

# {Restez Libres, Soyez Unix }

## Petite philosophie de l'informatique moderne

publié par Guillaume le 2006-10-29 21:45:00



On pourrait résumer le post par "**Comment on nous mène en bateau !**"

Un petit article sans prétention sur une vision que l'on peut avoir de l'informatique.

Ayez le courage de lire tous au jusqu'au bout, et dites-moi si vous êtes d'accord, car je sais l'idée peut être sujet à polémique...

### REMONTONS A LA SOURCE :

Etant étudiant en informatique, une palette de cours nous présente l'informatique e sous ses différentes formes. Un cours particulier, "l'analyse", nous permet de voir la programmation sous sa forme conceptuelle.

Les deux grands concepts de l'analyse sont bien entendu [Merise](#) et [UML](#) (je ne devrais d'ailleurs pas employer concept mais langage en parlant de l'[UML](#) ). [Merise](#) est une "invention" française se basant sur l'esprit cartésien des français qui l'utilisent. Même si cela peut vous paraître idiot, le fait que j'insiste sur l'origine française des utilisateurs a pourtant toute son importance.

En effet, contrairement à [Merise](#), [UML](#) est un langage inventé par les américains , dans le but d'être utilisé par les anglo-saxons.

Comme vous avez pu vous en rendre compte, nous ne réfléchissons pas de la même manière selon nos origines. Ceci est surtout dû au fait que les méthodes d'apprentissages sont différentes selon les pays.

Tous ceci pour vous dire qu'[UML](#) à été élaboré de tel sorte que l'on se fasse une image globale, donc non exhaustive du projet sur lequel nous travaillons.

C'est pourquoi nous français, avec nos esprits cartésiens avons tant de mal à comprendre cette méthode => ***il suffit de regarder les copies de contrôle de mon département informatique ;-)***

Le problème est qu'[UML](#) s'est fortement démocratisé dans les entreprises. C'est d'ailleurs une évidence car [UML](#) est bien plus rapide et moins cher à mettre en place que [Merise](#).

Mais cette utilisation d'[UML](#) a entraîné **la politique du bug**.

## LA POLITIQUE DU BUG :

Cette politique m'a été mise en évidence par une personne que je ne citerai pas mais qui se reconnaîtra sûrement si elle lit cet article.

Je vous explique le principe.

- Développez un projet avec [UML](#) , donc de manière globale. Au lieu de réitérer les opération d'affinage des problèmes, arrêtez vous à 80% de l'analyse.
- Programmez votre application. et vendez-la à un prix créant une marge convenable.
- L'application n'étant pas terminée, certains, voire de nombreux bugs sont possibles. Aucun problème, expliquez au client que moyennant une somme forfaitaire il peut s'abonner aux mises à jour du programme, permettant une meilleur sécurité, fonctionnalité...
- Le client paye donc les corrections de bug, et l'implémentation des fonctionnalités qui n'ont pas été introduites dans votre application.

**Conclusion : les 20% restant du programme ont été financé par le client, vous n'avez dépensé aucune pièce.**

## ANALOGIE AVEC LE "MONDE REEL" :

Imaginez que cette technique soit mise en place pour d'autres secteurs que l'informatique.

Vous allez chez un concessionnaire réputé de voitures. Vous achetez l'automobile de vos rêves mais au moment de vous donner les clefs le vendeur vous dit :

***"Bon il y aura peut-être un souci ou deux dans 150km mais vous inquiétez pas, on a l'habitude, vous la ramenez on verra ce que l'on peut faire".***

Vous l'accepteriez ?

Maintenant vous achetez une machine à laver le linge. vous installez l'engin dans votre buanderie et lors de la lecture de la notice vous voyez

***"Il se peut qu'au bout d'un mois ou deux le tambour se détache et détruit toutes vos affaires".***

Vous laveriez votre linge à la main non ?

## CONCLUSION :

Même si vous avez trouvé ces exemples stupides, il faut se dire que c'est ce que la plupart des consommateurs subissent lorsqu'ils achètent un logiciel. Attention, je ne veux pas du tout dire que [UML](#) est mauvais, au contraire il est très performant. C'est

son utilisation dans le seul but de faire du bénéfice que je remet en cause.

L'informatique a une chance énorme, elle peut vendre des produits défectueux et non finis sans que personne ne lui reproche rien. Et en plus, cela lui rapporte de l'argent.

J'espère qu'un jour nous n'accepterons plus cette politique du bug, même si un programme sans erreur est impossible, si les entreprises pensaient moins au business, elles pourraient limiter grandement la casse...