Nathan DUCREY
Arnaud JELMONI
Céphise LOUISON
César MARCHAL
Vincent POULAIN
Xavier RODRIGUEZ

Plan de management de projet

Groupe IHM Connexion

Responsable : **M. Benjamin LUSSIER**

Sommaire

Découpage du projet de notre module

Dans un premier temps, nous avons commencé notre projet en nous répartissant en binômes. Ce découpage semblait logique étant donné que nous n'avions que 3 codes d'accès aux machines. De plus, certains membres de notre groupe n'ayant pas une grande expérience en JAVA, la répartition en binôme permit de faire des couples complémentaires pour que chacun puisse prendre ses marques.

Cependant, dès la première séance de développement, il est devenu évident que ce système ne fonctionnerait pas car le groupe ne tournait qu'à 50% se ses capacités. Il a donc était décidé que, pendant les vacances, chaque membre du groupe s'occuperait de réaliser une frame particulière et qu'il en serait responsable jusqu'à la fin du projet. Cela a été possible car toutes les classes principales dont héritent nos frames ont été développées lors de la première séance.

Le découpage du projet pour notre groupe s'est fait de la façon suivante :

- Réalisation de la conception
- Réalisation des classes principales
- Réalisation graphique des frames
- Réalisation des interfaces
- Mise en place des méthodes dans les frames
- Intégration
- Rédaction de documentation

Estimation des charges

Réalisation de la conception (3-4 Séances)

Cette partie est un travail commun entre les modules. Il était important de définir les interfaces dont chaque module aura besoin vis-à-vis des autres et cela avant le développement afin de ne pas avoir trop de surprises lors de l'intégration. Cette phase est essentielle lors de la création d'un projet où plusieurs acteurs interviennent car elle permet de réaliser un document qui fait référence lors de litiges entre deux acteurs. De plus, elle permet d'imposer des bases de développement pour chaque module afin que tous aient une syntaxe identique (ces bases sont rappelées plus bas dans le rapport)

Pour cette partie, l'ensemble des membres du groupe est sollicité car le travail est conséquent et cela permet à tous d'être au courant de la conception avant de passer à la réalisation. Les documents à fournir ont été les suivants :

- le diagramme de classes
- le diagramme de séquences
- les *use-case*
- l'estimation des taches

Réalisation des classes principales (1 séance)

Dans un premier temps, il est nécessaire de réaliser ces classes qui serviront de base à toutes les vues que nous allons réaliser après, à la fois pour le groupe IHM Connexion que pour IHM Grille. Comme je l'ai dit plus haut, nous avons travaillé par binôme afin de réaliser ces classes. L'impératif étant de les avoir avant les vacances afin de ne pas ralentir la tache suivante qui consiste à créer les vues graphiques qui héritent de ces classes.

Les classes réaliser par cette tache sont :

- GameFrame(3h/homme)
- View(3h/homme)

Initialement, nous souhaitions réaliser plus de classes durant cette tache mais le travail en binôme a ralenti le développement.

Réalisation graphique des frames (1 séance)

Afin de rattraper le retard constaté sur la première séance, il a été décidé de répartir la réalisation graphique des vues entre les membres du groupe afin qu'ils les fassent pendant les vacances et que cela n'impacte pas le développement.

Ainsi, chaque membre s'est vu attribuer une ou plusieurs vues en fonction de ses capacités de développement. La MainView a cependant était laissée de côté car très complexe et nécessitant que plusieurs membres se penchent dessus, notamment en la divisant en plusieurs sous-panels.

La charge de travail à était répartie comme suit :

- CreateGamePopup (3h/homme)
- LogView (4h/homme)
- SetProfilView (4h/homme)
- OpponentProfileView(4h/homme)
- HistoricalView(4h/homme)
- WaitingResponsePopUp(2h/homme)
- ResponseResquestPopUp(3h/homme)
- MainView(3 personnes affectées)
    - ProfilPan(3h/homme)
    - OpponentProfilePan(3h/homme)
    - ListPlayerPan(5h/homme)

Réalisation des interfaces (1 séance)

Les interfaces avec les autres groupes sont un point important de la réalisation car ce sont elles qui assurent le bon fonctionnement entre les modules. Il est donc important d'en réaliser le squelette le plus vite possible afin que les autres groupes puissent les utiliser pour faire leur test. Elles ont été réalisées par une seule personne lors d'une séance.

Implémentation des méthodes dans les frames (2-3 séances)

Suite à la réalisation graphique des vues, nous avons commencé la mise en place des méthodes intervenant lorsque les utilisateurs interagissent avec elles. L'organisation a été la suivante, chaque personne ayant implémenté la vue en graphique s'est occupée d'ajouter les méthodes qui lui correspondent. Cas particulier : la MainView où nous avons dû ajouter d'autres développeurs tant cette vue est complexe. De plus, il a été décidé que le MainView et la HistoricalView seraient liées afin de simplifier la navigation. Un développeur a donc été affecté à cette tache.

- CreateGamePopup (3h/homme)
- LogView (6h/homme)
- SetProfileView (6h/homme)
- OpponentProfileView(6h/homme)
- HistoricalView(5h/homme)
- WaitingResponsePopUp(3h/homme)
- ResponseResquestPopUp(5h/homme)
- MainView(3 personnes affectées)
    - ProfilePan(1h/homme)
    - OpponentProfilePan(3h/homme)
    - ListPlayerPan(4h/homme)
    - Lien avec HistorcialView(4h/homme)

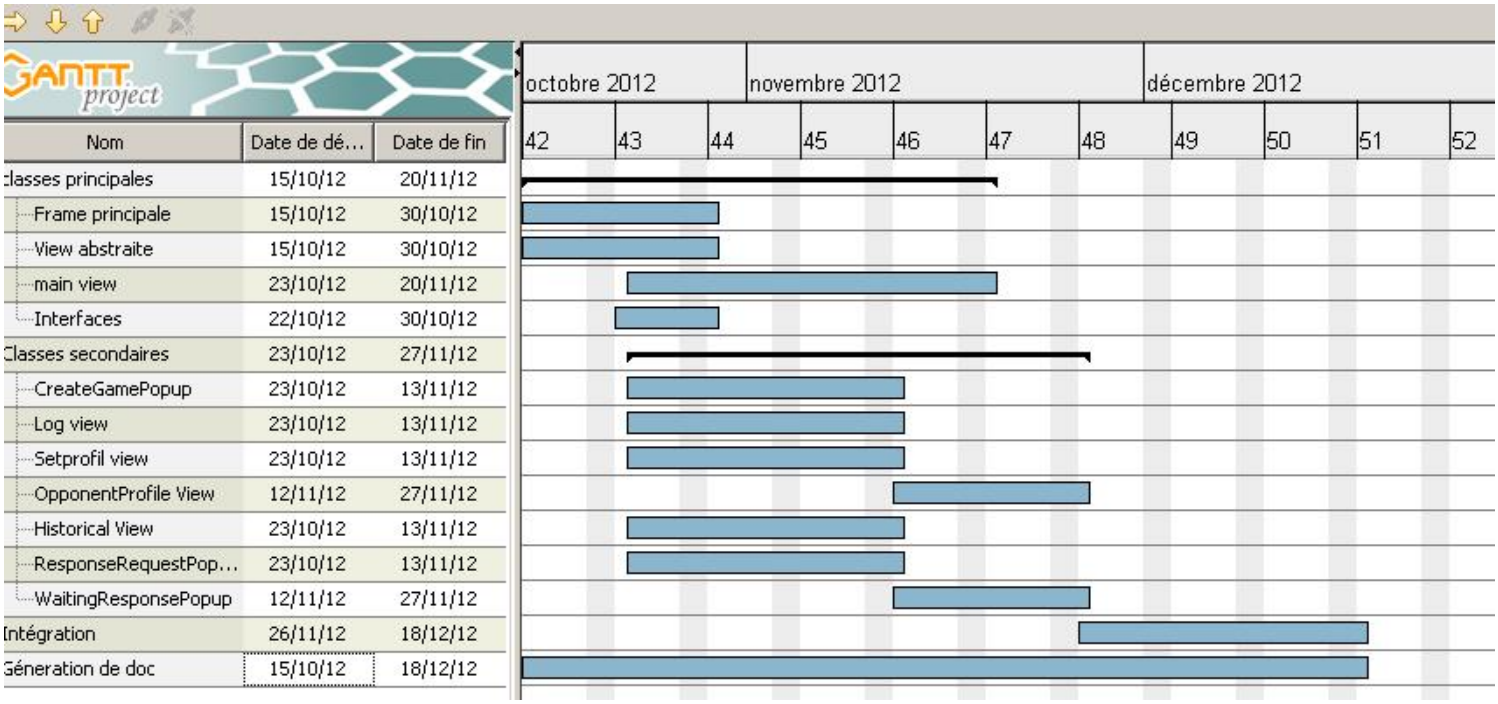
Intégration (3 séances)

