Développement coopératif

Samuel Tardieu

sam@enst.fr

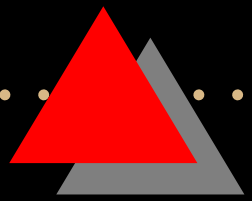
École Nationale Supérieure des Télécommunications

Introduction

- Les projets logiciels à Télécom Paris étaient: (souvent)
  - Originaux et différents à chaque trimestre
  - Proches de l'état de l'art
  - Fonctionnels et de bonne qualité
  - Perdus après la fin du trimestre
  - Rarement repris par une autre équipe
- On souhaite ne jamais retrouver les deux derniers points

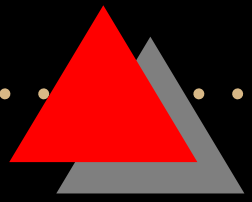
Sommaire

- Qu'est-ce que le développement coopératif?
- Quels outils utiliser?
- Les environnements intégrés



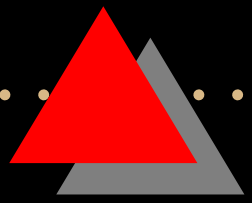
Développement coopératif

- Consiste à
  - Partager la connaissance de ce qu'il y a à faire
  - Partager le code source et la documentation
- À tout moment le projet doit compiler
- Aucun bug ne doit réapparaître
- Pour toute tâche ou problème, un responsable peut être identifié



Les versions

- Il faut distinguer
  - La dernière version qui compile et semble fonctionner (commune)
  - La dernière version testée intensivement, qu'il faut pouvoir produire rapidement à tout moment
  - La version de travail courante d'un développeur
- Intérêt immédiat: *diff* entre ces versions

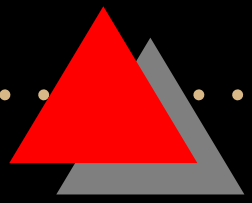


Les versions avec CVS

- Dans le dépôt, se trouvent:
  - La dernière version qui semble fonctionner
  - La dernière version testée intensivement, avec deux étiquettes (*tags*):
    - Un numéro de version (`R1_3` pour la version 1.3)
    - Une étiquette déplaçable (`STABLE`)
- Chez chaque développeur se trouve sa version de travail **personnelle**, qui ne fait référence qu'à des choses dans le dépôt

Les prereleases

- Version α:
  - Implémentation et documentation incomplètes
  - But: vérification des attentes du «client»
- Version β:
  - Implémentation complète
  - Documentation presque complète
  - But: découvrir de nouveaux problèmes

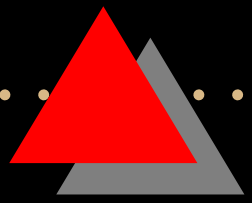


Les releases

- Version jalon
  - Implémentation et documentation partielles complètes
  - Les jalons sont idéalement posés avant le développement («roadmap»)
  - But: distribuer un logiciel utilisable
- Version définitive
  - Implémentation et documentation complètes
  - But: distribuer un logiciel parfait

Conséquences

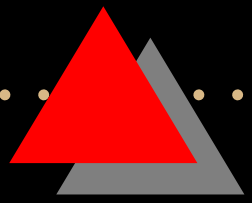
- Il faut
  - Tester ses changements avant de les rentrer dans le dépôt commun
  - Ne jamais échanger des fichiers autrement qu'en passant par le dépôt (courrier électronique, lecture directe du répertoire de travail, etc.)
  - Toujours indiquer clairement dans le message d'accompagnement pourquoi un changement a été fait



Attention!

Un gestionnaire de configuration comme CVS ne remplace **jamais** la discussion entre développeurs:

- courrier électronique
- messagerie instantanée (Jabber, ICQ, AIM, Y!Messenger)
- contact direct

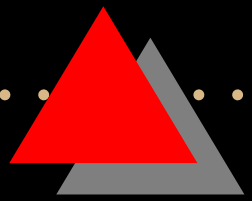


Qu'est-ce qu'un logiciel parfait?

Un logiciel parfait:

- répond totalement au cahier des charges
- implémente correctement chaque fonctionnalité annoncée
- n'en fait pas nécessairement plus qu'annoncé

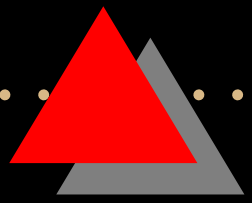
Un logiciel parfait doit avoir, **dans son domaine d'application**, zéro défaut.



