un score de défense et l'attaquant tire un score de dégâts en utilisant les capacités adéquates. Si le score de défense est supérieur au score d'attaque, le personnage s'en tire indemne. Dans le cas contraire, il est blessé et son niveau de blessures dépend de la différence entre les deux scores.

Chaque personnage dispose de 6 niveaux de blessure :

- 1 Blessures superficielles
- 2 Légèrement blessé
- 3 Blessé
- 4 Gravement blessé
- 5 Inconscient (dans ce cas, le personnage ne peut plus jouer : il doit attendre de recevoir une potion de soins ou de récupérer naturellement)
- 6 Mort (dans ce cas, le personnage est retiré de la partie, toutes ses possessions tombent au sol et le joueur ne joue plus)

Dès qu'un personnage est blessé, un niveau de blessure est perdu par tranche de 3 points de différence entre le score de dégâts et le score de défense. Les potions de soin permettent de regagner des niveaux de blessure selon la puissance de la blessure. Il faut attendre 2 cycles pour qu'un niveau de blessure soit regagné naturellement.

### 2.2.7 PNJ

Le comportement et les capacités des PNJ restent à définir. Si un PNJ meurt et qu'il possède des objets, ceux-ci restent au sol sur la case de sa mort. Dans le cas d'un monstre, un nombre aléatoire d'objets tirés au hasard apparaissent au sol.

### 2.2.8 Expérience

Chaque fois qu'un PJ réussit une attaque, il gagne 1 point d'expérience (PX). Il en gagne plus dès qu'il parvient à tuer un adversaire. Son total de PX est comparé à celui de son ennemi et un ratio est calculé entre les deux (les monstres et PNJs ont un nb de PX fixe et déterminé par les concepteurs du jeu). Si un PJ A tue un personnage B, que A a 50 PX et que B en a 100, le ratio est alors de 2. De ce ratio dépend le nombre de PX gagnés par l'attaquant victorieux.

Ces PX peuvent être à tout moment dépensés par les joueurs pour améliorer les caractéristiques de leur personnage. Une caractéristique avec un haut niveau sera bien plus difficile à faire progresser encore qu'une caractéristique avec un niveau assez faible.

La manière de calculer les PX attribués et le coût des améliorations sont laissées à votre convenance.

### 2.3 Fonctionnalités attendues

L'application développée devra permettre au moins ...

- Aux administrateurs de créer un monde ;
- Aux administrateurs de créer de nouveaux types de monstres ;
- Aux administrateurs d'administrer les comptes des abonnés ;
- Aux joueurs de s'inscrire ;
- Aux joueurs d'observer ce qui se trouve dans un rayon de quelques cases autour de leur personnage ;
- Aux joueurs de consulter l'état de leur personnage et ce qui lui est arrivé récemment (attaques à son encontre, potions...);
- Aux joueurs de choisir leurs actions ;
- Aux joueurs de gérer leur inventaire (sac à dos, mains droite et gauche, éléments d'armure...);
- Aux joueurs de s'envoyer des messages ;
- Automatiquement aux PNJs d'agir selon leurs caractéristiques (Intelligence Artificielle sommaire) ;
- Automatiquement aux monstres d'apparaître.

Cette liste n'est pas exhaustive et on pourra en outre imaginer toute fonctionnalité secondaire que l'on estimera souhaitable.