Bilan Gestion de Projet Projet GL

Groupe 44

COLIN-SOHY Mathéo

DIMITRIOU Tristan

EL HIT Oumayma

GAUX Antoine

HARRAUD Paul-Louis

Table des matières

1 Introduction **2**

2 Présentation de l'équipe **2**
2.1 Compétences individuelles . . . 2
2.2 Compétences collectives . . . 3

3 Organisation de l'équipe **3**
3.1 Les règles imposées . . . 3
3.2 Rôles et Responsabilités . . . 3
3.3 Communication au sein de l'équipe . . . 4
3.4 Gestion des risques pour les rendus . . . 4
3.5 Construction du planning prévisionnel . . . 4

4 Historique du projet **5**
4.1 Étape A . . . 5
4.2 Étape B . . . 5
4.3 Étape C . . . 5
4.4 Extension . . . 5

5 Retour sur notre gestion de Projet **6**
5.1 Observations générales . . . 6
5.2 Rendu de l'extension . . . 7

6 Bilans Personnels **7**
6.1 COLIN-SOHY Mathéo . . . 7
6.2 DIMITRIOU Tristan . . . 7
6.3 EL HIT Oumayma . . . 8
6.4 GAUX Antoine . . . 8
6.5 HARRAUD Paul-Louis . . . 8

1 Introduction

Ce document a pour but de présenter la gestion de projet effective durant le projet Génie Logiciel. Il présentera une analyse critique sur la gestion du projet d'un point de vue a posteriori. Les points suivants seront abordés :

— Présentation de l'équipe à partir de la charte
— Organisation adoptée dans l'équipe
— Historique du projet
— Critique de nos méthodes de gestion de projet
— Bilans

2 Présentation de l'équipe

Cette section a pour but de présenter brièvement les membres de l'équipe GL44, à partir des éléments de la charte d'équipe. Ce rappel nous semble primordial afin de mieux saisir le retour fait a posteriori

2.1 Compétences individuelles

Suite à une évaluation individuelle, on obtient les niveaux suivants pour les compétences utiles au projet GL :

Antoine Gaux
2A MMIS ENSIMAG
groupe 8, gl44

Leadership	3
Planification	2
Esprit d'équipe	3
Organisation	2
Persévérance	4
Ponctualité	
Créativité	4
Débrouillardise	3
Orateur	3
Communication écrite	3

Java	4
Assembleur	2
Qualité du code	3
Clarté du code	3
Testing	2
Git	3
TL	2

Tristan Dimitriou
2A SEOC
groupe 8, gl44

Leadership	4
Planification	3
Esprit d'équipe	3
Organisation	3
Persévérance	5
Ponctualité	4
Créativité	3
Débrouillardise	3
Orateur	3
Communication écrite	4

Java	4
Assembleur	2
Qualité du code	3
Clarté du code	3
Testing	3
Git	2
TL	4

Mathéo ColinSohy
2A MMIS ENSIMAG
groupe 8, gl44

Leadership	2
Planification	4
Esprit d'équipe	3
Organisation	3
Persévérance	5
Ponctualité	4
Créativité	4
Débrouillardise	4
Orateur	2
Communication écrite	4

Java	4
Assembleur	3
Qualité du code	3
Clarté du code	3
Testing	3
Git	2
TL	3

El Hit OUMAYMA
2A MMIS
groupe 8, gl44

Leadership	3
Planification	4
Esprit d'équipe	4
Organisation	3
Persévérance	4
Ponctualité	4
Créativité	3
Débrouillardise	2
Orateur	4
Communication écrite	4

Java	3
Assembleur	2
Qualité du code	3
Clarté du code	4
Testing	3
Git	3
TL	1

Paul-Louis Harraud
2A SEOC
groupe 8, gl44

Leadership	3
Planification	3
Esprit d'équipe	2
Organisation	2
Persévérance	2
Ponctualité	4
Créativité	4
Débrouillardise	4
Orateur	3
Communication écrite	3

Java	4
Assembleur	1
Qualité du code	4
Clarté du code	3
Testing	4
Git	4
TL	1

FIGURE 1 – Compétences individuelles des membres l'équipe

On peut voir que les compétences sont proches et les profils se ressemblent un peu. La lacune principale de l'équipe repose surtout sur la partie assembleur. On peut alors de manière naturelle définir la matrice SWOT.