Un fois toutes les vues créées et prêtes a l'utilisation, nous avons pu attaquer la période d'intégration entre modules. L'intégration s'est donc diviseée entre chaque module de la manière suivante :

- IHM Grille (2 personnes affectées) :
    - Lancement d'une nouvelle partie (3h/homme)
    - Retour à la liste des joueurs après une partie (2h/homme)
    - Chargement d'une partie existante (5h/homme)
- Réseau (2-3 personnes affectées) :
    - Connexion du joueur sur le réseau (6h/homme)
    - Envoi de demande d'une partie (2h/homme)
    - Réception d'une demande de partie (2h/homme)
    - Réception d'une réponse à une demande de partie (4h/homme)
- Data (2 personnes affectées) :
    - Enregistrement d'un nouveau profil (3h/homme)
    - Vérification de la connexion avec un profil déjà existant (2h/homme)
    - Importation de profil (2h/homme)
    - Exportation de profil (2h/homme)
    - Récupération des parties enregistrement (4h/homme)
    - Modification d'un profil (2h/homme)

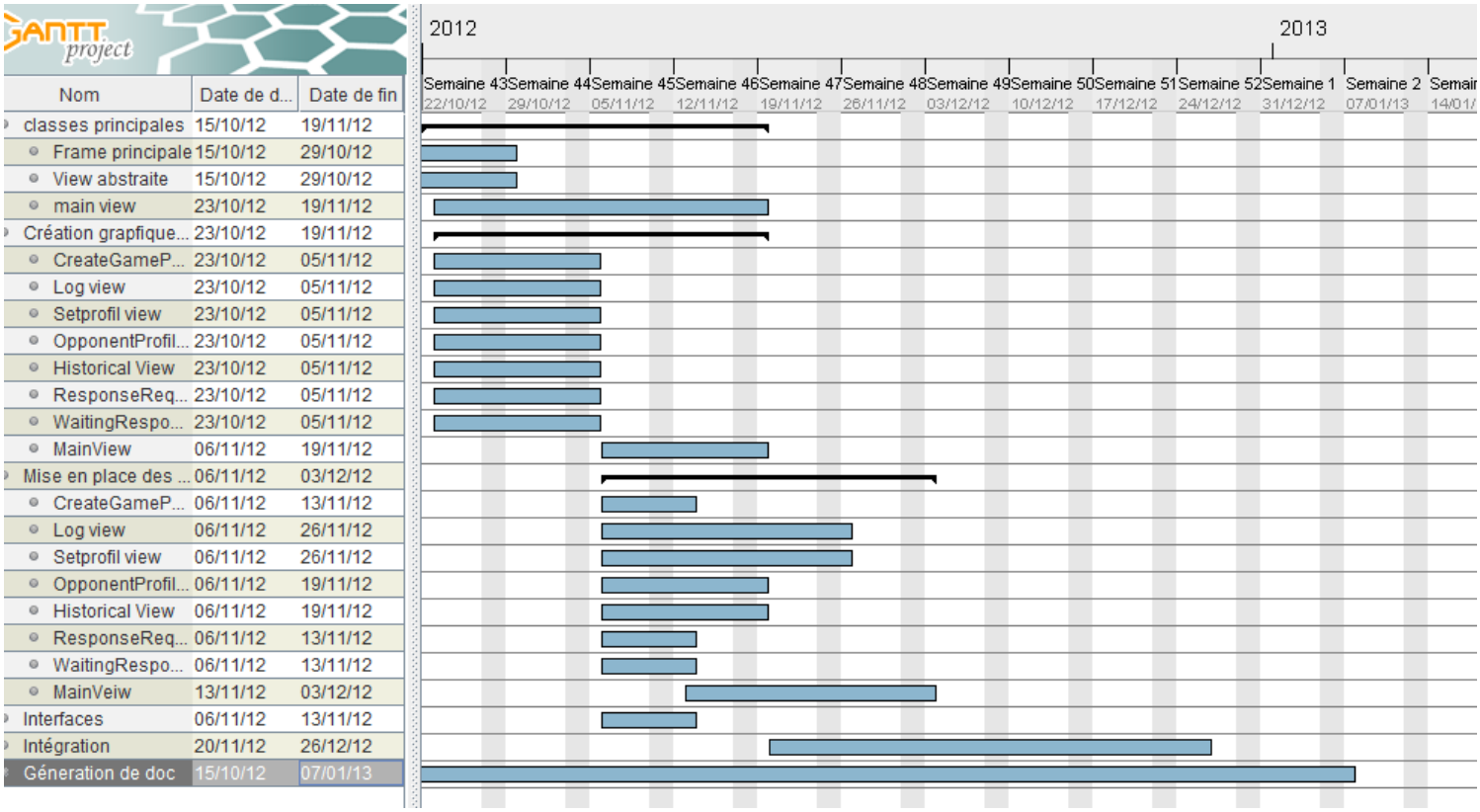
Planification

Gantt initial



Nom	Date de dé...	Date de fin
lasses principales	15/10/12	20/11/12
Frame principale	15/10/12	30/10/12
View abstraite	15/10/12	30/10/12
main view	23/10/12	20/11/12
Interfaces	22/10/12	30/10/12
Classes secondaires	23/10/12	27/11/12
CreateGamePopup	23/10/12	13/11/12
Log view	23/10/12	13/11/12
Setprofil view	23/10/12	13/11/12
OpponentProfile View	12/11/12	27/11/12
Historical View	23/10/12	13/11/12
ResponseRequestPop...	23/10/12	13/11/12
WaitingResponsePopup	12/11/12	27/11/12
Intégration	26/11/12	18/12/12
Géneration de doc	15/10/12	18/12/12

Gantt final



Nom	Date de d...	Date de fin
classes principales	15/10/12	19/11/12
Frame principale	15/10/12	29/10/12
View abstraite	15/10/12	29/10/12
main view	23/10/12	19/11/12
Création grapfique...	23/10/12	19/11/12
CreateGameP...	23/10/12	05/11/12
Log view	23/10/12	05/11/12
Setprofil view	23/10/12	05/11/12
OpponentProfil...	23/10/12	05/11/12
Historical View	23/10/12	05/11/12
ResponseReq...	23/10/12	05/11/12
WaitingRespo...	23/10/12	05/11/12
MainView	06/11/12	19/11/12
Mise en place des ...	06/11/12	03/12/12
CreateGameP...	06/11/12	13/11/12
Log view	06/11/12	26/11/12
Setprofil view	06/11/12	26/11/12
OpponentProfil...	06/11/12	19/11/12
Historical View	06/11/12	19/11/12
ResponseReq...	06/11/12	13/11/12
WaitingRespo...	06/11/12	13/11/12
MainVeiw	13/11/12	03/12/12
Interfaces	06/11/12	13/11/12
Intégration	20/11/12	26/12/12
Géneration de doc	15/10/12	07/01/13

Pilotage

Organisation externe et communication

Communication des informations

- Communication formelle (mails et comptes-rendus) :

La communication formelle est organisée de manière à limiter la taille des groupes de communications pour plus d'efficacité. En effet, les groupes de communications sont composés des 4 groupes de développement et du groupe de directeurs. Il aurait été inefficace de créer une *mailing-list* globale qui aurait surchargée les membres d'informations qui leur auraient été la plupart du temps inutiles.

