INF5153 – Génie logiciel : conception
Projet de session – Été 2017

Le travail doit être fait en équipe de 3 (+/- 1) personnes. Idéalement, les équipes devraient rester les mêmes durant toute la session. Aucun retard ne sera accepté.

Bataille navale

Vous devez concevoir un logiciel pour jouer à Bataille navale.

Règles du jeu

Le jeu implique 2 joueurs. Avant le début de la partie, chaque joueur positionne 5 navires sur une grille de 100 cases (10 colonnes numérotées de A à J et 10 rangées numérotées de 1 à 10). Les navires ont des tailles variables :

- le porte-avion utilise 5 cases;
- le croiseur utilise 4 cases;
- le contre-torpilleurs utilise 3 cases;
- le sous-marin utilise 3 cases;
- le torpilleur utilise 2 cases.

Chaque joueur garde le positionnement de ses navires secret.

À tour de rôle, les joueurs vont lancer une torpille sur une case de la grille de l'autre joueur. L'autre joueur peut donner 3 réponses :

- Dans l'eau : signifie qu'aucun navire n'est présent sur cette case.
- Touché : signifie qu'un navire a été touché mais qu'il n'a pas coulé.
- Coulé : signifie qu'un navire a été touché et qu'il a coulé.

Un navire coule si toutes ses cases ont été touchées. Le gagnant de la partie est le joueur qui coule tous les navires de son opposant en premier.

Démarrer une partie

Lors du démarrage d'une partie, le logiciel demande à l'utilisateur s'il désire jouer contre un humain ou contre un joueur artificiel.

S'il choisit de jouer contre un humain, l'application se connecte à un serveur qui lui affectera un autre joueur par Internet.

S'il choisit de jouer contre un joueur artificiel, il devra ensuite choisir un niveau de difficulté.

Niveau de difficulté

Le logiciel offrira 2 niveaux de difficulté. Chaque niveau de difficulté correspond à un algorithme d'intelligence artificiel différent.

1er algorithme : Niveau débutant

Le joueur artificiel effectue des coups au hasard.

2ème algorithme : Niveau avancé

Le joueur artificiel effectue des coups en fonction de l'algorithme minimax.

Visualiser une partie

Lorsqu'une partie est terminée, il est possible de visualiser le déroulement de la partie en rejouant les tours.

Sauvegarde et chargement

Pendant une partie contre un joueur artificiel, le jeu peut être sauvegardé dans un document XML. Une partie sauvegardée peut être chargée également.

Meilleurs temps

Lorsqu'un utilisateur joue une partie contre un joueur artificiel, le logiciel doit conserver le meilleur temps pour chaque niveau de difficulté. Ces données sont conservées par chaque installation du logiciel. Autrement dit, les meilleurs temps ne sont jamais transmis sur le réseau.

Lorsqu'un utilisateur bat un record de temps, le logiciel doit lui demander son nom et le conserver avec son temps et son niveau de difficulté. Il doit être possible de consulter les meilleurs temps dans l'application.

TP1 – 30 mai

Vous devez faire la conception architecturale du jeu. Vous devez remettre un rapport détaillant l'architecture projetée du système. Le rapport doit contenir :

- le(s) diagramme(s) de cas d'utilisation modélisant les fonctionnalités du système;
- le(s) diagramme(s) de classes du système;
- les diagrammes de séquence ou de communication modélisant les différentes fonctionnalités du système;
- le diagramme de packages du système;
- le diagramme de composants;
- le diagramme de déploiement du système.

Le rapport doit contenir une page de présentation et une table des matières. Le rapport doit contenir le texte nécessaire pour introduire la modélisation et ce qui est présenté dans chaque section du document. Le rapport doit être remis en format PDF par Moodle. Un seul membre de l'équipe doit faire la remise.

Le rapport doit être remise avant le 30 mai 2017 à 13h30.

TP2 – 27 juin

Vous devez faire la conception détaillée du jeu. Vous devez remettre un rapport détaillant la conception projetée du système. Le rapport doit contenir :

- le(s) diagramme(s) de classes illustrant la conception détaillée;
- les diagrammes de séquence montrant les relations entre les classes pour chaque cas d'utilisation;
- le diagramme de packages du système (en version détaillée, c'est-à-dire en indiquant les classes contenues dans les packages);
- chaque responsabilité/action des classes doit être justifiée par un pattern GRASP; cette justification doit être documentée (une justification par méthode publique; une justification pour la création des instances d'un classe).

Le rapport doit contenir une page de présentation et une table des matières. Le rapport doit être remis en format PDF par Moodle. Un seul membre de l'équipe doit faire la remise.

Le rapport doit être remise avant le 27 juin 2017 à 13h30.

TP3 – 18 juillet

Vous devez modifier votre conception détaillée afin d'y introduire des patrons de conception GoF. Le nombre de patrons à introduire correspond au nombre de membres dans l'équipe (ex. 3 patrons pour une équipe de 3 personnes). Pour chaque patron, vous devez fournir un diagramme de classe et un (ou plusieurs, si nécessaire) diagramme de séquence illustrant les relations entre les classes du patron et les autres classes du projet.

Vous devez également développer l'application dans le but de valider votre conception. Toutes les fonctionnalités du jeu doivent être développées, à l'exception de la partie contre un joueur humain (affectation d'un adversaire par Internet). Toute différence entre l'implémentation et la conception doit être documentée dans le rapport que vous allez me remettre. Pour chaque problème de conception que vous allez corriger, vous devrez décrire comment vous avez détecté le problème de conception et comment vous l'avez corrigé.

Le projet doit être développé en Java et toutes les librairies doivent être gérées avec Maven. Vous devez me remettre le code source du projet et le rapport (en format PDF). La remise doit être faite par Moodle au plus tard le 18 juillet à 13h30. Un seul membre de l'équipe doit faire la remise. N'oubliez pas d'inclure une documentation (readme) indiquant comment exécuter votre logiciel.