Licence Informatique 2e année

Programmation Objet I

Sujet du projet 2016-2017

1 Organisation

Le projet est à réaliser en binôme. En fin de semestre, lors de la dernière séance de TP, les binômes présenteront leur travail aux encadrants de TP. Lors des présentations, chaque binôme rendra un rapport qui présentera le programme développé, les résultats obtenus, les problèmes rencontrés, les solutions apportées et tout élément permettant d'évaluer le travail réalisé. Chaque binôme devra également, au moment des soutenances, envoyer le code des programmes écrits aux responsables de TP (*christine.irastorza@u-picardie.fr* et *frederic.furst@u-picardie.fr*).

La note du projet dépendra en bonne partie du respect des principes de génie logiciel mis en œuvre en programmation objet, en particulier la modularité, l'abstraction, l'encapsulation et la documentation.

2 Sujet

Le projet consiste à réaliser une simulation, éventuellement interactive, de réseau de transport urbain. La simulation comporte des lignes composées d'arrêts. Un arrêt peut éventuellement appartenir à plusieurs lignes. Une ligne possède aussi des véhicules, avec une capacité (nombre maximum de passagers). Une ligne de transport ne comporte que des véhicules de même type. La simulation doit comporter au moins deux types de véhicules :

- des bus, dont le déplacement est susceptible d'être freiné voire stoppé temporairement par des véhicules particuliers.
- des tramways, ayant une capacité supérieure à celle des bus, et dont le déplacement est régulier (on suppose que les tramways sont prioritaires sur les véhicules particuliers).

Les véhicules se déplacent en suivant leur ligne, et, à chaque arrêt, des passagers montent ou descendent. Chaque arrêt est plus ou moins fréquenté, c'est-à-dire qu'il génère plus ou moins vite des passagers.

Les passagers générés par les arrêts ont un arrêt d'arrivée et un trajet qu'ils vont suivre en montant et descendant dans autant de véhicules que nécessaire. Pour chaque passager qui apparaît à un arrêt, on peut générer son trajet entier (tous les arrêts par lesquels il va passer), ou bien fixer l'arrêt qu'il veut atteindre et calculer le plus court chemin à l'aide d'algorithmes comme l'algorithme du plus court chemin de Dijkstra, ou bien combiner ces deux approches.

3 Gestion du temps

Le jeu fonctionne en mode continu (pas au tour par tour) et peut être mis en pause. Pour temporiser le jeu, on peut utiliser l'instruction *try{Thread.sleep(n);} catch(InterruptedException e){}*.

On peut aussi utiliser la classe *javax.swing.Timer* qui déclenche l'appel d'une fonction à intervalle régulier.

4 Interface graphique

Pour l'affichage et le contrôle du programme, une classe *GUI_for_Displayable* est fournie. Elle permet d'afficher n'importe quel objet implémentant l'interface *Displayable* fournie également. Cette interface impose que chaque objet affichable donne la forme qui le représente (instance de *java.awt.Shape*), la couleur utilisée pour l'afficher (instance de *java.awt.Color*) et le texte associé avec sa position. Pour avoir un texte sur plusieurs lignes, il faut ajouter des \n dans la chaîne.

La classe *GUI_for_Displayable* offre des constructeurs pour créer une interface avec un fond coloré ou avec une image de fond, des méthodes pour ajouter et retirer un *Displayable* de l'interface, et une méthode *getClic()* qui renvoie une instance de *java.awt.MouseEvent* correspondant au dernier clic non traité de l'utilisateur (les clics sont stockés dans une file d'attente). Un *MouseEvent* décrit un clic avec une position, le bouton appuyé, les touches appuyées au moment du clic (Shift, Alt, ...).

La classe offre également une méthode *setBottomFieldText(String s)* qui permet de spécifier le texte affiché dans le composant textuel en bas de la fenêtre, et une méthode *displayMessage(String s)* qui ouvre une fenêtre de dialogue pour afficher la chaîne de caractères. Il est possible d'afficher ainsi le ratio production/consommation, mais ce ratio peut aussi être affiché à l'aide de courbes par un objet de type *Displayable*.

L'interface peut être modifiée ou améliorée à votre convenance, mais ce n'est pas sur cet aspect du programme que vous serez notés.

5 Pour aller plus loin ...

Les extensions du projet décrites dans cette partie sont optionnelles, mais en traiter au moins une augmentera très sensiblement la note attribuée, SI LA PARTIE OBLIGATOIRE A ÉTÉ TRAITÉE !

Il est possible de gérer l'aspect financier du réseau, en calculant les sous récupérés à chaque fois qu'un passager monte dans un véhicule, et les sous dépensés chaque jour pour les salaires des conducteurs, le carburant et l'entretien des véhicules. On peut aussi modéliser le fait que certains passagers ne payent pas leur ticket, et ajouter des passagers particuliers, appelés contrôleurs, qui se déplacent aléatoirement d'un véhicule à l'autre et ont pour effet quand ils sont dans un véhicule que tous les passagers sans ticket payent leur place plus une amende.

On peut ajouter d'autres types de véhicules : bus volants (très rapides et réguliers, mais très gourmands en carburant), bateau (déplacement régulier mais lent), ...

On peut gérer des pannes de véhicules. Si un véhicule tombe en panne, il reste sur place le temps de la réparation, ou bien disparaît temporairement de la simulation, et les passagers du véhicule sont repositionnés à l'arrêt le plus proche.

On peut gérer les flux de passagers en fonction des types de zone desservies par les arrêts (résidentiel, entreprises, commerces, ...) et des heures de la journée.