Projet de Programmation Objet : Implémentation d'une version simplifiée du jeu ***Advance Wars***

1 Présentation générale du projet

Dans ce projet, vous devez concevoir et implémenter une version simplifiée d'un jeu existant: ***Advance Wars***. Il s'agit d'un jeu de stratégie où les joueurs jouent à tour de rôle sans limite de temps. Nous nous concentrerons sur la dernière version du jeu, sortie sous le nom ***Dark Conflict*** en Europe et ***Days of Ruin*** en Amérique. Vous pouvez trouver des images du jeu sur YouTube, ainsi qu'un simulateur de ce jeu ici (il faut changer les touches dans les paramètres pour avoir la touche *Entrée* qui contrôle A et la touche *Echap* qui contrôle B).

Advance Wars est un jeu de stratégie militaire où l'objectif de chaque joueur est de capturer le Quartier Général (QG) adverse. Dans ce projet, il y aura deux joueurs, rouge et bleu, qui joueront à tour de rôle sur le même ordinateur. Le joueur rouge joue toujours en premier.

Les joueurs contrôlent chacun leur armée. Les armées sont composées d'unités militaires qui occupent chacune une case (il ne peut jamais y avoir deux unités par case). Une unité peut se déplacer sur les différents types de terrain, attaquer les unités ennemies ou effectuer d'autres actions spécifiques expliquées plus tard.

Le jeu se joue sur une grille rectangulaire composée de cases, à la manière d'un jeu de plateau. Chaque case correspond à un type de terrain (forêt ou montagne par exemple). Chaque unité pouvant se déplacer d'un certain nombre de cases par tour et le terrain affecte ces déplacements.

Une unité peut attaquer une unité adverse au plus une fois par tour. Chaque unité a un certain nombre de points de vie (PV), de valeur maximale 10. L'efficacité d'une attaque dépend du type d'unité, par exemple une infanterie n'est pas très efficace contre un tank, et ne peut pas du tout attaquer un avion.

À chaque tour, chaque joueur peut effectuer des actions avec chacune de ses unités. Au début du tour d'un joueur, toutes les unités de ce joueur sont marquées comme *disponibles*. Le joueur peut alors sélectionner une unité et lui faire réaliser des actions. Une fois les actions réalisées, l'unité est marquée comme non disponible (visuellement, elle change de couleur) et ne peut plus rien faire jusqu'à la fin du tour du joueur. Durant son tour, le joueur peut également produire des unités avec ses usines, ce qui sera expliqué plus tard dans le sujet.

Nous conseillons de lire les parties 2 et 3 attentivement avant de commencer à travailler.

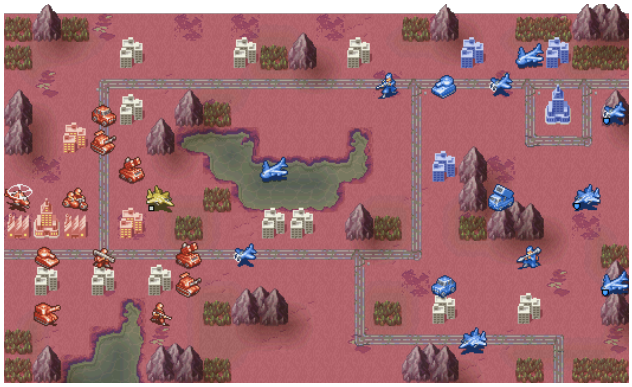

Figure 1: Exemple de carte du jeu.

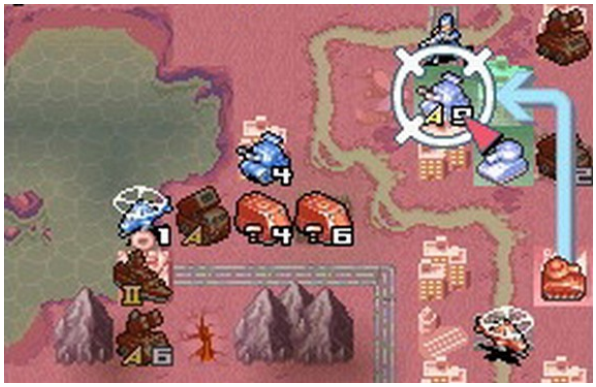

Figure 2: Le tank rouge qui a 10 PV attaque l'unité DCA bleue qui a 9 PV.

2 Explication des règles de base

Vous trouverez ci-dessous des explications plus détaillées sur les règles que vous devez implémenter. **Avant d'écrire du code, commencez par réfléchir à la conception de votre projet** (vous trouverez des informations là-dessus dans la section 3). Nous vous conseillons ensuite d'implémenter les différentes mécaniques du jeu dans l'ordre dans lequel elles sont présentées dans cette partie. Vous trouverez des détails sur le squelette qui vous est fourni dans la partie 4: lisez bien tout le sujet avant de commencer.

unités attendues	
Nom	**Caractéristiques**
Infanterie	Unité la plus fragile du jeu, mais qui peut capturer les propriétés adverses. Déplacements plutôt lents mais compatibles avec des terrains accidentés. Peu efficace en combat.
Bazooka	Similaire à l'infanterie mais équipé d'un bazooka, rendant l'unité très lente mais efficace contre les unités blindées.
Tank	L'unité offensive par excellence, rapide, robuste et puissante. Fortement ralenti par les terrains accidenté, il est également vulnérable aux attaques à distance et aux unités aériennes.
DCA	Abréviation de "Défense contre l'aviation", cette unité est particulièrement efficace contre les avions et hélicoptères, mais peu efficace contre les blindés.
Hélicoptère	Unité aérienne la plus basique, son déplacement n'est pas affecté par le terrain. Efficace contre les blindés.
Bombardier	Avion qui peut rapidement réduire les unités terrestres en miettes. Ne peut pas attaquer les autres avions.
Artillerie (bonus)	Attaque à distance, ce qui la rend particulièrement efficace défensivement. Vulnérable aux assauts de front (ne fait pas partie du projet de base, cf. partie 6).
Convoi (bonus)	Ne peut pas attaquer mais possède des capacités spéciales (ne fait pas partie du projet de base, cf. partie 6).

