Bilan de gestion d'équipe et de projet

Projet réalisé par
Boulahfa Adam
Errazki Mohamed
Gaoua Marouane
Lachiri Ilias
Sekkal Amine

Encadré par Mr.Mathias Ramparison et Mr.Tarik Larja

Table des matières

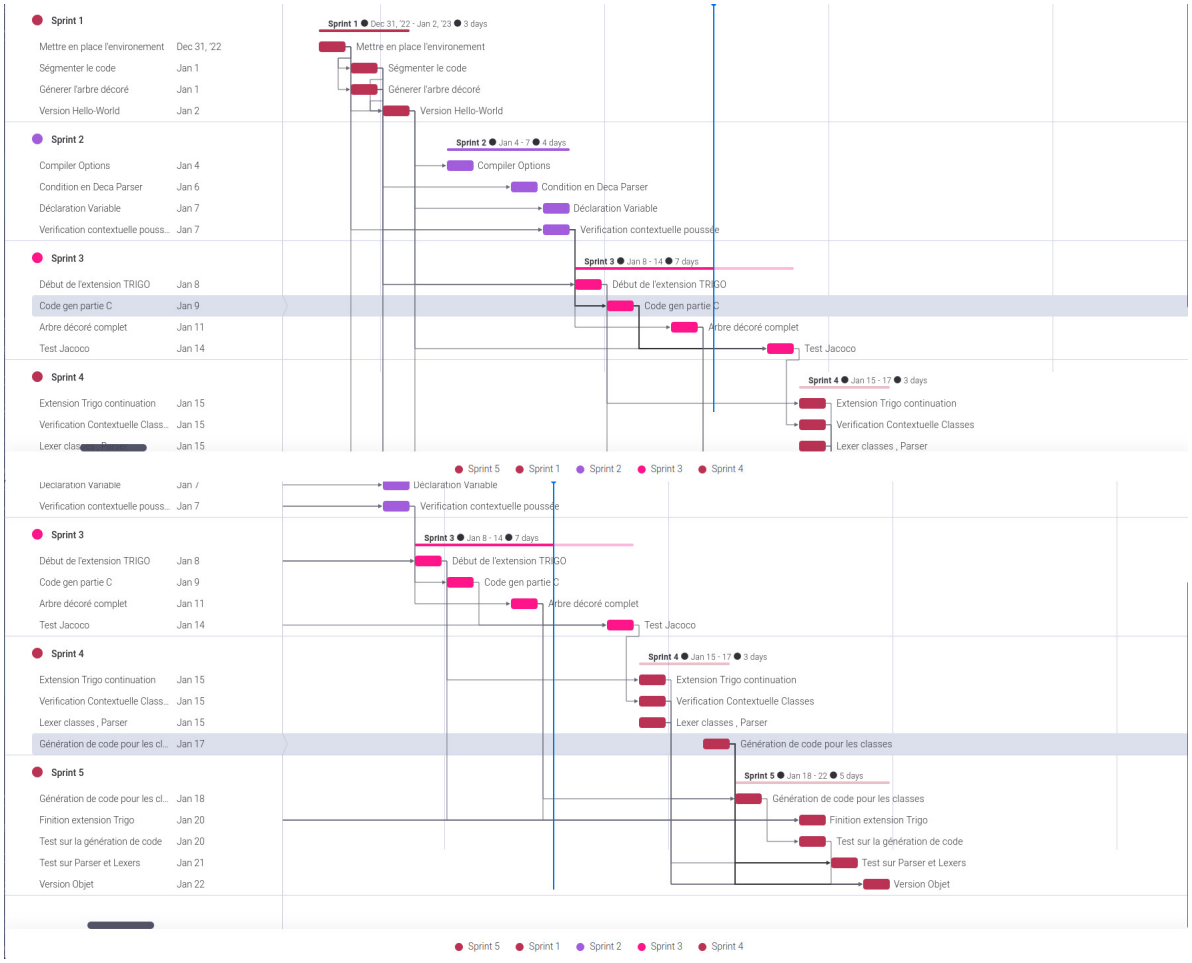
1 Description de l'organisation adoptée par l'équipe

1.1 Première approche

La première réaction de l'équipe face à la documentation proposée était celle de définir le cadre de notre sujet et de mettre en place l'environnement nécessaire au travail.
Nous nous sommes donc organisé pour mettre en place l'environnement Maven ainsi que Ima afin de commencer à travailler sans plus tarder.
Une fois cet environnement installé nous nous sommes rué vers une lecture en diagonale de la documentation afin de reconnaître clairement les objectifs et livrables qu'on devait rendre au plutôt.
Une fois cette lecture en diagonale faite , nous nous sommes concerté lors d'une réunion où nous avons établit tout d'abord les dates importante de notre projet :

- Rendu de la version Hello world 06/01
- Rendu de la version sans objet 16/01
- Rendu de la version avec objet 23/01

Une fois cela fait nous avons pu établir un Gantt provisionnel pour commencer à voir plus clair dans l'organisation du projet.
Voici donc la version première version de notre Gant :

