LO21 - Automne 2025 - Projet Akropolis

Dans ce projet, il s'agit de concevoir et développer une application permettant de jouer au jeu de société AKROPOLIS créé par Jules Messaud, et édité par *Gigamic*. Dans ce jeu de pose de tuiles, les joueurs prennent le rôle d'architectes qui s'affrontent en créant chacun une cité à l'aide de « *tuiles cité* ». Chaque tuile cité est composé de 3 « *hexagones construction* », chaque hexagone représentant un « *quartier* », une « *place* » ou une « *carrière* ». Il existe plusieurs types de quartier. Chaque quartier rapporte des points de victoire s'ils sont correctement placés selon sont type :

- les *habitations* (quartiers bleus) : seules les habitations faisant partie du plus grand groupe d'habitations adjacentes font gagner des points ;
- les *marchés* (quartiers jaunes) : ils ne doivent pas être adjacents à un autre marché pour rapporter des points ;
- les *casernes* (quartiers rouges) : elles doivent se trouver en périphérie de la cité du joueur pour rapporter des points ;
- les *temples* (quartiers violets) : ils doivent être complétement entourés pour raporter des points ;
- les *jardins* (tuiles vertes) : ils n'ont pas de contrainte de placement pour rapporter des points.

Les *carrières* ne rapportent pas de point lors du décompte final, mais permettent d'obtenir des « *pierres* » qui rapportent des points à la fin ou permettent d'acheter des facilités de choix des tuiles cité lors de son tour. Quand un Architecte recouvre une carrière à l'aide d'une nouvelle tuile, il prend une pierre de la réserve.

Les *places* sont des multiplicateurs, qui augmentent le nombre de points rapportés par les quartiers de même type. Le multiplicateur est représenté par des *étoiles*. Si un joueur possède plusieurs places de même type, leurs étoiles sont cumulables. Il n'est pas nécessaire qu'une place soit adjacente aux quartiers de même type.

À chaque tour, les joueurs vont choisir une tuile cité selon des contraintes données et l'ajouter à leur cité de manière adjacente aux autres. Une tuile cité peut être posée au niveau du sol (niveau 1) ou la superposer sur des tuiles existantes (niveaux supérieurs).

Le jeu se déroule en plusieurs tours jusqu'à épuisement des tuiles cité dont le nombre dépend du nombre de joueurs. Le joueur qui possède le plus de point de victoire à la fin de la partie l'emporte.

Ce jeu se joue de 2 à 4 joueurs mais propose un mode solo avec des règles précises simulant un adversaire virtuel appelé « *Illustre Constructeur* ». Trois niveaux de difficulté vous sont proposés, faisant varier la façon dont les points adverses sont comptabilisés

Le jeu propose quelques variantes qui permettent de corser le jeu. Dans ce cas, la mise en place et le déroulé de la partie ne changent pas. Chaque variante propose une condition de placement supplémentaire pour rapporter des points. Il est possible de jouer avec plusieurs variantes dans la même partie.

Pour découvrir ce jeu, vous pouvez consulter les règles en ligne et la façon de jouer en solo. Il est conseillé de jouer quelques parties pour bien comprendre les différents mécanismes du jeu. Vous pouvez apprendre et jouer en ligne sur le site Board Game Arena. Vous pouvez aussi regarder les nombreux tutoriels vidéos sur ce jeu.

1 Fonctionnalités attendues

L'application devra permettre de jouer des parties de 1 à 4 joueurs en utilisant éventuellement une ou plusieurs variantes proposées dans les règles.

2 Éléments d'interface

Lors de son ouverture, l'application doit permettre de paramétrer une partie :

- reprise ou abandon d'une éventuelle partie en cours ;
- nombre et nom(s) du ou des joueurs ;
- choix du niveau de difficulté de l'Illustre Constructeur si mode solo ;
- nombre de tuiles cité (standard ou augmenté) ;
- activation d'une ou plusieurs variantes.

L'application doit permettre de jouer la partie en fonction du mode (solo ou multijoueurs), de vérifier la validité des actions des joueurs (choix des tuiles cité, utilisation des pierres, placement des tuiles cité), de déterminer la fin de la partie, de compter les points et de déterminer le gagnant.
Tant qu'un joueur ne confirme pas un placement définitif, il peut revenir en arrière (finalement choisir une autre tuile cité ou un autre emplacement).
Il doit être possible de jouer soit en mode console, soit en mode graphique (avec une interface `Qt`).
Vous êtes libre d'organiser votre interface du moment qu'elle permet de piloter facilement l'application. Les visuels doivent être suffisants pour comprendre l'état du jeu. Il n'est pas demandé d'utiliser les graphismes originaux du jeu (ce n'est pas la priorité).

3 Evolution de l'application

L'architecture de votre application devra permettre d'intégrer de nouveaux composants. Les choix de conception devront donc permettre de rendre l'application évolutive sans impacter le reste du programme. Ils devront notamment garantir la facilité d'ajout des composants suivants :

- nouvelles tuiles cité ;
- nouvelle variante officielle ou inventée ;
- nouveau niveau de difficculté en mode solo.

