Université de Technologie de Compiègne

Projet NF28 – Coinch'UTC

Application de coinche multi support

CLERMONT Rémi
DEVAUX Bruno
LANCELOT Simon
WANG Yiou

13

CONTENU

REPRISE DU CAHIER DES CHARGES

Après l'enquête réalisée auprès des utilisateurs, nous avions conclu que le besoin le plus important pour notre application était la restitution de l'ambiance présente lors d'une partie de cartes réelles. La partie la plus importante de ce projet était donc de faire en sorte que l'utilisateur s'adapte facilement et apprécie de jouer une partie via notre application android. Cela rentre donc bien dans le cadre de l'uv NF28 puisque nous devions optimiser l'interface dans ce but.

Un autre objectif de notre application était la gestion du tournoi organisé chaque semestre par l'association de Coinche à l'utc.

Les fonctionnalités que nous avions prévu de développer étaient :

- Visualisation des règles du jeu
- Création d'une partie
- Rejoindre une partie
- Sélectionner les paramètres de jeu
- Jouer une partie
- Espace Coinch'UTC
- Statistiques et fiche profil des joueurs
- Chat
- Visualisation des joueurs connectés
- Création d'une application
- Connexion à l'application

Vis-à-vis de notre cahier des charges initial et de notre maquette d'avant conception, notre projet final est sensiblement différent, et ce pour plusieurs raisons (qui seront explicitées plus loin). Cependant, nous avons maintenu les fonctionnalités relatives à l'UV NF28.

La modification la plus importante par rapport au cahier des charges initialement prévu a été de coupler le projet de NF28 avec le projet d'IA04, afin de gérer la communication entre joueurs et de jouer la partie.

ARCHITECTURE DE L'APPLICATION

ARCHITECTURE RESEAU JADE

Ce projet a donc été couplé avec IA04, et possède une architecture multi-agents sous JADE, qui tourne sur Android.

Le principe est le suivant : un conteneur principal est créé sur un ordinateur, et un agent capable de gérer le déroulement de la partie. Il va gérer la distribution des cartes, le déroulement du jeu et le comptage des points.

Lors de l'arrivée dans l'application, il y a tout d'abord la possibilité de créer un compte ou de se connecter à un compte existant, grâce à un *login* et un mot de passe.

Un agent ***LogAgent*** va interroger une base de données RDF, qui contient les joueurs qui possèdent un compte, à l'aide d'une requête SPARQL de type ***ask***, qui va permettre de déterminer si le joueur dont le *login* et le mot de passe sont entrés en paramètres est inscrit dans l'application.

Si les informations entrées correspondent à un joueur existant, alors la requête retourne *true* et l'agent ***LogAgent*** va pouvoir créer un nouveau joueur pour la partie. Un agent ***Joueur*** est ainsi créé.

Lorsqu'il a rejoint la partie, il se retrouve, comme prévu dans le cahier des charges, dans la salle d'attente du jeu. On rappelle qu'il faut obligatoirement 4 joueurs pour jouer à la coinche, répartis en deux équipes de deux joueurs. Dans cette salle d'attente, il peut chatter avec les autres joueurs présents sur la partie.

Une fois que quatre joueurs ont rejoint la partie, le jeu en lui-même peut commencer. Chaque agent possède un *behaviour* séquentiel qui permet d'évoluer proprement dans une manche. Il faut également noter que les équipes sont définies en fonction de l'ordre dans lequel les joueurs rejoignent la partie : le joueur qui rejoint la partie en premier est dans l'équipe 1, celui qui la rejoint en deuxième est dans l'équipe 2, le troisième dans l'équipe 1 et le quatrième dans l'équipe 2.

Une manche de coinche est composée tout d'abord d'un système de distribution des cartes, qui est effectuée par l'agent ***Partie*** de manière aléatoire. Cet agent distribue 8 cartes différentes à chaque joueur.

