Rapport de conception du projet de programmation orientée objet

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

Study Week Simulator

présenté par

Aida Amerkhanova
Maty Diop
Aymen El Ouagouti
Enzo Mantzoutsos

le 11/11/2022

encadré par
Anastasia VOLKOVA et Laurent GRANVILLIERS

1. Cahier des charges

Notre projet consiste à créer un jeu de simulation d'une vie étudiante. L'utilisateur sera l'étudiant. L'étudiant aura ses propres statistiques (faim, attention, fatigue) qui évolueront tout au long du jeu. Puis, il aura des statistiques liées à chaque matière et chaque professeur. En effet, tout au long de la simulation il devra faire des choix sur les éventements auxquels il va (ou non) assister. Les éventements possibles sont : Pause/Révision/Cours.

Un bilan de sa semaine sera alors donné à la fin du jeu. L'objectif de l'utilisateur est de la réussir, c'est à dire d'avoir les statistiques les plus élevés.

Scenario nominal :

Au début du jeu, l'utilisateur saisit son nom et prénom, on lui donne son emploi du temps de 5 jours et il sera confronté à son premier choix.

Par exemple, son premier cours Y avec un professeur X. S'il veut y aller il aura à cocher la réponse « oui », ainsi il assistera à ce cours. Après ce cours, il aura un petit quiz afin de vérifier s'il a bien été attentif. Ses statistiques dans cette matière augmenteront et sa relation avec le professeur concerné aussi. Quant aux statistiques sur son état personnel, elles varient :
sa faim et fatigue augmente, son attention baisse.

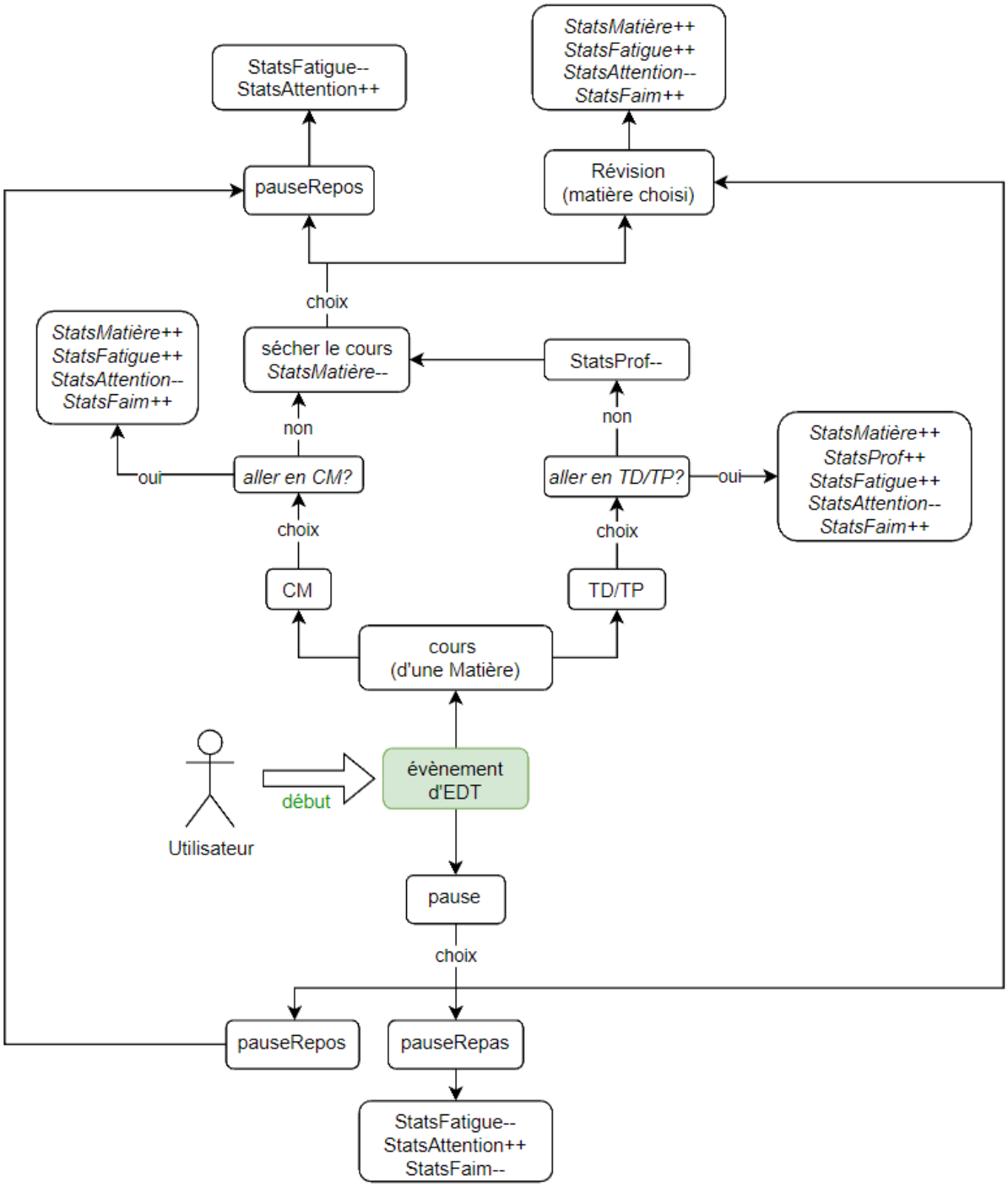
Dans un second cas, si l'utilisateur choisit de ne pas y aller :
Ses statistiques dans la matière concerné baissent. Quant à la relation avec le professeur, si c'est un TD ou TP elle baissera aussi.
Le choix de ne pas aller à un cours se suit d'un autre choix à effectuer. Que veut-il faire alors ?
Il pourra choisir de faire une pause de repos ou repas, ou bien réviser une autre matière.
Dans le cas d'une pause repas, il aura les changements suivants dans ses statistiques personnelles : attention augmente, fatigue et faim baisse.
S'il choisit de faire une pause repos : attention augmente, fatigue baisse. La faim restera inchangée, en supposant qu'il pourra manger un peu sans que ça devient une pause repas où sa faim diminuera de façon importante.
Enfin pour le choix d'une révision, ses statistiques dans la matière choisi augmenteront. L'attention va baisser puis la fatigue et la faim vont augmenter.