Vous démontrerez la pertinence de votre architecture dans votre rapport en expliquant comment ajouter ces éléments (classes à créer, intégration dans l'architecture, éventuelles modifications de code, ...).

4 Consignes

- Le projet est à effectuer en groupe de 4 ou 5 étudiants (du même groupe de TD) ;
- vous êtes libres de réutiliser et modifier du code déjà élaboré en TD pour l'adapter à votre architecture ;
- en plus des instructions standards du `C++/C++11/C++14/C++17/C++20`, vous pouvez utiliser l'ensemble des bibliothèques standards du `C++11/C++14/C++17/C++20/C++24` ;
- l'interface graphique est à réaliser en utilisant le framework `Qt`.

5 Livrables attendus

5.1 Rapports intermédiaires

Pour mener ce projet, vous devrez vous organiser de manière efficace. Il sera donc important de tenter de réfléchir aux tâches que vous allez devoir réaliser en vous posant pour chacune d'elles les questions

suivantes :

- Cette tâche peut-elle être encore découpée en sous-tâches ?
- Quelle est la complexité de la tâche ?
- Quelle est la durée estimée pour réaliser cette tâche ?
- Quelles sont les relations de dépendance entre cette tâche et les autres ?

Vous devrez ensuite prioriser vos tâches (indispensable, importante, utile, bonus), les répartir entre les différents membres du groupe de projet et construire des rétros-plannings de manière à organiser votre projet en fonction des livrables attendus.

Au cours du projet, il vous est demandé de rendre compte de votre organisation. Vous devrez ainsi rendre 3 comptes rendus (10 pages maximum) lors des semaines suivantes :

1. la semaine du 22 septembre ;
2. la semaine du 4 novembre ;
3. la semaine du 1er décembre.

Chaque rapport comportera :

- la liste des tâches mises à jour (qui seront des tâches *a priori* macros pour le rapport 1) avec l'estimation de leur durée (en heures de travail) en mettant en évidence les nouvelles tâches par rapport au précédent compte-rendu (rapports 2 et 3) : ces estimations et leurs mises à jour sont obligatoires ;
- l'affectation (répartition *a priori*) des tâches (restantes) aux différents membres du groupe ; indiquez clairement les changements d'affectation des tâches et leurs raisons ;
- pour les comptes rendus 2 et 3, l'état d'avancement par rapport au précédent compte rendu : la répartition entre les membres du groupe et la durée *a posteriori* des tâches déjà effectuées, ou un pourcentage d'état d'avancement pour les tâches en cours avec la durée de travail déjà fournie par chacun des membres de groupe. Vous êtes particulièrement attendu sur ce retour qui devra être complet à chaque fois.
- un bilan sur la cohésion de groupe et l'implication de chacun.

La liste des tâches s'affinera au cours du temps. Vous serez précis sur les tâches à court terme (jusqu'au prochain rendu).

1. Le premier compte-rendu proposera une première analyse des différents concepts qui apparaissent dans le jeu et qui devraient *a priori* apparaître *a minima* dans votre architecture. À ce stade, il n'est pas demandé d'expliciter toutes les relations qui existent entre les différents modules et il n'est pas demandé d'élaborer cette architecture. Il n'est pas non plus demandé de proposer un modèle qui intègre des éléments d'implémentation. Il s'agit d'élaborer un modèle conceptuel qui exige néanmoins d'avoir bien compris le jeu et ses constituants (il faut savoir jouer, il faut donc jouer !).
2. Le deuxième compte-rendu proposera un début d'architecture qui intègre déjà des modules liés à l'implémentation. Il faudra reporter les associations principales ainsi que les prémisses des hiérarchies de classes existantes dans votre système. Cette architecture ne représentera qu'un état intermédiaire et pourra encore largement évoluer par la suite en fonction de votre maîtrise des concepts orientés objet.
3. Le troisième compte-rendu rapportera l'architecture courante en indiquant les éventuelles modifications depuis le deuxième compte rendu. Il détaillera de manière complète le ou les modules qui permettent de jouer et les interactions que ce ou ces modules ont avec le reste de l'application. À ce stade du projet, il est attendu que la plupart des modules soient fonctionnels. Vous devez donc en faire la preuve. Il devrait normalement être possible de jouer en mode console.

Les rapports devront être déposés sur moodle dans la partie prévue à cet effet avant la fin de la semaine mentionnée (avant samedi 23h59 partout dans le monde).

Vous aurez peu ou pas de retour sur ces rapports intermédiaires, mais il sera tenu compte de la qualité de ces comptes-rendus dans la notation (le manque de précision dans les éléments demandés sera pénalisé).

