Université Claude Bernard Lyon 1
Master Informatique

2015 - 2016

MIF-TI5

Projet PERF

Conception d'une plateforme pour l'alignement d'entités par flux

Document Post-sprint 1

Alexandre Millot
Corentin Lonjarret
Antoine Hintzy
Murat Halat
Felician Raphael Matinya

Table des matières

Projet PERF 1

1 Préambule 3

2 Pré-sprint 3

2.1 Travail effectué pré-sprint 3

2.2 Décomposition en tâches 3

2.2.1 Tâches et attentes sprint 1 3

2.2.2 Planning du sprint à la demi-journée 4

2.3 Évaluation des charges et répartition des tâches 5

2.4 Livrables techniques 6

3 Post-sprint 6

3.1 État d'avancement des tâches & écarts par rapport aux prévisions 6

3.2 Résultats du sprint et modifications apportées au projet 7

3.3 Objectifs de la prochaine itération 7

1 Préambule

Ce document présente la synthèse des activités réalisées ainsi que des méthodes employées pour mener à bien le sprint 1 du projet PERF.

Celui-ci décrit tout d'abord le travail effectué avant le sprint, les livrables techniques à produire pour la première recette ainsi que la décomposition, la répartition et l'évaluation de la charge des tâches devant être réalisées au cours du sprint. Ensuite, une comparaison post-sprint est présentée de façon à évaluer les écarts entre les prévisions faites avant le sprint et les résultats réels fournis en sortie de ce dernier. Enfin, nous discuterons des résultats obtenus et modifications apportées au projet ainsi que des objectifs de la prochaine itération

2 Pré-sprint

2.1 Travail effectué pré-sprint

Outre la rédaction du dossier d'initialisation, plusieurs tâches ont été réalisées pré-sprint de façon à être opérationnels dès le début du sprint et à assurer le bon déroulement du lancement du projet.

Parmi les tâches effectuées, on retrouve notamment :

- La mise en place du projet de base en utilisant MEAN.JS ;
- L'étude de l'utilisation de MEAN.JS, notamment du fonctionnement des interactions avec la base MongoDB ;
- La création du dépôt forge ;

2.2 Décomposition en tâches

Pour rappel, voici les lots de tâches que nous avions prévu de réaliser au sprint 1 lors de la réalisation du dossier d'initialisation D0.

2.2.1 Tâches et attentes sprint 1

- Ajout et récupération de données dans la base MongoDB ;
- Implémentation basique client et serveur avec connexion entre ces derniers ;
- Échange simple d'entités au format JSON via *sockets* ;
- État d'avancement du travail fourni dans les tâches précédentes et tests de fonctionnement du système unifié ;
- Démonstration prévue : Une première version simple et fonctionnelle du système en local.

2.2.2 Planning du sprint à la demi-journée

Lundi Matin

- Prise en main et conception papier de la base de données MongoDB.
- Création de la *socket master* et de la classe java (client) – Permettre l'échange du client java avec le serveur Node.js.
- Échange simple d'entités au format JSON via *sockets*.

Lundi Après-midi

- Fonctions de *merge* et d'ajout de *datasets* au format CSV.
- Création de la *socket master* et de la classe java (client) – Permettre l'échange du client java avec le serveur Node.js.

Mardi Matin

- Fonctions de *shuffle* du *dataset* et de *parsage* en JSON.
- Création de la *socket master* et de la classe java (client) – Permettre l'échange du client java avec le serveur Node.js.

Mardi Après-midi

- Mise en place des documents administrateurs et performances.
- Création de la *socket worker* et envoi de données au client.

Mercredi Matin

- Fonctions de *merge* et d'ajout de *datasets* au format XML.
- Mise en place des documents administrateurs et performances.
- Connexion du client à la *socket worker* – protocole d'échange client-*worker*.
- Implémentation d'un algorithme d'alignement simple.

Mercredi Après-midi

- Fonction d'ajout de documents avec MEAN.JS.
- Connexion du client à la *socket worker* – protocole d'échange client-*worker*.

Jeudi Matin

- Récupération liste documents et affichage d'un document avec MEAN.JS.
- Améliorer API pour pouvoir retourner un résultat au serveur.

Jeudi Après-midi

- Fonctions basique d'authentification et d'ajout d'administrateurs en base avec MEAN.JS.
- Améliorer API pour pouvoir retourner une résultat au serveur.

Vendredi Matin

- *Merge* l'ensemble des parties du projet et tester.
- Préparation de la machine virtuelle et déploiement.
- Document post-sprint 1.
- Préparation du DAL.

2.3 Évaluation des charges et répartition des tâches

La charge des tâches est exprimée en demi-journée.

