HULSEN Sorn GROUPE 13

PETIT Martin

VANDERDONCK Nathan

Rapport de projet

« Shared Calendar »

1) Introduction

Voici le rapport qui sonne le terme de ce travail de groupe. Notre groupe, portant le numéro 13, travaillait sur un projet de calendrier appelé « Shared Calendar ».

Ce document, en plus de cette brève introduction, est composé de :

- ✓ Une présentation projet comprenant
  - Les consignes du projet,
  - Le cahier des charges,
  - Le diagramme UML final du projet ;
- ✓ Une explication sur les problèmes auxquels nous avons dû faire face ;
- ✓ Une explication des choix que nous avons dû faire ;
- ✓ Une description des moments marquants que ce projet laisse en nous ;
- ✓ Une séquence d'idées pour le futur de ce projet ;
- ✓ Une conclusion personnelle de chaque membre du groupe avec leur ressenti sur le projet.

2) Présentation du projet

a. Consignes

Le thème du projet est laissé au choix des étudiants, pourvu qu'ils répondent aux contraintes suivantes :

- Il s'agit soit d'une application utilitaire, soit d'un jeu (autres possibilités discuter avec l'équipe enseignante)
- L'application doit respecter l'architecture MVC avec deux interfaces utilisateurs (une interface console et une interface graphique)
- L'application doit comporter une communication Socket (les sockets seront vus au cours) OU une interaction JDBC avec une base de données (attention, les étudiants devront alors découvrir JDBC par eux-mêmes).
- L'application doit utiliser au moins une structure de données du Framework Java Collection (`HashMap`, `List`, ...).

b. Cahier des charges

Nous voulons créer un calendrier partagé. L'utilisateur peut faire partie d'un ou plusieurs groupes, chaque groupe possédant son propre calendrier indépendant (dont la gestion peut être gérée selon les autorisations). Les calendriers (et donc les différents évènements) des groupes dont l'utilisateur fait partie se synchronisent avec son calendrier personnel, dans lequel il peut ajouter des évènements indépendants de tout groupe. De base, le créateur du groupe en est l'administrateur et possède ainsi les permissions pour ajouter/supprimer des évènements et donner la permission aux utilisateurs qui sont dans le groupe d'en faire de même.

c. Diagramme UML

UML

3) Difficultés rencontrées

Ce travail fut pour nous une vraie épreuve, tant au niveau programmation qu'au niveau de la gestion de nous-même.

La toute première épreuve fut la construction de notre diagramme UML. Nous avons directement eu une idée de ce que nous voulions et nous sommes parti dans la direction qui nous semblait la plus logique. En avançant dans la réalisation du projet, nous avons remarqué que nous étions partis sur une mauvaise base, trop compliquée et surtout non-réalisable. Cette erreur peut s'expliquer par notre manque de connaissance de l'API au début du projet. Une fois nous être rendu compte que notre UML ne tenait pas la route, nous nous sommes renseigner sur ce que nous offre réellement Java. Fort heureusement pour nous, l'API Java est très riche et nous avons compris comment gérer nos dates grâce à elle. Cette découverte nous a permis de recréer un diagramme UML plus simple, plus petit et plus compréhensible.

Nous avons construit notre cahier des charges selon notre « rêve » d'application. Nous avons donc tenté d'implémenter de belles idées certes, mais pas des moindres. Notre premier problème fut ici, nous avons misé la barre haute pour des étudiants sans aucune connaissance en Java. Nous aurions peut-être dû alléger notre cahier des charges afin d'avoir un projet fini et complet. En effet, pour le moment notre Calendrier fonctionne mais il manque de vue graphique, comme demandé dans l'énoncé. Il y a également quelques fonctionnalités que nous n'avons pas eu le temps d'implémenter.

Ce qui nous amène à notre troisième difficulté : le temps. Le début du projet a commencé de manière presque « trop simple » (création de classes avec quelques méthodes telles que `equals`(), `toString`(), `compareTo`(), construction d'un diagramme UML, …). Le vrai travail étant survenu plus tard, nous avons commencé de manière relax et sereine sans réellement se rendre compte de ce qui nous attendait par la suite. Tel que le travail est à ce stade, Il nous manque notre vue GUI. Ce manque ne s'explique pas par une trop grande difficulté car notre groupe est solide et peut surmonter ce genre de challenge, mais pour

cause de manque de temps. Nous sommes persuadés que nous aurions pu réaliser une GUI fonctionnelle si nous avions eu une ou deux semaines de plus.