Figure 3: Tableau des unités

2.1 Plateau de jeu

Le plateau de jeu est une grille rectangulaire dans laquelle chaque case a un type de terrain. Les types de terrain possibles sont les suivants : Plaine, Forêt, Montagne, Eau, auxquelles s'ajoutent les propriétés : QG, Ville et Usine. Des images vous sont fournies pour les différents types de terrain, comme expliqué dans la partie 4.2.

Le plateau de jeu est chargé à partir d'un fichier d'extension ".awdcmap" sélectionné au moment du lancement de la partie. Vous trouverez plus de détails sur le chargement des cartes dans la partie 4.1.

2.2 Unités

Chaque joueur contrôle des unités militaires qui constituent son armée. Il y a au plus une unité par case du plateau. Ici encore, des images vous sont fournies pour les unités, comme expliqué dans la partie 4.2.

Chaque unité a des points de vie (PV) : initialement, toutes les unités ont 10 PV. Une unité à 0 PV est détruite. Les unités que vous allez implémenter dans ce projet sont décrites dans la figure 3.

2.3 Déplacements

Sélection du chemin et affichage d'une fleche à l'écran Le joueur actif peut sélectionner une de ses unités pour la déplacer. Pour cela, commencez par coder un curseur qui peut se déplacer sur les cases de la grille à l'aide de touches directionnelles. Pour déplacer une unité, le joueur met le curseur sur l'unité, appuie sur *Entrée*, puis dessine avec le curseur le chemin souhaité pour l'unité. Le joueur choisit donc la totalité du déplacement de l'unité en une seule fois. La figure 4 illustre l'interface souhaitée : le joueur vient de finir de tracer le chemin qu'il veut faire parcourir au tank rouge. Une flèche indique le chemin choisi, cette flèche arrêtant de suivre le curseur quand l'unité n'a plus de déplacement. La surbrillance en vert de la figure 4, qui représente les cases où le tank aurait pu se déplacer, n'est pas demandée dans ce projet.

Confirmation du déplacement et choix d'action Une fois le chemin choisi, le joueur doit confirmer le déplacement en appuyant sur *Entrée* ; s'il appuie sur *Echap*, le déplacement est annulé et l'unité est désélectionnée. Une fois le déplacement confirmé, les actions possibles par l'unité seront choisies sous la forme d'une fenêtre popup. Pour l'instant, il n'y aura qu'une seule action possible, qui est *Attendre*. Une fois le choix de mouvement et d'action confirmé par le joueur, l'unité doit être déplacée sur sa case destination (faîtes simplement apparaître l'unité sur sa nouvelle case). Ensuite, l'action sélectionnée par le joueur est effectuée. Dans le cas de l'action *Attendre*, il n'y a rien à faire. Si l'utilisateur presse *Echap* au moment de la sélection de l'action, la fenêtre popup de choix de l'action est fermée et le déplacement est annulé.

Une unité ne peut se déplacer qu'une seule fois par tour : une unité qui s'est déplacée est indisponible même s'il lui restait des points de déplacements à la fin de son déplacement.

Choix d'implémentation du trajet Il y a plusieurs manière d'implémenter le déplacement. Certaines implémentations rapporteront plus de points de d'autres.

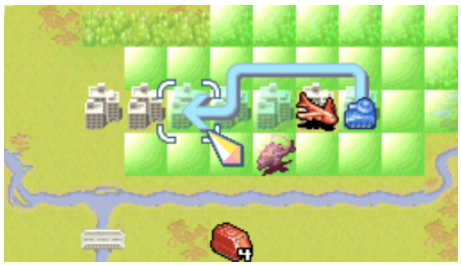

Figure 4: Le tank bleu est sélectionné et le joueur est en train de choisir son déplacement.

Une version simple serait sans mémoire des cases parcourue par le curseur, la flèche se contente de suivre le curseur jusqu'à ce qu'il n'y ait plus de points de déplacement restants. Cette version peut causer des boucles dans le trajet si le joueur repasse plusieurs fois sur la même case avec le curseur pendant le choix d'un trajet. Le joueur doit complètement annuler le trajet et recommencer pour éliminer la boucle.

La version plus complète (qui vous rapportera le maximum de points) consiste à conserver en mémoire les cases par lesquelles le curseur est passé jusqu'à la validation finale du trajet. Cette implémentation permet au joueur de revenir en arrière pendant le choix de déplacement, comme illustré sur la figure 5

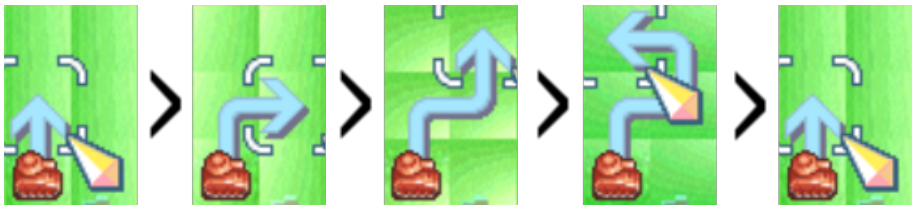

Figure 5: Ici, le joueur revient en arrière sur le déplacement qu'il commençait à tracer avec son curseur.