Intitulé de la tâche	Évaluation de la charge	Personnes affectées
Prise en main et conception de la base MongoDB	1	Alexandre, Corentin
Ajout et récupération de données aux formats CSV et XML dans la base de données avec MEAN.JS	4	Alexandre, Corentin
Mise en place document performances et affichages	2	Alexandre, Corentin
Échange entités format JSON entre *sockets*	1	Murat
Création API java et *socket Master*	3	Antoine, Murat, Felician
Création *socket Worker* et envoi de données au client	2	Antoine, Murat, Felician
Connexion client à la *socket Worker*	1	Antoine, Felician
Implémentation algorithme alignement simple	1	Murat, Corentin, Felician
Amélioration API	1	Antoine, Murat
Merge des différentes parties et tests	1	Corentin
Préparation VM et déploiement	1	Antoine, Felician
Gestion de projet	8	Alexandre, Corentin
Dossier Post-sprint 1	1	Alexandre
Document Architecture Logiciel	1	Alexandre

2.4 Livrables techniques

Parmi les livrables techniques que nous devons réaliser au cours du premier sprint, on retrouve :

- Le produit logiciel : Version minimale fonctionnelle du système mettant en œuvre l'ensemble des technologies choisies – Il devra notamment être possible de réaliser des échanges simples d'entités entre serveur et client. Les entités auront été ajoutées à la base et récupérées via le système.
- La première version du document architecture logiciel (DAL).

Ces deux livrables seront présentés aux porteurs du projet, c'est à dire M. DUCHATEAU et M. LUMINEAU lors d'une réunion le lundi 26 octobre

3 Post-sprint

3.1 État d'avancement des tâches & écarts par rapport aux prévisions

Intitulé de la tâche	Temps prévu	Temps réel	État d'avancement
Prise en main et conception de la base MongoDB	1	1	100 %
Ajout et récupération de données aux formats CSV et XML dans la base de données avec MEAN.JS	4	4	100 %
Mise en place document performances et affichages	2	2	100 %
Échange entités format JSON entre *sockets*	1	1	100 %
Création API java et *socket Master*	3	3	100 %
Création *socket Worker* et envoi de données au client	2	1	100 %
Connexion client à la *socket Worker*	1	2	100 %
Implémentation algorithme alignement simple	1	2	100 %
Amélioration API	1	2	100 %
Merge des différentes parties et tests	1	1	100 %
Préparation VM et déploiement	1	1	75 %
Gestion de projet	8	8	100 %
Dossier Post-sprint 1	1	1	100 %
Document Architecture Logiciel	1	1	100 %

3.2 Résultats du sprint et modifications apportées au projet

Comme on peut le voir dans le tableau d'état d'avancement des tâches à la fin du sprint décrit précédemment, l'ensemble des tâches que nous avions prévu de réaliser ont pu être implémentées. De ce fait, la première version de la plateforme que nous nous proposons de développer est entièrement fonctionnelle. Ainsi, la plateforme permet actuellement d'ajouter des *datasets* aux formats CSV et XML en base, d'envoyer ces *datasets* à un client effectuant une demande d'envoi et de renvoyer les résultats du client au serveur après que celui-ci ait effectué les opérations nécessaires sur les entités. Ensuite, une fois les résultats du clients récupérés, la plateforme est capable de calculer les performances de ces résultats.

En plus d'avoir mis en place la version de base de la plateforme, nous avons pu ajouter certaines des fonctionnalités que nous avions prévu d'ajouter dans le futur, notamment :

- La gestion des débits des flux d'entités entre *worker* et client. Lorsque le client envoie une demande au *worker* pour recevoir un *dataset* spécifique, celui-ci peut également choisir le débit du flux d'entités qu'il souhaite recevoir.
- La gestion des classements des performances par *dataset.* Ainsi, chaque client peut se positionner au niveau de la performance produite par rapports aux autres clients. Pour chaque *dataset*, seule la meilleur performance d'un client apparaît dans le classement.

Concernant les modifications apportées au projet durant ou suite à ce sprint, la seule modification notable concerne la façon de gérer les classements des performances. Nous souhaitions à la base gérer les classements via un document dédié en base MongoDB. Cependant, il a finalement été décidé de générer les classements de façon dynamique à chaque demande de visualisation d'un des classements. Nous avons pour cela implémenté une architecture MapReduce.

3.3 Objectifs de la prochaine itération

On retrouve ci-dessous la liste des tâches prévues pour la prochaine itération du projet. Globalement, le but est d'obtenir une version complète et fonctionnelle du système en local. Pour le prochain sprint, nous souhaitons mettre en place :

- Répartition des tâches avec plusieurs machines (un *master* et deux *workers* un premier temps).
- Création de l'interface administrateur et de l'interface client.
- Création de la page d'accueil qui se présentera comme un tutoriel d'utilisation pour les clients.
- Tableau de bord pour les administrateurs.
- Premiers tests de la partie « container ».
- Démonstration prévue: Version entièrement fonctionnelle du système en local, sans améliorations ni tests de montée en charge.