BTS

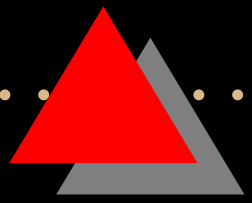
Un *Bug Tracking System* sert à:

- enregistrer les rapport de problèmes (*bugs*);
- faire des demandes d'amélioration (*wishlist* ou *feature request*);
- permettre à l'auteur de suivre l'évolution de sa demande ou de son signalement (ticket);
- permettre aux développeurs de suivre l'évolution d'un problème.



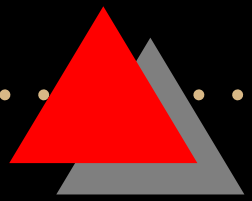
BTS: cycle de vie

- Les états possibles d'un tickets sont typiquement:
  - Ouvert ou Fermé
  - Analysé
  - En attente d'information complémentaire
  - Suspendu
- Un ticket fermé peut être réouvert



Tests unitaires

- Les tests unitaires permettent de:
  - Valider une unité à partir de sa spécification
  - Vérifier qu'une modification ne casse pas une unité
  - Faire confiance au code d'un autre développeur pour se concentre sur celui que l'on développe
- Idéalement, les tests unitaires sont écrits par un autre développeur



TELECOM PARIS

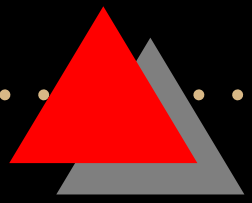
Tests de non-régression

Les tests de non-régression, qui couvrent plus qu'un module,

- Sont créés lorsqu'un problème est identifié
- Si possible avant que le problème soit corrigé
- Doivent être vérifiés **systématiquement**:
  - Tout problème, aussi évident soit-il, peut se reproduire
  - Il est extrêmement désagréable pour un client ou un développeur de voir se reproduire un problème déjà rencontré

BTS et code

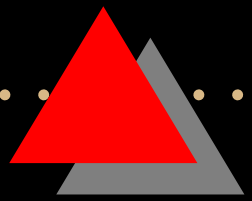
- L'idéal est de lier les tickets et le code source (original et tests de non-régression):
  - En regardant l'historique d'un ticket, on extrait les changements qui y sont liés
  - En regardant un changement, on sait ce qu'il corrige
  - En regardant un test de non-régression, on sait pourquoi il a été créé



Documentation

La documentation est nécessaire pour

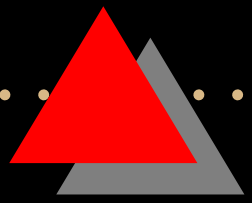
- Savoir comment utiliser le logiciel
- Documenter les choix faits
- Documenter les choix architecturaux
  - Pour les développeurs actuels
  - Pour les développeurs futurs



Quels outils utiliser?

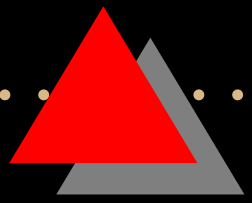
Typiquement, les outils suivants sont utilisés:

- Gestion des versions: CVS, GNU Arch, Perforce (propriétaire), Subversion (en α-test)
- Gestion des tickets: RT2, BTT, Gnats
- Documentation: texinfo (pour générer du texte, de l'HTML, du PDF, du PS, ...), docbook (basé sur XML), LaTeX, Microsoft Word, texte seul



Serveurs coopératifs

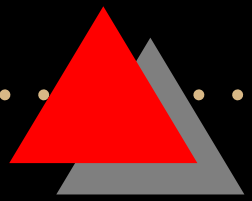
- Idée: intégrer tous les outils et s'affranchir de l'étape d'installation
- Interface WWW pour les outils
- Principaux serveurs existants:
  - Savannah (FSF)
  - SourceForge (VA)
  - PiCoLibre (GET)



Principes

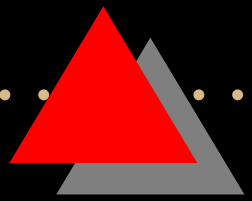
Les étapes sont:

- Choisir un nom pour le projet (le plus dur)
- Créer un compte développeur
- Créer un projet avec ce compte développeur
- Créer d'autres comptes développeur
- Ajouter ces développeurs au projet
- Écrire les pages correspondantes



Intérêts

- Pour les élèves:
  - suite d'outils à disposition
  - «obligation» de structurer le développement
- Pour les enseignants:
  - facilité de suivi du projet
  - facilité de continuation du projet



Conclusion

Le choix d'un bon outil de gestion de version et de documentation facilitera le travail de tous, à commencer par le votre.