Ensuite, on assiste à un ou plusieurs tours d'annonces, les joueurs devant, chacun leur tour, annoncer combien de points ils pensent réaliser avec quelle couleur d'atout. Ce tour se termine lorsque 3 joueurs ont passés leur tour (il faut au moins une annonce) ou qu'un joueur a annoncé un *capot* (l'équipe qui annonce un capot indique qu'elle pense réaliser tous les plis de la manche).

Lorsque le(s) tour(s) d'annonce sont terminés, le jeu peut commencer, et chaque joueur joue chacun son tour une carte, en respectant l'ordre dans lequel ils sont inscrits dans la partie. Cette carte est envoyée sur le tapis, elle apparaît ainsi dans la fenêtre relative à chaque joueur.

Lorsque les quatre joueurs ont joué une carte, le score est calculé et ajouté au score global de l'équipe qui a remporté l'annonce. Le tour de jeu recommence 7 fois ensuite (il y a donc

en tout 8 tours de jeu, ce qui est équivalent au nombre de cartes présentes dans la main des joueurs).

Ces huit tours de jeu correspondent donc à une manche. A la fin d'une manche, l'agent ***Partie*** vérifie si le score de l'équipe 1 ou de l'équipe 2 est supérieur ou égal à 1000. Si c'est le cas, la partie s'arrête. Sinon, on relance une nouvelle manche.

ARCHITECTURE DE L'APPLICATION

LES LAYOUTS

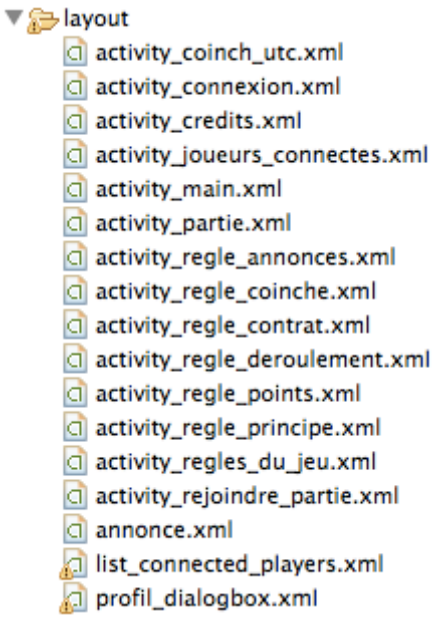

Figure 1 - Liste des layouts de l'application

Comme toute application Android, notre application se compose de différents layouts. Nous allons détailler les principaux dans cette partie.

ACTIVITY_CONNEXION.XML

Ce layout correspond à la vue de la page de connexion

ACTIVITY_CREDITS.XML

Ce layout correspond à la vue de la page « crédits » qui détaille les créateurs de l'application.

ACTIVITY_MAIN.XML

Ce layout est le layout principal qui contient le menu avec les différents boutons pour rejoindre une partie, voir les règles…

ACTIVITY_PARTIE.XML

Ce layout correspond à la vue d'une partie en cours.

ACTIVITY_REGLE_*.XML

Ces layouts correspondent aux différentes vues lors de la visualisation des règles. Nous voulions réaliser une liste déroulante avec les différentes catégories : la coinche, les annonces, … qui aurait affiché le texte correspondant à chaque item de la liste mais nous n'avons pas réussi à implémenter cela. Nous sommes donc restés sur un simple menu.

ACTIVITY_REJOINDRE_PARTIE.XML

Ce layout est celui qui est affiché lorsque l'on clique sur « Rejoindre une partie ». Il correspond aux chats entre les joueurs avant de débuter la partie.

LES CLASSES JAVA

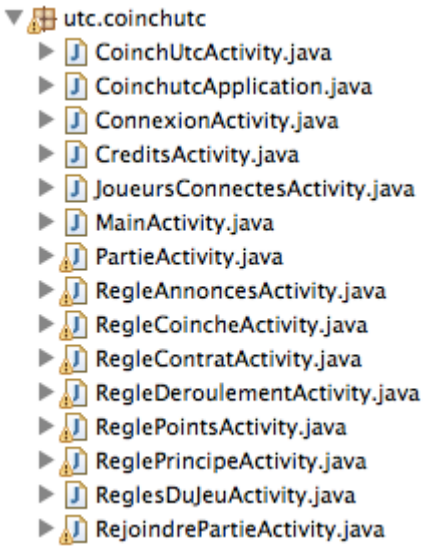

Figure 2 - Liste des classes java

Dans cette partie, nous allons détailler les différentes classes de notre application.