Dans notre schéma, si un directeur trouve qu'une information est importante à transmettre, il la fait passer sur le canal des directeurs qui eux-même transmettent à leur groupe si nécessaire. On a ainsi une bonne propagation et un bon ciblage de l'information.

Les décisions étaient au début actées par le rapport de conception. Par la suite, au sein du groupe, le mail a pris ce rôle de compte rendu même si on peut regretter la progressive raréfaction de ces comptes-rendus et le manque d'organisation du mail.

- Communication informelle (en TD) :

Cette communication est, par définition, assez peu organisée mais on note qu'elle suit le même principe que celui utilisé dans notre communication formelle : de petits groupes efficaces de deux ou trois dont le but est commun.

Partage des documents de travail

Le partage se réalisa grâce à deux Dropbox. Une pour l'ensemble du TD, divisé en 4 pour chacun des groupes et une pour notre groupe, pour éviter de polluer celle du TD pour les travaux intermédiaires.

Organisation Interne

Comme explicité un peu plus haut les échanges internes au module se sont fait à la fois en communication formelle et informelle. De façon régulière, le manager a demandé de façon informelle l'état d'avancement de chaque membre sur ses tâches afin de pouvoir leur déléguer des ressources supplémentaires en cas de besoin ou de leur affecter de nouvelles tâches s'ils étaient en avance.

A la fin de chaque séance, une petite réunion de 5 minutes était organisée entre les membres du module pour faire le point sur les tâches à réaliser avant la prochaine séance. Un suivi par mail était alors mis en place pour s'assurer que ces tâches étaient bien réalisées.

Indicateur d'avancement

Grâce aux réunions de fin de séance organisées par l'enseignant, chaque module a pu donner un état d'avancement du développement en pourcentage, ce chiffre étant calculé par le manager du module. Ces réunions ont permis aux autres modules de suivre l'avancement des uns et des autres afin de comparer leur retard. Fort heureusement, l'état d'avancement des modules était plutôt homogène.

De façon interne, l'indicateur d'avancement était détenu par le membre affecté à la tache et la consigne étant d'atteindre le niveau d'avancement souhaité avant la prochaine séance par un travail hors de la séance.

Capitalisation de l'expérience

Cette partie essaie de soulever les problèmes de communication et d'organisation que l'on remarque au niveau du poste de Directeur et tente de proposer des solutions.

- Il aurait fallu ouvrir un wiki en début de semestre, c'est actuellement le meilleur outil de partage d'informations. Il permet d'organiser l'information en rubriques liées les unes aux autres, ce qui n'est pas possible avec les mails. Il offre un support collaboratif de rédaction (bien supérieur à un document Word). Les pages rédigées sont plus efficaces pour capitaliser l'information au fur et à mesure et nous auraient facilité la rédaction des rapports finaux.
- Noter ce que l'on fait et les choix que l'on prend de la manière la plus complète possible pendant la phase de développement. En utilisant un wiki pour la documentation, ceci reviendrait juste à compléter la conception.
- Chaque tâche doit avoir un responsable car si une tâche n'a pas de responsable ou un groupe de responsables alors chacun attendra que l'autre le fasse.

- Ne plus utiliser Word dans un projet de groupe. En effet, tout le monde a une version différente et le logiciel gère très mal les gros rapports. Il y a actuellement des solutions web collaboratives en latex.

Analyse de Risques

Absence d'un ou plusieurs membres de l'équipe
Effet : Cela peut entrainer des retards dans certaines tâches du projet.
Réduction des effets : Binômage pour éviter qu'une tâche devienne bloquante en cas d'absence ou de départ d'un membre du groupe.

Coupure Réseau
Effet : Perte de données dans le pire cas, indisponibilité de travailler pendant l'incident sinon.
Réduction des effets : Il est donc important de réaliser régulièrement des sauvegardes sur différents supports (notamment localement) pour permettre une poursuite d'activité ou une reprise d'activité immédiate même en cas d'imprévu.

Serveur SVN indisponible
Effet : Indisponibilité de travailler sur une version à jour, modifications en parallèle d'un même fichier entraînant des confilts
Réduction des effets : Avoir une sauvegarde récente en local ou mettre en place un serveur de secours contenant des sauvegardes régulières à utiliser en cas de panne du serveur principal.

Intégration des différent modules (interfaces non-adaptées, incompréhension, etc.)
Effet : Retard sur le projet
Réduction des effets : Organiser des réunions de mise au point au début de chaque séance, valider clairement les différentes interfaces à réaliser avant implémentation.

Compétences techniques limitées
Effet : Prises de retard sur la réalisation de certaines fonctionnalités
Réduction des effets : Binômage, Auto-formation

Incompréhension avec le client sur le cahier des charges
Effet : Perte de temps sur des fonctionnalités non-nécessaires ou mal-comprises
Réduction des effets : Rédiger un cahier des charges clair et lever toute ambiguité avant la conception.

Actions qualité

Formatage

Cette partie décrit les normes d'écriture de code en vigueur dans le projet. Leur respect est nécessaire pour le bon déroulement du projet pour toute l'équipe.

Langue utilisée

La langue choisie pour l'écriture des commentaires et du code est l'**anglais**. Il est donc nécessaire de veiller au nommage des classes, méthodes, variables, constantes ou bloc commentaires dans cette langue.

Packages

Le nom des packages est composé de 3 éléments :

1. Un préfixe : `fr.utc`
2. Le nom du module auquel il fait référence :
    a. `data` pour données
    b. `network` pour réseau
    c. `grid` pour IHM grille
    d. `lobby` pour IHM connexion
3. (Optionnel) un identifiant de sous package en minuscule, composé d'un seul mot. (exemple : `board`)

Par exemple le package regroupant les classes concernant l'interface de l'échiquier seront regroupés dans le package : `fr.utc.grid.board`.

Classes

Les noms des classes doivent respecter les conventions établies par SUN :

1. La première lettre est toujours en majuscule
2. Si le nom est composé de plusieurs mots, la première lettre de chaque mot est en majuscule
3. Donner des noms simples et descriptifs

Utiliser uniquement des chiffres et des lettres**,** ne pas utiliser de caractères spéciaux.

Exemple d'un nom de classe : `GameManager`.

Variables et attributs de classe

Les noms des variables et attributs de classe doivent respectent les conventions établies par SUN :

1. La première lettre est toujours en minuscule
2. Si le nom est composé de plusieurs mots, la première lettre de chaque mot est en majuscule
3. Donner des noms simples et descriptifs
4. Préfixer les attributs de classes par le caractère *underscore* (_)

Utiliser uniquement des chiffres et des lettres, ne pas utiliser de caractères spéciaux.