2.2 Compétences collectives

Matrice SWOT

Forces	Faiblesses
Equipe motivée Sereine face à la difficulté que peut présenter un code Bonne communication	Heures de travail décalées entre les étudiants (distanciel)
Opportunités	**Menaces**
Mener un bon projet, efficace et robuste Apprendre le travail d'équipe (Méthode agile)	Temps imparti face à la taille du projet Les difficultés en Assembleur

FIGURE 2 – Matrice SWOT

3 Organisation de l'équipe

3.1 Les règles imposées

Les règles suivantes ont été énoncées dès le début du projet. Nous reviendrons sur ces règles dans la partie critique

- Pouvoir parler de tout sans porter de jugement irrespectueux, mais plutôt répondre de manière constructive afin d'alimenter le débat / solutionner le problème.
- Ne pas rester bloqué dans un problème seul, mais demander de l'aide aux autres membres afin d'en recevoir dans la bienveillance.
- Tout problème, même résolu, doit être consigné dans le log prévu à cet effet sur le salon discord.
- Être présent aux réunions d'équipe : 1 fois par jour en distanciel et 1 tous les 3 jours en présentiel, selon l'emploi du temps.
- Respecter le coding Style de java (CamelCase), ainsi que produire le code nécéssaire à la création de JavaDoc (en anglais)
- Partager ses ressources et son savoir-faire.
- Le travail le week-end n'est pas imposé, mais sera probablement nécessaire.

3.2 Rôles et Responsabilités

Pour ce projet, nous avons décidé d'avoir deux chefs de projets : Tristan et Paul-Louis. Chaque décision importante est alors prise à deux, ce qui permet de confronter au mieux les avis afin de choisir la bonne option lors des conflits. Paul-Louis à également été nommé responsable Git, justifié par son savoir-faire dans ce domaine. Tout au long du projet, chaque membre a été amené à travailler sur plusieurs partie du projet (A, B, C, Test & Extension) selon l'avancement de ce dernier. C'est pour cela que pendant la première semaine, chaque membre s'est focalisé particulièrement sur une partie afin d'en devenir le responsable :

- A : Tristan
- B : Mathéo
- C : Antoine
- Tests : Paul-Louis
- Extension : Oumayma

Chaque personne aura alors la responsabilité du développement dans sa partie. Les chefs d'équipes auront pour rôles de guider les membres de l'équipe dans leurs tâches. Les rôles de chacun sont considérés fixes, mais des changements ne sont pas impossibles (même probables) en cas de difficultés sur certaines parties.

3.3 Communication au sein de l'équipe

La communication au sein de l'équipe est principalement assurée via Messenger ou Discord. Les réunions quotidiennes servent à relater les différentes difficultés rencontrées afin que chacun puisse donner son avis sur la question. Les réunions à distance sont réalisées via Discord, et les réunions en présentiel ont lieue à l'ENSIMAG. Le partage de la documentation type Latex est assurée via le site Overleaf, qui permet une édition commune des différents documents facilitant le flot d'écriture et d'apport d'informations. Les documents un peu moins formels mais utiles à tous sont dispensés sur le Discord dans un salon attribué. Si besoin est de l'éditer, alors nous pourrons utiliser Google Drive qui permet également l'édition simultanée.