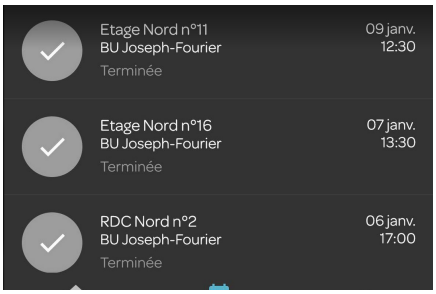


Comme on peut le voir sur le Gantt nous avons choisi de travailler avec une méthode agile afin de pouvoir avancer dans le projet par sprint de 4 jours et effectuer une implémentation continue de notre travail au fur et à mesure des version réalisées. Après avoir établit ce diagramme de Gantt nous avons réparti les taches en fonction de la préférence des membres et de leur compétences dans certains domaines ; ainsi nous avons réparti les tâches comme suit :

- Ilias et Adam prennent l'étape C et l'étape A partie Parser
- Marouane et Amine de l'étape B et l'étape A partie Lexer
- Mohamed s'occupe de l'extension

1.2 Communication

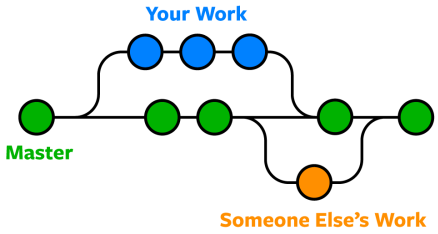
Pour ce projet GL nous avons décidé de communiquer principalement par Messenger et Discord.
Nous avons également décidé de travailler tous les jours en présentiel à l'Ensimag , ou de réserver des salles de travail dans la Bibliothèque Universitaire Joseph Fourier afin d'y travailler durant les Week-end :



En plus nous travaillions chez un membre de l'équipe lorsque un rendu doit être fait le soir même.

1.3 Organisation de l'environnement de travail

Concernant l'implémentation de notre compilateur nous avons décider d'utiliser Git afin de pouvoir travailler en équipe et implémenter graduellement les fonctionnalités .
Nous sommes parti du principe que pour une étape = une branche
Nous avons donc mis en place pour chaque étape du projet 3 branches ainsi que la branche de développement principal qui est la branche master comme suit :



Nous avons donc plusieurs branches organisée comme ci-dessous

```
Master
Version sans objet
|-> A
|-> B
|-> C
Version avec objet
|-> A
|-> B
|-> C
```

2 Présentation de l'historique du projet

2.1 Historique version Hello World

Le premier rendu dût était celui du compilateur qui affichait le String Hello-World. Pour cette première version nous nous sommes organisé comme ci-contre :

1. Ilias et Adam : partie A parser Analyse Syntaxique
2. Marouane : partie A lexer + options -P
3. Amine : partie B Analyse de contextuelle
4. Mohamed documentation pour l'extension TRIGO

Après avoir attribué les taches afin de réaliser cette première partie de notre compilateur Decac , nous avons entamé le visionnage des vidéos présentées individuellement .
Une documentation et une lecture plus détaillés a été faite par les membres en fonction de l'étape attribué. Une

fois l'étape A terminée nous nous somme rendu compte que la partie B était indépendante de cette dernière. Pour cela nous avons redistribué la force de travail sur la partie B et nous avons essayé d'avancer le plus possible sur la partie Sans-Objets La partie C ne nécessitait aucune action de notre part , car elle était déjà implémentée dans cette version "Hello World" .

2.2 Historique version Sans Objet

Pour la partie sans objet nous avons réparti le travail de la manière suivante :

1. Ilias et Adam partie A Parser + partie C
2. Marouane et Amine partie B
3. Mohamed début de l'extension TRIGO sur java

Lors de la mise en place de cette partie nous avons décider de travailler par pair sur un seul ordinateur pour pouvoir à la fois comprendre et rectifier les erreurs si l'un des membres fait une erreur d'inattention ou qu'un problème se présente. Cela était hautement efficace afin de réfléchir à deux sur un problème donné.
En prenant du recul dans le travail nous avons compris que la partie C était dépendante de la partie B. C'est pour cela que l'équipe se chargeant de la partie B a demandé à la partie s'occupant de prendre le temps de bien implémenter la partie A et de se documenter sur la partie C afin de conclure avec l'analyse contextuelle dans les temps et passer à la génération de code.

Après finition de la partie B et A . Nous nous sommes reparti de cette manière :

1. Ilias et Adam partie C
2. Amine test partie B
3. Marouane test Jacoco partie B

Les tests effectués par Amine ont montré plusieurs bug et erreur au niveau de la partie et notamment au niveau du convfloat qui contenait plusieurs erreur au niveau de la conversion de int vers un flottant.
Cette erreur devait etre rapidement réglée pour ne pas avoir de probléme au niveau de la partie C dans la génération de code et afin d'optimiser au mieux la gestion de flottant.
Après avoir fini cette partie sans objet , nous l'avons soumit pour une corrections de rendu intermédiaire.
Après le feedback de l'encadrant nous avons remarqué que les erreurs effectuées étaient mineures et qu'elles pouvaient être rapidement fixée.
Nous nous sommes donc fixé comme objectif de valider complètement le compilateur sans objet en debbugant tout le nécessaire et en faisant en sorte que toute les remarques données par l'encadrant soit prise en compte avant d'entamer la partie avec objet.

