Bilan de gestion d'équipe et de projet

Janvier 2022

TEIMUR ABU ZAKI, ADRIEN BOUCHET, TROY FAU, PAUL MARTHELOT, HUGO ROBERT

1 Organisation

1.1 Méthode de travail

Dès le premier cours de méthode de gestion de projet, nous avons pris conscience de l'importance du choix d'une méthode de travail adaptée au projet ainsi qu'à l'équipe. Deux membres de l'équipe étaient membres de la Junior-Entreprise et étaient déjà sensibilisés aux méthodes Agile. Avant de commencer à réfléchir à la méthode que nous allions mettre en place, nous avons décidé que chacun prendrait connaissance de l'intégralité du sujet avant le début du projet.

Une fois que chacun eu pris connaissance du sujet, nous avons décidé d'adopter une méthode de travail à la frontière entre **cycle en V** traditionnel et méthode **Agile**. Les méthodes Agile sont particulièrement intéressantes dans le cas des projets pour lesquels la spécification n'est pas définie dès le début ou est vouée à être modifiée au cours du développement. Ce n'est pas du tout le cas du mythique projet GL, qui, malgré quelques modifications mineures, reste fidèle à sa réputation. Nous avons donc considéré que les spécifications étaient suffisamment détaillées et qu'il n'était pas nécessaire de refaire un travail de séquençage des tâches sous forme de backlog et user stories.

Cependant, les professeurs ont insisté sur l'importance de développer le projet par incréments successifs. Chaque incrément représente alors un sous-ensemble du langage Deca. Le sujet ainsi que le rendu intermédiaire nous poussent aussi à développer le compilateur par sous-ensemble.

C'est pourquoi nous avons choisi de réaliser le projet sous forme de cycles en V itératifs de courte durée. Chaque cycle se compose de 4 phases :

- **Analyse** ;
- **Ebauche de code** ;
- **Tests** ;
- **Validation** .

Les membres de l'équipe ont été assignés par binôme sur chacune des phases. Les phases de tests et d'écriture de code n'étaient jamais réalisées par les mêmes personnes, ce qui nous a permis de mieux respecter les spécifications. Les tests de la partie Sans-Objet qui se sont révélés efficaces à l'issue du rendu intermédiaire ont donc été repris pour tester l'implémentation de l'extension et la génération d'assembleur ARM. L'étape de validation se finissait lorsque les résultats aux tests étaient concluants et que le code avait été relu.

1.2 Outils mis en place

Pour suivre le projet, nous avons choisi de mettre en place un tableau Trello. Voici le bilan que nous en faisons à l'issue du projet :

Points positifs :

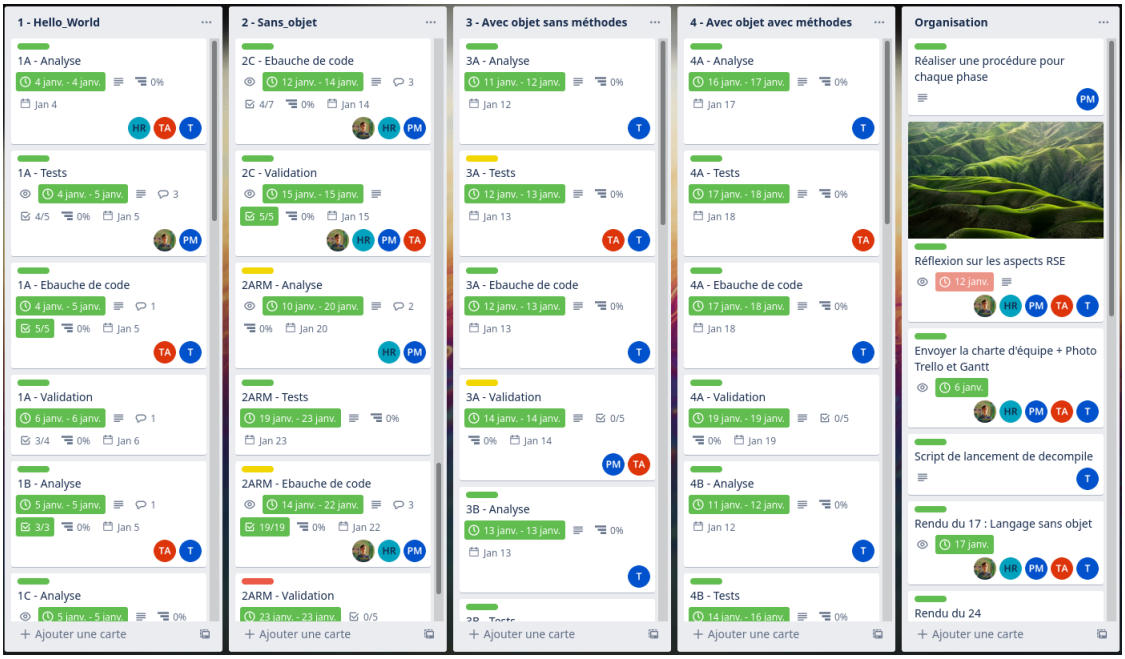

FIGURE 1 – Tableau Trello, équipe 49 au 25/01/2022.