Limite au déplacement Une unité donnée a un certain nombre de points de déplacement qui représente le maximum de cases qu'elle peut franchir en 1 tour. Le terrain peut ralentir les unités : par exemple un tank se déplace sur chenilles, il doit donc payer deux points de déplacement pour rentrer dans une case forêt. Certains terrains sont également infranchissables pour certaines unités, comme la montagne pour les tanks et l'eau pour les unités terrestres. De plus, une case occupée par une unité ennemie est considérée comme infranchissable. Une case occupée par une unité alliée peut être traversée mais il n'est pas possible de s'arrêter dessus.

Lorsqu'une unité se déplace d'une case A à une case B, seul le type de terrain de la case B influe sur le coût du déplacement. Entrer dans une case coûte donc le coût de déplacement correspondant au moyen de locomotion de l'unité et au terrain (présenté en détail dans le tableau de la figure 6). L'infanterie et le bazooka se déplacent à pied, les autres unités terrestres se déplacent à chenilles. Les unités aériennes ne sont pas sensibles au type de terrain.

Chaque unité a sa propre quantité de déplacement. Le bazooka a 2 points de déplacement ; l'infanterie 3 ; l'artillerie 5 ; le tank, la DCA, l'hélicoptère et le convoi 6 ; et le bombardier 7. Ces valeurs sont répertoriées dans la table 8.

2.4 Gestion des combats

Points de vie Chaque unité a un certain nombre de points de vie (PV) ; intuitivement, cela correspond à l'effectif de l'unité, donc une unité avec 1 PV fera par exemple moins de dégâts en attaquant qu'une unité avec 10 PV. Le nombre de PVs est une valeur flottante, mais seul l'arrondi au supérieur de la valeur des PVs de chaque

	Moyen de locomotion	
Terrain	**À pied**	**Sur chenilles**
Propriété	1	1
Plaine	1	1
Foret	1	2
Montagne	2	Inaccessible
Eau	Inaccessible	Inaccessible

Figure 6: Coût de déplacement des unités terrestres

Arme	Infan.	Bazoo.	Tank	DCA	Hélico.	Bomba.	Convoi
Mitrailleuse légère	60 %	55 %	15 %	10 %	30 %	-	40 %
Canon	45 %	45 %	55 %	60 %	30 %	-	70 %
Mitrailleuse lourde	100 %	80 %	30 %	30 %	110 %	70 %	50 %
Missiles	50 %	50 %	70 %	40 %	70 %	70 %	70 %
Bombes	100 %	100 %	100 %	70 %	-	-	100 %

Figure 7: Efficacité des armes de bases contres les types d'unités de base

unité est utilisé dans les calculs comme celui des dégâts ou de la capture des propriétés (par exemple 4,1 sera arrondi à 5 PV). Lorsque les PVs d'une unité tombent à 0, la unité est détruite. Les PV (arrondis) d'une unité sont affichés dans un coin de la case où se trouve l'unité. Si cette unité a 10 PV, rien n'est affiché.

Attaques et ripostes L'attaque est implémentée par l'action *Attaquer*, qui est proposée à la fin d'un déplacement quand une unité est à portée. Le joueur doit ensuite sélectionner l'unité à attaquer. Une unité peut uniquement attaquer les unités ennemies qui sont dans des cases adjacentes. Lorsqu'une unité en attaque une autre, l'attaquant inflige en premier ses dégâts au défenseur, puis le défenseur, s'il n'a pas été détruit, riposte automatiquement en infligeant ses dégâts à l'attaquant. Le défenseur a donc déjà perdu ses PVs lors de la riposte. La riposte a lieu durant le tour de l'attaquant et ne compte pas comme une action pour l'unité qui défend, laquelle pourra attaquer en retour lors de son propre tour.

Armes Une unité inflige plus ou moins de dégâts aux autre unités selon l'arme qu'elle utilise. Les armes ont leurs forces et leurs faiblesses, par exemple la mitrailleuse lourde de la DCA est dévastatrice face aux unités aériennes et troupes à pied, mais ne fait que très peu de dégâts aux tanks. Certaines armes ne peuvent pas atteindre certains types d'unités. Par exemple la mitrailleuse légère ne peut pas atteindre un bombardier et le bombardier ne peut pas larguer ses bombes sur d'autres unités aériennes.

La table 7 liste les armes du jeu de base et leur efficacité contre les types d'unités de base. Cette efficacité indiqué en pourcentage sert au calcul des dégâts expliqué plus bas (partie 2.5).

Dans la partie principale du projet, une unité n'aura qu'une seule arme. L'infanterie a la mitrailleuse légère, le bazooka et le tank ont le canon, la DCA a la mitrailleuse lourde, l'hélicoptère a les missiles, et le bombardier largue des bombes.

La capacité d'une unité à attaquer une autre unité et la puissance de son attaque dépend de l'arme de l'unité. Pour infliger des dégâts, une unité doit avoir une arme pouvant tirer sur l'unité adverse. Par exemple un tank n'a pas d'arme pouvant atteindre les bombardiers. Il ne peut donc pas attaquer un bombardier, ni riposter s'il se fait attaquer par un bombardier.

2.5 Calcul des dégâts

Les dégâts infligés par une unité A à la unité B dépendent de l'arme de l'unité A, du type de l'unité B et des PV de l'unité A. Plus précisément, lorsqu'une unité A attaque une unité B, les dégâts que A inflige à B sont égaux à l'efficacité de l'arme de A multipliée par les PVs de A (qui sont arrondis à l'entier supérieur). La riposte de B suit la même règle, sauf que les PVs de B utilisés par le calcul sont ceux après l'attaque de A.