Les réunions seront animés par Tristan et/ou Paul-Louis. Le terme "animé" sous-entend de prévoir l'ordre du jour afin de ne pas négliger les points importants. L'ordre du jour sera établit en fonction du précédent (résolution des problèmes) et chacun pourra en ajouter de nouveaux si besoin est.
Concernant les tensions qui auraient pu apparaître sur une divergence d'opinion, nous avons opté pour la manière suivante : L'ensemble du groupe discute à propos du problème et de ses différentes facettes pendant un temps fixe, et à l'issue de ce délai, le groupe réalise un vote pour savoir quelle solution sera retenue (le fait d'être un groupe de 5 assure une décision).

3.4 Gestion des risques pour les rendus

Afin de réduire les risques d'erreurs lors des différents rendus, nous avons opté pour la stratégie suivante :

Risque	Conséquence	Action
Louper la date de rendu	Branche Master avec une version/vide	Chaque date de rendu a été notée dans le Planning. De plus, un événement à l'heure du rendu est mit en place sur Discord
Erreur(s) bloquant la compilation	Compilateur inutilisable	Le responsable de rendu effectue un clone sur un dossier différent de son répertoire de travail, afin de simuler un clonage du côté professeur/client. Il vérifie alors que la compilation fonctionne sans erreur et que le script de test général se déroule sans problème
Branche GIT de rendu qui n'est pas la dernière version	Manque de fonctionnalités ou Compilateur inutilisable	La branche master est réservée pour les rendus : Un seule fusion de branche par rendu. Si une fusion à eu lieu, c'est que la branche master possède la dernière version. Il suffira alors de d'assurer que cette fusion à eu lieu.
Print de debug qui n'ont pas été effacés	Messages intempestifs	On interdit l'utilisation du "System.out.Println()" pour utiliser le Logger. Via le logger, on peut désactiver les messages affichés via la config du fichier log4j.properties.On est alors assuré qu'aucun message de debug ne sera affiché lors des différents tests/exécutions.
Non réussite d'un programme fourni	Fonctionnalité exigée non implémentée	Couverture par le script de test général qui est censé couvrir tous les tests.
Dépendance(s) par rapport à des fichiers locaux	Certaines fonctionnalités	Le responsable de rendu effectue un clone sur un dossier différent de son répertoire de travail, afin de simuler un clonage du côté professeur/client. Il vérifie alors que la compilation fonctionne sans erreur et que le script de test général se déroule sans problème

FIGURE 3 – Gestion des risques adoptée par l'équipe

3.5 Construction du planning prévisionnel

Afin de construire le planning prévisionnel, chaque responsable présentait un découpage des différentes tâches associées à sa partie avec un nombre de la suite de Fibonacci représentant l'importance à accorder (en fonction de la complexité) à cette tâche. Le responsable devait alors ensuite justifier son choix de pondération aux autres membres de l'équipes en expliquant les différents aspects techniques. Voici quelques exemples :

— Lexer : 1
— Parser (toutes parties confondues) : 5
— Etape B Passe 1,2 & 3 : 13
— Etape C (Avec Object) : 21

Tout au long du document, on note les semaines de la manière suivante : Semaine 1(03/01 au 10/01), Semaine 2 (10/01 au 17/01) & Semaine 3 (17/01 au 24/01)

Au fur et à mesure des discussions, 3 sprints d'une semaine ont été crées :
— Language HelloWorld : Semaine 1
— Language sans Objet : Semaine 2
— Language avec Objet : Semaine 3

En terme de quantités de fonctionnalités, ses sprints à durées égales peuvent paraître déséquilibrés. Mais nous savions que notre compréhension du projet au fur et à mesure de notre avancement nous rendrait de plus en plus efficace dans le développement. Le découpage de ses sprints en tâches a alors été saisi sur Planner, en fonction des coefficients attribués à chacune d'elles.

