MIAGE 1
Projet Développement Logiciel –
Responsable : Mathieu Acher
Contact : mathieu.acher@irisa.fr
2019/2020

Gestion de projet – Projet Développement Logiciel (PDL)

Ressources : https://github.com/acherm/PDL1920

Votre équipe a la responsabilité de mener à bien un projet logiciel avec des technologies et données ouvertes. De nombreux défis sont à relever, nécessitant des compétences en gestion de projet, en modélisation, et en programmation.

Cette mise en situation doit permettre de mieux comprendre et appréhender la difficulté de développer du logiciel dans un contexte extrêmement concret. Des techniques et outils (git, github, Maven, JUnit, etc.) de développement logiciel, bien connus de l'industrie, seront utilisés. Des choix technologiques devront également être effectuées. Il est attendu de la part de chaque étudiant de PDL de démontrer qu'elle ou il est capable :

- de comprendre l'existant (exigences, spécificité des technologies et du domaine d'application, architecture, format des données, etc.)
- de contribuer très concrètement à un projet à la fois au niveau du code (cas de tests, ajouts de fonctionnalités, refactoring, etc.), de la documentation, ou de la mise en œuvre de l'intégration continue
- de maîtriser un ensemble de technologies (Java et son écosystème, CSV, JSON, JUnit, API Web, etc.) et de techniques (e.g., test) importantes
- de s'adapter à l'évolution d'un projet et de ses exigences
- de travailler collectivement
- de tenir les dates de rendu
- de valider de manière continue les exigences et l'implémentation[1]

1 Contexte général du sujet

Un seul sujet, commun à la promotion, est proposé cette année : « Wikipedia Matrix : The Truth ». Ce projet est la suite du sujet de l'année dernière : vous allez

[1] La « validation » continue a différents impacts sur le déroulement du projet : il s'agit de valider avec le client les exigences, de démontrer avec différents prototypes la valeur de son travail et sa conformité par rapport aux attentes, de tester l'implémentation pour démontrer sa robustesse, etc.

reprendre les projets de vos collègues et votre objectif sera de démontrer leurs faiblesses, et surtout de les améliorer tant au niveau de la documentation que du code et du test.

La note pourra être individualisée si on observe un manque évident de travail à l'intérieur d'un groupe. A ce titre, l'activité sur Github sera observée. Par défaut, la note du groupe s'applique à chaque membre.

2 Travail demandé

Au cours du projet vous devrez rendre différents « livrables ».

Eliciter des exigences (EX)

Vous allez écrire trois documents sur votre dépôt Github:

- Un document README.md qui permet à tout visiteur du projet de comprendre rapidement le but du projet, les fonctionnalités supportées, les fonctionnalités à développer dans le futur, ainsi que la licence du projet, les technologies utilisées, les participants et le contexte du projet, etc.
- Un document d'architecture DESIGN.md à la racine du github qui reprend les grands éléments d'architecture incluant des modèles statiques (organisation des packages, descriptions des classes principales et de leurs responsabilités, etc.) ainsi que des modèles dynamiques (flux des événements, scénarios nominaux et exceptionnels, etc.). Il est fortement conseillé d'utiliser UML (diagramme de cas d'utilisation, diagramme de classes/d'objets, machines à états, etc.). L'objectif de ce document est qu'un potentiel contributeur externe puisse comprendre l'architecture et les fonctionnalités du projet pour pouvoir éventuellement reprendre le code et l'étendre/l'améliorer
- Un document INSTALL.md décrivant comment on construit le projet à partir du code source, comment on exécute les suites de tests, comment on exécute le logiciel

Tous les documents seront écrits en anglais.

Si besoin, vous devez *également justifier le choix d'une technologie et le faire valider par les clients*. De fait une étude de l'existant est à réaliser, ce qui permettra d'expliquer pourquoi certaines solutions n'ont pas été retenues.

Il y a un « client » : Mathieu Acher (mathieu.acher@irisa.fr). Vous devez discuter avec le client tout au long du projet pour valider vos choix.
La date de rendu pour les trois documents est fixée au **15 octobre 2019**

Attention : le livrable EX implique de *dores et déjà maîtriser les technologies existantes et de développer une solution logicielle* ; ce travail est indispensable pour pouvoir discuter avec le client des exigences, des choix technologiques, et de valider avec lui certains prototypes.
Il est possible qu'au cours de l'avancée du projet certains éléments (fonctionnalités, choix techniques, etc.) soient remis en cause : c'est parfaitement normal! Les groupes décriront et justifieront ces changements lors de soutenance (cf PR ci-dessous).