Exemple d'un nom de variable locale : `boolean sentMessage`
Exemple d'un nom d'attribut de classe : `private int _numberOfKnights`

Constantes

Les noms des constantes est défini de la manière suivante :

1. Toutes les lettres sont en majuscule
2. Si le nom est composé de plusieurs mots, les mots sont séparés par le caractère *underscore* (_)
3. Donner des noms simples et descriptifs

Exemple d'un nom de constante : `static int BOARD_SIZE = 8`.

Méthodes

Le nom des méthodes utilise la même convention que pour le nom des variables.

1. La première lettre est toujours en minuscule
2. Si le nom est composé de plusieurs mots, la première lettre de chaque mot est en majuscule
3. Donner des noms simples et descriptifs

Exemple d'un nom de méthode : `public void setupChessboard()`

Indentation, espaces et accolades

A chaque nouveau bloc, le niveau d'indentation doit être augmenté. Pour rappel, un niveau d'indentation correspond à une tabulation ou 4 espaces.
Dans le cadre de la déclaration d'une méthode, l'accolade ouvrante se situera sur la prochaine ligne.
Pour un bloc de type *if*, *for*, *switch, while*, *do* l'accolade ouvrante se situera sur la même ligne. Pour chacun de ces mots-clés, un seul espace sépare ce dernier et la condition.

Exemple :

```
public void setupChessboard()
{
	if (!started) {
		…
	}
}
```

Commentaires JAVADOC

Le projet utilise l'utilitaire de génération de documentation JAVADOC, en respectant le format suivant :

```
/**
 * Description détaillée du comportement de la méthode.
 *
 * @param nomParametre Description du paramètre
 * @return Description de la valeur de retour
 */
public boolean legalMove()
{
      …
}
```

Le bloc de commentaire débute donc par `/**` et chaque mot-clé JAVADOC par un arobase (@).

Pour une documentation complète de la syntaxe JAVADOC ainsi que ses différentes fonctionnalités, consulter le site d'Oracle :
http://www.oracle.com/technetwork/java/javase/documentation/index-jsp-135444.html

Gestion des versions

Etant donné le nombre relativement important de développeurs travaillant sur le projet, il est important de mettre en place un dispositif permettant à chacun de travailler sur la dernière version du code et de garder un historique des changements apportés afin d'écarter tout risque lié à l'erreur humaine.

Pour cela, nous avons mis en place le logiciel de gestion de versions « SVN ». Le dépôt est hébergé par le service en ligne « Google code » et un guide de configuration de NetBeans a été mis à disposition de tous les développeurs. Ce guide se nomme « aide au démarrage svn.docx » et se trouve sur la dropbox dans le répertoire « Dossier Plan de Management ».

Les bonnes pratiques d'utilisation de SVN sont par ailleurs données dans le document « Tutoriel SVN (A2012).pdf » disponible sur l'espace moodle de LO23.

Suivi de la qualité

Pour permettre de suivre le projet de manière simple à travers différents indicateurs Qualité, notre module, sur idée de Céphise, a ajouté le projet sur le site Ohloh.net (https://www.ohloh.net/p/utchess).
Il propose des statistiques telles que le nombre de commits par personne, le pourcentage de commentaires, la répartition des commits dans le temps, etc.
Cependant, les chiffres de commits par personne sont un peu biaisés dans la mesure où les comptes GoogleCode paramétrés sur les machines de l'UTC ne correspondent que rarement aux utilisateurs réels. De plus, le pourcentage de commentaires est également approximatif car il prend en compte les blocs de code commentés et non uniquement les commentaires Javadoc.

FICHES PERSONNELLES

Xavier Rodriguez : Directeur

CONCEPTION

- La structure de la fenêtre principale :

  Conception d'une fenêtre commune à toutes les vues, la taille de la fenêtre étant fixe pour faciliter le développement graphique. Elle possède une méthode de chargement

- La structure en vues chargeables :

  La vue définit la taille de la fenêtre et les propriétés au chargement.

- Interface avec IHM grille :

  Mise à disposition de la classe View qui offre à IHM grille un environnement pour s'intégrer dans la fenêtre principale et ainsi facilité le passage de Connexion à Grille et inversement.

- Interface avec Réseau :

  Mise en place d'un pattern observer pour la récupération des utilisateurs connectés.

- Participation à la rédaction du dossier de conception :

  Réalisation du diagramme de classes

  Rédaction de la description des classes

Temps de réalisation : 2 séances de 3 heures

DEVELOPPEMENT

Chaque point de développement ci dessous inclut l'implémentation et le test.

- Formation :

  Avant de commencer le développement, il a fallu un temps de formation sur la gestion des layout en java.

Temps de réalisation : 30 minutes

- InformationField :

  Le développement de ces champs fut motivé pour éviter de devoir gérer séparément un champ d'information et son label. On combine ainsi les deux, formant un seul composant plus facile à positionner, à dimensionner et réduisant par 2 le nombre de pointeurs dans la vue.

Temps de réalisation : 30 minutes

- MainFrame :

  C'est la fenêtre principale de l'application, l'implémentation inclut le système de chargement des vues avec prise en compte des propriétés et la mémorisation de la vue précédemment chargé.

Temps de réalisation : 45 minutes

- View :

  Cette classe représente les panels qui sont chargés dans la fenêtre principale. Ce sont juste des classes héritées de Jpanel auxquelles on a ajouté des propriétés.

Temps de réalisation : 15 minutes

- SetProfilView :

  Cette classe est celle que j'ai principalement développée. Elle a deux fonctions : celle de créer un nouveau profil et celle de modifier un profil existant. Le contrôle des champs se fait dynamiquement lorsque l'on le curseur rentre ou sort du champs avec un FocusListener. L'utilisateur connaît les spécifications de ce qu'il doit rentrer dans le champ car un message d'aide s'affiche quand il rentre dans le champ. Un dernier contrôle est réalisé en fin pour ne laissez rien au hasard.

  Lors de la modification d'un profil, les champs non modifiables sont grisés pour éviter de laisser croire à toutes modifications non autorisées.

Temps de réalisation : 4 heures

INTEGRATION

- Ajouter un profil

  Intégration avec data puis avec réseau

  Tests

Temps de réalisation : 3 heures

- Modifier un profil

  Passage de la vue en Frame suite à une erreur de conception.

  Intégration avec data puis avec réseau

  Tests

Temps de réalisation : 3 heures

Explication des retards :

L'intégration de ces parties a pris du retard, les fonctions de sauvegarde du profil en local devant être corrigé au fur et à mesure : Problèmes de sérialisation, problème de Htable.

ORGANISATION

Pour le TD :

- Mise en place d'une organisation en étoile entre les groupes : les groupes communiquent au travers leurs directeurs vers les autres groupes
- Passerelle d'information entre mon groupe et les autres.
- Création Dropbox générale : cela implique la récupération des adresses mails de chaque membres du TD et la mise à disposition pour tous.

Pour le groupe :

- Dropbox de groupe.
- Mailling liste de groupe.

Temps de réalisation : première séance de 3 heures

PILOTAGE

- Autorisation de la mise en place du Konami code.

Temps de réalisation : 1min

- Motivation de l'équipe.