4 Historique du projet

4.1 Étape A

Le déroulement de l'étape A n'a posé aucun problème. Le lexer a été terminé dès la première semaine (quelques modifications ont été nécessaire par la suite, comme par exemple la mise en place d'erreurs). La partie sans objet a été terminée au milieu de la semaine 2. Il a été décidé, contrairement au planning prévisionnel d'enchaîner directement avec la partie avec-objet afin que la partie B et C, nécessitant les sources de l'arborescence Tree (DeclClass, DeclMethod, This...), puissent attaquer au plut tôt après la fin du sans-objet. Le parser (et ainsi l'étape A) était alors terminé le 13/01.

4.2 Étape B

Le déroulement de la partie B, bien que conséquent, n'a également pas posé de problème majeur.

Pour le langage hello-world, il a été terminé à temps après une période de compréhension de l'architecture relativement longue, mais nécessaire et qui s'est avérée très utile par la suite.
Durant la seconde semaine, avec l'échéance du langage sans-objet, le développement de l'analyseur contextuel a pu suivre le planning prévisionnel et atteindre l'objectif du rendu le vendredi 14/01, permettant alors la création de nombreux tests, de scripts pour les exécuter et ainsi la détection de bugs durant tout le week-end.
Pour l'échéance finale du langage avec-objet à la fin de la 3ème semaine du projet, l'extension de l'analyse contextuelle qui paraissait importante au début du projet, fut globalement aussi complexe que le reste du travail sur les deux semaines précédentes, grâce à l'expérience et la connaissance acquise sur le corps du compilateur. Les plus grandes difficultés qui se sont donc présentées furent plus des temps de réflexions à propos des environnements, leurs définitions et leurs interactions que le reste. Dès lors que ces points ont été éclaircis, et après un ajout au squelette de code fourni, le reste de l'implémentation a pu se terminer dans la journée du samedi précédent le rendu, permettant à nouveau une phase importante de tests et de débogage avant le rendu final.

4.3 Étape C

La partie C s'est développée en accord avec le planning prévisionnel, à un retard constant de 2 jours près. Le début a été assez compliqué, car il était dur de deviner le fonctionnement général du code. Il nous a fallu environ 5 jours à partir du début du projet pour générer les premiers fichiers.
Ensuite, étant donné que nous avons fonctionné en méthdoe AGILE, la répartition du travail était linéaire : génération de code pour le sous-langage "hello-world", puis sans-objet, puis avec-objet. Puisque implémenter une partie du langage peut révéler des bugs sur des parties ultérieures, on ne peut pas vraiment parler de version "finale" pour autre chose que le rendu final, mais on peut considérer qu'une partie est "globalement finie" lorsque les tests centrés sur cette partie passent. De ce point de vue, nous avons globalement fini le hello-world le 12/01 au soir, ce qui est un retard de 3 jours par rapport à nos prévisions. Ensuite, nous avons globalement fini le sans-objet le 18/01, ce qui est un retard de 2 jours compte tenu de nos prévisions. Il manque très peu à notre compilateur sur le langage avec objet, ce que représente ce retard.

4.4 Extension

Pendant la première semaine, après négociation nous avons fait le choix de l'extension : Trigo. Nous avons commencé à nous documenter sur les différentes façons d'implémentation des fonctions trigonométriques, nous nous sommes décidé de s'appuyer sur l'algorithme de Cordic pour aborder l'implémentation en Java puis deca. Il y avait des fonctions intermédiaires à implémenter également à l'instar de pow, sign, abs. Au final, on a

implémenté la fonction Ulp à l'aide de la méthode Kahan.
La deuxième semaine a consisté a coder CORDIC en Java, pour les quatre fonctions trigonométriques, c'était de toute facon la seule chose a faire puisque la partie objet de deca n'était pas finie, on a rencontré des difficultés, surtout avec arctan car les conditions et les comparaisons changaient. Une fois le problèmé trouvé, arcsin a été fait presque directement.
Enfin on a recopié le code java en deca dans la classe Math, les fonctions utiles pour cet algo, qui étaient relativement simples. On a aussi implémenté ulp en flottant simplé précision d'après un papier.
Tout ces efforts pour finalement aucun résultat dans le rendu final (cf 5.2)

5 Retour sur notre gestion de Projet

5.1 Observations générales

Premièrement, on peut remarquer que notre planning effectif (Realisation.pdf) est plutôt proche de notre planning prévisionnel. L'estimation des tâches et de leurs difficultés semblent alors avoir été un outil efficace pour répartir temporellement les jalons. Les principaux retards concernent la partie C, comme on s'y attendait.