REJOINDREPARTIEACTIVITY.JAVA

Cette classe permet de créer la listView que l'on voit dans la vue de « Rejoindre une partie » en créant un adapter qui permet de transformer les données en liste selon le layout prévu pour l'affichage de celle-ci. De plus, elle implémente les listeners sur les différents items

composants la listView. En effet, lors d'un clic sur un profil, une dialogbox s'ouvre affichant les informations du profil du joueur.

Elle implémente aussi la fonctionnalité de chat entre les joueurs grâce aux méthodes fournies par Jade.

Lorsque tous les joueurs cochent la case « Prêt », la partie commence.

PARTIEACTIVITY.JAVA

Il s'agit là du cœur de l'application : cette activité gère le déroulement d'une partie une fois que les 4 joueurs se sont connectés à la partie. Une partie se déroule en plusieurs manches qui se décomposent chacune en deux phases : une phase d'annonce et une phase de jeu. Lors de la première, l'utilisateur pourra saisir son annonce dans une boite de dialogue en fonction du jeu qui lui a été distribué au début de la manche et des annonces de son partenaire et de ses adversaires (affichées dynamiquement à la réception des messages réseaux). Puis lors de la deuxième il pourra jouer les cartes par un système de Drag&Drop. Les points sont affichés et mis à jours tout au long de la partie en haut à droite, l'annonce retenue à la fin de la première phase est affiché en haut à gauche

Cette activité exécute en parallèle un agent Jade s'occupant des communications réseaux nécessaires aux annonces et aux échanges de cartes.

STRUCTURES DE DONNEES ET AUTRES ELEMENTS IMPORTANTS

STRUCTURE DE DONNEES

La communication réseau s'effectue avec la librairie Jade. Les données sont échangées entre les joueurs via le format de donnée Json.

Nous avons inclus une base de connaissances RDF comprenant les profils des différents joueurs inscrits.

CHARTE GRAPHIQUE

La charte graphique utilisée correspond à celle indiquée dans la maquette initiale de notre application. Nous avons voulu garder un aspect sobre à l'application car elle pourrait être utilisé par la suite par l'association Coinch'Utc. La couleur noire permet de garder un style moderne. La couleur bleue utilisée lors d'un clic sur un bouton représente le calme et la serenité, idéal donc pour l'effet sérieu voulu.

Le plateau de jeu est classiquement de couleur verte afin de recréer une ambiance jeu de carte de type « poker ».

INTERFACE HOMME MACHINE

Tout d'abord, comme explicité dans le cahier des charges, l'interface est réalisée sous Android.

Les deux principales vues intéressantes du point de vue interface homme machine sont :

- La fenêtre pour rejoindre une partie
- La fenêtre de jeu

Lorsque l'utilisateur clique sur « rejoindre une partie », il voit les joueurs connectés dans l'espace supérieur. En cliquant sur un des joueurs, une box apparait qui lui indique les différentes informations du profil du joueur (nom, email, devise…).

L'utilisateur a ensuite la possibilité de chatter avec les autres joueurs en attentes grâce à un système de chat classique.

Lorsque celui-ci est prêt, il coche la case « Prêt » et attend que les quatre autres joueurs la cochent à leur tour. Une fois les quatre joueurs prêts, un bouton apparaît pour lancer la partie. Cette interface rappelle par exemple le jeu Age Of Empire lors d'une partie en réseau.

Une fois la partie lancée…. (todo bruno)

MULTI SUPPORT

De par l'architecture Jade, nous avons donc réalisé un prototype dans lequel nous pouvons jouer depuis un ordinateur ou depuis un appareil Android. Cela optimise donc l'interface homme machine puisqu'il est possible de jouer via deux supports différents.