Temps de réalisation : Permanent

Arnaud Jelmoni : Manager

CONCEPTION

- **Familiarisation avec les méthodes de conception :**
  N'ayant pas suivit d'UV sur la conception d'un projet, j'ai du me familiariser avec les méthodes telles que les Use Cases, les diagrammes de Classe et Diagrammes de séquence
  Temps : 2h au travers des séances de conception

- **Aide à l'élaboration des Uses Cases pour la partie IHM Connexion :**
  Définition des différentes fonctionnalités auxquelles le logicielle devra faire face.
  Temps : 2h

- **Aide à la réalisation des diagrammes de classes :**
  Réflexion sur les classes à mettre en œuvres avec les autres membres du groupe.
  Temps : 1h

- **Aide à la réalisation des diagrammes de séquence :**
  Dessin au papier des différentes interactions à la fois interne au modules mais aussi entre les modules.
  Temps : 2h

- **Réalisation du diagramme prévisionnel de Gantt :**
  Répartition des taches entre les différents membres du groupe
  Mise en place de jalons pour vérifier l'avancer du projets
  Temps : 30min

DEVELOPPEMENT

- **Formation** :
  N'ayant pas fait LO21 je n'étais pas très au fait de la programmation objet et encore moins du développement en JAVA, j'ai donc consacré une partie de mes vacances à apprendre ce langage afin de pouvoir participer au mieux dans la phase de développement.
  Temps : 6h

- **Réalisation de la frame WaitingResponsePopUp** :
  Cette popUp est la plus simple graphiquement car elle ne fait qu'afficher un message au joueur qui fait une demande de partie.
  Cependant dans le but d'empêcher le joueur d'effectuée une autre demande de partie, j'ai dut implémenter un blocage des fenêtres qui l'appelle . Et faire attention que le blocage ne soit plus actif quand la PopUp se ferme.
  Temps : La réalisation de la frame m'a prit 1h30 mais le blocage m'a pris prêt de 3h.

- **Mise en place et test des enchainement de Frame et PopUP**
  Comme expliciter plus haut j'ai fait en sorte que certaine PopUp empêchent le joueur d'effectuer d'autre action dans le programme. Le programme pouvant ouvrir plusieurs niveau de Frame à la fois, j'ai rajouter un attribut "Parent" de type View à celle-ci afin de pouvoir activer facilement les fonctions de blocage et de déblocage des frames appelantes.
  De plus j'ai assuré l'enchainement des différentes frames.
  Temps : 2h

INTEGRATION

- **Création de l'initialisation de l'objet "Game":**
  En effet lors de l'intégration avec IHM Grille ils nous ont annoncer que pour démarrer une partie nous devions leur fournir un objet *Game* déjà initialisé avec les paramètres saisi par les joueurs. Sachant que pour heu le joueur 1 était toujours le joueur local. N'ayant pas prévu ce case de figure nous avons du créer de nouvelles fonction ainsi qu'un nouvelle attribue pour répondre à leur attentes.
  Temps : 3h

- **Réalisation de test de partie en collaboration avec Réseau :**
  Afin de découvrir les Bug possible nous avons effectué un grand nombre de test de lancement de nouvelles parties. Une fois un bug découvert nous le remontions au module correspondant. Si le bug était du côté de mon modules je m'en chargeais.
  Temps : 3h

ORGANISATION

- **Définition et répartitions des tâches :**
  Afin de pouvoir profiter au maximum des talents de chacun, j'ai attribués les taches les plus critiques aux personnes les plus douées pour le développement.

- **Suivit du travail effectuer :**
  Afin de s'assurer que le groupe ne prend pas de retards j'ai demandé régulièrement à chacun où il en était et en cas de problème comment il envisageait de le résoudre pour la séance prochaine

  Temps : Quelques dizaines de minutes à chaque séance.

- **Management du Konami Code**
  Temps : 1min

- **Rédaction de document divers :**
  Plan de management, manuel utilisateur, ...
  Temps : 7heure

Nathan Ducrey : Responsable conception

CONCEPTION

Etant responsable conception dans notre groupe, je me suis donc plus particulièrement intéressé à la phase de conception du projet. Au cours de la première séance, j'ai donc réalisé le diagramme de cas d'utilisation avant de le soumettre aux membres de mon groupe puis ensuite au reste des protagonistes du projet. C'est d'ailleurs mon diagramme qui a été choisi pour servir de base et être sujet aux modifications par les autres responsables de conception. Après cette phase de concertation et surtout de validation tous les membres et le client, nous nous sommes donc divisés les taches de travail avec tous les protagonistes du projet, à savoir l'élaboration des diagrammes de séquences récapitulant les phases d'utilisation du jeu d'échecs. Pour ma part, je me suis penché sur la réalisation des diagrammes de séquences concernant la création d'une nouvelle partie, le chargement d'une partie sauvegardée et la sauvegarde d'une partie en cours.

Ensuite, je me suis intéressé à la description complète des maquettes des interfaces graphiques dessinées par Céphise Louison. Je les ai donc prises une par une et j'ai détaillé les éléments qu'elles comportent, c'est à dire leurs utilités et leurs fonctionnements. En ce qui concerne les boutons de chaque interface, j'ai également renseigné dans le rapport de conception vers quelles vues ils redirigeaient l'utilisateur.

Pour finir, j'ai également participé à la rédaction du rapport, à la phase de conception des interfaces nécessaires au fonctionnement de notre module avec les autres et à la réflexion de la structure de nos classes.

Tâche	Temps	Description
Diagramme use case	~2h	Réalisation du modèle de base + modifications conjointes avec le groupe
Diagrammes de séquence	~2h	Création de partie + chargement de partie + sauvegarde de partie
Description des maquettes	~1h30	Détails des éléments et de leurs fonctionnement

DEVELOPPEMENT

Une fois la phase de conception terminée et validée, nous nous sommes divisés les tâches de développement des interfaces entre les membres de notre module. Pour ma part je me suis vu attribué le développement de deux interfaces graphiques :

- **Pop-up de création d'une nouvelle partie :** Cette vue instanciée dans une autre fenêtre permet le paramétrage d'une création d'une nouvelle partie (couleur + timer).
- **Panel du joueur courant :** Ce panel contient toutes les informations personnelles du joueur (pseudo, nom, ratio parties gagnées/perdues…) et les boutons permettant de visualiser l'historique de ses parties, de modifier son profil et de l'exporter.
- **SetProfilePopUp :** En compagnie de Xavier Rodriguez, nous avons repris sa vue *SetProfileView* pour en créer une popup quand le joueur souhaite modifier son profil lorsque ce dernier est déjà connecté.

Après avoir fini le développement des interfaces graphiques, je me suis mis à réfléchir à quelques améliorations qui pourraient s'avérer sympathiques pour notre projet. La première

d'entre elles consiste à gérer proprement la fermeture intempestive de la fenêtre principale par l'utilisateur (clic sur la croix). Pour ce faire j'ai donc mis en place une fonction de callback lors du clic de fermeture. Cette fonction a pour but de demander à l'utilisateur s'il souhaite vraiment quitter l'application et si c'est le cas alors j'appelle la fonction de clôture du réseau fournie par le module réseau afin de proprement couper la communication réseau du joueur avec les autres. Cette fonction a également été mise à disposition pour l'IHM grille pour demander, en plus de la clôture réseau, la demande de sauvegarde de partie.

De plus, en raison de mon bon avancement sur le développement de mes IHM, j'ai également eu le temps de penser à mécanisme limitant l'exécution du jeu d'échecs à une seule instance par machine afin notamment d'éviter des bugs dans l'échange de paquets sur le réseau. Pour ce faire, je me suis basé sur le tutoriel proposé par un développeur sur developpez.com utilisant les sockets. Cette technique consiste en fait à bloquer un port de la machine au lancement de l'instance du programme. Quand l'utilisateur exécute le programme, alors ce dernier va tout d'abord vérifier si ce port dédié est libre, et si ce n'est pas le cas alors le processus du programme est tout simplement tué avant qu'une connexion réseau ne soit établie.

