upmf Grenoble
Université Pierre Mendès-France
Sciences sociales & humaines

IUT 2 Grenoble
Université Pierre-Mendès-France

GUILDe

Vortex

Développement et menée d'un projet libre à titre personnel

Avant de commencer

- Qui sait programmer?
- Qui a déjà pensé à lancer son projet de logiciel libre?
- Qui a effectivement pris cette initiative?
- Qui est allé jusqu'au bout?

Projets personnels

- Dans le monde du libre, un énorme pourcentage de logiciels sont le fruit de projets personnels.

- Au fil des années, le travail devient énorme, rassemblant énormément de développeurs, et apte à concurrencer des logiciels professionnels.

- Linux: initié en 1991 par un seul étudiant
- Aujourd'hui, plusieurs milliers de développeurs y travaillent.
- Utilisé en entreprise, par des particuliers, etc.
- Coût estimé du projet si il avait été réalisé par une entreprise: 10 milliards de dollars, pour 60 000 années-hommes de travail.

(http://www.linuxfoundation.org/publications/estimatinglinux.php)

- Qui sait programmer?

- Qui a déjà pensé à lancer son projet de logiciel libre?

- Qui a effectivement pris cette initiative?

- Qui est allé jusqu'au bout?

A qui s'adresse cette conférence?

- Aux informaticiens de formation qui ne se sont jamais lancés de leur propre chef
- Aux développeurs étrangers au monde du libre
- Aux curieux

- Ce n'est pas un cours de management ou de gestion de projet, et encore moins de code: seulement un retour sur expérience.

De quoi va t-on parler?

- Le projet Vortex
  - En quoi a consisté le projet?
  - Gestion du projet
  - Retour sur le développement
- Gestion de projet
  - Méthodologies Agile
  - TDD
  - Gestion d'équipe
- Outils
  - Communication: Wikis & Mailing Lists
  - Code: SVN & Bug Trackers


VORTEX

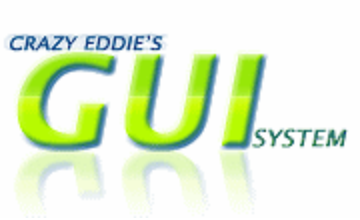
Contexte du projet

- Réalisé dans le cadre des “Projets Tutorés”, deuxième année de DUT Informatique
- Equipe de 6
- 16 semaines
- Soutenance effectuée fin janvier

- Note obtenue: 16.37/20

Uniquement des outils libres

C++



Mozilla Public License

Projet sous MPL: la licence n'impose des restrictions que sur le fichier qu'elle couvre

Contrairement à des licences telle que la GPL posant des restrictions de licence sur l'ensemble de l'application.

