Première session des LIFTech'

F. Jaillet[1] and D. Arrivault[2]

[1]Laboratoire d'Informatique Fondamentale
UMR 7373
[2]Laboratoire d'Excellence Archimède
Aix Marseille Université

05 Novembre 2015 / LIFTech' 1ère session

Outline

Qu'est-ce que c'est une journée du développement ?

Objectifs

- Apporter de l'aide au chercheurs qui souhaitent programmer. L'ordre du jour dépend des demandes.
- Mutualiser les pratiques.
- Rencontres autours de thématiques techniques. Favoriser les échanges avec les ingénieurs développement mais aussi entre chercheurs.
- Faire connaître les bonnes pratiques et les outils qui vous feront gagner du temps et de la qualité.

Précisions

Le développement logiciel

Wikipédia :

> *Le développement de logiciel consiste à étudier, concevoir, construire, transformer, mettre au point, maintenir et améliorer des logiciels.*

Un logiciel

Wikipédia :

> *En informatique, un logiciel est un ensemble de séquences d'instructions interprétables par une machine et d'un jeu de données nécessaires à ces opérations.*

Toute ligne de code, instruction, interface est un logiciel et celui qui la produit fait du développement logiciel.

Précisions

Qui fait du développement logiciel ?

Précisions

Qui fait du développement logiciel ?

Le principe de cette journée.

- 4 exposés d'une demi-heure questions incluses + un temps à la fin pour les retours, les questions sur des problématiques personnelles et les suggestions.
- Tout le monde est le bienvenu pour proposer des interventions. Si vous avez une expérience avec un langage, un logiciel, une API (Interface de Programmation), un outil d'aide au développement. . .
- La fréquence n'est pas encore fixée, cela dépendra des propositions d'interventions. Une journée / trimestre serait bien.

Le Labex Archimède.

- 4 Laboratoires (I2M, LIF, LSIS, CPT) + CIRM.
- 4 Missions :
  - renforcer les collaborations inter laboratoires,
  - favoriser les échanges internationaux,
  - développer les échanges de compétences et les transferts technologiques,
  - améliorer les liens entre la recherche et l'enseignement au sein de l'AMU.
- 1 Cellule d'expertise en développement de logiciels. Projets :
  - MACAON, traitement automatique de la langue, indexation morpho-syntaxique de textes.
  - LtfatPy, bibliothèque d'outils temps/fréquence en Python.
  - RR, plateforme de Recherche Reproductible pour les chercheur du labex.
  - GOOL, traduction de code objets (java, C++, C#, python).
  - GOOL translate, le traducteur en ligne de GOOL.
  - Scikit-Gilearn, boîte à outil python pour l'inférence grammaticale.
  - e-IDS, calculateur d'indices du sommeil en ligne.
  - SXP, plate-forme de troc peer to peer.
- Un appel à projets 2/3 fois par an.

Qu'est-ce qu'un bon logiciel ?

Du point de vue de l'utilisateur.

- **Utile** : il répond aux besoins.
- **Ergonomique** : facile à utiliser.
- **Fiable** : il ne plante pas dès qu'on change un paramètre, il passe le test de l'utilisateur novice.
- **Performant** : optimisez, si vous ne le faites pas pour vous, faites le pour la planète...

Qu'est-ce qu'un bon logiciel ?

Du point de vue de l'utilisateur.

- **Utile** : il répond aux besoins.
- **Ergonomique** : facile à utiliser.
- **Fiable** : il ne plante pas dès qu'on change un paramètre, il passe le test de l'utilisateur novice.
- **Performant** : optimisez, si vous ne le faites pas pour vous, faites le pour la planète...

Qu'est-ce qu'un bon logiciel ?

Du point de vue de l'utilisateur.

- **Utile** : il répond aux besoins.
- **Ergonomique** : facile à utiliser.
- **Fiable** : il ne plante pas dès qu'on change un paramètre, il passe le test de l'utilisateur novice.
- **Performant** : optimisez, si vous ne le faites pas pour vous, faites le pour la planète...

Qu'est-ce qu'un bon logiciel ?

Du point de vue de l'utilisateur.

- **Utile** : il répond aux besoins.
- **Ergonomique** : facile à utiliser.
- **Fiable** : il ne plante pas dès qu'on change un paramètre, il passe le test de l'utilisateur novice.
- **Performant** : optimisez, si vous ne le faites pas pour vous, faites le pour la planète...

Qu'est-ce qu'un bon logiciel ?

Du point de vue de l'utilisateur.

- **Utile** : il répond aux besoins.
- **Ergonomique** : facile à utiliser.
- **Fiable** : il ne plante pas dès qu'on change un paramètre, il passe le test de l'utilisateur novice.
- **Performant** : optimisez, si vous ne le faites pas pour vous, faites le pour la planète...

