Rapport projet IHM

Le groupe a décidé d'implémenter le jeu du UNO. Nous avons gardé les mêmes modules obligatoires, et nous avons spécifié dès le début que l'on ajouterait un module réseau, différents types d'IA et une traduction en Anglais.

Nous avons répartit chaque « gros » module pouvant se faire en parallèle par groupe de 2.

Module	Equipe	Langage
Noyau	Nicolas Vergnes Adrien Fleith	C++
Interface	Nicolas Hoerter Nathan Roth	QML , C++
Réseau	Constantin Divriotis Govindaraj Vetrivel	Python , C++

Vous trouverez en annexe le diagramme de PERT/CPM (figure 1) associé à notre projet. Nous avons essayé de respecter au mieux les délais, bien que ce ne fût pas tout le temps possible.

Le jalon au temps 28 constituait une fusion du noyau avec l'interface et le réseau. Celui-ci aurait dû être plus tôt, car il nous a pris beaucoup de temps et a nécessité beaucoup de modifications des modules.

Nous avons trouvé ce projet conséquent pour le peu de temps que nous avions (34 jours après l'élaboration des groupes). Si nous devions établir le périmètre du projet, nous avons beaucoup misé sur les délais, d'où le schéma suivant :

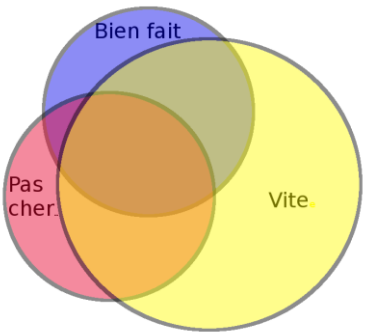

Figure 2 : Notre périmètre pour ce projet

Noyau

Le noyau se décompose en plusieurs classes que sont : Partie, Manche, Joueur, Pioche et Carte. Pour jouer il faut créer une Partie, dans laquelle on peut changer quelques paramètres, notamment le caractère aléatoire de la partie afin de reproduire la même partie au besoin. Afin d'évoluer dans le jeu, il faut récupérer les messages du noyau, qui indiquent si une partie/manche commence/se termine ainsi que si un joueur doit jouer, ou choisir une couleur. Tant qu'aucun joueur n'as remplit les conditions de victoires, une nouvelle manche est créée et les joueurs concourent pour la remporter. La partie contient une manche (quand il y en a une de créée), une Pioche (constituée des 108 cartes du jeu), ainsi que tous les joueurs participant à la Partie. Les interactions avec le noyau passent par la Partie pour réceptionner les messages, mais principalement par l'appel des méthodes des Joueurs en fonction de leurs choix d'actions.

A ce noyau s'ajoute une classe JoueurIA, qui hérite de la classe Joueur, et qui peut donc être substituée à un Joueur dans une Partie afin de jouer contre des Intelligences Artificielles. Seulement deux niveau d'IA ont été implémentés (simple et moyen), mais il est relativement aisé d'ajouter une nouvelle IA dans le jeu.

Pour certifier le noyau, des tests ont été effectués, mais certaines classes ou fonctionnalités n'ont pas eu le temps d'être testées.

Réseau

La programmation réseau en python est très simple proposant une bibliothèque de bas niveau. A la base, nous avons choisi Python car nous voulions associer le noyau (C++) et le serveur (Python), mais nous avons abandonné cette idée assez rapidement. Nous nous sommes aidés d'Internet pour coder le réseau. Les tests des programmes ont été effectués assez rapidement avec un client de debug en python puis en qt et ils se sont très bien déroulés, mais il a fallu l'appliquer par la suite à l'interface. Nous avons terminé ce module en moins de 20 jours (fixé initialement dans le diagramme) et nous n'avons rencontré presque aucune difficulté. La charge de travail a été mal anticipée, nous pensions qu'il y aurait beaucoup plus de travail sur le réseau. Nous voulions notamment faire tourner le noyau en réseau. De base, nous

voulions que les étudiants du cursus réseaux s'occupent du réseau, mais nous aurions pu en placer un sur une autre partie du projet. Malgré tout, nous avons dû résoudre des problèmes liés à la robustesse du serveur, étant donné que l'interface peut éventuellement imprévisible (crash, arrêt systèmes). C'est ainsi que la création d'un salon depuis l'interface peut être faite sans aucun problèmes, nous n'avons malheureusement pas pu mettre le jeu en réseau liée notamment au plateau de jeu qui nous a causé de très grand problèmes (cf interface). La partie de réseau étant faite souvent pendant les périodes de cours nous avons souvent travaillé ensemble ce qui explique la différence du nombre de commits.

Interface

Afin de réaliser l'interface, nous avons décidé d'utiliser le QML car celui permet une expérience utilisateur bien plus intéressante que la création de fenêtre Qt avec Widget. Ce choix a aussi été motivé par la mise à jour récente de Qt5.10 qui permet de nombreuses nouvelles fonctionnalités en QML. Mais ce choix nous a très vite mis à mal du fait du peu de tutoriel et exemple compatible avec les nouveaux standards, sans compter que même certains éléments de la documentation officielle n'étaient pas toujours à jour. L'un des problèmes majeurs sur lequel nous avons été confrontés fut la synchronisation de l'interface avec le code source de l'application, notamment sur la synchronisation des différentes listes (liste de serveurs, de joueurs dans un salon…). Ces problèmes nous ont donc fait perdre beaucoup de temps afin de nous former correctement aux mécanismes à utiliser. En l'état, l'application est capable d'enregistrer les paramètres de l'utilisateur, mais aussi de créer et visualiser les salons de jeux disponibles sur le serveur, la synchronisation des différentes variables de salon sont mise à jour en direct mais un bug lié au plateau de jeux nous a pris de court et donc empêché de finaliser celui-ci mais aussi sa mise en réseau.