Gestion du projet et coordination

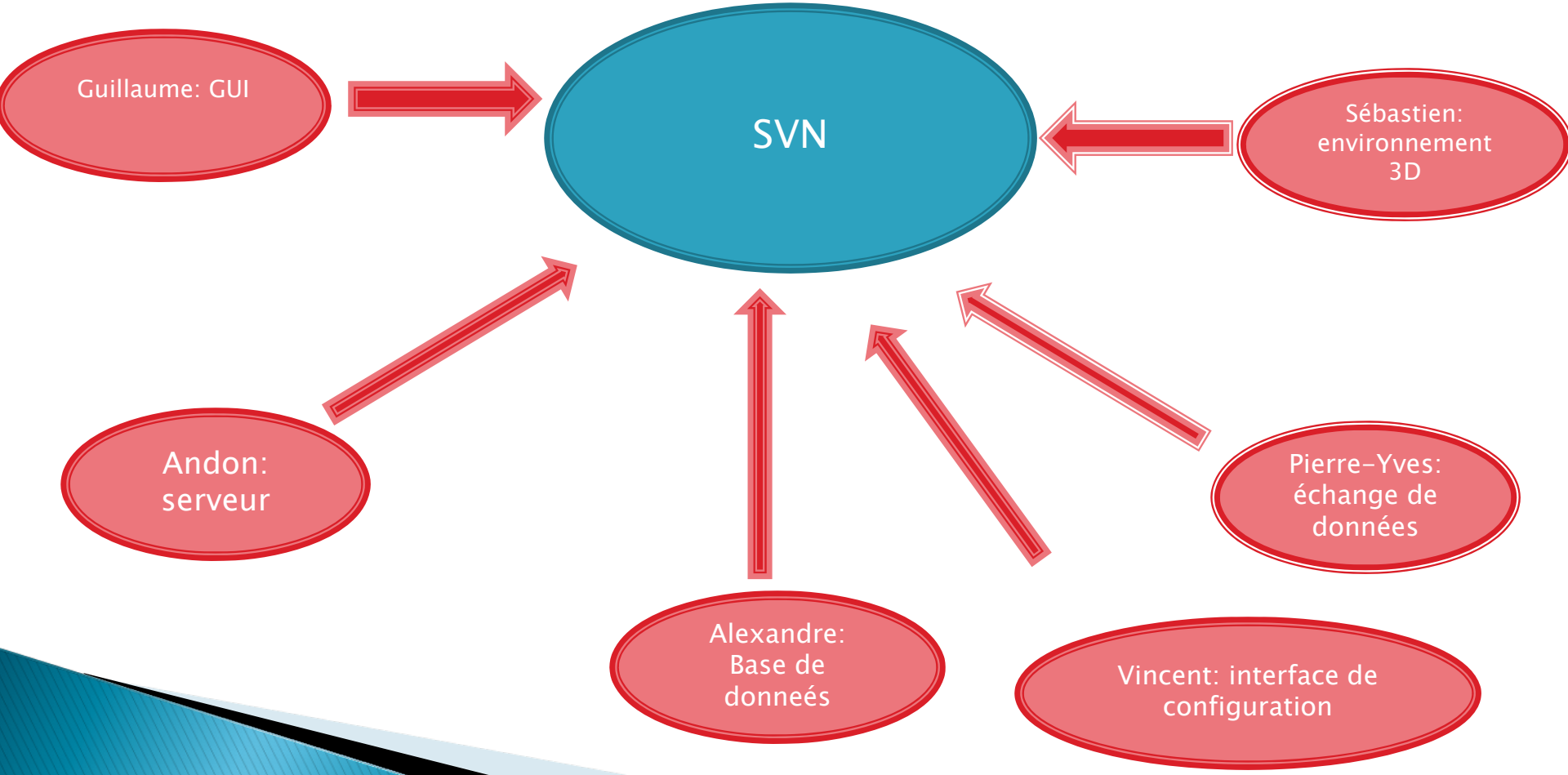
Vue générale du projet

- 16 semaines (cours, autres projets, partiels,… inclus)
- Technologies complètement nouvelles à l'équipe
- Projet ambitieux

1 : Mise en place d'outils

- Wiki
- SVN
- Mailing list

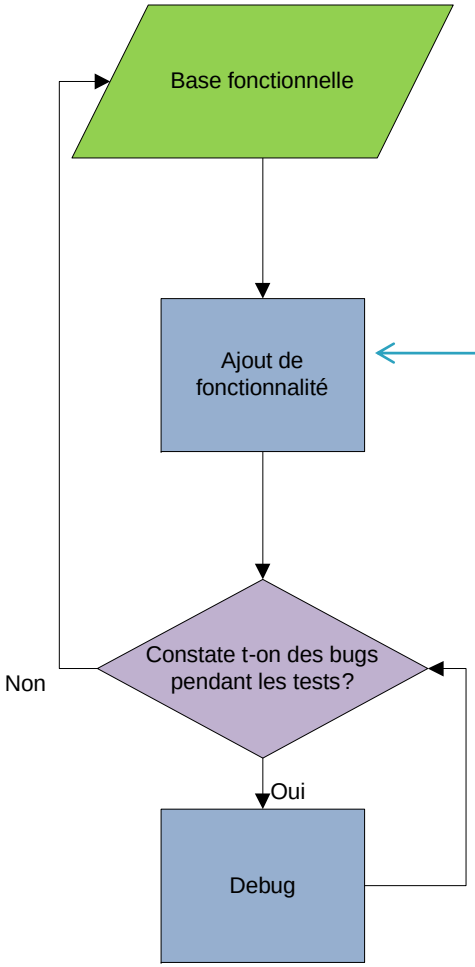
2: Première répartition des tâches



Dysfonctionnements

- Ce système a rapidement montré ses limites
- Une base fonctionnelle a vite été établie, mais la progression était problématique.
- Besoin d'une réorganisation

Méthode de développement



Chacun continue sur son domaine

Approche itérative

Au final

- Approche hybride (TDD, Agile, par tâches)
- Efficace car a réussi à gérer un projet d'une taille conséquente sur le court terme.
- Eût-elle été adoptée dès le début, nous aurions sans doute été plus efficace.
- La mise en place de tests unitaires aurait pû être intéressante si le projet avait été plus long.