Enfin, pendant la dernière séance de la phase de développement je me suis attelé à la rédaction des interfaces requises par les autres modules à savoir :

- **newGameEvent :** C'est la méthode qui sera appelée par le module réseau quand un joueur adverse lancera une nouvelle partie contre le joueur courant.
- **loadGameEvent :** C'est sensiblement la même méthode que newGameEvent à la différence qu'un identifiant de partie est également passé en paramètre.
- **receiveResponse :** C'est la méthode qui sera appelée par le module réseau quand le joueur adverse enverra une réponse à propos d'une demande de partie envoyée par le joueur courant.
- **receiveErrorSendingMessage :** C'est une méthode spécialement demandée par le module réseau pour faire apparaître des pop-up de messages d'erreur.

Pour finir, ayant un peu de temps à disposition pendant la phase d'intégration j'ai également mis en place un système de déconnexion, chose qui a été maladroitement oubliée pendant la phase de conception. Disposant de peu de temps devant moi, j'ai donc opté pour la mise en place d'une barre de menu à un bouton de déconnexion. Le code de déconnexion qui est appelé lors du clic est sensiblement le même que lorsque l'utilisateur clique sur le bouton de fermeture, mis à part que l'on rappelle la fenêtre de connexion au lieu de fermer le programme.

Tâche	Temps	Description
Formation gestion des layout	~1h	Tutoriel du siteduzero
createGamePopup	~2h	Créer une nouvelle partie + tests
profilePlayerPanel	~2h	Visualisation du profil du joueur + tests
Gestion de la fermeture de la fenêtre principale	~1h	Implémentation de la fonction callback + tests
Limiter le programme à une instance	~1h30	Compréhension du fonctionnement + création de la classe + tests
Rédaction des interfaces	~1h30	Développement des interfaces requises par le module réseau
Ajout du bouton de déconnexion	~1h	Barre de menu + tests

INTEGRATION

Durant cette phase, j'ai essentiellement passé le temps alloué à échanger avec le module réseau afin de faciliter l'intégration des interfaces que j'ai écrites dans leur code source. J'ai également donné des coups de main par ci par là lorsque que les autres modules avaient des questions sur l'intégration de l'IHM Connexion.

Tâche	Temps	Description
Intégration des interfaces	~3h	Intégration répartie sur plusieurs séances

ORGANISATION

Dans le projet LO23, je me suis occupé de créer l'espace sur Google Code et mis en place le gestionnaire de versions avec Subversion. De plus, pour tous les membres du projet j'ai rédigé un petit tutoriel pour la prise en main facile à travers Netbeans.

Tâche	Temps	Description
SVN Google code	~1h	Création de l'espace + tutoriel d'utilisation

Enfin, en ce qui concerne la qualité j'ai effectué des tests après le développement de chacune de mes interfaces graphiques. De plus, après l'écriture des interfaces pour le module réseau j'ai mis en place des scénarios de simulation les testant afin d'attester du bon fonctionnement de ces dernières. Enfin, comme nous avons pu voir en cours, j'ai apporté un grand intérêt à la rédaction de la Javadoc et des commentaires aux endroits sensibles dans mon code. Cela a porté ses fruits notamment lorsqu'il a fallu le relire et le comprendre pendant les phases d'intégration, cela m'a fait gagné un temps certain.

Tâche	Temps	Description
Javadoc + commentaires	~2h	Répartie sur plusieurs séances

CONCLUSION

En conclusion, j'ai vraiment apprécié participer à ce projet. Evoluer dans ce dernier m'a confirmé l'orientation professionnelle que je souhaite emprunter en sortant de l'UTC, c'est à dire le management et le développement de projet informatique. Ce projet et le cours associé m'a conforté dans mon idée que la gestion d'un projet d'une certaine taille n'est pas une mince affaire, et qu'il faut bien se concentrer sur la phase de conception si on veut que ce dernier arrive à son terme. Travailler, communiquer et surtout s'organiser avec autant de personnes m'a beaucoup appris. Cela m'a procuré une expérience certaine et qui me sera utile dans ma vie professionnelle si j'ai la chance de travailler dans des projets de grande ampleur.

Céphise Louison : Responsable développement

CONCEPTION

- Diagramme de séquence

  Réalisation de certains diagrammes de séquence avec le logiciel Modelio.

Estimation du travail : une séance et demie de travail

- Diagramme de classe

  Réflexion sur l'architecture du module IHM-Connexion avec les membres du groupe et aide à la réalisation du diagramme de classe avec le logiciel Modelio.

Estimation du travail : une demi-séance de travail.

- Conception du design des fenêtres d'interface du module IHM-Connexion

Estimation du travail : une demi-séance de travail.

- Réalisation du logo (Figure 1) et d'avatar

FIGURE 1 : LOGO DE L'APPLICATION

Estimation du travail : une demi-heure de travail.

DEVELOPPEMENT

- **Classe PasswordFieldCustom et InformationField :**

Ce sont des champs de saisie de texte et de saisie de mot de passe modifier qui intègre en même temps un label permettant de décrire le champ de saisie. Ils ont été utilisés dans plusieurs vues, par exemple dans LogView, SetProfileView.

Estimation du travail :

- Création de la classe PasswordFieldCustom et test de bon fonctionnement, une demi-heure de travail,
- Création de la classe InformationField et test de bon fonctionnement, une demi-heure de travail

- **LobbyConstant :**

Cette classe comporte toute les variables constantes et méthodes statiques qui peuvent être utiles au groupe IHM Connexion. Cette classe a notamment été utilisée pour définir les tailles de chaque fenêtre, les chemins d'accès aux fichiers de ressources... Cette classe a été complète au fur et à mesure de l'ajout des Vues par l'ensemble des membres du groupe IHM Connexion.
Estimation du travail : création et implémentations des premières variables un quart d'heure de travail.

- **LogView** :

Cette vue correspond à l'interface de connexion. C'est la fenêtre qui s'ouvre lorsqu'on lance l'application. Elle permet aux utilisateurs de se connecter et offre la possibilité de créer un profil ou de l'importer.
Estimation du travail :

- Réalisation visuelle de la vue : deux tiers d'une séance de travail,
- Test de bon fonctionnement : un quart de séance de travail.

- **Gestion de l'importation et de l'exportation de profils :**

Gestion des événements des boutons liés à l'importation (bouton de LogView) et l'exportation (bouton de ProfilePlayerPan) de profil.
Estimation du travail :

- Le travail comprend trois étapes : ouverture de fenêtre de dialogue permettant de sélectionner un dossier (pour l'importation) ou un emplacement (pour l'exportation). Puis vérification du contenu du dossier pour être sûr qu'il contienne bien un profil. Pour finir, interfaçage avec les fonctions de Data réalisant les imports/exports en fonction du chemin fourni. Charge : un quart d'une séance de travail.

- **HistoricalDetail :**

Le panel HistoricalDetail est un composant de la MainView. Il permet d'afficher les détails d'un historique de partie et de lancer la visualisation.
Estimation du travail :

- Réalisation visuelle du panel : un tiers de séance de travail,
- Interconnexion avec la liste d'historiques (récupération de la partie sélectionnée dans la liste et mise à jour des champs en fonction), un tiers de séance de travail.
- Test de bon fonctionnement, un tiers de séance de travail.