Pour chaque nouvelle journée les statistiques personnels vont se réinitialiser en supposant un cas idéal : faim et fatigue à 0, attention à 100.

Au début du jeu la relation avec chaque professeur et la maîtrise de chaque matière sera de 50 (/100), soi-disant « neutre ».

A la fin de la simulation, un bilan de sa semaine sera donné.

Voici un arbre de décision qui illustre le fonctionnement général du jeu :


StatsFatigue--
StatsAttention++
StatsMatière++
StatsFatigue++
StatsAttention--
StatsFaim++
pauseRepos
Révision
(matière choisi)
choix
StatsMatière++
StatsFatigue++
StatsAttention--
StatsFaim++
sécher le cours
StatsMatière--
StatsProf--
non
non
oui
aller en CM?
aller en TD/TP?
oui
StatsMatière++
StatsProf++
StatsFatigue++
StatsAttention--
StatsFaim++
choix
choix
CM
TD/TP
cours
(d'une Matière)
Utilisateur
début
évènement
d'EDT
pause
choix
pauseRepos
pauseRepas
StatsFatigue--
StatsAttention++
StatsFaim--

2. Architecture

A) Description générale

Décrire avec des phrases les différentes classes et les relations entre ces classes
Nous aurons une classe ***Personne*** qui associera les étudiants et les professeurs à **Game**. Cette classe aura en attribut nom et prénom et en méthode les getter et setter associés.

La classe **Étudiant** aura en attribut une liste de statistiques personnelles, une liste de statistiques sur les matières que l'étudiant aura dans la semaine et un EDT propre à lui.
En effet étudiant aura un EDT qui diffère de celui imposé par le jeu en fonction de choix d'évènements qu'il va effectuer.
Pour méthode cette classe aura tous les getter et setter associés.

La classe **Professeur** aura en attribut un entier affection, c'est à dire l'état de la relation entre le professeur et l'étudiant, cette classe constituera en partie la classe Matière (agrégation). On aura bien sur un setter et un getter lui associés.