Exemple : Par exemple une DCA à 3,5 PV qui attaque un hélicoptère à 10,0 PV infligera 4,4 dégâts (4 * 1,1, le 3,5 étant arrondi à 4 pour le calcul des dégâts). Il reste donc 5,6 PV à l'hélicoptère. Puisque l'hélicoptère survit à ces dégâts, il riposte selon la même formule, infligeant 2,4 points de dégât à la DCA (6 * 0,4), qui tombera à 1,6 PV.

2.6 Propriétés

Les propriétés sont des types de terrains particuliers qui peuvent être capturés pour changer de possesseur. Une propriété est soit neutre, soit possédée par un joueur. Les propriétés peuvent changer de possesseur via la capture.

Capture *Capturer* est une action qui n'est possible que pour les unités à pied : infanteries et bazookas. Pour faire cette action, il faut que l'unité à pied termine son déplacement sur une propriété neutre ou ennemie. Il n'est pas possible pour une unité d'attaquer et de capturer dans le même tour.

Toutes les propriétés ont initialement 20 points de résistance. Lorsqu'une unité choisit l'action *Capturer*, la résistance de la propriété diminue du nombre de PVs de l'unité qui capture, encore une fois arrondie au supérieur. Lorsqu'un joueur fait tomber les points de résistance d'une propriété à 0, il prend le contrôle de cette propriété. Une unité à 10PV met donc deux tours pour capturer une propriété, et une unité à 6PV met quatre tours. Une capture doit être intégralement réalisée par une même unité. Si l'unité qui capture quitte la case de la propriété

ou est détruite avant d'avoir fini de la capturer, la capture est interrompue et la propriété remonte à 20 points de résistance. De même, une propriété capturée regagne ses 20 points de résistance en changeant de camp.

Types de propriétés Il y a trois types de propriétés : les usines, les villes et les QGs. Les villes ne jouent aucun rôle particulier. Il y a exactement deux QGs, un pour chaque joueur, et la partie est terminée quand l'un des deux joueurs capture le QG de l'autre joueur. Les usines peuvent produire des unités en utilisant du crédit.

Usines et crédits Chaque joueur commencent avec 0 crédits, et toute propriété possédée par un joueur lui rapporte 1000 crédits au début de son tour. Un joueur peut utiliser ses usine pour produire des unités, à condition d'avoir les crédits nécessaires pour produire l'unité ; dans ce cas, le prix de l'unité est soustrait à ses crédits. Vous trouverez les prix des unités dans la figure 8.

Pour produire une unité, une usine doit n'avoir aucune unité dessus (ni alliée ni ennemie). Le joueur met alors le curseur sur l'usine et appuie sur *Entrée*, ce qui ouvre une fenêtre popup lui proposant les différents unités pour lesquelles il a assez de crédits. L'unité produite apparaît alors sur la case de l'usine ; elle n'est pas disponible donc elle ne peut pas agir avant le tour suivant. Il n'est ainsi possible de produire qu'une seule unité par tour et par usine.

Unité	Déplacement	Armes	Prix
Infanterie	3, à pied	Mitrailleuse légère	1 500
Bazooka	2, à pied	Canon	3 500
Tank	6, sur chenilles	Canon	7 000
DCA	6, sur chenilles	Mitrailleuse lourde	6 000
Hélicoptère	6, aérien	Missiles	12 000
Bombardier	7, aérien	Bombes	20 000
Convoi	6, sur chenilles	Désarmé	5 000

Figure 8: Récapitulatif des unités

3 Conception

Pour développer votre jeu, il faut que vous réfléchissiez soigneusement à la structure des classes. Rappelez-vous des notions apprises durant le TP4-5 portant sur le Zoo : par exemple, un terrain peut être une propriété ou non, et une propriété peut être une ville, une usine ou un QG. De même que dans le TP sur le Zoo, votre structure doit rendre facile l'ajout de nouveaux terrains ou de nouvelles unités à posteriori. **Discutez avec votre enseignant de la structure de vos classes.**

Gardez en tête lors de la conception de cette structure de classes qu'un bout de code doit être écrit une seule fois : comme dans le TP4-5, utilisez les outils de la programmation orientée objet pour limiter la duplication de code, c'est-à-dire la présence de codes identiques ou très similaires à plusieurs endroits de votre projet. Par exemple, toutes les propriétés peuvent être capturés, donc la capture ne doit être codée qu'une seule fois dans votre projet et non indépendamment pour les villes, QG et usines. Les projets répondant bien à cette exigence seront particulièrement valorisés par la notation.

Lorsque votre enseignant de TP vous a donné le feu vert, vous pouvez vous lancer dans l'implémentation de votre projet. Prenez alors le temps de comprendre les fonctionnalités présentes dans le squelette qui vous est fourni, expliquées dans la partie 4.

4 Squelette fourni

Nous vous fournissons un squelette qui vous sera utile pour vous lancer dans le projet. Ce squelette est commenté ; la documentation du squelette est disponible ici. Le squelette est commenté selon la norme Javadoc et la documentation du lien ci-dessus a été générée automatiquement par Java. Pour information, il est possible d'obtenir la documentation d'un tel projet sous Eclipse en cliquant sur le projet puis en allant sur «Project → Generate Javadoc ».

4.1 Fichiers cartes

Les fichiers de la forme *.awdcmap* codent les cartes du jeu. Quelques exemples de cartes vous sont déjà fourni. Au lancement du jeu, il sera demandé de choisir la carte de jeu parmi celles figurant dans le dossier *maps*. Si

vous souhaitez créer de nouvelles cartes, vous pouvez regarder et essayer de comprendre le fonctionnement des cartes fournies ; nous vous conseillons cependant de ne pas passer trop de temps à cela.

Un parseur vous est fourni qui transforme les fichiers ".awdcmap" en un tableau de `String` ; à vous derrière de travailler avec ce tableau à l'aide de la méthode `.split()`.