Qu'est-ce qu'un bon logiciel ?

Du point de vue du développeur

- **Evolutif** : la grosse fonction unique atteint toujours ses limites. . .
- **Ouvert** : utiliser de préférences des technologies (langages, interfaces) standards, ouvertes (~~Matlab~~) et maintenues.
- **Maintenable** : commenter le code.
- **Exploitable** : facile à installer, facile à mettre à jour, permettre de sauvegarder et de récupérer les données.
- Les 11 facteurs qualité de Mc Call : exactitude, fiabilité, efficacité, conviviabilité, intégrité, maintenabilité, flexibilité, testabilité, portabilité, reutilisabilité, interoperabilité.

Qu'est-ce qu'un bon logiciel ?

Du point de vue du développeur

- **Evolutif** : la grosse fonction unique atteint toujours ses limites. . .
- **Ouvert** : utiliser de préférences des technologies (langages, interfaces) standards, ouvertes (~~Matlab~~) et maintenues.
- **Maintenable** : commenter le code.
- **Exploitable** : facile à installer, facile à mettre à jour, permettre de sauvegarder et de récupérer les données.
- Les 11 facteurs qualité de Mc Call : exactitude, fiabilité, efficacité, conviviabilité, intégrité, maintenabilité, flexibilité, testabilité, portabilité, reutilisabilité, interoperabilité.

Qu'est-ce qu'un bon logiciel ?

Du point de vue du développeur

- **Evolutif** : la grosse fonction unique atteint toujours ses limites. . .
- **Ouvert** : utiliser de préférences des technologies (langages, interfaces) standards, ouvertes (~~Matlab~~) et maintenues.
- **Maintenable** : commenter le code.
- **Exploitable** : facile à installer, facile à mettre à jour, permettre de sauvegarder et de récupérer les données.
- Les 11 facteurs qualité de Mc Call : exactitude, fiabilité, efficacité, conviviabilité, intégrité, maintenabilité, flexibilité, testabilité, portabilité, reutilisabilité, interoperabilité.

Qu'est-ce qu'un bon logiciel ?

Du point de vue du développeur

- **Evolutif** : la grosse fonction unique atteint toujours ses limites. . .
- **Ouvert** : utiliser de préférences des technologies (langages, interfaces) standards, ouvertes (~~Matlab~~) et maintenues.
- **Maintenable** : commenter le code.
- **Exploitable** : facile à installer, facile à mettre à jour, permettre de sauvegarder et de récupérer les données.
- Les 11 facteurs qualité de Mc Call : exactitude, fiabilité, efficacité, conviviabilité, intégrité, maintenabilité, flexibilité, testabilité, portabilité, reutilisabilité, interoperabilité.

Qu'est-ce qu'un bon logiciel ?

Du point de vue du développeur

- **Evolutif** : la grosse fonction unique atteint toujours ses limites. . .
- **Ouvert** : utiliser de préférences des technologies (langages, interfaces) standards, ouvertes (~~Matlab~~) et maintenues.
- **Maintenable** : commenter le code.
- **Exploitable** : facile à installer, facile à mettre à jour, permettre de sauvegarder et de récupérer les données.
- Les 11 facteurs qualité de Mc Call : exactitude, fiabilité, efficacité, conviviabilité, intégrité, maintenabilité, flexibilité, testabilité, portabilité, reutilisabilité, interoperabilité.

Qu'est-ce qu'un bon logiciel ?

Du point de vue du développeur

- **Evolutif** : la grosse fonction unique atteint toujours ses limites. . .
- **Ouvert** : utiliser de préférences des technologies (langages, interfaces) standards, ouvertes (~~Matlab~~) et maintenues.
- **Maintenable** : commenter le code.
- **Exploitable** : facile à installer, facile à mettre à jour, permettre de sauvegarder et de récupérer les données.
- Les 11 facteurs qualité de Mc Call : exactitude, fiabilité, efficacité, conviviabilité, intégrité, maintenabilité, flexibilité, testabilité, portabilité, reutilisabilité, interoperabilité.

Qu'est-ce qu'un bon logiciel ?

Du point de vue du développeur

- **Evolutif** : la grosse fonction unique atteint toujours ses limites. . .
- **Ouvert** : utiliser de préférences des technologies (langages, interfaces) standards, ouvertes (~~Matlab~~) et maintenues.
- **Maintenable** : commenter le code.
- **Exploitable** : facile à installer, facile à mettre à jour, permettre de sauvegarder et de récupérer les données.
- Les 11 facteurs qualité de Mc Call : exactitude, fiabilité, efficacité, conviviabilité, intégrité, maintenabilité, flexibilité, testabilité, portabilité, reutilisabilité, interoperabilité.