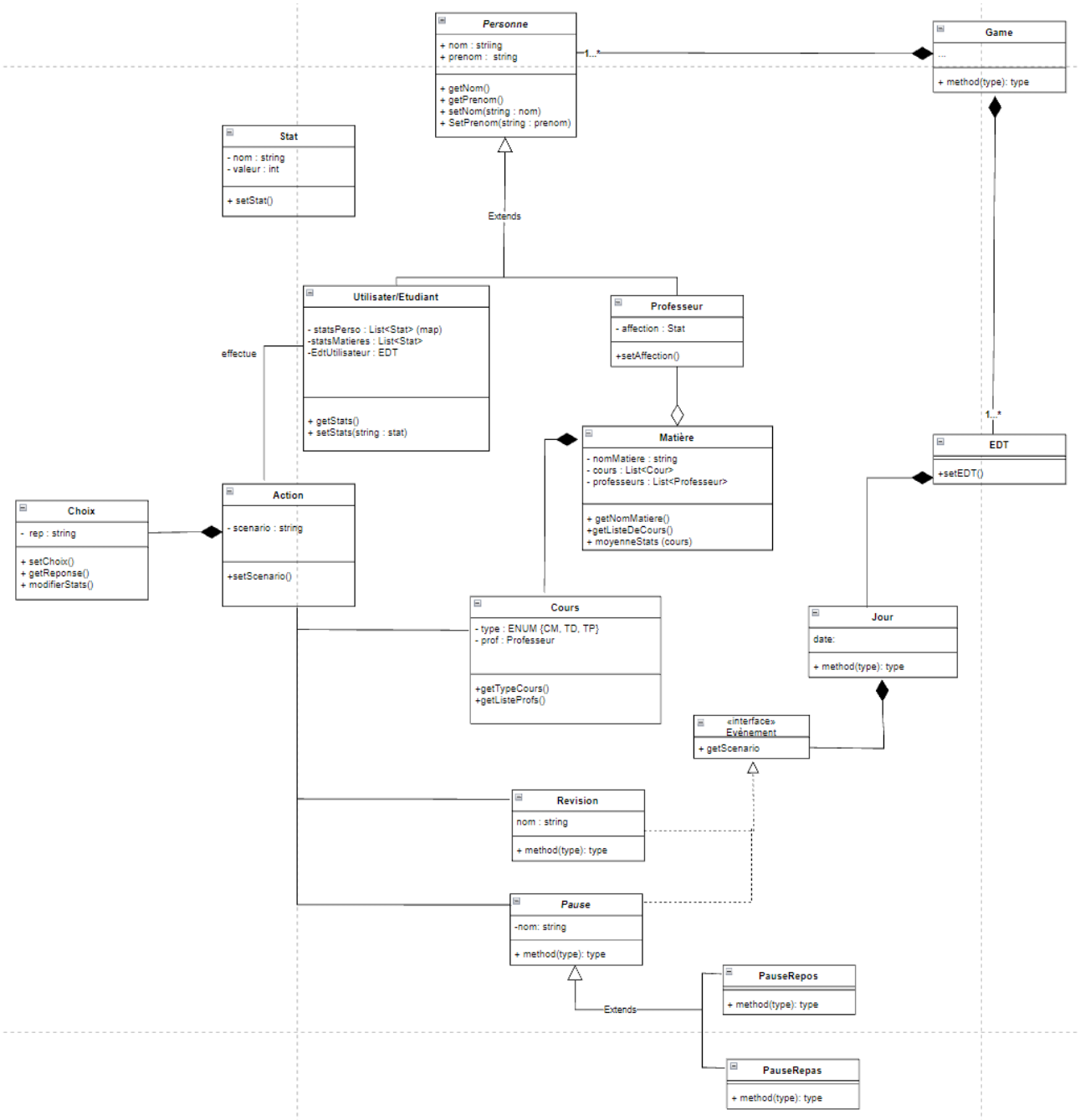
La classe **Matière** aura en attribut un nomMatière, une liste de cours et une liste de professeurs. En méthode elle aura les getter et setter associé et une méthode de moyenne des statistiques liées à chaque cours.

La classe **Cours** aura en attributs le type de cours qui sera une énumération et le professeur associé. Elle aura pour méthode toujours les getter, setter et éventuellement l'affichage d'un quizz.

La classe **Action** qui composera en partie les cours (compositions), qui aura en attribut une liste de Choix et un scénario.

La classe **Choix** qui composera en partie les actions (composition) aura en attribut une réponse. En méthodes il y aura les getter et les setter habituels.
Une Interface ***Évènement***, permettra de faire le lien avec la classe **Pause** et la classe **Révision**, la classe Pause sera composé de la classe PauseRepas qui permettra de simuler un repas pour restaurer la faim et la classe PauseRepos qui restaurera l'attention et la fatigue.
Enfin nous aurons une classe EDT (Emploi du temps) qui agrégera une classe Jour qui elle même sera relié à l'interface Événement.

B) Diagramme de classes



Le diagramme n'étant pas très lisible, voici le lien pour le visualiser : https://uncloud.univ-nantes.fr/index.php/s/2wtFXofq2ZQcxQ7

Il faut admettre qu'il n'est pas finalisé. Nous avons eu du mal à le faire sans avoir commencé la programmation. On pourra le remplir au fur de développement de jeu en regardant les interactions possibles entre les différentes classes.