- Facilité d'utilisation ;
- Possibilité d'attribuer des tâches aux membres de l'équipe ;
- Plug-in de diagramme de Gantt relié au Trello ;

Points négatifs :
- Difficile de s'y retrouver parmis toutes les sous-tâches ;
- Edition du diagramme de Gantt limitée à 3 utilisateurs en version gratuite ;

La figure 1 montre notre tableau à l'issue du projet.

2 Historique du projet

Nous avons scindé le projet GL en plusieurs grandes phases qui correspondent à des sous-ensembles du langage Deca.
- HelloWorld ;
- Sans-Objet ;
- Avec Objet sans méthodes ;
- Avec Objet et méthodes ;

Pour chacun des sous-ensemble s'appliquait la méthode en 4 phases décrite ci-dessus. La rédaction de la documentation a commencé une semaine avant le rendu final et s'est poursuivi jusqu'à la soutenance.
Analyse : 50 jours cumulés.
Tests : 31 jours cumulés.

Code : 37 jours cumulés.
Documentation : 27 jours cumulés.

Nous avons donc a 5 un total de 145 jours de travail ce qui correspond a 29 jours de travail par étudiant ce qui revient à peu près à la durée du projet GL. On remarque que c'est de loin l'analyse qui nous a pris le plus de temps. Les tests ont pris moins de temps que l'écriture du code, ce qui n'est pas un point très positif au vu de l'importance des tests pour détecter les bugs et valider le code. Cela s'est ressenti notamment au niveau de la partie du langage avec objets et méthodes où nous avons investi beaucoup de temps sur l'analyse au détriment des tests.

3 Critiques

Dans l'ensemble, nous pensons que nous avons bien défini notre organisation du travail, et qu'elle a été le bon choix pour mener à bien le projet. Les points faibles de notre gestion de projet ne concernent pas l'organisation générale mais plutôt notre respect des règles établis, et une sous-estimation du niveau de spécialisation qui allait se développer parmi les membres de l'équipe.

3.1 Spécialisation avec manque d'échanges techniques

Même en ayant pris connaissance en amont du sujet, il était difficile de comprendre le projet dans sa globalité avant de commencer l'implémentation. Cela impliquait un certain niveau de spécialisation des membres de l'équipe. Ceci n'était pas un problème en soi, d'autant plus qu'il était prévu que les revues de code et les tests d'une fonctionnalité ne soient pas effectués par les mêmes personnes qui avaient codé cette fonctionnalité.

Ce qui s'est révélé problématique a été le manque d'échanges sur le plan technique entre les membres s'occupant de différentes parties du projet, ce qui pouvait ralentir le développement, en particulier pour l'étape C de la compilation en bout de chaîne. Les réunions du matin permettaient de se tenir au courant de l'état d'avancement du projet et des tâches de chacun, mais ne rentraient pas dans les détails techniques.

Solution : à l'avenir, faire une réunion technique tous les deux/trois jours, pendant laquelle les membres de l'équipe échangent sur l'implémentation de leurs parties respectives.

3.2 Pas assez de revue de code

Nous avions mis en place un protocole de revue de code qui n'a été respecté qu'au début du projet. Une fois qu'un membre de l'équipe ou un binôme avait codé une partie du projet (*e.g.* la partie A Sans Objet), un autre membre de l'équipe devait relire le code et le valider.

Mais lire avec attention du code écrit par quelqu'un d'autre est long et ennuyeux, et donc cette étape a eu tendance à passer à la trappe. Nous avons commencé à utiliser les tests écrits en parallèle comme unique outil de validation, ce qui a bien fonctionné en général, mais certains problèmes plus subtils n'ont pas été détectés par les tests. Une bonne relecture du code aurait peut être éliminé le mal à la racine.

Solution : Être plus rigoureux sur la revue de code, et la faire systématiquement. Cela permettrait aussi à une personne s'occupant d'une autre partie de mieux connaître une partie dont elle valide le code.

3.3 Pas assez de tests de la partie Objet

Nous avons été moins systématique sur la génération de tests pour cette partie du projet. Certes, c'était une partie plus complexe, mais on pouvait automatiser la génération de tests pour certains aspects spécifiques, comme les initialisations d'objet, les appels de méthodes avec des paramètres différents, des surcharges de méthodes avec différentes signatures, etc. Nous ne l'avons pas fait, et les tests écrits à la main n'ont pas couvert tous les cas.