- **MainView :**

La fenêtre principale MainView est une fenêtre toujours présente une fois que le joueur est connecté et tant qu'il n'est pas en partie. Elle est divisée en plusieurs panels (cinq panels différents : ProfilePlayerPan, ListOpponentPan,ProfileOpponentPan, HistoriqueList et

HistoricalDetail) dont chacun a une fonction définie. La MainView se distingue en deux affichages : affichage de l'accueil, affichage des historiques de parties. Dans chacun de ces deux affichages, seulement trois panels sont visibles. L'un des panels est commun aux deux affichages. La MainView s'occupe de la communication entre les panels et l'organisation dans l'espace des panels. Cette formation permet de répartir le travail : chaque panel pouvant être développé par une personne du groupe.
Estimation du travail :

- Organisation des panels dans l'espace : une demi-heure de travail,
- Communication entre les panels : un tiers de séance de travail,
- Gestion du changement d'affichage : un tiers de séance de travail,

- **Konami code**

Ajout d'un Konami code permettant de débloquer un avatar et un logo spécial.
Estimation du travail :

- Ajout de la détection de Konami code, une dizaine de minutes de travail,
- Ajout d'éléments additionnels (avatar, logo), une dizaine de minutes de travail,
- Tests de bon fonctionnement, une dizaine de minutes de travail,

INTEGRATION

Participation à l'intégration de différente fonctionnalité dont :

- Importation et Exportation de profils:

Intégration avec les interfaces de Data.
Estimation du travail : une demi-heure de travail

- Connexion :

Intégration avec les interfaces de Data et de Réseau.
Estimation du travail :

- participation à l'intégration : une demi-séance de travail,
- participation aux tests de bon fonctionnement une demi-séance de travail.

- Revu de partie :

Intégration avec les interfaces de IHM Grille.
Estimation du travail : un quart d'heure de travail

- Changement des statuts des joueurs :

Passage de l'état du joueur 'En attente' à 'En jeu' et inversement et communication aux autres joueurs de l'état, au moment de création/chargement de parties et de fin de partie et aux moments de début et de fin de revisualisation de partie terminée. Intégration avec Réseau.
Estimation du travail :

- Changement d'état et communication aux autres joueurs : un quart de séance de travail,
- Mise à jour du champ dans la liste des joueurs : un quart de séance de travail,

Participation aux tests de bon fonctionnement : un quart de séance de travail.

AUTRES

- Travail sur la qualité :

  Vérification de la qualité dans le code du groupe IHM-Connexion, notamment concernant la JavaDoc.
  Ajout du projet sur le site https://www.ohloh.net/p/utchess permettant d'avoir un suivi des commits : nombre de commit, fréquence, des informations sur le code : nombre de lignes, nombre de lignes de commentaire, langage utilisé et sur les contributeurs...

Estimation du travail : une séance de travail répartie en plusieurs fois.

- Travail de management

  Management de la production, suivie du travail effectué au sein du module IHM Connexion pour avoir une vision globale de l'avancement du développement et en prévision de la phase d'intégration : pour me permettre de répondre aux questions des membres des autres modules sur la construction et le développement du module.

Estimation du travail : dizaine de minute à chaque séance de travail.

César Marchal : Responsable qualité

CONCEPTION

- Participation à la création de plusieurs diagrammes de séquences avec l'ensemble des membres du module.
  Durée estimée : 1 séance
  Durée réelle : 1 séance

- Participation à la réflexion commune sur l'architecture du module et la création du diagramme de classes.
  Durée estimée : 1 séance
  Durée réelle : 1 séance

- Rédaction de la 1ère version du plan de management pour le module IHM Connexion: estimation des charges, planification.
  Durée estimée : 0,5 séance
  Durée réelle : 0,5 séance

- Rédaction du dossier de conception: choix de conception, description de quelques diagrammes de séquence.
  Durée estimée : 0,5 séance
  Durée réelle : 0,5 séance

- Correction du travail du module, intégration dans le rapport final commun et envoi au client.
  Durée estimée : 1 heure hors-séance
  Durée réelle : 2 heures hors-séance

DEVELOPPEMENT

- Création du squelette de la classe GameFrame en binôme avec Nathan. Cette classe est utilisée comme fenêtre principale et les différentes vues utilisées sont chargées dans cette GameFrame.
  Durée estimée : 0,5 séance
  Durée réelle : 0,5 séance

- Implémentation du Pop-Up de réponse à une demande de nouvelle partie. A partir de la maquette effectuée par Céphise, j'ai créé le Pop-Up permettant la réponse à une demande de nouvelle partie par un joueur. Elle contient les informations concernant l'éventuelle partie: le profil de l'opposant, la couleur du pion et la présence ou non d'un timer.
  Durée estimée : 2/3 séance
  Durée réelle : 1 séance

- Création d'une JTable et de son TableModel permettant l'affichage des joueurs connectés. Cette JTable est utilisée dans la MainView et est actualisée à chaque connexion/déconnexion d'un joueur, ou lorsqu'un joueur connecté modifie des informations dans son profil (via le pattern Observer-Observable).
  Le TableModel permet de simplifier la création à partir d'une structure de données et l'accès aux données présentes dans la JTable.
  Quelques difficultés ont été rencontrées sur ce point: la JTable est au départ prévue pour afficher des données statiques, le TableModel a été nécessaire pour contourner ce problème. N'ayant jamais implémenté cette structure, j'ai eu des difficultés à comprendre son fonctionnement et les interactions avec le TableModel.
  De plus, il a été impossible de conserver la ligne sélectionnée après mise à jour de la JTable. En effet, la fonction permettant de retourner la ligne sélectionnée ne fonctionne pas (renvoie constamment -1, soit "aucune ligne sélectionnée").
  Durée estimée : 1 séance
  Durée réelle : 1,75 séances

- Création d'une JTable affichant la liste des parties à revoir avec son TableModel.
  Cette JTable fut moins complexe car elle ne se met pas à jour dynamiquement. De plus, j'avais déjà réalisé la JTable précédente, donc l'implémentation a été assez rapide.
  Durée estimée : 1 séance

Durée réelle : 0,5 séance

INTEGRATION

- Intégration de la liste des joueurs connectés avec le module Réseau : déclaration et utilisation de leurs interfaces aux endroits adéquats pour pouvoir récupérer la liste des joueurs connectés et être notifié à chaque mise à jour de cette liste via le pattern Observer-Observable.
  Durée estimée : 0,5 séance
  Durée réelle : 1 séance

- Intégration des différents échanges avec Data et Réseau, notamment lors de la mise à jour du profil en local et de l'envoi de l'information sur le réseau, et la mise à jour du statut des joueurs (En jeu/En attente d'une partie).
  Durée estimée : 0,5 séance
  Durée réelle : 1 séance

- Participation ponctuelle sur d'autres parties de l'intégration.

ORGANISATION

- Suivi des indicateurs Qualité pour le module via OhLoh, l'outil mis en place par Céphise.
  Durée estimée : 2 heures réparties sur l'ensemble du projet
  Durée réelle : 2 heures réparties sur l'ensemble du projet

- Rappels réguliers sur les actions Qualité à effectuer: mettre des commentaires régulièrement, tester les différentes fonctionnalités au fur et à mesure, etc.
  Durée réelle : 2 heures réparties sur l'ensemble du projet

- Reporter les différentes erreurs trouvées par les membres d'IHM Connexion aux autres modules car j'avais déjà de bons rapports avec certains membres des autres modules.

- Rédaction de la partie *Analyse de Risques* du plan de management du groupe.

- Correction globale des rapports de développement et plan de management pour le groupe.