C) Interfaces

Si on aura le temps, nous aurons une interface graphique en forme d'emplois du temps où on verra l'emploi du temps que on était censé suivre. Sur ce système d'emploi du temps on pourra voir une barre défiler tout au long de la journée et qui nous indiquera où nous en sommes.

On aura aussi des fenêtres « Pop-up » avec les quizz et décisions possibles.

Ensuite l'utilisateur pourra cliquer sur des boutons pour accepter ou non par exemple d'assister à un cours.

D) Aspects spécifiques

Nous allons utiliser un modèle MVC pour la programmation de notre jeu .

Aussi on aura à trouver un algorithme de calcul de variation des statistiques. Par exemple, la baisse de l'attention sera plus ou moins importante en fonction de la faim et de la fatigue.

3. Regard critique

Sur la mise en œuvre des concepts de la POO

Notre jeu implémente bien les notions vues dans le cours. On a des classes qui héritent d'une classe abstraite Personne. On a aussi une interface Evenement qui est hérité par les autres classes. Tout cela donne une possibilité d'implémenter la notion de polymorphisme entres les interfaces et classes concernées.

Sur le calibrage du projet et la quantité de travail fournie

Sur la quantité de travail fournie jusqu'ici elle nous semble plutôt cohérente bien qu'elle puisse être augmenté à l'avenir lorsque nous nous lancerons dans la conception. Justement pour la conception, le projet parait plutôt imposant et il va falloir être très attentif dans notre suive de projet pour tenir les dates. Mais nous pensons que sa taille est adapté pour le temps qu'il reste. Grâce au diagramme UML nous rendons mieux compte du travail à fournir avant d'envisager les extensions et nous avons réussi à définir le cadre que nous voulions pour la partie graphique. Notre travail est donc pour l'instant bien délimité et si nous le répartissons correctement tout devrait se passer correctement.

Nous envisageons d'utiliser un outil Monday.com de gestion de projet pour bien être organisé avant la date du rendu final.

Sur le travail, la motivation et l'organisation de l'équipe

L'investissement du groupe peut être améliore. Nous avons pour l'instant travaillé majoritairement en groupe en salle de TP sur nos créneaux libres et nous avons bien avancé à chaque fois, de plus le fait d'être tous là en présentiel facilite les prises de décisions et la communication. L'équipe est toujours autant motivé qu'au départ et a hâte de passer à la conception. On s'organise à chaque réunion sur ce que l'on doit faire pour la fois suivante.

Sur les compétences de l'équipe au début du projet et le développement de nouvelles

Nos compétence au début du projet étaient moyenne par rapport à la grille fournie. Nos compétence sur l'architecture d'une application on progressé ainsi que sur la gestion du projet avec Gitlab. Nous avons aussi progressé dans la communication entre nous.

Sur des perspectives d'extension du projet et sur l'extensibilité de la solution actuelle

Si le temps nous le permet, nous avions penser à rajouter un budget, donné à l'utilisateur en début de semaine/partie, avec lequel il pourra aller au restaurant universitaire. Ainsi, sa faim diminuera et sa performance en cours augmentera.

Toujours lié au budget, l'étudiant pourra se faire des petites sorties avec ses camarades et pourra se rendre aux fêtes étudiantes. Mais cela constituera peut-être un moyen de distraction pour ses cours et l'empêchera d'être assez concentré durant ces derniers (par exemple le lendemain d'une fête).

Cependant, il nous faudra créer des personnages, agrandir et améliorer notre interface graphique, rajouter encore plusieurs méthodes, sans doute en modifier plusieurs déjà implémentée, il nous faudra aussi plus de déplacement dans notre plateforme graphique et surtout plus d'interactions.

On pourra ajouter plusieurs professeurs pour un même cours : C'est à dire pour un même TD on aura deux professeurs par exemple.

Sur le point d'extension possible : avec le schéma actuel d'UML on n'est pas sûr qu'il est bien cohérent si on voudra ajouter des nouvelles fonctionnalités. Cet aspect nous reste à retravailler et à revoir en cours de développement.

Sur la qualité de l'encadrement et du suivi par nos chefs de projet

Nous avons trouvé que le nombre de séance de suivi du projet était pour le début un peu limité par rapport aux questions que nous avions au début. De plus le TD sur les diagrammes UML est arrivé juste avant la date de rendu du Livrable 3 donc nous n'avons pas eu beaucoup de temps pour le préparer après la compréhension de UML qu'a apporté le TD.