Ces retards sont en partie liés à notre gestion de projet (et non pas forcément à l'aspect technique de l'assembleur). En effet, notre organisation en termes de "responsable" de chaque partie a eu pour effet un léger isolement de chacun vis a vis des autres parties. Il était compliqué pour quelqu'un d'autre qu'Antoine de venir l'aider sur la partie C qui nécessite une grande compréhension de la partie assembleur et de tout ce qui gravite autour.

Nous avions également décidé de travailler en majeure partie en distanciel, avec des réunions quotidiennes via Discord. Il s'est vite avéré que nous préférions le présentiel, qui apportait une efficacité bien plus importante, grâce notamment à une meilleure communication , plus simple et lus rapide que par messages interposés. C'est donc à partir du 12/01 (soit 1 semaine et demi après le début du projet) que nous nous sommes imposés les séances en présentiel quotidiennement. Généralement, nous nous donnions rendez-vous à 13h30 pour travailler plusieurs heures dans une salle de l'ENSIMAG.

Nous pouvons également revenir sur la génération de JavaDoc qui n'a pas été respectée par l'ensemble de l'équipe (contrairement à ce qui avait été énoncée au commencement du projet). Cette JavaDoc aurait pu permettre une meilleure compréhension des différentes parties pour les autres membres de l'équipe. Nous notons également une absence de code review par les pairs lors des différents commits, ce qui aurait permit une détection plus facile des différents bugs et donc un gain de temps sur les parties de débogage.

Concernant le rendu intermédiaire, il nous a permit de mieux rebondir pour la suite. En effet, suite aux échanges effectués avec le professeur, nous nous sommes rendus compte que notre base de test était insuffisante pour tester suffisamment notre compilateur. La partie A étant terminée au moment du constat, l'accent à alors été poussé sur les tests afin d'identifier un maximum de bugs au sein de notre code. Lors du rendu, la couverture de code testée évaluée par Jacoco était de 38%. Lors du rendu final, nous sommes parvenus à la monter jusqu'à 80% (ce qui a permis la résolution de nobmreux bugs). Nous estimons que ce rendu, par le retour qu'il nous en a été fait, à été très bénéfique pour l'aspect final du projet.

Cette équipe est issue d'un tirage aléatoire, nous n'avons donc pas choisi d'être ensemble. Malgré tout, nous gardons un très bon souvenir de notre entente. Aucun conflit n'est a déclarer, l'ambiance était très bonne ce qui nous a permis d'être efficace. Le rôle de chef de projet ne s'est pas vraiment fait "sentir", nous étions en général tous d'accord sur les différents points, le travail à exécuter en priorité. La maturité et la motivation de l'équipe fait que nous n'avons pas eu besoin de rappeler, ou de donner une date d'échéance pour que le travail soit fait à temps, ce qui a naturellement réduit le risque de tensions au sein du groupe.

Finalement, d'un point de vue gestion de projet, notre organisation semble un peu confuse et mélange quelques ingrédients des différents framework que l'on peut trouver en gestion de projet. Ceci est probablement du au fait que ce projet est notre première approche d'une réelle gestion de projet, et que nous ne maîtrisions pas vraiment les outils. Cependant, avec du recul, nous sommes satisfaits de notre organisation qui nous a semblé plutôt efficace. Notre planning prévisionnel n'était pas vu comme quelque chose à impérativement respecter, mais plutôt à essayer d'imiter au maximum pour éviter la montée de la pression. Le résultat est plutôt bon en comparant le planning prévisionnel et le planning effectif. On trace une tendance du burn down chart :

