Conduite de projet : Quelques conseils.

Mo Foughali
(IRIF, U-Paris) – foughali@irif.fr
(d'après transparents originaux de Yann Régis-Gianas et Aldric Degorre)

26 novembre 2021

Echéances à venir

- Examen écrit (compréhension du cours) - **jeudi le 6 janvier 2022 de 16h00 à 17h00, dans les amphis 1A, 4C et 6C**
- Rendu TP (vidéo) - deadline **vendredi le 14 janvier 2022 à 23h59**
  - C'est une vidéo décrivant **votre produit** et non la contribution de chacun (que nous connaissons déjà)
  - Toutes les fonctionnalités doivent être décrites et montrées
  - Tout le monde doit parler
  - Durée **maximale** de la vidéo : n minutes ($n \in \mathbb{N}$ étant le nombre d'étudiants constituant le groupe)

Séance de cours d'aujourd'hui

Quelques conseils pour conclure cette partie du cours :

Comment devenir un meilleur développeur ?

1. Comment bien travailler dans de bonnes conditions ?
2. Comment bien collaborer ?
3. Comment bien programmer ?
4. Comment progresser ?

Comment travailler dans de bonnes conditions ?

L'importance des outils

Utilisez les bons outils

- d'ailleurs, qu'est-ce qu'un bon outil ?

Connaissez-les

- au delà du cours et des TP. Soyez curieux

Une chose à la fois

Concentrez-vous

- ... sur une seule tâche à la fois
- mieux vaut une fonctionnalité opérationnelle que cinquante buggées

Comprenez ce que vous avez à faire

- ... avant de vous embarquer dans du code

Une course de fond

Dormez

- vous n'êtes pas surhumains...

Repoussez les deadlines

- les échéances sont souvent sous-estimées

De la bonne paresse

Automatisez les tâches répétitives de développement

- Ecrivez e.g. des scripts, des générateurs de code

Exemple ?

Restez à jour

Terminez ce que vous commencez...
Ne produisez pas des vaporware

- d'ailleurs, Bill Gates est lauréat du prix Vaporware (Vaporware award) en 1985

Comment bien collaborer ?

Soyez honnêtes

The greatest of all weaknesses is the fear of appearing weak.
J. B. Bossuet, Politics from Holy Writ, 1709

Prenez vos responsabilités
Admettez vos erreurs
Explicitez vos faiblesses

puis

Proposez des solutions, pas des excuses.

L'histoire de la soupe aux cailloux

Ne soyez pas celui/celle avec le caillou (ni les autres, d'ailleurs)

Votre message

Communiquez

- Réfléchissez à ce que vous voulez dire
- Réfléchissez à comment vous voulez le dire
  - ... en vous mettant à la place de votre interlocuteur
- Dites-le (sujet, verbe, complément)

Comment bien programmer ?

Du perfectionnisme (version positive)

Ne laissez pas une fenêtre cassée dans votre code

- les fameux FIXME, éventuellement déguisés en TODO

Du perfectionnisme (version négative)

Apprenez à fixer un niveau de qualité suffisant pour votre projet...

De l'endurance

Retravaillez votre code

Du débogage

Reproduisez les bugs

- ... en écrivant des tests

Ne supposez rien, prouvez-le...
Ecrivez du code qui s'observe
Ecrivez du code qui se teste facilement

- exemple de code qui se teste difficilement ?

Embrasser le changement

Apprêtez-vous à tout jeter...

Atteindre le but, très vite

Construisez d'abord le squelette complet du produit

- c'est à dire ?

Programmez incrémentalement

- idem, c'est à dire ?

Faites, si vous savez comment faire

- sinon ?

La modularisation

Une fonction, un composant = une phrase
Paramétrez
Structurez l'enchaînement des calculs

On programme pour des humains

Communiquez avec votre code (oui, soyez un peu fous)
Commentez
Documentez (prochain cours)

La complexité des logiciels

Supprimez du code
Gardez-le simple (et stupide, s'il le faut)
Luttez contre la complexité avec de la généralité

Comment progresser ?

Que sais-je ?

Fixez-vous des compétences à acquérir à courts et à moyens termes
N'utilisez jamais du code que vous ne comprenez pas

Extraire l'information

Posez des questions
Lisez du code

Qu'est-ce qui fait un bon développeur ?

Résumé

Au risque d'être un peu normatif, un bon développeur :

- **s'adapte rapidement** à de nouvelles technologies, de nouvelles idées, de nouvelles équipes ;
  - ça vous rappelle quelque chose ?
- **veut comprendre** et pose donc continuellement des questions pour savoir "comment cela fonctionne" ;
- **a un esprit critique**, ne se contente pas d'un "c'est comme ça !" et cherche des justifications ;
- **est réaliste**, en essayant d'affronter les difficultés sans les éluder ;
- **a un large spectre de compétences** qui lui permettent de construire une compréhension profonde et une approche raisonnée des problèmes de développement logiciel.

Résumé (suite)

mais surtout le plus important :

- Un bon développeur accorde une grande importance à ce qu'il fait.
- Un bon développeur réfléchit à ce qu'il fait.

Pour résumer

Pour résumer

REFLECHISSEZ!