NB : Dans le menu principal on peut voir une fonctionnalité nommée « Arrive bientôt », celle-ci devait être utilisée pour une extension d'une base de données permettant de stocker les statistiques du joueur ….

Nous reviendrons sur tous ces points lors de la soutenance.

Qualité

En termes de qualité, nous estimons les caractéristiques suivantes :

- Capacité fonctionnelle : NON
    - De nombreux bug (surtout au niveau du plateau) n'ont pas pu être résolus
- Fiabilité : NON
- Ergonomie : OUI
    - L'application est intuitive et facile à prendre en main
- Rendement : OUI
    - Le jeu accepte sans problème jusqu'à 4 joueurs en réseau.
- Maintenabilité : OUI
    - Conception modulaire, noyau indépendant
- Portabilité : NON
    - Nous ne pouvons pas jouer sur smartphone
- Compatibilité : OUI
    - Nous pouvons jouer en mode fenêtré avec d'autres tâches en parallèle.
- Sécurité : NON

Govindaraj Vetrivel

Constantin Divriotis

Nicolas Hoerter

Nicolas Vergnes

Nathan Roth

Adrien Fleith

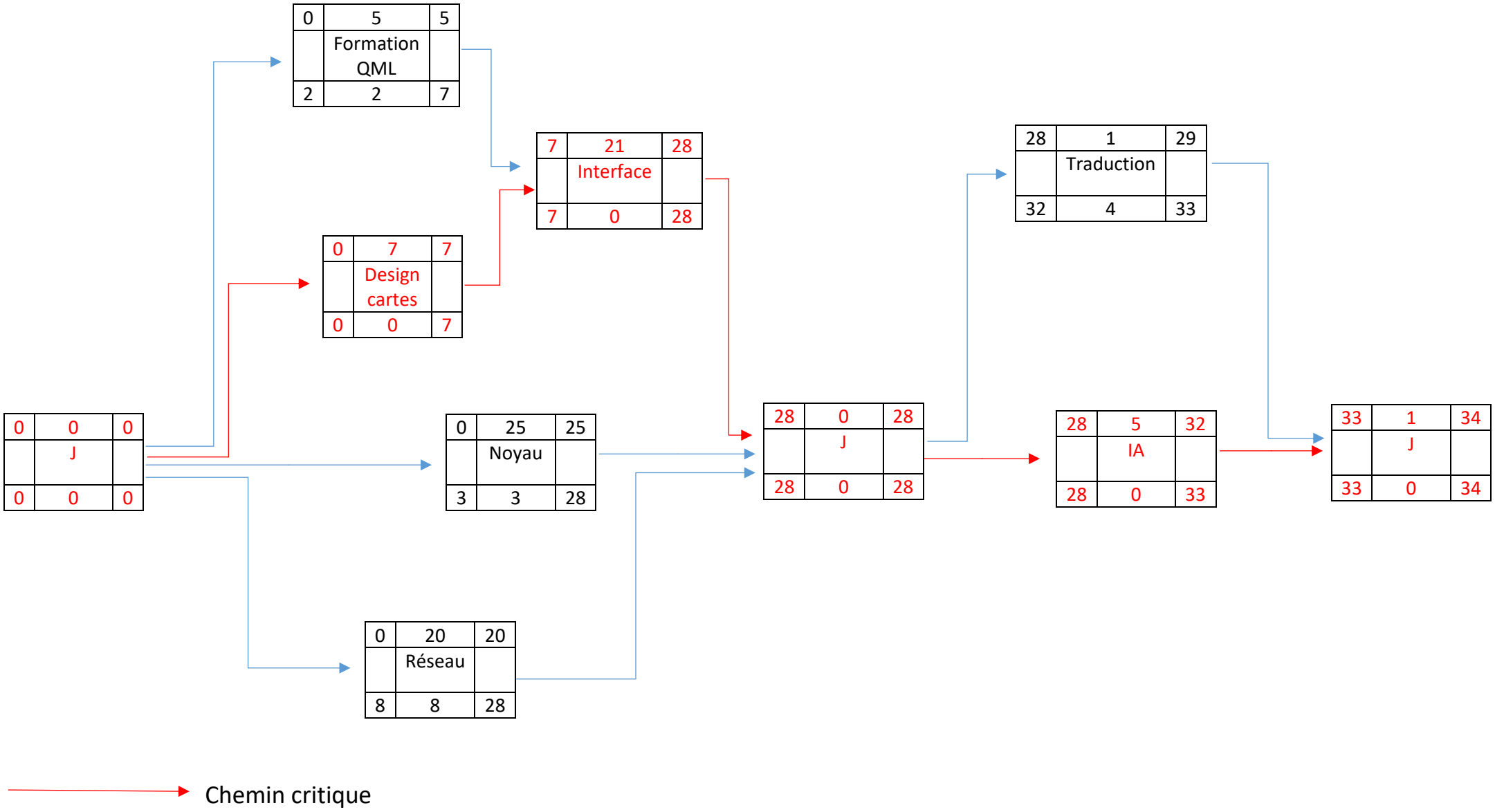

Figure 1 : Diagramme de PERT/CPM

Noyau Uno - Class Diagram