ISIMA F2 – ZZ2
TP de simulation Multi-agent

Le développement du TP de simulation multi-agents peut s'inspirer du cadriciel présenté en cours et des exemples proposés sur les ouvrages indiqués en cours (cf. url sur les slides).

La catégorie d'agents choisie est de votre ressort (simulation économique, simulation de réseau, d'écosystème, jeu,…). Vous prendrez cependant en compte les contraintes suivantes:

- Mise en œuvre de 2 catégories d'agents avec un comportement social. Ce comportement est libre : compétition pour des ressources, compétition en agents,… Restez modeste dans vos choix, il s'agit d'un TP et pas d'un projet.
- Le comportement des agents utilise une source de hasard. Chaque agent doit disposer de sa propre source : générateur de nombre pseudo-aléatoires (reproductibilité, trace et suivi d'un agent). La reproductibilité
- Le cadre d'évolution des agents se fait dans un espace à 2 dimensions représenté par une matrice 20 x 20 affichée en mode texte. Une lettre peut représenter un agent ou une ressource passive si vous en avez introduit une.
- Vous proposerez vos règles de déplacement, de vie, de mort, de naissance, les champs de vision des agents (voisinages etc).
- L'activation 'en parallèle' des agents ne reposera pas sur les threads système (reproductibilité) mais se fera avec un tirage aléatoire afin de simuler le parallélisme et afin que chaque agent activable à un instant 't' (temps virtuel de la simulation) ait la même probabilité d'être choisi (certaines implémentations peu sérieuses prennent toujours les 1er agents en tête de la structure de données représentant l'échéancier de la simulation).
- Le développement d'une interface graphique n'est pas nécessaire. Ce type de développement peut se faire dans un 2ème temps uniquement si cela vous intéresse et que vous avez du temps libre. Attention au temps nécessaire pour ce type de développement qui peut représenter plus de 70% du temps de développement. Ce n'est pas le sujet du TP qui devrait se focaliser sur le développement d'une petite intelligence artificielle distribuée.

Exemple d'affichage en mode texte :

```
. . . . . .
. . L . . . . . . . . . . . .
. . . . . .
. r . . R . . . . . . . . . .
. . . . . .
. . . . . L . . . . . . . . .
```

Le TP de multi-agent donnera lieu à une rédaction en 2 temps :

(1) Une partie d'analyse et de conception est à rédiger pour le jour de l'examen de simulation (contre émargement – binôme ou monôme).

Dans votre rédaction vous présenterez d'une part votre cahier des charges avec votre analyse des besoins, vos choix et spécifications, les classes d'agents et les règles retenues… Vous utiliserez UML, mais vous ne rentrerez pas encore dans la conception.

Vous donnerez aussi les éléments de planification des principales tâches de développement de votre TP. Comme sur un projet réel on commence petit, et on avance étape par étape en les validant au fur et à mesure. Vous fournirez si possible un Diagramme de Gant.

Dans la 1ère phase de rédaction, vous n'avez pas en entrer dans la conception détaillée, et encore moins dans l'implémentation (cela fera l'objet de la 2ème partie de ce TP pour mi-février avec démonstration et évaluation du code source).

TD/TP – Architecture logicielle et qualité – lien avec le TP long de SMA

(2) Le développement et la deuxième partie de la rédaction se feront pendant le temps du cours d'Architecture Logiciel et Qualité. Le langage de développement n'est pas imposé (C++ proposé mais pas obligatoire, la STL n'est pas nécessaire dans ce cas, mais reste bien sur autorisée). Les contraintes pour la conception et l'implémentation sont les suivantes :

- Utilisation d'au moins 3 patrons de conception. Patrons candidats : le patron composite (groupe d'agent), le patron stratégie (comportement d'agents). Les patrons MVC/Observer, state et singleton peuvent être considérés ainsi que d'autres.
- Un outil de documentation automatique de type DOxygen est à utiliser.
- Vous utiliserez au moins 2 autres outils du génie logiciel pour ce TP (cas d'un binôme)

La rédaction finale comporte au maximum 10 pages de présentation du développement en binôme et un complément de 5 pages – à rédiger individuellement – sur un outil du développement logiciel de votre choix et que vous souhaitez découvrir de façon plus approfondie. Les présentations synthétiques de ces comporteront, les points suivants :

- Résumé.
- Procédure d'installation (url, etc…).
- Présentation des fonctionnalités.
- Mini-guide d'utilisation sur les principales fonctionnalités dans le cadre du TP Multi-agents.
- Les points clés du logiciel.

Vous pourrez ensuite à la fin de cette 2ème partie présenter dans la finale du rapport de TP tout ce qui a été changé par rapport à votre dans l'analyse/conception initiale lors de la 1ère rédaction. Ces changements (qui sont normaux) ont induit des délais de développement, un décalage des différentes tâches que vous commenterez.

Ce TP vous initie de façon modeste à une gestion de projet de développement. Pour les binômes : proposition de mise en œuvre de la méthode eXtreme Programming. Documentez-vous, il d'agit d'un cycle testeur/développeur. On parle aussi de « Test Driven

Development ». Dans ce cas vous pourrez donner un pourcentage de couverture de test du logiciel. Tout comme la présentation de l'analyse, la présentation de la conception se fera avec UML.

Pour le rendu final de cette 2ème partie : imprimer un dossier papier comprenant partie conception et implémentation du TP SMA et imprimer vos présentations individuelles d'un outil. Attention : éviter les copier/coller pour les présentations d'outil et citer toutes vos sources (URL). L'université dispose de l'outil « compilatio » pour l'anti-plagiat.

Exemples d'outils du développement logiciels que vous pouvez choisir (liste non exhaustive) :

Ateliers UML	Objecteering/UML Personal Edition Poseidon for UML Community Edition Visual Paradigm for UML Community Edition Rational Rose Poseidon (Argo) Bouml Dia Fujaba tool suite O mondo…
Gestion de projets	Etude approfondie de make Etude de cmake Etude de Ant Etude de Maven – (Apache Software Foundation) Utilise un paradigme connu sous le nom de Project Object Model (POM) pour décrire un projet logiciel et ses dépendances avec des modules / bibliothèque externes …
Gestion des versions	CVS, Tortoise CVS SVN, Subversion Mercury Git...
Etude des performances	Prof / Gprof Jprofiler AQtime…
Etude d'un débuggeur	gdb – ddd dbx – dbxtool Eclipse débuggeur .Net débugger …

Etude des tests	CppUnit, Junit, Valgrind, Kcachegring, …
Logger de traces d'exécution	Log4J…
Production de documentation	Doxygen Javadoc …