Rapport de conception du projet de programmation orientée objet

Licence d'informatique – 2ème année
Faculté des sciences et techniques de Nantes

Dice Dungeon (DD)

présenté par :
- MERLET Raphaël : raphael.merlet@etu.univ-nantes.fr (380A)
- CADIOT Dorian : cadiot.dorian@etu.univ-nantes.fr (385)
- LE CAP Nathanaël : lecap.nathanael@etu.univ-nantes.fr (385)
- M'BAYE Samba : m-baye.samba@etu.univ-nantes.fr (385)
- MORINEAU Ludovic : morineau.ludovic@etu.univ-nantes.fr (385)

le 14 11 2023

encadré par
Christophe LINO

Nantes Université

1 Cahier des charges

Le but du projet est de créer un jeu vidéo fonctionnel, jouable dans la console dont le genre se situe entre le jeu de rôle sur table et le roguelike. Il devra si possible avoir une bonne prise en main et donc être agréable.

Les différentes interactions possibles avec le jeu devront être de pouvoir lancer une partie, via un menu. Ce dernier devra également permettre au joueur de changer la difficulté de sa ou ses prochaines parties. Dans celles-ci, il devra être capable d'explorer un donjon composé de salles contenant différents évenements parmis lequels se trouvent des combats, ainsi qu'un magasin ou marchand.

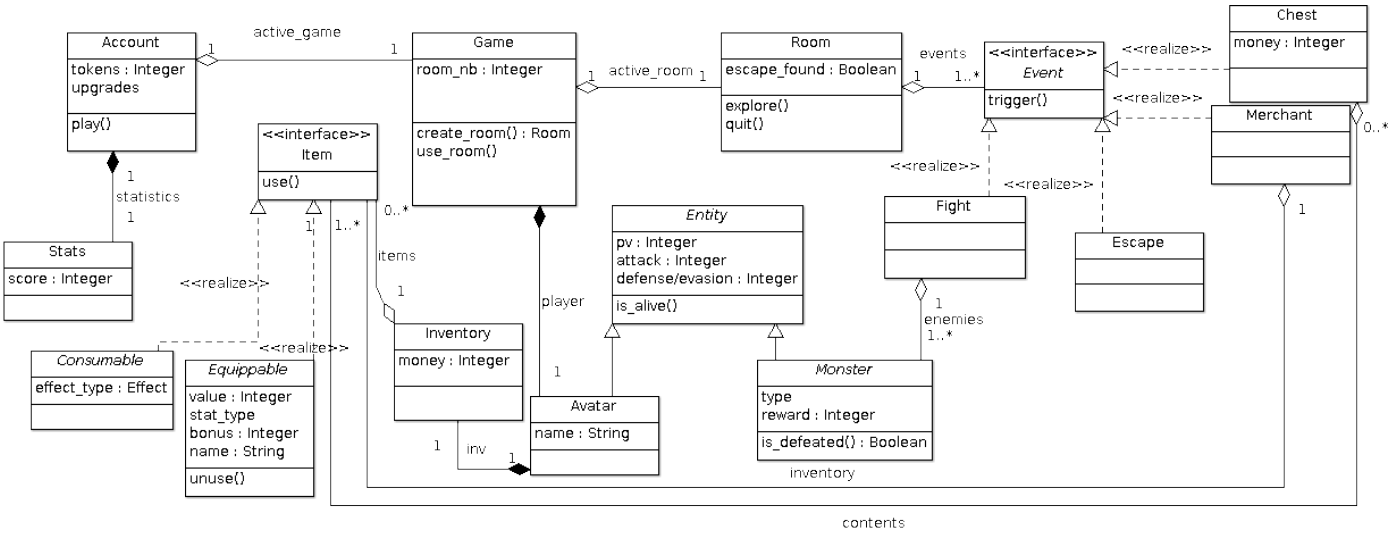
Afin de progresser dans le jeu, le joueur aura l'opportunité de faire plusieurs parties pour obtenir des bonus sur le long terme. Ces derniers combinés aux objets équipables ou non qu'il récupèrera lui permettront d'aller de plus en plus loin.

2 Architecture

2.1 Description générale

Un Account possède des Stats (qui dépendent des parties de l'utilisateur). C'est l'utilisateur qui lance une Game. Chaque Game comporte plusieurs Rooms. Une Room contient des Event et se voit attribuée une difficulté ; Un Event peut être un Fight (un combat) , une Finding (un marchand) ou une Escape (permettant de sortir du jon). Une Entity est soit un Monster soit un Character. Le Character se déplace de Room en Room. Une Entity possède des hp, une attackDamage et une Defense. Parmi les Entity, le type Character possède un Name tandis que le type Monster a un Type. L'utilisateur peut avoir avec lui des objets de type Equippable. Les Equippable sont soit des Consumable, soit des Equippable. La différence étant que Un Consumable possède des stats instantanées et sont à utilisation unique, tandis que Un Equippable possède des stats à long terme

2.2 Diagramme de classes



2.3 Interfaces

Pour notre projet, nous avons décidé d'interagir avec l'utilisateur via la console. L'interface console devra donc permettre la progression du joueur dans le donjon. On aura donc accès à une représentation graphique de la salle parcourue : cela incluera les combats(monstres et attributs associés), les marchands(produits vendus, possibilité de rachats), les sorties de salles et les coffres et leur contenu.

En bas de la console, il y aura les attributs du joueur et les actions possibles sous forme de liste à choix comme par exemple l'accès à l'inventaire ou les actions de combats.

2.4 Aspects spécifiques

Parmis les fonctions particulières à programmer, il y en aura une permettant de faire une sauvegarde d'un compte joueur en enregistrant l'avancée dans un fichier texte que l'on pourra importer dans le jeu plus tard pour reprendre notre progression. Il y aura aussi la gestion de l'interface qui sera un peu plus délicate à coder que le reste.

3 Regard critique

La réalisation du projet requiert l'application de nombreux concepts vu en POO, notamment les notions d'interfaces, d'héritage et de classes abstraites. Les Items et les Evenements par exemple seront décomposés en plusieurs types (ex : items consommables et équipables), implémentés à partir d'interfaces. Cela nous permettra de pouvoir étendre simplement leur nombre sans changements majeurs de code ainsi que de les manipuler en grande quantité plus facilement avec des listes.

Chaque membre du groupe devra programmer entre 1 et 4 classes selon leurs quantité de travail requise, celle-ci étant définie par leurs attributs et méthode respectives(cette répartition du travail a été abordée dans le rendu 1). L'UML permettra à chacun de d'implémenter ses classes sans avoir besoin d'accéder à celle des autres ce qui devrait permettre une progresion fluide dans le développement.

Le travail de ce projet étant équilibré et intéressant, nous sommes donc motivés à mener ce projet à fruition. Chaque membre du groupe étant novice en java, le projet devrait nous permettre d'améliorer nos compétences individuelles tout en coopérant grâce à l'utilisation de gitlab.

Si la quantité de travail envisagée est bien moindre que celle qu'il nous faudra réellement pour terminer le projet, alors on pourra potentiellement ajouter un système de drop d'items par les monstres, ou encore la possibilité de choisir un autre système de combat ou une difficulté alternative. Aussi l'implémentation d'un système de sauvegarde est envisageable.