M1 Informatique AIGLE

HMIN204

Conduite de Projet

Méta-Rapport du TER

Groupe Bajonim

Bachar Rima,
bachar.rima@etu.umontpellier.fr
Joseph Saba,
joseph.saba@etu.umontpellier.fr
Tasnim Shaqura,
tasnim.shaqura@etu.umontpellier.fr

Responsable de l'UE :
Eric Bourreau

22 Mai 2019

Table des matières

Chapitre 1

Sujet

Le logiciel constitue une partie trés importante de nos connaisances scientifiques, culturelles, et technique. Les logiciels sont présents dans tous les aspects de notre vie quotidienne. Il est donc important d'archiver les logiciels. Des efforts ont déja été fait pour la préservation des logiciels, tel que The Internet Archive et UNESCO Persist, mais ils se concentrent sur la préservation des éxécutables. Software Heritage est un projet qui a comme but la préservation des codes source des logiciels disponibles publiquement. Les codes source sont importants parcequ'ils peuvent être façilement compris par des humains, et peuvent être façilement transformés en éxécutables.
L'équipe de Software Heritage on crée une architecture qui permet de retrouver les sources codes d'un dépôt et de les placer dans l'archive. Les **Listers** sont une partie centrale à cette architecture. Ce sont des crawlers configurés pour parcourir des dépot de code et retrouver leurs contenu. Les differents dépôts de code ont des structures bien différentes l'un de l'autre, ce qui necessite la création d'un Lister dédié à chaque platforme qu'on souhaite archiver. L'équipe de Software Heritage a déja crée des Listers pour quelques dépots populaires, tel que Github et Bitbucket, avec succés ; mais jusqu'à présent, aucune équipe externe a crée un Lister. Le dépôt que nous avons choisi est Launchpad.
Les objectifs de ce TER sont :

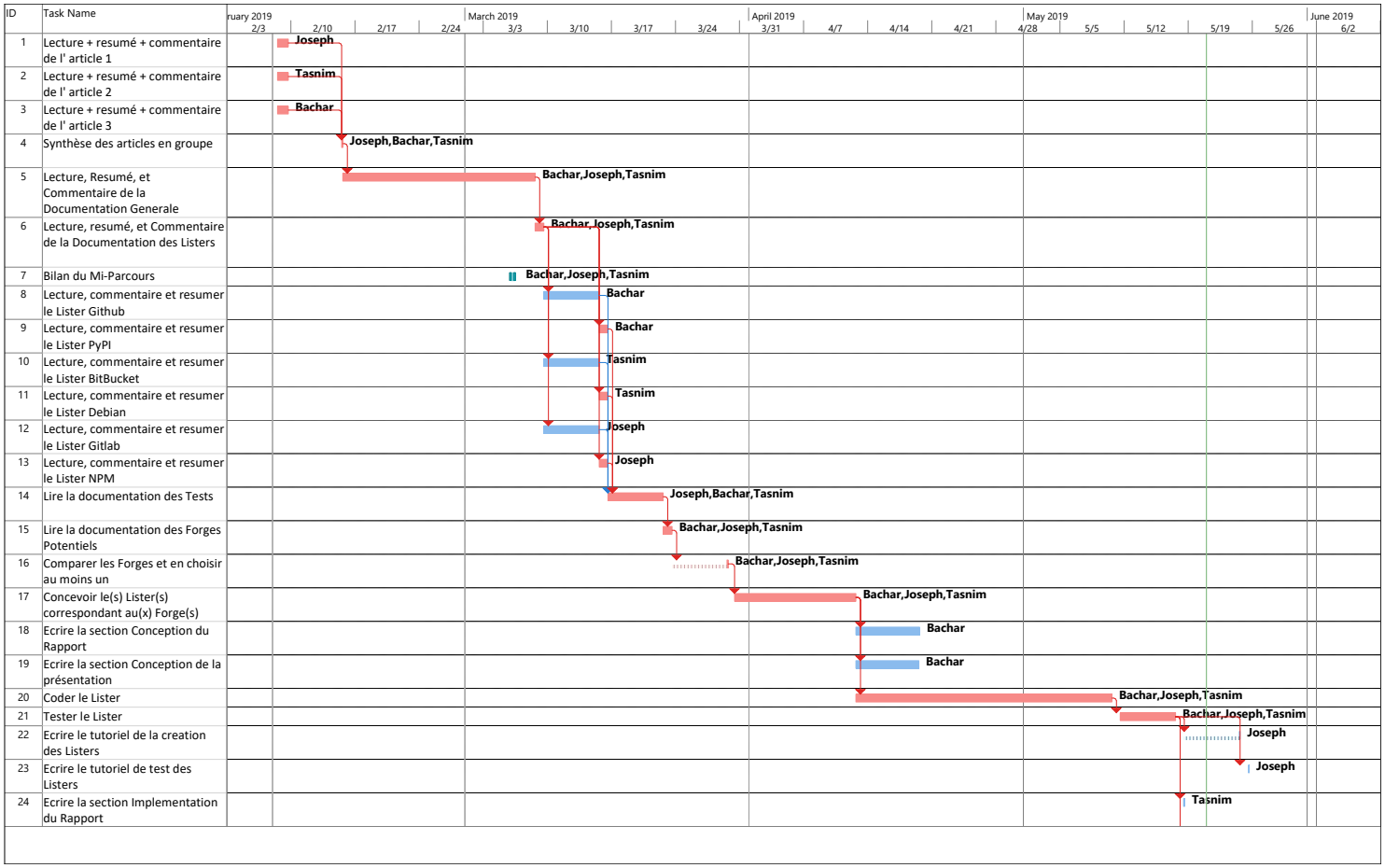
- Lire et comprendre les articles publiés par l'équipe de Software Heritage
- Lire les tutoriels écrits par l'équipe de Software Heritage
- Tester differents dépot de code
- Écrire un Lister pour le dépot choisis
- Répliquer localement l'environnement de Software Heritage et tester le Lister
- Faire une Pull Request pour ajouter le Lister à Software Heritage