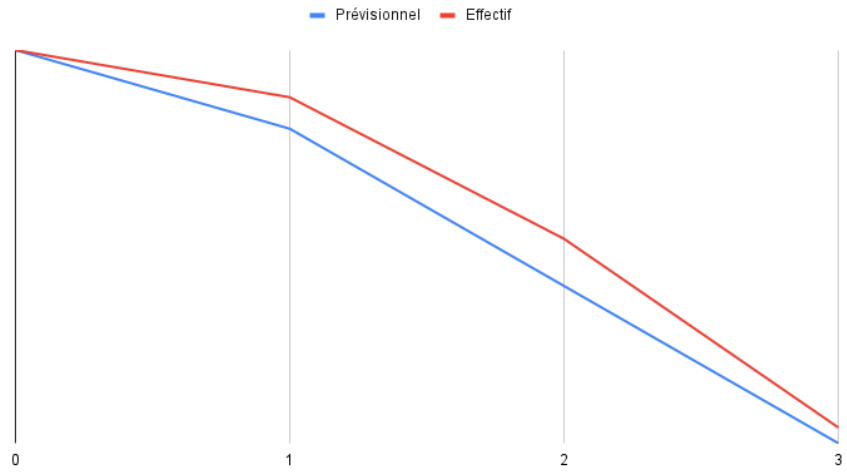

FIGURE 4 – Burn Down Chart

Le retard qui a été pris sur la partie C nous a poursuivi tout le long du projet, mais à pu être rattrapée partiellement avant la fin. La différence des deux courbes à la fin de la troisième semaine n'est pas nulle car certains tests n'étaient pas opérationnels lors du rendu.

Notre compilateur, qui n'est certes pas parfait, fonctionne plutôt bien avec les spécificités imposés par le Deca avec objet. Le gros point noir du rendu final réside dans le rendu de l'extension.

5.2 Rendu de l'extension

Après avoir merge toutes les branches et les quelques commits restants avant le rendu. Deux personnes ont cloné master et testé tout le projet (mvn clean compile verify), aucun problème n'a été relevé.
A cause du manque de tests de l'extension, le seul test de l'extension (dans provided) a été noyé dans les erreurs des autres tests et n'a pas été remarqué.
C'est seulement après le rendu que nous avons constaté que rien dans l'extension ne fonctionnait plus, alors qu'elle était terminée et fonctionnelle dans une autre branche vingt minutes plus tôt.
Pour mitiger ce risque, nous aurions pu vérifier plus consciencieusement, merge plus tôt, ou faire plus de tests sur l'extension. Ou bien les trois en même temps, ce qui nous aurait permis d'éviter cette erreur.

6 Bilans Personnels

6.1 COLIN-SOHY Mathéo

A posteriori, si ce projet m'a permis de me rendre compte de l'importance du travail de groupe, de la communication entre les membres et donc de la création de documentation afin de comprendre plus facilement le fonctionnement d'une partie du code pour des projets de grande envergure, il m'a également fait me rendre compte de mon affection pour le travail seul et indépendant, d'un bloc complet de code, faisant éventuellement partie d'un tout, mais sans avoir à dépendre d'autres morceaux de codes.
J'ai également beaucoup apprécié l'écriture des messages d'erreur du compilateur qui devaient être le plus clair possible, à la fois pour l'utilisateur afin de corriger simplement le code deca mais également pour les concepteurs en se rappelant à la grammaire contextuelle.
Une autre chose qui a été particulièrement agréable fut la correction de bugs : malgré le fait qu'on en souhaite évidemment le moins possible, j'ai trouvé que le fait de se voir indiqué une faiblesse depuis l'extérieur et de devoir aller reproduire, comprendre et corriger cette erreur très satisfaisant.
Enfin, également au niveau de la communication, j'ai apprécié apporter mon aide, même de manière marginale aux autres membres du groupes lorsqu'ils en avaient besoin, à l'instar d'Antoine afin de simplifier sa compréhension des environnements et de leur implémentation lorsqu'il en a eu besoin.

