Projet Java : Développement d'un programme permettant à la machine de jouer le jeu *Xou Dou Qi* contre les humains et contre d'autres machines.

2ème Année Génie Informatique
Année Universitaire 2018/2019

L'objectif du projet est la réalisation d'une application JAVA qu'on nommera *JungleApp* et qui pourra jouer le jeu *Xou Dou Qi* contre un utilisateur humain ou contre une autre application *JungleApp* installée sur une autre machine.

Chaque groupe d'étudiant développera son propre algorithme pour *JungleApp*. On désigne par *JungleApp_X* l'application *JungleApp* d'un groupe X.

1. Machine contre un utilisateur humain

Lorsque la machine joue avec un utilisateur humain le contrôle des règles du jeu est fait par l'application elle-même. Voici les fonctionnalités qu'il faut implémenter dans ce cas :

1. L'application et le joueur pourront déplacer leurs pièces sur le plateau en respectant les règles du jeu.
2. En cas d'un déplacement non autorisé par le joueur, l'application doit afficher un rappel textuel de la règle non respectée.
3. L'application doit permettre de sauvegarder une partie non terminée avec possibilité de la reprendre.
4. *JungleApp* pourra développer une certaine intelligence en se basant sur les situations déjà rencontrées en jouant avec les humains.

2. Machine contre une autre Machine

Soit X et Y deux groupe d'étudiant, l'application *JungleApp_X* doit pouvoir jouer automatiquement sans intervention d'un utilisateur avec une autre application *JungleApp_Y* installée sur une autre machine sur le même réseau local. Dans ce cas, les règles du jeu sont contrôlées par un programme qu'on nommera *JungleApp_EVAL* qui sera installé sur une machine tierce. Ainsi, la communication entre *JungleApp*_X et son adversaire passera toujours via *JungleApp_EVAL*.

Il n'est pas demandé de développer l'application *JungleApp_EVAL* elle vous sera fournie.

Ci-dessous un scénario d'exécution d'un déplacement d'une pièce effectuée par *JungleApp_X*:

1- *JungleApp_X* déplace une pièce, ceci engendrera une communication réseau avec *JungleApp_EVAL* pour lui envoyer un message indiquant la pièce déplacée et le déplacement effectué
2- *JungleApp_EVAL* vérifie que le déplacement effectué est possible selon les règles du jeu. Il y a trois cas qui se présentent :
    1.2. Si le déplacement est autorisé, elle fait les calculs nécessaires des points de chaque application et informe *JungleApp_X* et *JungleApp_Y* par un message via le réseau.

Les deux applications mettront à jour leurs interfaces affichant l'état de l'échiquier et les points obtenus.

1.3. Si le déplacement n'est pas autorisé, *JungleApp_EVAL* va enlever un point pour l'application *JungleApp_X* et informe les deux applications.

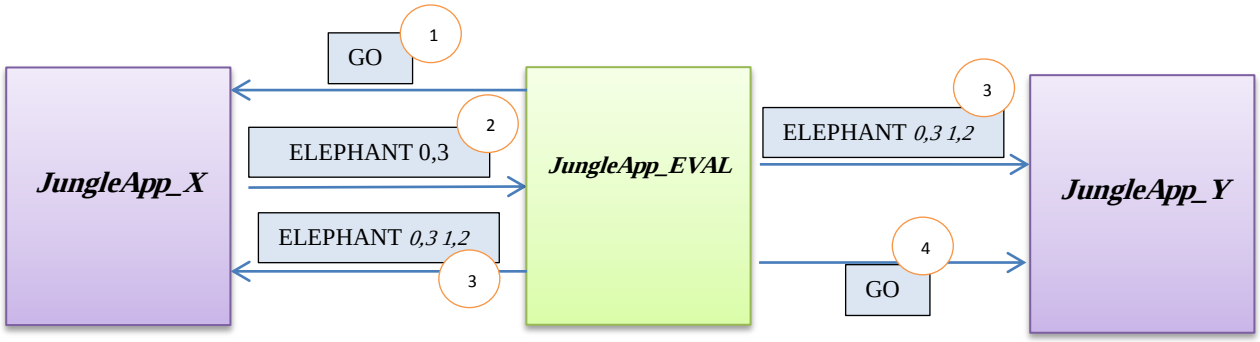
1.4. Si le solde d'une application (nombre de points) devient négatif, l'application adversaire est déclarée gagnante.