Chapitre 2

Planning

Dans ce chapitre, nous montrons le diagramme de Gantt initiale et le diagramme de Gantt final, puis nous analyserons les differences entre les deux.

2.1 Planning Prévisionel



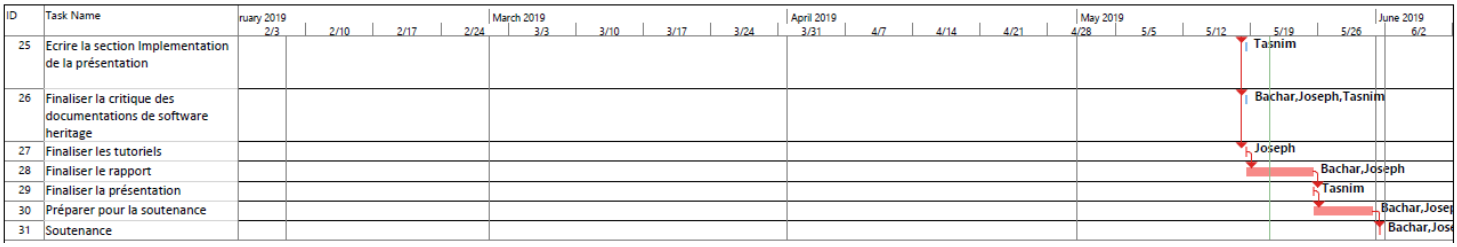
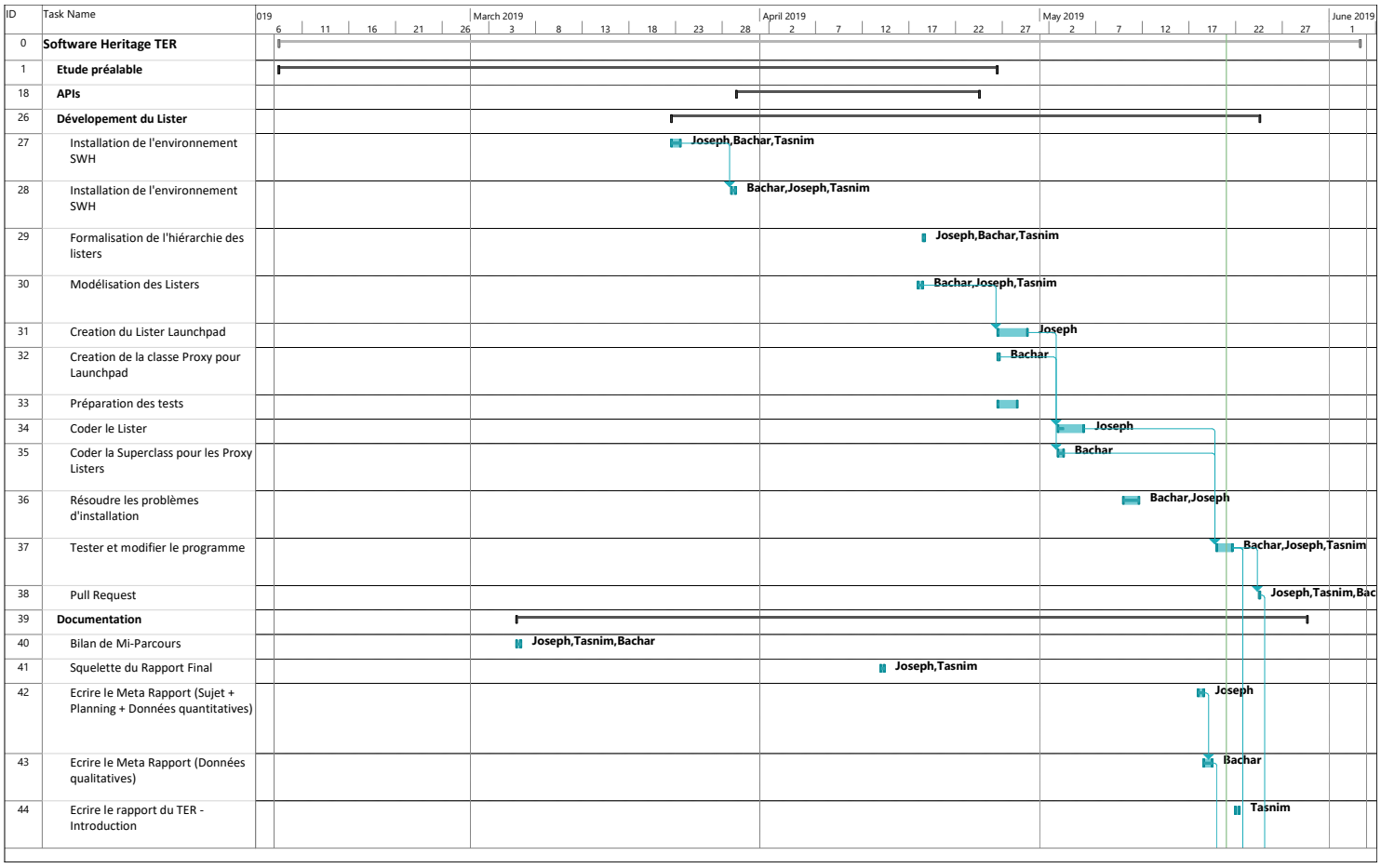