Comme dit plus haut, notre attitude calme face au début de ce projet nous a ralenti, mais il n'y a pas que ça qui a joué en notre défaveur. Notre calendrier contient une base de données pour stocker nos personnes, nos groupes et nos évènements. La création de cette base a été pour nous un jeu d'enfant, mais qui dit base de données en Java dit JDBC et voilà la source de longues heures d'études et de casse-tête pour implémenter des patterns inconnus et, à l'époque, incompréhensibles. Nous avons passé beaucoup de temps à les étudier pour qu'ils fonctionnent au mieux dans notre cas et nous y sommes finalement parvenu. Mais cette étape nous a fait perdre un temps considérable car, en plus de devoir étudier et tester JDBC, il a fallu par la suite l'implémenter dans notre code déjà existant, ce qui nous a fait remettre en question une bonne partie de notre code. Ce fut long et compliqué.

Un dernier point qui ne nous a pas aidé au niveau du timing est l'examen d'Infrastructure réseau de base qui a eu lieu ce lundi 19 décembre. Ce test était, appelons un chat un chat, un gros morceau. Ce genre de bilan ne s'étudie pas en quelques heures compte tenu de la charge assez importante de travail qu'il contenait. C'eût été une vraie tâche en soi de trouver du temps pour travailler notre projet avec ce « monstre » approchant à grands pas. Peut-être que sans lui nous aurions su construire une interface graphique valable, ou au moins amélioré notre code au maximum pour le rendre plus propre, plus clair et peut-être moins redondant.

4) Choix des implémentations

Tout au long de notre aventure, nous avons dû faire un grand nombre de choix.

Le premier fût la question du stockage de données. Les consignes du travail stipulaient que le projet contienne une connexion réseau. Les deux solutions qui s'offraient à nous étaient une connexion socket ou une connexion JDBC. Dans notre cas, nous avons préféré la seconde proposition. Les raisons qui ont motivé notre choix fût simples et évidentes. Premièrement, il nous fallait une façon de partager nos données avec plusieurs utilisateurs de manière simple. Notre formation de première année nous a initié au langage SQL et à la gestion de base de données. Nous avons directement et de manière unanime pensé qu'il serait très intéressant et enrichissant de faire des liens avec des notions déjà connues pour les approfondir davantage. La seconde raison est simplement qu'une base de données est exactement le format que nous souhaitions pour stocker nos données. Nous avons donc opté pour cette solution malgré le besoin de recherche et d'apprentissage du JDBC et du Pattern DAO.

Nous avons choisi d'implémenter notre base de données comme étant la représentation la plus complète possible de notre modèle en Java. Nous avons donc opté pour créer une base contenant une table de personnes, une de groupes et une d'évènements, qui sont une sorte de miroir de nos classes Person, Group et Event. Nous avons commencé notre projet sans la base de données, ce qui implique que notre code engendrait de bases les liens entre les différentes classes. Au moment de créer et d'implémenter notre base, nous nous sommes

rendus compte qu'il aurait peut-être été plus intéressant au niveau de la légèreté de gérer nos liens en SQL plutôt qu'en Java. Malheureusement, le manque de temps a fait que nous n'avons pas eu le choix de garder notre code tel qu'il était et donc de garder un client lourd. Il est clair pour chaque membre du groupe que si c'était à refaire, nous privilégierons une approche plus légère de notre partie client, et donc gérer les liens au sein de notre base de données. Notre structure se résume donc en une simple base de stockage de nos objets Java que notre programme lie lui-même. Les liens sous-entendus ici sont le lien entre les groupes et les personnes, le lien entre les évènements et les personnes et celui reliant les évènements et les groupes.

Le second choix auquel nous avons été confrontés est celui du type de donnée de stockage en Java. Nous étions au préalable partit sur la création de tableau car il s'agissait d'une structure connue par tous les membres du groupe. Bien vite, nous avons changé d'avis et opté pour des `HashMap` et des `ArrayList`, les `ArrayList` étant elles-mêmes contenues dans une `HashMap`. La raison première de ce choix est la simplicité. En effet, la gestion de ce type de donnée est beaucoup plus simple une fois le principe global assimilé. La seconde raison est l'adaptabilité et la puissance, ce qui rejoint la première raison. Les deux structures de données utilisées sont beaucoup plus modulaires qu'un simple tableau. Les méthodes applicables sur ce type de tableau sont très variées et faciles à comprendre, ce qui nous offre un plus large accès à nos données et une gestion plus performante de notre stockage.

Un autre dilemme auquel nous avons dû faire face est celui du choix des commandes. Le problème était d'arriver à gérer un système fonctionnant avec des commandes et des arguments de commande différents en nombre notamment. La solution a été au final de récupérer tout ce que l'utilisateur avait entré sous forme d'un tableau de string et de gérer la commande principale en faisant un switch sur le premier élément du tableau et ensuite en fonction de la longueur du tableau (et donc du nombre d'arguments à la commande) d'utiliser les fonctions nécessaires à ce que l'utilisateur a demandé.
Ce n'est sans doute pas le système le plus optimal au niveau performance et surtout au niveau clarté du code (le niveau d'imbrication de condition est parfois limite), mais cela fonctionne et à plus long terme, nous pensons qu'il serait

5) Points marquants

Bien que délicat et laborieux, ce projet a aussi été une aventure complète.