Un autre problème a été la non réutilisation de tests basiques de contexte en tests de génération de code. Ainsi, des fonctionnalités qui fonctionnaient parfaitement au niveau de l'étape B (*parsing* et décoration de l'arbre abstrait représentant le programme Deca) pouvaient avoir des problèmes au niveau de l'étape C (génération de code), mais nous l'avons découvert avec retard, ou même dans quelques cas seulement après la date de rendu du compilateur.

Solution : rester systématique sur les tests : élaborer des scripts générant des tests pour *tous* les cas de figure. Une « revue de code » des tests serait également utile : les tests ne doivent pas être écrits par ceux qui ont codé, mais ceux-ci devraient passer tous les tests en revue. Comme ils ont implémenté la fonctionnalité, ils en connaissent les subtilités et peuvent repérer des points précis n'ayant pas été testé.

4 Conclusion

Teimur

Vu qu'on a commencé a lire le poly du projet pendant les vacances, j'ai eu une image globale de ce qu'on va faire. Mais au début du projet je me suis trouvé devant les parties A et B en train de chercher les trous, à identifier ce qui a était déjà implémenter dans le code fourni ducoup c'était intéressant, parce que vu la vision des tâches (HelloWorld -> SansObjet -> Objet) ça m'a permis d'avoir une vue technique de l'implémentation du compilateur. Donc après avoir fini la partie A et B de la partie SansObjet, je me suis mis sur les tests pour bien valider ce qu'on a fait,

notamment des scripts pour créer des tests en deca, mais enfin je trouve que je n'ai pas utilisé ce script assez à cause de la peur d'avoir beaucoup de tests sans buts concrets, mais après la correction de plusieurs bugs je pense ça aurait été plus judicieux de passer un plus de temps de créer des scripts qui cibles exactement des problèmes (qui peuvent être identifiés en revue de code).
Pendant la dernière semaine, je me suis mis sur la partie ARM, parce que je n'avais pas assisté assez au developpement de la partie C, et l'extension était surtout sur cette partie là. Après quelques heures de recherches et 2 jours de developpement en binôme je trouve qu'on était super efficaces avec ce qu'on a pu faire .
À la fin de ce projet, je trouve que l'importance de communication (surtout technique) entre les personnes qui travaillent dans des tâches différentes mais qui sont en relation (comme A et B avec C) est très élevée, et que le developpement en binôme doit être aussi mis en priorité à un certain point vu que ça peut beaucoup augmenter l'efficacité et la qualité du code. Sinon je suis satisfait avec le niveau d'adaptation et cohérence de l'équipe.

Adrien

Même après avoir lu la spécification avant le début du projet, j'ai eu du mal à comprendre l'architecture du projet et l'articulation des parties A, B et C. J'avais le rôle de chef de projet, ce qui consistait à travailler sur l'organisation temporelle, la mise en place des outils de gestions de projet ainsi que la rédaction de comptes rendus des réunions que j'animais quotidiennement. Ce travail au début du projet a été utile au groupe pour se structurer et s'organiser, ce qui s'est vu lors des premières séances de suivi. Ce temps m'a permis de prendre du recul sur le projet et de ne pas être plongé dans le code directement sachant que je ne comprenais à ce moment là pas assez bien, ce qui aurait donc pu ralentir l'équipe.

J'ai fait l'erreur de commencer à coder la partie ARM seul au milieu du projet car nous avions du retard sur cette partie. Ce choix n'a pas été judicieux tout de suite car un membre de l'équipe était alors seul sur la partie de génération de code pour les objets et méthodes. Si c'était à refaire, je pense que j'aurai favorisé le codage en binôme à ce moment là du projet en privilégiant la partie commune pour ensuite revenir sur l'extension. Cependant, en fin de projet, après avoir mis en place l'architecture de génération de code pour ARM, j'ai pu tirer profit du temps investi en compréhension de l'assembleur ARM pour coder les fonctionnalités du langage Sans-Objet. J'ai aussi pu m'appuyer sur mon travail sur l'extension lors de la rédaction de la documentation de l'extension.

Troy

Pendant mes études j'ai eu l'occasion de participer à pas mal de projets informatiques, mais tous se préoccupaient quasi exclusivement des aspects techniques. C'était donc intéressant d'aborder une UE qui imposait de réfléchir à une vraie gestion de projet, de définir une organisation à l'avance et de s'y tenir le plus possible.

Après avoir codé les étapes A et B de la partie Objet, je me suis attelé, en plus de la documentation, à la résolution de bugs. Mais j'ai fait cela de façon un peu passive, en attendant le feedback de mes collègues chargés des tests. Il aurait probablement été plus judicieux, comme il a été dit dans la section « Critiques » de faire une revue active des tests pour combler les manques, car je connaissais les subtilités de l'implémentation. J'ai pu néanmoins en toute fin de projet corriger quelques bugs qui étaient passé entre les filets.