- Idée de la mise en place du Konami Code.

- Bonne humeur et création d'une équipe soudée (contre moi)

Vincent Poulain : Responsable Etude

CONCEPTION

Nous avons tout d'abord réfléchi aux questions d'étude car elle pouvait être bloquante pour la conception. Des choix ont rapidement été fait, le langage fut le Java (car peu utilisé à l'UTC), l'IDE fut NetBeans (même raison), le logiciel de suivit de version fut alors SVN car supporté par un plugin NetBeans… Le choix du logiciel UML fut plus laborieux mais la décision de Modelio fut votée avec une grande marge.

Pendant la phase de conception nous nous somme réparti les taches en binôme, j'ai alors travaillé en coopération avec Nathan sur le diagramme use case auquel nous avons intégré petit à petit les suggestions des autres membres du groupe.

Le travail sur les diagrammes de séquence fut identique, répartition en binôme avec des points de réflexion avec l'ensemble du groupe pour ne rien oublier et choisir les meilleures idées. J'ai principalement travaillé sur la négociation de partie entre deux joueurs, partie qui c'est très rapidement agrandi en de nombreux sous cas : demande de partie (nouvelle et chargement), réception de la demande de partie (nouvelle et chargement) et réception de la réponse à la demande de partie. J'ai alors défendu nos choix (à Nathan et moi) à l'ensemble du groupe puis du projet : la demande de partie est bloquante (pour éviter d'avoir à traiter le cas de plusieurs demandes en même temps) et d'un timer d'environ 30s pour débloquer l'utilisateur. Ils permettent en effet de grandement simplifier la négociation de partie (surtout les effets de bords) et nous avions toujours la possibilité de le modifier si nous étions en avance (hypothèse bien sur irréaliste). Je me suis ensuite charger de les réadapter et modifier au changement dus à des modifications dans le projet (convention de nommage, langue,…).

J'ai ensuite travaillé sur le désigne des fenêtres avec Céphise et Nathan, décrire les principaux écrans, les boutons et fonctions associées. Puis de la critique des maquettes faite par Céphise pour être sûr de n'avoir rien oublié et d'avoir le même fonctionnement que décrit dans les diagrammes de séquence. Enfin j'ai participé à la création du diagramme des classes, surtout pour qu'il soit conforme aux fenêtres que nous avions définies et aux diagrammes de séquence.

J'ai participé au rapport en rédigeant la description concernant les diagrammes de séquence que Nathan et moi avions réalisé et concernant les choix de conception associé (timer et fenêtre bloquante).

Tâche	Temps estimé	Temps réel	Description
Diagramme use case	~2h	~2h	Création d'un use case puis modification avec le groupe.
Diagrammes de séquence	~2h	~5h	Demande de partie + réception de demande + réception de la réponse. Retard à cause de la prise en main du logiciel puis des nombreuses retouches et modifications.
Désigne des fenêtres	~1h	~1h30	Détails des fenêtres et de leurs fonctionnements
Participation au diagramme des classes	~1h	~1h	Participation et vérification des classes des fenêtres.
Rédaction du rapport	~2h	~2h	Rédaction de la description des diagrammes de séquence et des choix de conception

DEVELOPPEMENT

La première séance à commencer par la répartition du travail, création de l'ensemble des classes puis répartition de celle-ci. Je me suis ensuite mis en binôme avec Céphise pour pouvoir travailler sur un ordinateur et avons commencé à travailler sur la logview. Nous n'avons pas beaucoup avancé mais nous sommes familiarisés avec Java - swing et avons modifié la classe InformationField en PasswordFieldCustom pour qu'elle soit adaptée à la logview.

J'ai eu la charge de la fenêtre OpponentProfileView qui doit afficher les informations détaillées de l'adversaire sélectionné et le panel ProfileOpponentPan qui va de même afficher les informations de l'adversaire sélectionné dans la MainView. J'ai pris beaucoup de temps à m'adapter à swing de manière à obtenir le rendu voulut, sachant que le panel serait plus ou moins identique à ma vue. J'ai par exemple rajouté une fonction pour modifier la taille des images et plus précisément de l'avatar dans LobbyConstant. J'ai pour finir créer les liens entre les boutons de mes fenêtres à celle des autres et opéré aux faibles modifications nécessaire à leur fonctionnement. J'ai testé pas à pas au cours du développement le bon fonctionnement de mes fenêtres en utilisant de faux profiles.

J'ai eu la mauvaise surprise de devoir modifié ma View en une JFrame pour qu'elle puise apparaitre en pop –up et non en replaçant la MainView qui contient l'interface réseau. Beaucoup de sous-problème sont apparu, principalement lors de l'interfaçage avec la pop-up de création de partie.

Tâche	Temps estimé	Temps réel	Description
LogView	~2h	~2h	Création et premier jet avec Céphise
OpponentProfileView	~5h	~9h	Création et modification de la vue, puis transformation en JFrame et liaison avec les autres fenêtres.
ProfileOpponentPan	~1h	~1h30	Création du panel dans le même genre que celui déjà réalisé. Liaison avec les autres fenêtres.

INTEGRATION

J'ai intégré la partie concernant la négociation des parties, c'est-à-dire le lancement d'une nouvelle partie et la réception de cette demande.

La première partie a été réalisée avec le groupe Réseau et à principalement consisté à la création des pop-ups avec les informations dont ils avaient besoin pour appeler leurs fonctions. J'ai passé beaucoup de temps à bloquer les fenêtres lors de la réception d'une demande de partie en sachant que l'utilisateur peut avoir ouvert plusieurs pop-up à la suite. Cette intégration c'est très bien passé, la majorité du travail à faire devant être fait de notre côté.

La deuxième partie fut d'intégrer avec IHM grille pour lancer la partie à proprement parler, intégrer leur code ne fut pas très dur. Uniquement leur demander d'appeler une fonction pour initialiser la taille de la fenêtre. Le principal problème fut de leur retransmettre l'ID de la partie, chose non défini dans la phase de conception. Il a donc fallut lors de la demande de nouvelle partie créer un objet Game, travailler avec le groupe réseau pour modifier les interfaces et envoyer l'ID de la Game à l'adversaire. De même, il a fallu créer chez l'adversaire un objet Game avec les bons paramètres et en modifier l'ID (et donc demander au groupe data de rajouter cette fonction). Le principal problème est venu lors de la création de l'objet Game chez l'adversaire, les joueurs étant mal configurés.

Tâche	Temps estimé	Temps réel	Description
Intégration du lancement de partie avec réseau	~2h	~3h	Majorité du travail fait sur les pop-ups.
Intégration du lancement de partie avec IHM Grille	~3h	~4h	Modification des pop-ups, modification avec réseau et avec data.

PACKAGING

Pour finir, je me suis chargé de réaliser le packaging de l'application. Tout d'abord j'ai eu l'occasion de remettre ma casquette de responsable d'étude et de trouver une solution pour réaliser cette tâche. J'ai commencé par tester « launch4j » mais qui m'a donné peu de résultat, j'ai donc rapidement cherché autre chose et j'ai trouvé « jsmooth ». Un logiciel libre très intuitif qui dispose même d'une fenêtre de débug ! Il crée un .exe qui a tout de même besoin des répertoires de ressources. Un setup basique va être conçu en utilisant Inno Setup que j'ai déjà utilisé dans le passé.

Tâche	Temps estimé	Temps réel	Description
Packaging du jar en .exe	~3h	~2h	Le logiciel fonctionne bien !
Packaging du projet dans un setup	~2h	~2h	Création d'un setup grâce à Inno Setup pour installer l'application.