5.2 Livrable final

Le livrable final est composé des éléments suivants :

- **le code source**, *i.e.* l'ensemble du code source du projet. Attention, **ne pas fournir d'excutable ou de fichier objet** ;
- **une vidéo de présentation avec commentaires audio**, *i.e.* une video de présentation dans laquelle vous filmerez et commenterez votre application afin de démontrer le bon fonctionnement de chaque fonctionnalité attendue (max 10 min, 99 Mo) ;
- **un rapport en format `.pdf`** (de 20 à 25 pages) composé des parties suivantes :
  - une synthèse de ce que permet votre application (en précisant parmi les opérations attendues celles qui ont été implémentées et celles qui ne l'ont pas été) ;
  - la description précise de votre architecture en justifiant vos choix ;
  - une argumentation détaillée où vous montrerez que votre architecture permet facilement des évolutions (voir explications ci-dessus). Il est attendu de présenter des diagrammes UML qui illustrent correctement (et au bon niveau de détails) les sous-parties que vous décrivez ;
  - une description détaillée du planning effectif de votre projet ;
  - une description détaillée de la contribution personnelle de chacun des membres du groupe sur les différents livrables (cette partie sera notamment utilisée pour la notation). Vous évaluerez en % la part de contribution de chaque membre **sur l'ensemble du projet**. Chaque membre devra aussi reporter une évaluation du nombre d'heures de travail qu'il a consacré au projet.

  Pour ce rapport, vous pouvez réutiliser sans limite les parties pertinentes qui ont déjà été rédigées dans les compte-rendus pour les compléter.

L'ensemble des livrables est **à rendre avant le 21 décembre 23h59 au plus tard (partout dans le monde)**. Les éléments du livrable devront être déposés sur moodle dans la partie prévue à cet effet.

6 Responsabilités des livrables

Tous les membres du groupe doivent participer à chacun des livrables en partageant le travail de manière équitable et en fournissant tous les éléments dont ils ont la charge. Un des membres du groupe sera désigné comme auteur responsable (au niveau du rendu) d'un des 4 rapports (comptes rendus intermédiaires et rapport final) et de la vidéo. Chacun de ces rendus devra avoir un auteur responsable différent. La répartition de la responsabilité des livrables devra être reportée dans le rapport final.

7 Évaluation

Barême de l'évaluation du projet sur 20 :

- **couverture des fonctionnalités demandées au moins en mode console** : 5 points ;
- **intégration dans une application GUI Qt** : 5 points ;
- **choix de conception et architecture** : 5 points ; en particulier sera évaluée la capacité de l'architecture à s'adapter aux changements (extensions, nouvelles IA) ;
- **évaluation des livrables** : 5 points (video, code source, rapport, compte rendus intermédiaires, respect des consignes).

Remarque : une note inférieure ou égale à 9/20 au projet est éliminatoire pour l'obtention de l'UV. **Il est conseillé de consulter la grille critériée (disponible sur moodle), partie projet, qui est utilisée pour évaluer votre projet. Elle précise les attentes sur plusieurs aspects.**

La note finale attribuée est individuelle : elle dépend du rendu global et des contributions de chacun. Un membre du groupe inactif peut obtenir une mauvaise évaluation (qui peut être éliminatoire sur l'évaluation globale de l'UV) sur un projet considéré comme excellent pour les autres membres du groupe.

8 Conseils

Vous devriez commencer à sérieusement réfléchir au projet dès le début du semestre, les premiers jours servant à découvrir le jeu et apprendre à jouer.

Le temps de travail attendu est de l'ordre de 5 à 6h de travail en moyenne par semaine et par membre sur chacune des 8-10 semaines actives du projet. Il est donc important d'avancer régulièrement et de ne surtout pas attendre les 2-3 dernières semaines.

Il est fortement recommandé d'utiliser un logiciel de gestion de versions afin de faciliter le travail collaboratif.

Plusieurs TDs utilisent certains concepts plus ou moins communs au projet afin de commencer à vous familiariser avec les différentes entités de l'application qui est à développer. On ne perdra pas de vue que les questions développées dans ces TDs ne constituent pas une architecture forcément pertinente pour le projet.

La partie difficile du projet est la conception de votre architecture : c'est là-dessus qu'il faut concentrer vos efforts et passer le plus de temps au départ. Une des difficultés est l'acquisition des concepts orientés objets au fur et à mesure du semestre qui remettra certainement en cause votre architecture. C'est normal.

Il est conseillé d'étudier au moins les *design patterns* suivants qui pourraient être utiles pour élaborer l'architecture de votre projet : decorator, factory, abstract factory, builder, bridge, composite, iterator, template method, adapter, visitor, strategy, facade, memento. En plus de vous donner des idées de conception, cette étude vous permettra de vous approprier les principaux modèles de conception. Attention, cela ne signifie pas qu'ils doivent forcément tous être utilisés.

L'apparence de l'application ne sera pas prise en compte dans la notation. Soyez avant tout fonctionnels. Cela peut être très moche du moment que c'est jouable. En particulier le design des différents éléments visuels n'ont pas à être esthétiques : il faut simplement que l'on puisse jouer.