La communication entre les membres de l'équipe responsables de l'étape B et ceux responsables de l'étape C n'a pas été assez continue. J'aidais quand un problème se manifestait, mais là aussi il aurait fallu avoir un rôle plus actif, en s'imposant par exemple des réunions techniques pour expliquer plus en détail l'*output* de l'étape B.

Dans l'ensemble, je suis quand même satisfait de notre organisation du travail, que nous avons respecté dans les grandes lignes jusqu'à la fin.

Paul

Lors de cette période, j'ai surtout joué un rôle d'encadrant dans l'équipe : je n'ai que rarement touché aux parties A, B, C et à l'extension mais j'ai pu apporter de l'aide en identifiant la plupart des classes dont nous allions avoir besoin et en les intégrant dans la structure de code fourni, le plus rapidement possible (par exemple : le gestionnaire d'erreurs, l'analyse syntaxique des options du compilateur, la gestion des registres, etc.). J'ai également initié l'écriture des scripts de tests avant que Teimur ne reprenne le flambeau et automatise encore plus ce processus. Sur la fin du projet, voyant que je ne pouvait plus être très utile aux membres déjà spécialisés sur leurs parties, j'ai rédigé des tests pour la génération de code et lancé les scripts de validation. D'autre part, j'ai veillé à m'informer le plus vite possible sur les spécifications globales du sujet (sauf peut-être la partie B) pour pouvoir répondre aux questions des membres qui n'étaient pas sûr d'eux. Enfin, lorsqu'Adrien est resté bloqué sur la gestion des flottants, j'ai essayé d'identifier des solutions en faisant des recherches sur Internet. Malheureusement, je n'ai réussi à faire fonctionner le compilateur croisé comme je l'aurais voulu et j'ai dû laisser ma place à Teimur.

Ces trois dernières semaines m'ont fait prendre conscience de l'étendue du travail à fournir sur un projet d'une telle envergure, assez différent des autres que nous avons pu vivre à l'Ensimag. D'abord, l'aspect gestion de projet et d'équipe a joué un rôle très important car il fallait d'abord répartir les tâches aux différents membres en essayant de satisfaire au mieux les différentes parties. J'ai toutefois pu être témoin des problèmes générés par l'ultra-spécialisation de Troy et Hugo sur leurs parties et donc de mon incapacité à résoudre des problèmes directement dans le code. En revanche, il s'est avéré possible de pouvoir discuter régulièrement des avancées sur le projet et de s'accorder avec Hugo sur la partie C quant aux structures de données que nous allions utiliser.

Si je devais reprendre ce projet depuis le début, je pense que je demanderais à l'équipe de prendre plus de temps pour prendre du recul sur le code et les spéci-

fication, et je reprendrais le concept de programmation en binôme sur les parties conséquentes (B, C et ARM).

Pour conclure, je suis heureux d'avoir travaillé avec cette équipe et j'ai pu avoir un aperçu de ce qu'est une équipe motivée et compétente. Mes remerciements à eux d'avoir respecté la charte d'équipe !

Hugo

Ce projet à été pour moi une première expérience pour me confronter directement à un problème technique que nous devions résoudre en équipe, et ce à temps plein durant environ 1 mois.

La première difficulté pour moi fut d'appréhender le sujet et de comprendre ce qui était attendu. Je retiens tout d'abord qu'un des bons réflexes que nous avons eu à été de lire les documentations et cours en amont pour se confronter directement au code dès le premier jour de projet. Nous avons par la suite su maintenir une organisation efficace en attribuant les rôles de chacun des membres chaque jours.

J'ai personnellement été en charge du code de la partie C et j'ai été directement confronté aux difficultés de communication entre la partie B et C, la principale leçon que j'ai donc appris de ce projet est l'importance de s'imposer des réunions techniques à intervalle régulier pour garder une compréhension globale de l'avancé du projet et empêcher de phénomène de spécialisation des membres du groupe dans un certain domaine.

Un de mes regrets cependant à été de m'être spécialisé en partie C au lieu de travailler en binôme sur la Partie C pour ensuite gagner beaucoup de temps sur la partie ARM.

Aujourd'hui je ressort de ce projet avec une vision plus claire de l'organisation au sein d'une équipe pour mener un projet et une capacité à comprendre d'un point de vue global une hiérarchie de code complexe. Enfin, Je retiens vis à vis de cette expérience passé l'importance d'une organisation solide avec des membres rigoureux sur leurs manières de respecter les délais tout en produisant un travail de bonne qualité.