Rapport De projet
Génie Logiciel

Réalisé par :
Boutin Yoann
Varandas Kévin
Benzarti Zied
Destefanis Marc

Année 2014/2015

Table des matières

Introduction

Ce TP long a pour but de nous faire pratiquer et acquérir les principaux concepts de génie logiciel. Pour cela nous avions à faire un refactoring de notre code de simulateur.

Dans un premier temps, nous avions pour tâche de faire compiler, tester et exécuter notre projet à l'aide de l'outil de gestion et l'automatisation de production des projets : Maven. Ensuite, nous avions à mettre en place un système de plugins avec les classes créatures et tests déjà disponibles. Les tests devaient être exécutés avant de charger les plugins. Ainsi, s'il y a une ou plusieurs erreurs dans un test, le plugin correspondant n'est pas chargé. Pour démontrer cela, nous avons utilisé la méthode fail() dans notre test de BouncingCreature, qui n'est pas chargé et donc pas visible dans l'environnement. Afin que l'utilisateur comprenne les erreurs survenues dans les tests, un rapport d'erreur est disponible à partir de l'environnement. Nous avions, ensuite, à faire le refactoring des classes des créatures existantes afin de bien séparer les créatures et leur comportement au bord de leur environnement ainsi que leur déplacement.

Enfin, nous devions modifier l'application principale afin qu'elle charge et associe le déplacement et le comportement ainsi que de créer des créatures variées et différentes en même temps dans le simulateur.

Nous avons décidé de prendre la version corrigée fournie par le professeur afin de partir d'une version stable du code.

Choix de conceptions

Afin de mettre en place un système de plugin pour les comportements et les déplacements, nous avons créé des interfaces, IComportement et IDeplacement, et des classes qui implémentent ces interfaces. Nous avons donc Toric, Circular et Closed qui implémentent notre interface IComportement et Smart, Stupid et Troupeau qui implémentent IDéplacement. Les plugins sont déposées dans des répertoires comportements et déplacement à côté des plugins des créatures déjà existants.

L'utilisateur a la possibilité de créer une créature customisée en lui donnant un comportement aux bords et un type de déplacement. Nous avons décidé de faire un pattern singleton pour le monde (comportement aux bords) afin de s'assurer qu'il n'y ait qu'un seul monde pour toutes les créatures customisées. Lors de la création des créatures, une popup apparaît afin que l'utilisateur puisse y saisir

le nombre de créatures qu'il souhaite créer. Si aucun chiffre n'est entré par l'utilisateur, le nombre de créatures crées est de 1.

Possibilité de créer des créatures préalablement configurées, qui ont leur propre déplacement :

- BouncingCreature : déplacement 'hasard'.
- SmartCreature : déplacement 'troupeau'.
- StupidCreature : déplacement 'stupid'.

Les comportements proposés sont les suivants :

- Circular: rebond des créatures au nord et au sud de l'environnement mais torique sur les côtés.
- Toric : réapparition de l'autre côté sur tous les axes.
- Closed : rebond sur tous les bords.

Les déplacements proposés sont les suivants :

Troupeau : Lorsqu'une ou plusieurs créatures entre dans les champs de vision d'une créature, sa vitesse et direction change en fonction de la moyenne des directions et vitesse de celles qui l'entourent.

Hasard : Chaque créature possède un cycle qui lui est propre, tous les 30 cycles on lui affecte une variation sur sa vitesse et sa trajectoire.

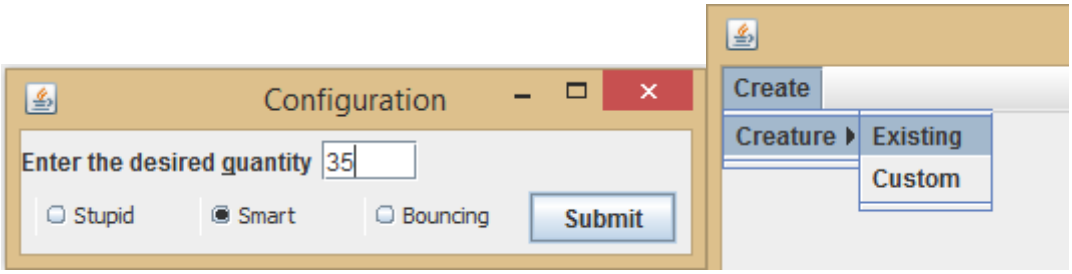
Stupid : Les créatures suivent la même trajectoire et gardent une vitesse constante tout au long de la simulation.

Lorsque l'utilisateur a choisi ou configuré son type de créature, il peut appuyer sur (Re-)start simulation. Les créatures apparaissent alors dans l'environnement. S'il souhaite ajouter un autre type de créatures, il peut choisir ou configurer une nouvelle fois des créatures et appuyer sur Add creatures. Il pourra donc avoir différents types de créatures dans le même environnement.

Nous avons fait en sorte que l'utilisateur ne soit pas perdu dans le simulateur et avons défini un ordre de tâches à effectuer pour le bon déroulement d'une simulation. A savoir, une CustomCreature ne peut être créée si l'utilisateur n'a pas préalablement donné un comportement. Après la sélection du comportement et de custom créature, l'utilisateur est amené à choisir un déplacement. Tout comme une simulation ne peut être lancée si aucune créature n'a été créée.

Problèmes rencontrés

Nous avons commencé par faire une interface plus élaborée avec des boutons radio, menu items. Mais il était difficile de gérer proprement la création automatique des éléments dans le menu et de récupérer les bons plugins lors d'un envoie de formulaire au travers d'un bouton dans une nouvelle fenêtre (images ci-dessous).



Ce qui nous a amené à faire une interface plus basique.

Le Refactoring des créatures nous a posé pas mal de problèmes, nous avons eu du mal à visualiser comment procéder pour utiliser nos plugins de comportements et de déplacements.

Conclusion

Ce TP long nous a permis de prendre du temps pour réfléchir à la manière d'organiser notre code afin qu'il soit plus maintenable et réutilisable grâce aux patrons conceptions et au génie logiciel. Nous avons également appris à mettre en place un système de plugin.

La réalisation de ce TP nous a aussi permis d'apprendre à utiliser la bibliothèque graphique Swing.

Les tests ont été très importants pour nous permettre de maintenir l'integrité de l'application. Nous avons en effet essayé de réaliser ce projet en utilisant la méthode AGILE.