FIGURE 2.1 – Planning prévisionel

2.2 Planning Final



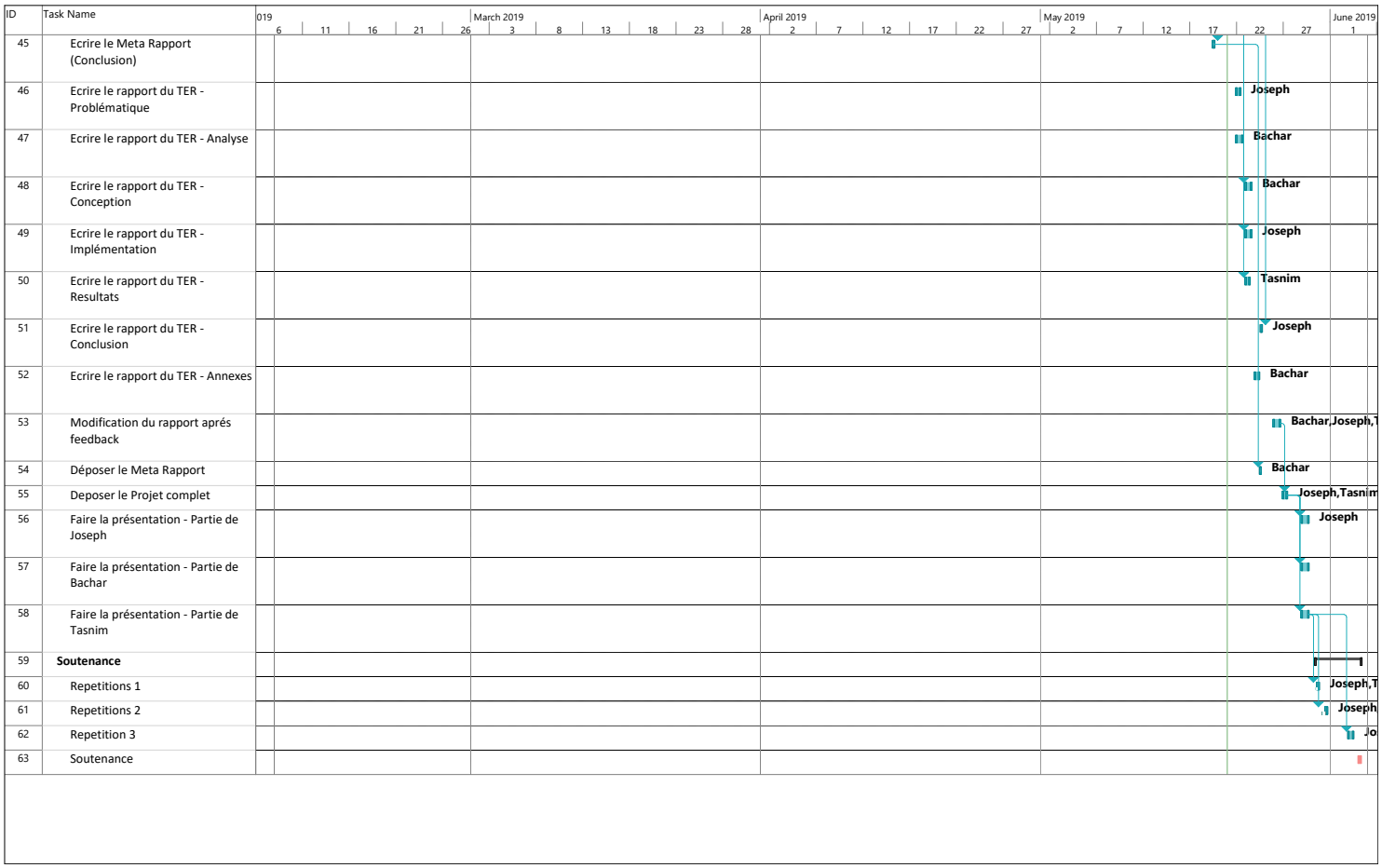


FIGURE 2.2 – Planning final

2.3 Notes sur le planning final

Aprés avoir déposer le planning prévisionnel, nous avons refait le planning et nous l'avons découpé en cinq parties pour mieux refléter la structure du projet :

- **Étude préalable :** Cette section forme une grande partie du projet. Elle inclut toutes les tâches de lecture et recherche qui nous ont permis de se familiariser avec l'architecture et l'environnement de Software Heritage, ainsi que les differents Listers préexistants.
- **APIs :** Cette section regroupe les tâches qui concernent la recherche d'un dépot de code avec une API adaptée, qui va être éxploitée dans

les tâches de développement.

— **Développement du Lister :** Ce groupe contient les tâches liées au développement du Lister, donc les tâches de modélisation et conception, de coding, et de testing.
— **Documentation :** Pour organiser la création des rapports et de la présentation.
— **Soutenance :** Pour organiser les répétitions et la soutenance.

2.4 Comparaison des plannings

Bachar

Chapitre 3

Données Quantitatives

Ce chapitre ce concentre sur les details quantitatives concernant le TER. Le projet est disponible sur le lien Git suivant : `https://github.com/joe11093/Software-Heritage-TER`

3.1 Commits

Depuis la date de la création du Git repo, jusqu'au XX/XX/XXXX, nous avons effectué XX commits.