De plus (comme certaines personnes nous l'ont dit lors de tests), tout le monde ne dispose pas d'un appareil équipé d'Android. Il était donc utile de pouvoir jouer également depuis un ordinateur.

La version sur PC reprend l'architecture JADE et utilise des agents qui implémentent leur propre vue. Chaque joueur implémente sa vue, qui évolue au cours du temps. On va également avoir l'interface de chat, et les cartes sont jouées au clic, à partir de la main du joueur (le joueur doit cliquer sur l'image de la carte). Une fois que le joueur clique sur une carte, celle-ci disparaît de sa main et apparaît sur le tapis de jeu, à côté de sa photo, pour que tout le monde puisse voir qui a joué quoi.

Une fois une manche terminée, une fenêtre indiquant le score de la partie apparaît.

MODIFICATIONS APPORTEES

Nous avons dû modifier notre cahier des charges initial qui a été estimé trop conséquent pour un temps de réalisation trop court. Nous avons donc concentré notre travail sur les principales fonctionnalités qui peuvent intéresser les intervenants de NF28.

Le joueur a donc toujours la possibilité de se connecter. Ensuite, la possibilité de créer une partie a été supprimé de part l'architecture réseau Jade utilisé. Nous avons juste maintenu la possibilité de « rejoindre une partie » qui n'est en faites qu'une partie unique gérée grâce à un serveur sur un PC. En effet, nous avons dû intégrer un ordinateur pour pouvoir utiliser JADE, afin d'avoir le système de partie centralisée. Il n'était pas possible de créer un *main conteneur* sur un smartphone Android.

Enfin, nous n'avons pas développé la partie Coinch'UTC qui ne rentrait pas dans le contexte de NF28.

EVALUATIONS AUPRES DES UTILISATEURS

Nous avons effectué une démonstration devant des étudiants de l'UTC. Il en ressort de bons points : une interface optimisée pour le joueur ainsi qu'un chat facilitant l'organisation de la partie. Globalement, le concept de notre application a beaucoup plu et il serait intéressant que d'autres étudiants améliorent notre prototype par la suite.

L'un des points importants concerne les interactions avec les cartes. En effet, après plusieurs tests par des utilisateurs potentiels, il est ressorti qu'il était plus commode, pour la coinche en tout cas, d'avoir un effet de *Drag and Drop* sur la plateforme Android, mais qu'un simple clic était préférable sur la plateforme PC. En effet, le Drag and Drop sur ordinateur a été jugé trop lourd pour un jeu tel que celui-ci.

LIMITES DU PROTOTYPE ET AMELIORATIONS POSSIBLES

Malheureusement, les délais impartis et la charge de travail ne nous ont pas permis de réaliser l'intégralité de l'application. Notre prototype se limite donc uniquement aux parties qui rentrent dans le contexte de l'UV NF28 et IA04.

Il y a plusieurs points qui pourraient être améliorés :

- Modifier l'architecture réseau afin de ne pas avoir besoin de pc pour faire fonctionner l'application
- Implémenter la gestion multi parties
- Mise en place d'une intelligence artificielle pour s'entrainer
- Mise en place d'un système d'aide à chaque tour de jeu pour les débutants

CONCLUSION

En conclusion, ce projet nous a donc permis de réaliser notre première application android. Nous avons découvert la puissance de ce système d'exploitation qui sera utilisé de plus en plus par le futur. En effet, le format XML utilisé facilite ainsi la résolution du problème principale de nos jours : la compatibilité et l'affichage multi support.

Nous avons énormément appris en réalisant ce projet, tant au niveau de la gestion de projets que de la programmation sous Android. De plus, nous avons ainsi pu étudier la mise en correspondance de Jade/ Android.

Nous tenons donc à remercier les intervenants des UVs NF28 et IA04 qui nous ont fait découvrir un fort intérêt pour cette dernière technologie.