4.2 Images du jeu

Un certain nombre d'images provenant du jeu vous sont fournis dans le dossier *pictures*. Vous y trouverez les images des unités et des terrains. La classe **Chemins**, expliquée ci-dessous, vous permet d'obtenir facilement les chemins des images.

4.3 Package *librairies*

Le package *librairies* n'est pas censé être modifié. Les classes qu'il contient sont utilisables telles quelles.

La classe **AssociationTouches** sert à lire les appuis de touches du clavier. Vous serez amener à utiliser cette classe pour interagir avec l'utilisateur. En particulier, la fonction `trouveProchaineEntree()` permet de trouver la prochaine entrée de l'utilisateur.

La classe **FiltreDeFichier** sert à forcer la sélection de fichier ".awdcmap" lors du lancement du jeu et est utilisée dans la classe **Main**, vous n'avez pas besoin d'utiliser cette classe directement.

La classe **SelecteurDeFichier** sert à ouvrir la fenêtre de sélection du fichier ".awdcmap" à charger. Là aussi, vous n'avez pas besoin d'utiliser cette classe directement.

La classe **StdDraw** sert à l'affichage (comme au premier TP). Avant d'utiliser directement **StdDraw**, vérifiez que l'affichage que vous voulez faire ne pas être réalisé directement avec la classe **Affichage** décrite ci-dessous.

4.4 Package ressources

Le package **ressources** contient plusieurs méthodes pour vous assister dans le projet. Vous pouvez modifier ces classes-là si vous en avez besoin ; pensez juste à expliquer vos modifications dans votre fichier *readme.txt*.

La classe **Affichage** vous donne accès à des méthodes d'affichage reposant sur la bibliothèque **StdDraw**, par exemple pour afficher une image dans une case, ou dans une partie d'une case, ou la grille de jeu toute entière. Nous vous conseillons de lire cette classe avant de commencer afin de ne pas recoder des fonctions d'affichage déjà fournies.

La classe **Chemins** contient les chemins d'accès vers les différents fichiers, comme les images fournies ou les fichiers ".awdcmap". Par exemple, `Chemins.getCheminTerrain(Chemins.FICHIER_FORET)` renvoie le chemin menant à l'image d'une case forêt. Si vous voulez ajouter vos propres images, par exemple si vous ajoutez des unités qui ne sont pas dans le sujet, vous devrez donner le chemin d'accès aux nouvelles images ici.

La classe **Config** contient les paramètres de configuration du jeu. Vous pouvez notamment modifier cette classe si vous souhaitez changer quelque chose aux grandeurs d'affichage. Si vous souhaitez rajouter des touches reconnues par le jeu, il suffit d'ajouter le caractère au tableau `TOUCHES_PERTINENTES_CARACTERES`, et vous pouvez tester si le contrôle de l'utilisateur vaut ce caractère avec le test `isCaractere(c)`.

La classe **ParseurCartes** permet de lire les fichiers ".awdcmap". Si vous voulez ajouter de nouveaux types de terrains ou d'unités, vous devez ajouter leurs encodages dans ce parseur. Sinon, vous n'avez pas besoin de comprendre ce parseur en détail. Vous allez néanmoins devoir travailler avec le résultat du parseur, qui est un tableau de `String`. Chaque `String` représentant le contenu de la case, c'est à dire un terrain et éventuellement une unité. Par exemple la `String` `‘‘Ville:0;Tank:1’’` signifie que cette case a comme terrain une ville neutre (0) et un tank appartenant au premier joueur (joueur rouge). Une méthode qui pourra vous être utile est la méthode `split` de la classe `String` : par exemple, si `s = "Plaine;Tank:1"`, alors `s.split(";")` renverra le tableau `{"Plaine", "Tank:1"}`.

5 Notation

Le projet est à réaliser en binôme. Il sera à déposer sur Moodle pour début janvier (cf. Moodle pour la date exacte). Vous devez déposer un dossier compressé *.zip* avec :

• Le **code source** de votre jeu (ainsi que les images), en tant que projet Eclipse. Vos évaluateurs utiliseront la fonctionnalité "File → Import... → Existing project into workspace" de Eclipse pour charger votre projet. Si ça ne marche pas car vous avez rendu sous le format d'un autre IDE, vous serez fortement pénalisés. Avant de rendre votre projet, faites vous-même la démarche de l'importer dans Eclipse afin de vérifier que tout fonctionne correctement.

Remarque : *Cela ne vous empêche pas d'utiliser votre IDE préféré, et il se peut que ce ne soit pas Eclipse. Vous devez juste réserver un peu de temps à la fin pour préparer un projet Eclipse pour le rendu final.*

• Un fichier **README.txt**, donnant vos noms et une description de votre jeu : ce que vous avez fait, les commandes de votre jeu, les bonus que vous avez ajoutés. Écrivez ici tout ce à quoi vous voulez que votre évaluateur fasse attention. Si plusieurs binômes rendent des projets *trop* similaires, tous les participants de la triche auront la note de **0/20** (cette note compte pour $\frac{1}{4}$ de votre moyenne totale de PO). Vous aurez la note de **14/20** si vous implémentez un **jeu de base** correspondant aux règles indiquées ci-dessus (cf. partie 2).

• Le jeu doit être totalement fonctionnel (aucune erreur). Il n'a pas besoin d'être joli, mais doit être jouable et simple à utiliser (Testez le vous-même ou faites-le tester par vos amis). Le code doit présenter une conception objet propre et des commentaires Javadoc pour les classes et méthodes principales. L'implémentation doit impérativement utiliser des collections Java.

• Si vous désirez avoir plus de 14/20, vous pouvez gagner des points bonus en améliorant le jeu de base. Laissez parler votre créativité ! En sachant que n'étant pas une formation artistique, nous récompenserons mieux les améliorations démontrant votre maîtrise de la programmation objet que votre capacité à coller de jolies images.