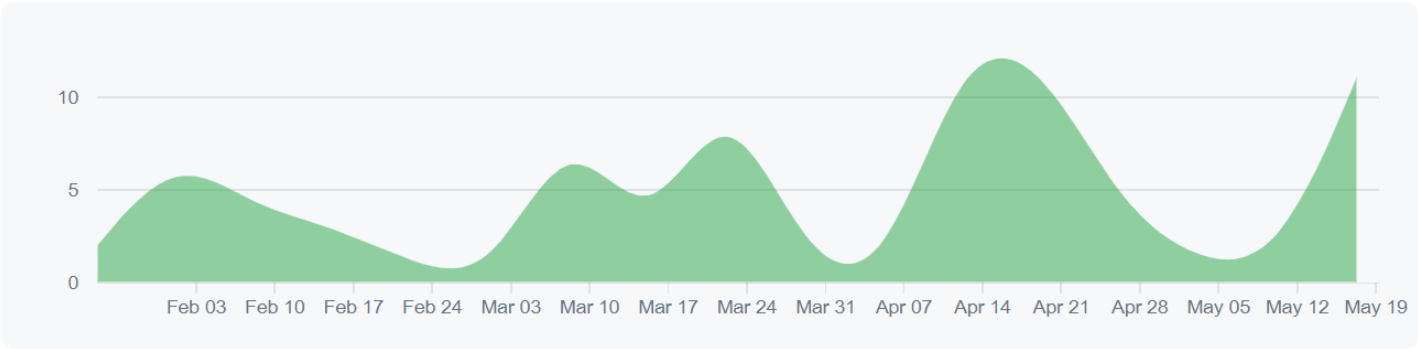


FIGURE 3.1 – Graph des commits

Nous observons qu'en premier temps, le nombre de commits est bas. Cela correspond à la période de l'étude préalable qui consistait surtout de tâches de lecture. Les deux points bas suivants (aux alentour de fin Mars et début Mai) sont dû à la période de rattrapage et la période des révisions des examens finaux. Les commits s'élèvent à la fin du TER parceque nous ajoutons au fur et à mesure le code sur lequel nous travaillons, ainsi que les rapports et leurs documents correspondants.

Nous observons que les commits ont été souvent éffectué les Vendredis. En effet, nous avons decidé de travailler sur le TER les Vendredis, qui est la seule journée oú les trois membres du groupe n'ont pas de cours. Les commits de la dernière période du projet sont une éxception ; cela est dû à l'arrêt des cours.

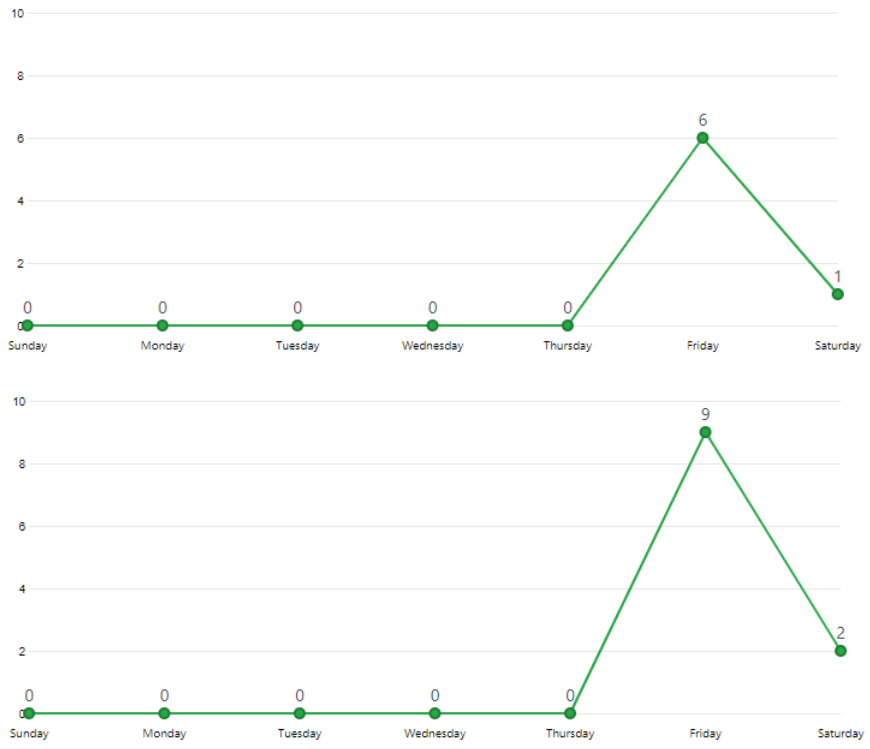


FIGURE 3.2 – Moyenne des commits par jours pour les semaines du 3 Février et du 24 Mars

3.2 Classes

— **LaunchpadProxy :** Cette classe éxploite l'API de Launchpad pour récupérer tous les projets Git disponibles publiquement.
— **LaunchpadGitModel :** Elle définit le modèle d'un repository en Git hébergé sur Launchpad.
— **LaunchpadGitLister :** La classe du Lister qui implemente toute les fonctions requises pour énumérer les origines hébergés sur Launchpad.
— **ProxiedLister :** Classe abstraite qui pourra être implementée dans un autre Lister qui a les éxigences que Launchpad.
— **WebApiProxy :** Une interface qui contient les déclarations des fonctions requise dans une classe Proxy utilisée par un Lister.
— **LaunchpadListerTester :** Une classe utilisée dans les tests

3.3 Temps de Travail

En ce qui concerne le temps de travail, nous estimons :
— Approximativement 110 heures sur l'étude préalable du projet.
— Approximativement 60 heures pour la recherche et les tests des APIs.
— Approximativement 110 heures pour le developpement du code
— Approximativement 20 heures pour la documentation et les rapports.
— Approximativement

Nous estimons encore autour de 30 heures de travail, qui portera surtout sur l'écriture du rapport du TER ainsi que la présentation.

Chapitre 4

Conclusion