6.2 DIMITRIOU Tristan

Personnellement, j'ai trouvé cette expérience enrichissante, plus d'un point de vue gestion de projet que d'un point de vu technique. J'ai pu découvrir les enjeux du travail d'équipe tout au long de la réalisation. Je pense que ces compétences développées pendant ce projet me seront utiles dans ma future vie professionnelle dans les

différentes réalisations. La gestion de projet me parait encore un aspect difficile, mais ça viendra avec le temps et l'expérience. J'ai été responsable de partie A, puis je suis passé sur la partie Test, ce qui m'a permis d'utiliser plusieurs langages : ANTLR4, Java, Shell... J'ai beaucoup aimé le fait d'tre "libre" dans ses heures de travail, répartir comme bon nous semble. Au final, je pense que ça a bien fonctionné.

6.3 EL HIT Oumayma

Quant à moi, vu que je suis responsable de la partie extension(Trigo) qui est la partie la moins documentée dans le projet, j'ai consacré assez de temps pour se documenter, pour chercher les différentes manières possibles pour implémenter les fonctions en question, mais surtout les plus rapides et les plus efficaces en terme de précision. J'ai découvert pas mal de nouvelles choses en cherchant du coup j'ai beaucoup apprécié que cette partie est trop ouverte. Je suis satisfaite du travail au sein du groupe, de la manière dont on s'est organisé mais surtout du fait qu'on s'entraide, on demande de l'aide en cas de besoin,etc.Paul-Louis m'a rejoint pour finir l'implém Enation de l'extension, j'ai apprécié son aide.En gros, je me suis rendu compte que le travail en équipe est tellement efficace et beaucoup mieux qu'un travail individuel surtout pour un grand projet comme celui-ci. J'ai pas bien organisé du temps parcqu'au final, j'ai manqué de temps pour tester l'extension. J'ai apprécié qu'en deca, faut tout coder, notamment les fonctions intermédiaires qui sont déja implémentées dans les autres languages à l'instar de pow,abs. On apprend à tout coder, tout comprendre et c'est agréable !

6.4 GAUX Antoine

En regardant le projet maintenant qu'il est fini, je me rends compte d'un point très important des projets de cette taille. Je ne peux pas du tout tout faire moi même, ni même tout comprendre du fonctionnement. Je dois laisser une partie importante du code à mes coéquipiers, et je dois donc travailler avec cette "boîte noire". De la même manière, ce que je code n'est non seulement pas aussi évident pour moi que pour mes coéquipiers, mais ils auront d'autant plus de mal à comprendre quels choix j'ai fait et pourquoi. Et c'est pour ça que je regrette a posteriori de ne pas avoir correctement documenté mon code (j'estime que le niveau de commentaire est le minimum pour qu'une personne comprenne mon code).

Néanmoins, je suis très satisfait de ce que j'ai fait, j'ai beaucoup appris sur le fonctionnement d'un ordinateur et de l'assembleur, et j'ai encore plus appris sur la génération de code, et les limites de la compilations, de ce qu'on peut vérifier avant la compilation et de ce que l'on ne peut pas.

6.5 HARRAUD Paul-Louis

Ce projet était une bonne première expérience en gestion de projet logiciel, le fait qu'on ait formé un groupe d'inconnus était aussi intéressant, ca représente mieux le cadre professionnel qu'un projet scolaire.
Au départ, j'avais choisi le rôle d'auteur de tests. Je n'ai jamais été très efficace dans cette tache, et j'ai finalement rejoint Oumayma sur l'extension.
Au final c'était trois semaines ou je faisais partie d'une équipe, avec un objectif commun. C'est un projet assez stimulant et je préfère ça aux cours classiques, bien que les deux aient évidemment leur utilité.