2.3 Historique version Avec-Objet

Avant d'entamer la partie avec objet nous avons tout d'abord effectuer un bilan général de notre compilateur afin de savoir ce qui n'allait pas dans la partie sans-objet et ce qu'il fallait améliorer.
Les principaux points à respecter au fur et à mesure étaient :

1. La partie A nécessite le travail d'une personne uniquement
2. Mettre en place des tests unitaires pour assurer une couverture maximale
3. S'assurer de la robustesse de notre compilateur en mettant plus de tests
4. Gérer les problèmes des grands nombre en déca

La partie du compilateur avec-objet une fois entamée nous nous sommes réparti de la manière suivante :

1. Ilias et Adam partie C
2. Amine partie B –> Déclaration des Fields
3. Marouane partie B –> Déclaration des Methodes
4. Mohamed partie A lexer + parser

Durant cette phase du projet , la partie C a été réalisée indépendamment de la partie B. En effet , une fois compris que la partie A pouvait être fait en une journée , ce qui rendait possible la mise en place et test de la partie B aussi tôt .
La partie C ,elle, dépendante de la partie B prenait d'autant plus de temps que cette dernière et pouvait être réalisée en parallèle sans test du compilateur.
Pour cela il a été nécessaire d'avancer le plus rapidement possible la partie B pour que l'équipe de la partie C puisse commencer à coder et tester en parallèle leur code. Nous avons donc opté de fournir la première sous-partie prête de la partie B (déclaration de field ou méthode) à l'équipe se chargeant de la génération de code pour qu'elle puisse entamer cette dernière étape et tester.
Quelques problèmes sont survenus au niveau de la mise en place de l'étape B , ce qui a retardé notre travail et a obligé l'équipe C de travailler à l'aveugle sur sa partie sans pouvoir réellement compiler et exécuter . Heureusement une bonne partie de ce qui a été réalisé pour l'étape C était correct et concordait avec ce qui était demandé.
Une partie de debuggage a néanmoins été nécessaire pour la génération de code et une implémentation complète a été faite avec des fois des retours vers la partie B (convfloat).

Lors de l'implementation de la partie C et de la compilation nous avions également remarqué que quelques éléments de la partie A devaient être modifié afin de bien prendre en compte les paramètre de visibilité. PROTECTED et PUBLIC En parallèle de son implementation la partie B a du être testé et validée en comparant nos résultats avec les résultats attendus dans la documentation client qui nous a été donnée. Cela nous a permit de comprendre ce qu'il était demandé de cette partie.

Les critères de validation étaient :

- Est ce que le test n'affiche pas d'erreur ? (Arrêt du test)
- Est ce que l'arbre obtenu est correct ? (Décoration , définition , type)
- Est ce qu'un exemple particulier pourrait compromettre notre implémentation ?

En amont une gestion de test automatisé a été mise en place ce qui a hautement facilité la phase de test avec plusieurs fichier.
Concernant les test et validation de la partie C , il fallait se poser les questions suivantes :

- Est ce que le code compile ?
- Est ce que le résultat de la compilation est correct ?
- Est ce que l'appel au méthodes est correct ?
- Est ce que le code est robuste ?

2.4 Extension

Concernant l'extension il a été trés important d'effectuer une recherche en amont du projet car cette partie ne pouvait être faite qu'avec implémentation de la partie avec objet qui était mis en place durant le sprint final de notre projet GL.
Pour palier à cela Mohamed s'est penché sur l'extension en se documentant et en mettant en place un code en Java ressemblant au language Deca afin de faciliter son implémentation en Deca.
Son implémentation en Deca dépendait de deux paramtre qui étaient :

- Fonctionnement de la partie doInclude
- Fonctionnement de la partie Objet et des opérations arithmétique avec flottant.

La partie doInlcude étant déjà faite dans la partie sans-objet , il ne fallait que la finition de la partie objets afin de pouvoir mettre en place cette extension .decah.

3 Conclusion

Lors de ce projet GL nous avions eu une idée de l'organisation autour d'un travail qui demandait l'implication de tout les membres de l'équipe.
La mise en place de code de manière incrémentale , l'automatisation des tests ainsi que leur mise en place et la validation du fonctionnement de notre compilateur ont été des points clefs à la réussite du projet.
Cependant il est à noter qu'il reste à améliorer quelque points notamment au niveau de l'estimation de la charge de travail sur une partie donnée et sur la prise de conscience au niveau de dépendance entre chaque partie du projet.
Au final ce projet a été une bonne expérience apportant un challenge à chacun d'entre nous et nous permettant de tous nous améliorer que se soit sur le plan technique ou relationnel.