
Gestion de projet – Projet Développement Logiciel (PDL)

Ressources : <http://mathieuacher.com/teaching/PDL/>

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 effectués. Il est attendu de la part de chaque étudiant de PDL de démontrer qu'elle ou il est capable :

- de comprendre un projet existant (exigences, spécificité des technologies et du domaine d'application, architecture, 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 (CSV, JavaScript, JSON, JUnit, 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'implémentation¹

1 Contexte général du sujet

Quatre sujets sont proposés :

- Projet #1 (OpenCompareReverseJSON)

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.

- Projet #2 (OpenFoodFacts2CSV)
- Projet #3 (MatrixSynthesizerWikipedia)
- Projet #4 (WikipediaAnalysis)

Les sujets proposés partagent un objectif pédagogique commun (cf ci-dessus) et ce sont sur ces différents critères que sera évalué le travail. Il est à noter que, bien que les objectifs de chaque projet diffèrent, la complexité de ce qui est demandé est proche. Aussi il n'y a pas de projet plus facile ou difficile que d'autres : la notation sera certes uniformisée (e.g., entre les groupes d'un même projet), mais il n'y aura pas moins d'attente en termes d'objectifs pédagogiques. La note pourra être individualisée si on observe un manque évident de travail à l'intérieur d'un groupe.

A noter que les sujets seront échangés le **10 octobre 2017** (Projet 1 ↔ Projet 4 ; Projet 2 ↔ Projet 3). Autrement dit, vous travaillerez à partir du cahier des charges fourni par un autre groupe.

2 Travail demandé

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

Eliciter des exigences (EX)

L'objectif est d'écrire un cahier des charges de votre application qui aura un double intérêt : (1) c'est un support pour *communiquer*: ce document permettra d'échanger avec le client et de valider avec lui certains aspects du projet. Il sera également lu par un autre groupe; (2) c'est un document qui permet de formaliser les attentes, de délimiter les champs d'application, et de se défendre contre des attentes disproportionnées ou inattendues: ce cahier des charges peut être vu comme un « *contrat* » qui sera vérifié à la fin du projet en fonction de l'implémentation réalisée.

Vous utiliserez les *formalismes de modélisation* (e.g., UML) pour expliciter les exigences et concevoir votre application, par exemple:

- Diagramme de cas d'utilisations
- Diagramme de classes
- Machines à états décrivant le comportement des principales classes
- Modèles d'interface graphique
 - Avec des modèles d'UI (sketchs), cf <https://balsamiq.com/>
 - Machine à états UML

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).

L'objectif de rendu est l'écriture d'un document, en anglais ou en français, d'exactly 20 pages (il est autorisé d'inclure un « appendix » de 5 pages maximum). Une version du document PDF sera envoyée à mathieu.acher@irisa.fr
La date de rendu pour ce document est le **9 octobre 2017**

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.

Le cahier des charges sera rendu le 9 octobre et sera utilisé par un autre groupe. Il est possible qu'au cours de l'avancée du projet certains éléments de votre document (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 (SP)

Au cours du projet, il est demandé d'effectuer une tâche précise, non triviale dans un laps de temps prédéfini (de mi-septembre au **20 décembre 2017**). Il est attendu :

- une implémentation
- des cas de test associés pour valider l'implémentation
- une documentation (README.md) du projet en anglais décrivant l'objectif, le résultat, la licence, les technologies utilisées, ainsi que l'architecture du projet
- une démonstration du résultat final : une vidéo (« screencast ») de 1' minimum (2' maximum)
- des instructions pour déployer le résultat (et notamment ré-exécuter la démonstration) seront également à inclure dans le README.md

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

Aussi le rendu se fera par un simple mail indiquant l'adresse du « repository » Github. Le travail sera évalué à partir du dernier commit précédent le **20 décembre 2017 (inclus)**.

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 : 9 octobre 2017 (23h59) ; « échange » de projets le 10 octobre

SP : 20 décembre 2017 (23h59)

PR : mi-janvier 2018

La note est sur 20: 5 points pour EX, 10 points pour SP, 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., les matrices de comparaison aka PCM)
- de *techniquement comprendre l'existant* et les technologies déjà utilisées (API Java, base de données, données existantes, etc.)
- de *comprendre l'objectif général* afin de raffiner les besoins par la suite

D'un point de vue technique, il faut être en mesure de répondre aux questions suivantes: quelles sont les fonctionnalités du projet? comment récupérer le code source? comment exécuter, installer et déployer une application? quelles sont les technologies utilisées? comment exécuter des cas de tests? Quelques ressources (pointeurs vers des bouts de code github, base de données, etc.) vous seront fournies.

Un exercice intéressant est d'écrire, dans le document de spécification, l'objectif général à atteindre avec vos propres mots. Cela peut valider votre compréhension du sujet (et de toute façon il sera nécessaire de le faire).

Un 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.**

Enfin ne pas hésiter à échanger avec la communauté open source (Wikipedia, OpenFoodFacts, OpenCompare) et à reporter des bugs ou manques (e.g., sur le github d'opencompare).

Un document spécifique par projet décrivant les besoins du client vous est également communiqué.

Bon projet