6 Suggestions d'améliorations

Voici des idées d'améliorations, certaines étant plus dures que d'autres. **Vous n'êtes pas limités aux améliorations citées ci-dessous :** toute amélioration sera valorisée, vous pouvez par exemple vous inspirer du jeu original ou simplement faire parler votre créativité! Il sera bienvenu de fournir avec vos améliorations des exemples illustratifs (typiquement sous la forme de nouvelles cartes ".awdcmap") quand cela est pertinent, . Il n'est pas du tout nécessaire de réaliser toutes les améliorations ci-dessous pour avoir la note maximale! Le barème fourni n'est qu'indicatif et peut changer sans préavis.

6.1 Touche pour lister les unités encore utilisables (1 point)

Ajouter une touche pour placer le curseur sur une unité du joueur courant qui est toujours prête à agir ce tour-ci. Si le joueur courant a N unités encore prêtes à agir ce tour-ci, chaque appui de cette touche placera le curseur sur une unité différente parmi ce groupe de N. Si le joueur appuie sur cette touche alors qu'aucune de ses unités n'est prête à agir ce tour-ci, le curseur n'est pas déplacé, un signal sonore ou un message avertissant le joueur que toute ses unités ont agit ce tour-ci apparaît.

6.2 Affichage des déplacements et attaques possibles d'une unité (2 point)

Lors de la sélection d'une unité, afficher vers quelles cases elle peut se déplacer. Ajouter une touche pour afficher quelles unités une unité peut attaquer (avec ou sans déplacement).

6.3 Armes multiples (1,5 point)

Dans la partie de base du projet, les unités n'avaient qu'une seule arme. Dans le jeu d'origine, une unité peut en réalité avoir plusieurs armes. Lorsqu'elle attaque ou riposte contre une autre unité, elle le fait avec l'arme qui infligerait le plus de dégâts (si elle a au moins une arme qui peut attaquer l'autre unité).

Par exemple, le tank aura maintenant une mitrailleuse légère en plus de son canon. Si un tank attaque un autre tank, il peut l'attaquer avec le canon ou la mitrailleuse ; le canon étant l'arme qui fait le plus de dégâts au tank, c'est celle qui sera utilisée. Néanmoins, si le tank attaque un hélicoptère, qui sera à présent inattaquable par le canon, il utilisera sa mitrailleuse. Le tank et le bazooka ont à présent une mitrailleuse légère en plus de leur canon, l'hélicoptère a à présent une mitrailleuse lourde en plus de ses missiles (qui deviennent des missiles exclusivement air-sol)

Les armes sont redéfinies selon le tableau 9.

Arme	Infan.	Bazoo.	Tank	DCA	Hélico.	Bomba.	Convo.
Mitrailleuse légère	60 %	55 %	15 %	10 %	30 %	-	40 %
Canon	-	-	55 %	60 %	-	-	70 %
Mitrailleuse lourde	100 %	80 %	30 %	30 %	110 %	70 %	50 %
Missiles Air-Sol	-	-	70 %	40 %	-	-	70 %
Bombes	100 %	100 %	100 %	70 %	-	-	100 %

Figure 9: Liste des armes lorsque les unités peuvent avoir plusieurs armes

6.4 Attaques à distances (2,5 point)

Les attaques à distance ne sont possibles que pour les unités qui possèdent une arme à distance. L'arme à distance à implémenter en premier est le mortier, qui sera utilisé par l'artillerie, une unité également à implémenter. L'attaque à distance n'est possible que si l'unité ne s'est pas déplacée lors de ce tour : pour attaquer à distance, le joueur doit donc sélectionner l'unité qui va attaquer, sélectionner le déplacement nul (l'unité reste au même endroit) puis choisir l'option *Attaque à distance*. Il n'y a pas de riposte à une attaque à distance.

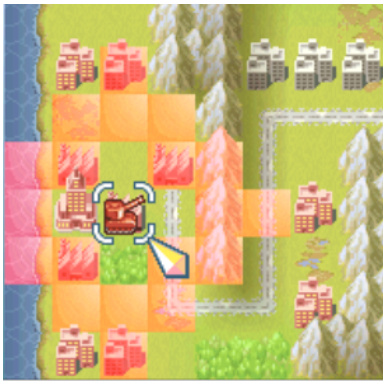

Figure 10: Portée du mortier, en rouge

Artillerie L'artillerie est un véhicule à chenilles avec 5 points de déplacement, qui coûte 6 000 crédits à déployer. Toutes les armes qui peuvent tirer sur un convoi peuvent également tirer sur une artillerie, avec exactement la même efficacité (cf. figure 2). L'artillerie possède une seule arme : le mortier.

Mortier Le mortier est efficace à 70 % contre les autres artilleries, les convois, les DCAs et les tanks, à 50 % contre les bazookas et à 40 % contre les infanteries. Il n'est pas possible de tirer au mortier sur les unités aériennes. Seul l'artillerie est équipée de mortiers. Le mortier a une portée minimum de 2 cases, et une portée maximum de 3 cases : il ne peut pas attaquer les unités qui lui sont adjacentes, mais celles à une distance de 2 ou de 3 cases. Par conséquent, l'artillerie ne peut pas riposter quand elle se fait attaquer par une attaque directe (par un tank, par exemple). La figure 10 illustre la portée d'attaque d'une artillerie (en rouge).

Alternative : Lance-missile Sol-Air Alternativement (ou en plus), vous pouvez implémenter le lance-missile Sol-Air, qui est exactement comme l'artillerie mais dont les coût est de 12 000 crédits. Au lieu d'un mortier, il lance des missile Sol-Air dont la portée minimum est de 3 cases et la portée maximum de 6 cases. Ses missiles ne peuvent atteindre que les unités aériennes, contre lesquelles elles sont efficaces à 120 %. La figure 11 illustre la portée d'attaque des missiles Sol-Air (en rouge).