Il a commencé avec des petites réjouissances telles que nos classes de départ qui fonctionnaient comme attendues, notre diagramme UML qui se construisait petit à petit, …. Les choses ont ensuite évolué et nous sommes vraiment rentré dans notre calendrier. Nous avons appris beaucoup de choses que le cours laissait floues tel que les `HashMap`, les `ArrayList` que nous connaissons dorénavant. Il est plaisant aujourd'hui de remarquer que nous maîtrisons ces principes et que nous jonglons avec alors que, quelques semaines plus tôt, nous grimacions face aux slides à leur sujet.

Un évènement qui nous a très fortement marqué est l'implémentation de notre JDBC dans notre code. Nous avions commencé l'apprentissage de cette matière dans un projet à part pour comprendre le principe te sélectionner ce qui nous intéressait pour notre projet et ce qui ne nous serait pas utile. La « rencontre » entre les deux codes a alors été une source de stress importante. Nous nous imaginions déjà tomber sur un travail remplit d'erreurs et de non-concordances mais nous nous trompions. Effectivement il y avait des erreurs de compatibilités entre les deux séquences mais nous les avons réglées plutôt rapidement. Le moment où nous sommes rendus compte que notre calendrier se connectait bel et bien à notre base de données, que nos méthodes ne devaient pas être modifiées et que le programme nous sortait les informations demandées a réellement été une source de joie et de fierté pour notre groupe.

6) Améliorations futures

Pour la suite des évènements, nous imaginons pas mal de nouveauté.

Tout d'abord, la GUI. En effet, notre programme n'en dispose pour le moment pas pour raisons citées plus haut. Nous avons cependant des idées assez précises de la forme que notre interface graphique pourrait avoir.

Ensuite, pour continuer dans les éléments manquants, nous pourrions implémenter tout ce que notre cahier de charges stipulait et qui ne fait pas partie de notre travail.

Une autre amélioration possible serait l'ajout de fonctionnalités telles que la connexion via une adresse mail, la possibilité de synchroniser le calendrier avec celle-ci, l'ajout de Material Design au niveau de la GUI, une connexion via Facebook ou un autre réseau social pour y partager et y recevoir des évènements.

Une fonctionnalité intéressante que nous pourrions implémenter serait une gestion de l'emploi du temps scolaire. Ce serait un système où on rentre des informations sur nos cours comme le lieu, les travaux à remettre, les examens et interrogations ainsi de, pourquoi pas, un système de prise de note directe ou de stockage de document intégré.

7) Conclusion

a. Hulsen Sorn

Mon ressenti personnel sur ce travail est assez mitigé.

J'ai réellement adoré travailler sur ce projet en tant que tel. C'était très intéressant pour moi de me lancer dans la réalisation de quelque chose de personnel qui me demande de me trouver des solutions par moi-même. Quand je regarde derrière moi, je trouve impressionnant de voir l'avancée que j'ai fait en Java. Ce projet m'a permis de tester et de comprendre une grande partie de la matière vue en cours alors que, durant les périodes en classes, je me demandais comment j'allais retenir tout ça. J'ai même eu la possibilité d'apprendre l'implémentation de JDBC dans un programme, chose que le cursus ne prévoit pas. Bien que cette partie du travail ait été un vrai challenge, je peux aujourd'hui dire haut et fort que je sais connecter un petit programme Java à une base de données. Je pense que cet aspect de langage va me resservir au cours de ma vie.

J'ai également adoré travailler en groupe même si toute collaboration laisse des moments de conflits d'idées. Cet aspect du travail est, je trouve, assez significatif de ce qu'est un vrai développement de programme. Je ne pense pas que je serais un jour confronté à réaliser un programme seul, ce qui me fait apprécier l'aspect groupe du projet. Je le vois comme un entrainement à la vie active assez réaliste.

Je n'ai cependant pas tout aimé dans ce projet. Nous avons dû choisir notre sujet au tout début de l'année, au moment où le Java pour nous ne consistait qu'à créer des classes et à faire des méthodes `toString()`. C'est très dur d'imaginer un projet sans savoir ce qu'on va être capable de réaliser par la suite. C'est aussi fort compliquer d'apprendre des notions d'un langage en même temps que notre projet avance, voir trop tard.