Le format des messages échangés entre les applications *JungleApp* et *JungleApp_EVAL* doivent respecter les spécifications ci-dessous :

- GO ce message est envoyé par *JungleApp_EVAL* à *JungleApp* pour lui demander de faire un déplacement d'une pièce.
- ERROR GAME_RULES_NOT_RESPECTED point1,point2 : ce message est envoyé par *JungleApp_EVAL* lorsque une application essaie de faire une action qui ne respecte pas les règles du jeu. point1 est le nombre de points de l'application qui a fait l'action erronée et point2 le nombre de points de son adversaire.
- ANIMAL_X a,b : ce message est utilisé par *JungleApp* pour demander à *JungleApp_EVAL* de déplacer l'animal ANIMAL_X vers la case ayant les coordonnées (a,b) sur l'échiquier. Exemple : RAT 2,2 : ce message est une demande de déplacement de l'animal Rat vers la case (2,2).
- ANIMAL_X a,b point1,point2 : ce message est utilisé par *JungleApp_EVAL* pour informer *JungleApp* que l'animal ANIMAL_X de l'adversaire a été déplacé vers la case (a,b) et que son nombre de points est point1 et celui de l'adversaire est point2.

Ci-dessous un exemple d'échange de messages:

1. *JungleApp_EVA*L donne la main à *JungleApp_X* pour jouer
2. *JungleApp_X* déplace l'éléphant vers (0,3)
3. *JungleApp_EVAL* accepte ce déplacement et informe les deux applications par ELEPHANT 0,3 1,2 pour qu'elles puissent mettre à jour leurs états.
4. *JungleApp_EVAL* donne la main à *JungleApp_Y* pour jouer



Architecture de l'application :

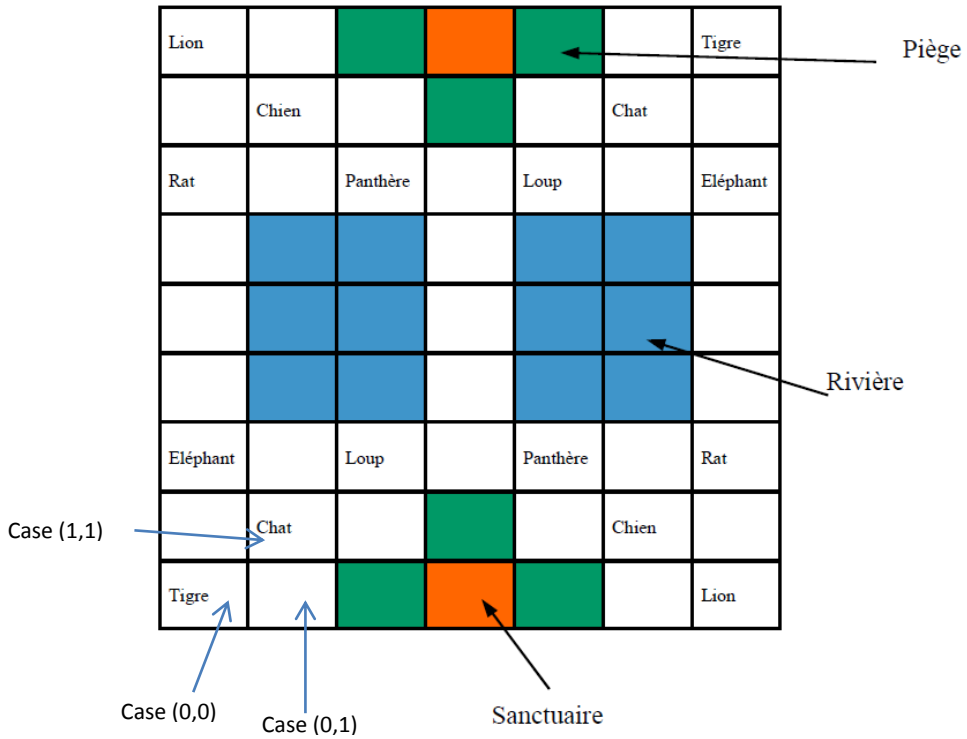
Il est demandé de mettre en place une architecture modulaire avec un couplage faible entre chaque couche : IHM, Logic layer, DAO Layer, Network Layer.