Figure 11: Portée du lance-missile Sol-Air indiquée en rouge

6.5 Ravitaillement et réparations (3 points)

Les armes consomment des munitions à chaque utilisation (que ce soit en temps qu'attaquant ou défenseur). Une arme à court de munitions n'est plus utilisable. On considère que les mitrailleuses ont des munitions à l'infinie.

La majorité des moyens de locomotions nécessitent du carburant. Pour ces unités, chaque point de déplacement utilisé coûte un point de carburant (par exemple, il faut deux points de carburant à un tank pour entrer dans une case de forêt, puisque cela lui coûte deux points de déplacement). Les unités aériennes consomment un montant fixe d'essence par jour (2 pour un hélicoptère, 5 pour un bombardier). Une unité ne peut pas se déplacer sans carburant, une unité aérienne sans essence est détruite au début de son tour.

Les unités à pied n'utilisent pas d'essence pour se déplacer. Vous êtes libres de choisir le nombre maximum de munition de chaque arme et la quantité maximum de carburant de chaque unité. (En général, une unité a entre 50 et 90 points de carburant, et une arme a entre 3 et 9 munitions).

Vous pourrez faire apparaître un indicateur sur les unités dont le niveau d'essence ou de munitions est bas (par exemple inférieur à un tiers de la capacité maximum).

Les unités récupèrent toutes leurs munitions et carburant lorsqu'elles commencent le tour sur la case d'une propriété alliée ou sur une case adjacente à un convoi allié. Une unité est également réparée de jusqu'à 2PV si elle commence le tour sur une propriété alliée et que le joueur a assez de crédits pour payer le coût de la réparation (proportionnel au coût de l'unité et au nombre de PVs à réparer).

Le convoi peut, après son déplacement, choisir l'action *Ravitailler* qui rend immédiatement toutes leurs munitions et carburant aux unités alliées adjacentes

6.6 Transport de unités à pied (1 point)

Certains véhicules ont la capacité de transporter les unités à pied. Les unités à pied peuvent monter à bord de véhicule de transports. Un véhicule de transport peut déposer l'unité qu'il contient vers une case vide adjacente traversable par une unité à pied. Le convoi et l'hélicoptère sont des véhicules de transport, au travers de nouvelles actions *Monter à bord* et *Déposer*.

6.7 Fin de tour automatique (0,5 point)

Un joueur peut activer ou désactiver l'option *Fin de tour automatique* à n'importe quel moment où il est le joueur actif. Lorsqu'un joueur qui a choisi d'activer cette option est le joueur actif et n'a plus aucune action possible, la fin de son tour est automatiquement déclenchée.

6.8 Couverture de terrain (1 point)

Les unités peuvent avoir une meilleurs défense sur certains types de terrains : Une unité subie des dégâts réduits de 20 % si elle se trouve sur une case de forêts d'une usine, de 30 % en ville, et de 40 % en montagne ou sur un QG. Les unités aériennes ne profitent pas de la couverture de terrain.

6.9 Unités navales (3 points)

Les unités navales se déplacent uniquement sur l'eau ou les propriétés. Elles ne peuvent être produites que par des usines à proximité de l'eau.

Unités navales basiques (1 point) :

- **Croiseur** : Equipé d'un canon et d'une mitrailleuse lourde. Se déplace de 6 cases en mer.
- **Corvette** : Equipé d'un canon. Se déplace de 7 cases, peut transporter une unité à pied.
- **Barge** : Se déplace de 6 cases, peut transporter deux unités terrestres. Ajouter le type de terrain "Plage" qui est traversable par les unités terrestres et par les barges. Le coût pour traverser cette case est de 1 pour les unités à pied, et de 2 pour les barges et les véhicules à chenilles.

Unités navales avancées (2 points) :

- **Porte avion** : Equipé d'une mitrailleuse lourde. Se déplace de 4 cases. Peut transporter deux unités aériennes. Peut produire une unité aérienne qui sera placé dans son espace de transport (s'il lui reste de la place).
- **Cuirasser** : Equipé d'une arme similaire au mortier, mais avec une portée minimum de 3 cases et maximum de 5 cases. Se déplace de 4 cases. Peut se déplacer et attaquer le même tour.
- **Sous-marin** : Equipé de torpilles ayant une efficacité de 55 % contre les croiseurs et les autres sous-marins et de 100 % contre toutes les autres unités navales. Se déplace de 6 cases. Le sous-marin a deux actions supplémentaires : "Plonger" et "Faire surface". Il ne peut plonger que lorsqu'il est à la surface, et il ne peut faire surface que lorsqu'il est en plongé. Tant que le sous-marin est en plongé, sa consommation de carburant est doublée, mais il est invisible à moins qu'une unité ennemie soit adjacente. Tant qu'il est en plongé, il ne peut être attaqué que par d'autres sous-marins ou par des croiseurs.