Tout code à vocation à être repris.

Qu'est-ce qu'un bon logiciel ?

Du point de vue du développeur

- **Evolutif** : la grosse fonction unique atteint toujours ses limites...
- **Ouvert** : utiliser de préférences des technologies (langages, interfaces) standards, ouvertes (~~Matlab~~) et maintenues.
- **Maintenable** : commenter le code.
- **Exploitable** : facile à installer, facile à mettre à jour, permettre de sauvegarder et de récupérer les données.
- Les 11 facteurs qualité de Mc Call : exactitude, fiabilité, efficacité, conviviabilité, intégrité, maintenabilité, flexibilité, testabilité, portabilité, reutilisabilité, interoperabilité.

Tout code à vocation à être repris.
Merci pour nous...

Les étapes du développement.

En matière de logiciel, il est conseillé de faire précéder l'action par la réflexion.

Dicton Populaire

Spécifier.

Ecrire ce que le logiciel doit faire.

- Les cas d'utilisation : comment il va s'utiliser.
- Les besoins fonctionnels : comment les données seront modifiées.
- Les besoins non fonctionnels : environnement, performance, sécurité, support, maintenabilité...
- Les limites du logiciel : ce qu'il ne fera pas.
- Chaque spécification doit pouvoir être démontrée/testée.

Fixer les règles.

Réfléchir à comment on va le faire.

- Le ou les langage(s) à utiliser,
- les bibliothèques externes à utiliser (influence sur la licence),
- la chaîne de compilation (compilateur, outils),
- la structure du dépôt source,
- le ou les paradigmes de programmation,
- les règles de nommage,
- la gestion des erreurs,
- la gestion des commentaires,
- ...

Rédiger un document d'architecture (expliquant les choix de programmation) qui sera mis à jour durant le développement.

Développer avec méthode.

Quelques conseils.

- Développer par l'exemple : on commence par construire un exemple complet (code, test, documentation) qui sert de template pour la suite.
- Faire de la revue de code si on développe à plusieurs.
- Documenter au plus près du code (Génération automatique de documentation).
- Rédiger des tutoriels et un document utilisateur.
- Ecrire des tests et les automatiser : unitaires (structurels), intégration, fonctionnels (système global), recette (acceptation).
- Ecrire des classes et des méthodes de taille raisonnable.

En résumé

- YAGNI : You Ain't Gonna Need It, pas de fonctionnalités qui ne sont pas spécifiées.
- KISS : Keep It Simple.
- DRY : Don't Repeat Yourself, le copier/coller est à éviter.
- Réutiliser du code, ne pas réécrire ce qui existe déjà.

Les gestionnaires de version.

Permet de stocker les fichiers source en conservant la chronologie de toutes les modifications qui ont été effectuées.

Les plus courants.

- GIT
- Subversion ou SVN
- Mercurial
- CVS

Les Forges.

Ensemble d'outils qui facilitent le développement collaboratif. Une forge intègre généralement : un gestionnaire de version ; un gestionnaire de listes de discussion (et/ou de forums) ; un outil de suivi des bugs ; un gestionnaire de documentation (souvent sur le principe du wiki) ; un gestionnaire des tâches.

Quelques exemples

- Souce Sup de Renater : https://sourcesup.renater.fr/
- GForge Inria : https://gforge.inria.fr/
- GitLab du LIF : https://gitlab.lif.univ-mrs.fr
- SouceForge : http://sourceforge.net/
- GitHub de GiHub Enterprise : https://github.com/

Pour les deux dernières le code doit être open source.

D'autres outils bien utiles

Intégration Continue

Adossée à une forge, elle compile, lance les tests et rapporte les résultats à chaque nouveau commit. Jenkins, GitLab CI, travis-ci.

Les outils de build

Gestion de toute la chaîne de compilation, des tests et de la distribution.

- Pour Java : Maven, Gradle.
- Pour C++ : CMake/CTest/CPack.
- Pour Python : distutils/pip

Et encore. . .

- tests unitaires : cunit, cppunit, junit, pyunit, Funit . . .
- IDE : emacs, Eclipse, Netbeans, Spyder. . .
- Debuggers : gdb, jdb, pdb.
- Profiling : gprof, valgrind.

Conclusions

- La qualité doit être un principe dès le début du projet.
- Faire simple.
- Le développement c'est : 1/3 de réflexion, 1/3 de programmation, 1/3 de tests.

Credits

- Uncle Sam (pointing finger) : http://commons.wikimedia.org/wiki/File:Uncle_Sam_(pointing_finger).jpg
  Picture under a Creative Commons licence from James Montgomery Flagg (1916-1917)