- *JungleApp* utilise une base de données embarquée Java DB pour y stocker les données nécessaires à son bon fonctionnement.
- La communication réseau doit se faire via les sockets avec Java.

- Les tâches longues doivent être déléguées à des threads séparés.
- L'application pourra fonctionner soit en mode console soit avec interface graphique.

3. Principe du jeu de la jungle :

Le Xou Dou Qi ou Doushou Qi ou le jeu de la jungle est aussi appelé les échecs des animaux. Il est originaire de Chine. Le plateau de jeu est de neuf cases sur sept, de part et d'autre, au centre, deux lacs; de chaque côté un sanctuaire et trois cases pièges. Cf. la figure ci-dessous :



Chaque joueur dispose de huit pièces hiérarchisées (animaux), le vainqueur est celui qui occupe le sanctuaire adverse, les prises sont réalisées par substitution, une pièce ne peut prendre qu'une pièce ennemie d'hiérarchie égale ou inférieure.

Hiérarchie des pièces :

1. ELEPHANT
2. LION
3. TIGRE
4. PANTHERE
5. CHIEN
6. LOUP
7. CHAT
8. RAT

Chaque pièce peut capturer une pièce de rang inférieur ou égal mais jamais une pièce d'un rang supérieur, sauf le Rat qui peut capturer l'éléphant. Le Lion et le Tigre peuvent sauter par-dessus la rivière (sauf si un rat nageant se trouve sur sa trajectoire) dans le sens de la largeur comme dans la longueur. Le rat est le seul à pouvoir se déplacer dans l'eau. Les pièces se déplacent toutes d'une case horizontalement ou verticalement mais ne peuvent pénétrer dans leur propre sanctuaire. Les prises sont réalisées par substitution (les diagonales sont interdites aussi bien pour le déplacement que pour la

prise). Les Pièges sont des cases où les pièces ennemies se retrouvent tout en bas de la hiérarchie, elles ne peuvent prendre aucune pièce et peuvent être prise par toutes les pièces adverses. Le rat ne peut prendre en sortant de la rivière. Le premier à atteindre le sanctuaire ennemi gagne. Chaque joueur dispose de huit pièces (Eléphant, Lion, Tigre, Panthère, Chien, Loup, Chat, Rat).

4. Evaluation et livrable

Le livrable doit contenir :

- Le code source
- L'exécutable
- Un résumé de 6 pages (police Arial 11) maximum décrivant le travail réalisé et expliquant la conception choisie et l'algorithme implémenté. Les pages supplémentaires à 6 ne seront pas lues.

L'évaluation va se faire en deux parties :

- **Evaluation 01 : concerne la partie machine contre un utilisateur**

Chaque groupe présentera une démonstration de son application. Et obtiendra une note note1 en se basant sur la qualité du travail réalisé et le nombre de fonctionnalités correctement implémentées. Il sera également tenu en compte de :

- ✓ la qualité de la conception orientée objet
- ✓ le faible couplage entre les couches de l'application
- ✓ l'extensibilité
- ✓ la documentation du code et la lisibilité (respect des normes de codage, JavaDoc, commentaires, …)

- **Evaluation 02 : concerne la partie machine contre machine**

Elle sera organisée sous forme d'un challenge entre toutes les applications réalisées.

- ✓ Chaque groupe évaluera son application en jouant 3 parties automatiquement avec chaque application parmi les applications réalisées. La moyenne des résultats obtenus permet de déterminer l'application gagnante.
- ✓ Un classement des applications réalisées aura lieu suite à ce challenge
- ✓ Une note note2 sera affectée à chaque application à partir de son classement

La note finale = 0.6*note1 + 0.4*note2

Bon courage