6.10 Brouillard de guerre (Difficile : 4 points))

Brouillard de guerre basique (2 points) :

- La carte est couverte de brouillard de guerre. Le joueur voit le terrain sous le brouillard de guerre, mais ne voit pas les unités qui s'y trouvent, il ne peut pas non plus voir qui contrôle les propriété se trouvant sous le brouillard de guerre (elles apparaissent neutres, à l'exception des QG dont l'appartenance est toujours connue).
- Les propriétés et les unités au sol ont une vision de portée 2, les unités aériennes et les unités en haut d'une montagne ont une vision de portée 4.
- Lorsqu'une case du terrain est révélée à un joueur, elle reste révélée jusqu'à la fin du tour de ce joueur. Lorsque le tour d'un joueur commence, toutes les unités et propriétés qu'il contrôle ou qui sont contrôlées par un allié lui révèlent les cases de terrain qui l'entourent avec une portée égale à leur score de vision. Lorsqu'une unité se déplace, elle révèle le terrain autour d'elle de manière similaire à chaque case qu'elle traverse ainsi qu'à la case où elle termine son déplacement. Si son déplacement devait passer par une case occupée par une unité ennemie, le déplacement prend fin immédiatement avant, un message "Piégé" apparaît à l'écran pendant 1 seconde, et la unité piégée ne peut plus agir jusqu'au tour suivant. Il n'est pas possible d'attaquer une unité ne se trouvant pas sur une case révélée.

Brouillard de guerre avancé (2 points) :

- Certains terrains sont propices aux embuscades : Les forêts, les propriétés et les ruines. Ces cases ne sont révélées que par une unité adjacente, ou si elles sont contrôlées par le joueur dans le cas des propriétés. La ruine est un nouveau type de terrain, qui est aussi simple à traverser qu'une plaine, et qui offre 10 % de couverture de terrain.

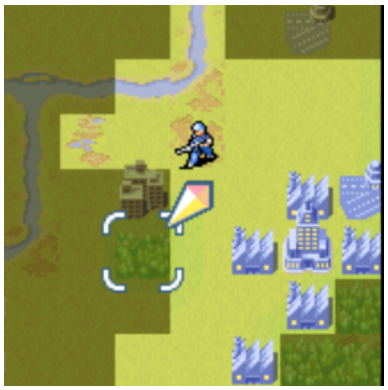

Figure 12: Jeu avec brouillard de guerre

- **Nouvelle unité :** le lance fusée-éclairante peut lancer une fusée qui révèle le terrain sur la case sélectionnée ainsi que sur les cases adjacentes. La case sélectionné ne peut pas être au-delà d'une portée de 5 du lance fusée. La zone sera également révélée aux autres joueurs pendant leur prochain tour. L'effet de la fusée éclairante cesse juste avant le début du prochain tour du joueur qui l'a déclenché. L'effet de la fusée éclairante révèle également les cases de terrains propices aux embuscades (forêts, ruines, propriétés). "Lancer une fusée" est une action. Elle ne peut pas être réalisée après un déplacement (comme les attaques à distance).

6.11 Météo (2 point)

Il y a 4 types de météo : *Beau temps (météo par défaut), Pluie (si le brouillard de guerre est implémenté), Neige, Vent violent.*

- Le beau temps n'a aucun effet.
- La pluie réduit la vision des unités au sol et des propriétés d'une case, et celle des unités aériennes et unité en montagne de deux cases.
- La neige affecte la vitesse des unités. La table 13 définie les déplacements des unités aux sols lorsqu'il neige. La vitesse de déplacement des unités aériennes lorsqu'il neige est réduite d'un tiers, arrondie à l'inférieur. Les navires ne sont pas affectés par la neige.

- Les vents violents réduisent la portée des attaques à distance d'une case, et réduisent leurs dégâts de 20 %.
- Le climat est configuré au début de la partie, soit sur l'un des 4 types de météo, soit en mode aléatoire, ce qui aura pour effet de changer la météo de manière aléatoire au cours de la partie.
- Les joueurs sont avertis un tour en avance d'un changement de météo à venir en mode climat aléatoire.

	Moyen de locomotion	
Terrain	**À pied**	**Sur chenilles**
Propriété	1	1
Plaine	1	2
Foret	2	3
Montagne	Inaccessible	Inaccessible
Eau	Inaccessible	Inaccessible

Figure 13: Coût de déplacement au sol lorsqu'il neige

6.12 Commandants (Difficile : jusqu'à 4 points)

Chaque joueur peut choisir un commandant en début de partie, il garde ce commandant pour toute la partie.

Pendant la partie, le joueur peut faire monter le commandant à bord de l'une de ses unités. Pour cela, l'unité doit terminer son déplacement sur le QG du joueur, puis prendre l'action "Commandement". Le commandant prend alors la tête de cette unité, ce qui donne des bonus à cette unités et à toutes les unités alliées proches. La nature de ces bonus et la taille de la zone où ils sont donnés dépendent du commandant. En général la zone a une portée allant de 1 à 5 cases. Les bonus peuvent par exemple être sur les dégâts infligés, le déplacement, la résistance, la régénération de PV, etc. Les commandants peuvent avoir des spécialités (par exemple un commandant pourrait booster uniquement les unités aériennes, ou uniquement les unités qui attaque à distance, etc.). Des projets implémentant des commandants avec des propriétés plus complexes seront valorisés (n'ayez pas peur d'innover sur les propriétés offertes par les commandants!).

Si l'unité du commandant est détruit, le commandant bat en retraite, il pourra monter à bord d'une autre unité à partir du tour suivant. Le commandant ne peut être à bord que d'une seule unité à la fois.

7 Liens utiles

- Lien vers le wiki du jeu : Wiki Advance Wars Dark Conflict
- Lien vers la javadoc du squelette.
- Lien vers le site contenant les sprites (images) du jeu Spriters Ressource.
- Lien vers le simulateur du jeu : Simulateur Advance Wars Dark Conflict

8 Contacts

Pour toute question et information complémentaire, n'hésitez pas à contacter les auteurs de ce projet :

- ***Nicolas WALDBURGER :*** nicolas.waldburger@irisa.fr
- ***Jean-Loup HATCHIKIAN HOUDOT :*** jean-loup.hatchikian-houdot@inria.fr
- ***Hasnaa OUADOUDI BELABZIOUI :*** hasnaa.ouadoudi-belazioui@ens-rennes.fr