Sprint (SP1 + SP2)

Au cours du projet, il est demandé d'effectuer deux tâches précises, non triviales dans un laps de temps prédéfini.

Deux sprints sont à réaliser, SP1 et SP2

SP1 (amélioration de l'existant)

L'objectif de SP1 est d'améliorer les extracteurs typiquement en corrigeant des bugs ou en étendant la prise en compte de certains cas. Pour réaliser ce travail, nous suivrons une méthodologie bien particulière :

- Ecrire une issue Github (en anglais) à chaque fois que vous identifiez un manque
- Ecrire des cas de tests automatiques (avec JUnit) qui démontrent les faiblesses des extracteurs actuels
- Ensuite, et seulement ensuite, corriger/modifier les extracteurs

En sortie de ce sprint, les extracteurs doivent être de meilleure qualité et la suite de tests associée démontrera votre axe d'amélioration : en d'autres termes, vous apporterez davantage de confiance dans les implémentations des extracteurs.

La date de rendu pour les trois documents est fixée au **15 novembre 2019**

SP2 (vérité terrain sur les résultats attendus)

L'objectif de SP2 est de construire une vérité terrain et donc une suite de tests plus aboutie. Assez rapidement vous allez vous rendre compte de la difficulté de tester les extracteurs : le résultat attendu n'est pas connu à l'avance par une machine. On pourrait spécifier le résultat attendu pour chaque tableau de chaque page Wikipedia, mais il y a potentiellement plus de 1000 tableaux à traiter. C'est assez fastidieux et coûteux en temps de spécifier la vérité terrain. Aussi l'idée est de développer une suite d'outils pour pouvoir graphiquement fabriquer un résultat attendu.
La date de rendu pour les trois documents est fixée au **20 décembre 2019**

Le code sera nécessairement hébergé sur github, dans un repository publique.

Présentation (PR)

- 20' de présentation
  - Rappel du contexte
  - Capture des exigences et validation avec le client (EX)
  - Expliquer l'implémentation, l'architecture du projet, les technologies utilisées, et son déploiement (SP)
  - Démonstration (SP)
  - Retour d'expérience (difficultés rencontrées, adéquation par rapport au cahier des charges, technologies et méthodes maîtrisées, etc.)
- 10' de questions par le jury

Rendu

Les dates de rendus sont strictes et imposées.
EX : 15 octobre 2019 (23h59) ;
SP1 : 15 novembre 2019 (23h59)
SP2 : 20 décembre 2019 (23h59)
PR : mi-janvier 2020
La note est sur 20:

- 5 points pour EX,
- 10 points pour SP dont 2 points pour le concours (cf description du projet « Wikipedia Matrix »)
- 5 points pour PR

Les notifications des rendus s'effectueront par un email du responsable du groupe (et évidemment via Github pour le code et les instructions, cf SP).

Comment commencer ? (E0)

Avant de commencer la rédaction du cahier des charges, l'implémentation, ou de simplement discuter avec le « client » il est nécessaire :

- de *comprendre le domaine d'application* (e.g., Wikipedia)
- *lire http://blog.mathieuacher.com/WikipediaMatrixChallenge/ sur le retour d'expérience de l'année dernière*
- de *techniquement comprendre l'existant* et les technologies déjà utilisées (APIs Java existantes et candidates, données existantes, etc.)
- de *comprendre l'objectif général* afin de raffiner les besoins par la suite

Un exercice nécessaire est d'écrire, dans le document README.md, l'objectif général du projet ***avec vos propres mots*****.** Cela doit valider votre compréhension du sujet.

Un autre objectif à atteindre le plus rapidement possible est que chaque membre du groupe soit à même de commiter/pusher du code qu'il aura développé, de prendre en compte les mises à jours de ces collègues, et bien sûr d'exécuter l'application et les cas de test. **Il faut donc mettre en place un « repository » Github très rapidement (en forkant le projet de l'année dernière dont vous aurez la charge).**

Enfin ne pas hésiter à échanger avec la communauté open source et à reporter des bugs ou manques (e.g., sur le Github des projets).

Un document décrivant le projet « Wikipedia Matrix : The Truth » vous est également communiqué.

Bon projet