Peut-être que le problème vient de nous, mais je pense que si on avait eu une connaissance un peu plus approfondie du Java nous aurions su mieux accorder notre cahier des charges. Nous avions imaginé un programme avec des fonctionnalités qui nous semblaient simples mais, en avançant, nous avons dû en laisser de côté.

Je tiens à remercier mes deux camarades, Nathan et Martin qui ont fourni un travail remarquable. J'ai quelque regret personnel au niveau de mon engagement dans le projet. J'ai réellement pris conscience de la charge un peu tard. J'espère que les manques de notre calendrier ne sont pas le résultat de ce retard que j'ai pris.

b. Vanderdonck Nathan

Plus que simplement mitigé comme mes celui de mes camarades, mon avis sur ce projet, ses implications dans notre apprentissage et les enseignements à en retirer, est plus tranché dans mon esprit.

Je suis fier des enseignements que j'ai pu tirer de ce projet, à commencer par la connaissance du code. Le cours théorique, bien que très intéressants, manque parfois de lien direct, de mise en pratique sûre et rapide, pour en tirer des implications précises et concises dans nos esprits. Mais, tout comme je l'imagine que cela se passera dans la vie professionnelle qui nous attend, quel serait l'amusement à avoir les réponses toutes faites ? C'est pour cela que je suis fier d'être arrivé à fournir un projet pareil, d'avoir appris à créer, gérer, modifier, implémenter du code, surtout à cette échelle.

Plus que les connaissances du langage utilisé, je tire aussi de nombreux enseignements de ce travail de groupe, justement parce qu'il était à faire en groupe. Apprendre à gérer son temps, ses relations, l'avancement des autres, ses propres avancées, les attentes des autres par rapport à son propre code, … Toute ces choses à apprendre à gérer au fur et à mesure de l'avancée du projet que je suis aussi fier d'avoir appris pendant le projet, même si j'imagine qu'il nous reste encore beaucoup à apprendre, autant individuellement qu'en groupe avec d'autres personnes.

Je regrette néanmoins, malgré la fierté de m'être impliqué autant dans le projet et d'en avoir tiré autant en tant que programmeur, le manque de préparation, pas tant au niveau du cours, que du manque de suivi parfois ressenti lors du lancement du projet. Je suis convaincu que, plus qu'un suivi sur l'avancement des informations générales sur le

projet, les élèves auraient profité d'un suivi sur le projet en lui-même, de son évolution, et de la gestion de l'UML. Je crois aussi que les élèves auraient pu profiter d'un survol de l'utilisation de Git, pour éviter tout « cafouillage » lors de sa mise en place.

Pour finir, je tiens à remercier mon groupe pour les efforts fournis lors de ce travail. Martin, pour sa mise en place acrobatique du modèle MVC, loin d'être simple à gérer, et son aide tout au long du projet. Et Sorn, pour ses recherches extensives sur la façon de connecter Java une DB, qui, même si elles n'ont pas été des plus rapides lors de leurs mises en œuvre, nous ont été critiques pour la réalisation du projet. Je les remercie pour leurs complémentarités caractérielles et leur travail.

c. Martin Petit

Tout comme Sorn, j'ai des avis partagés sur ce projet. Tout d'abord je suis embarrassé de rendre un projet sans GUI du tout … Nous avons manqué de temps pour implémenter cela mais je pense que c'est parce qu'on a pris du temps à trouver notre rythme.
Je crois que c'est dû au fait que nous avons dû apprendre trois nouvelles matières en parallèle : JAVA, git (et github) et simplement la gestion d'un gros projet en groupe.
De plus, la gestion de la DB avec java nous a coincé quelque temps et surtout des choix différents auraient été fait dans la structure du projet si on avait su comment travailler avec une DB avant.
Il était difficile d'appliquer la matière vue en TP sur un projet plus conséquent (peut-être trop conséquent pour un premier projet ?).
Cependant, la courbe de progression qu'on a eu s'est débloquée à un moment, sans doute trop tard pour le projet et nous finissons un peu frustrer de finir ce projet là-dessus car le concept de base nous plait vraiment et nous pensons qu'il peut être réellement intéressant.
En conclusion, je pense que nous n'avons pas rempli le cahier des charges fixés au départ, vu qu'il nous manque une interface complète et je trouve ça dommage car il nous aurait suffi de quelque jour en plus que pour que l'application soit complète. Cependant je suis relativement satisfait d'où on en est arrivé malgré tout, et surtout de l'ouverture que cela nous procure sur ce qu'il est possible de faire avec nos connaissances dans le futur.
Merci à mes deux collègues de groupe ! Je pense qu'on s'est révélé pas mal complémentaires au final même si ce n'est pas toujours facile de travailler à plusieurs, de séparer les tâches à faire et rassembler par après le travail effectué. Recommencer un projet à nous trois serait clairement plus facile maintenant !