Post-mortem

- « L'expérience est ce que l'on obtient quand on n'a pas eu ce que l'on voulait »
  - Randy Pausch

Objectifs de base

- Multiutilisateurs

- Environnement 3D « réaliste »

- Possibilité d’envoyer des messages

- Possibilité de consulter les médias des autres

Ce qui n'a pas marché

- Un membre en moins
- Eparpillement pendant la première moitié du projet: perte de temps
- Projet peut être trop ambitieux

Ce qui a marché

- Nous étions tous motivés par le projet
- Le résultat reste convenable pour un projet étudiant réalisé en si peu de temps
- Gestion d'équipe et phases de développement efficace sur la seconde moitié

Méthodologies Agile

Gestion de projet

- Traditionnellement, la gestion de projet dans les entreprises est très hiérarchisée, basée sur les coûts, les limites temporelles et les besoin du client.

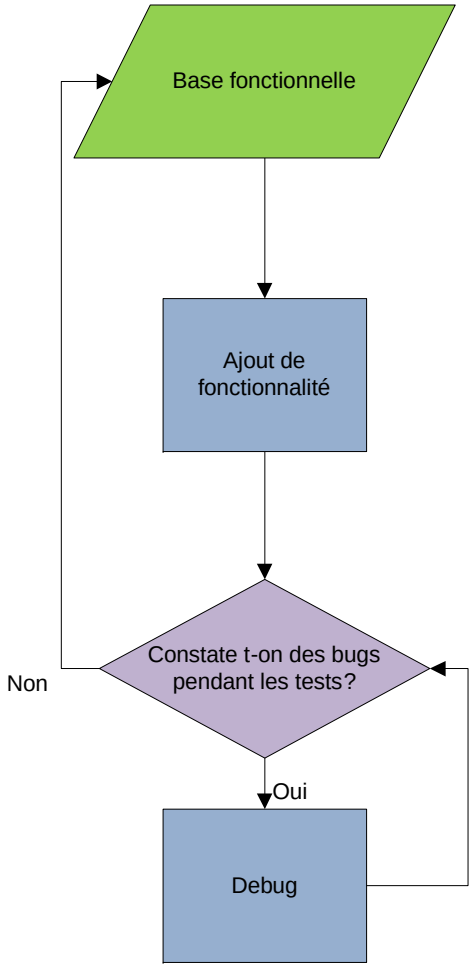
- Or dans le monde du libre:
  - On n'est pas payés
  - On n'a aucune limite de temps
  - On n'a pas de client au sens traditionnel
  - La hiérarchie est très flexible

Méthodologies Agile

- Récentes: datent de la fin des années 90.
- Ont pour but de s'adapter vite et efficacement plutôt que de suivre un plan potentiellement obsolète.
- Encouragent le travail en équipe et l'autonomie.
- Certaines de ces méthodes ne prévoient même pas de chef de projet.

Approche itérative

- Pierre angulaire des méthodologies agile.
- Le but est d'avoir, à tout moment du projet, une version opérationnelle.
- On ajoute à cette version une ou plusieurs fonctionnalités, puis on passe en phase de debug, pendant lequel on ne code rien de nouveau (freeze).



Typiquement en entreprise,
on a des itérations d'un mois.

Approche itérative

TDD: Test Driven Development

- On constate donc que dans cette approche, la phase de debug est critique.
- On dit donc que c'est une approche TDD: Développement Conduit par les Tests.
- Il est cependant très dur, voir impossible, pour des développeurs de passer leur temps à compiler et à faire la chasse au bug.

Build automatisés

- On va donc automatiser la compilation dans un premier temps.
- Cela est très aisé; sous Linux par exemple, un script bash et cron font l'affaire pour compiler un projet chaque nuit.
- Des solutions plus flexibles et puissantes existent: Ant pour Java, par exemple.
- Ces outils peuvent générer de la documentation, envoyer un mail à la fin de compilation, tester que le logiciel s'exécute bien, encrypter des fichiers, etc, etc.

	Cowon D2 - Boot	Cowon D2	Cowon D2 - Sim	Zen Vision	ZVM 30GB	ZVM 30GB - Sim	ZVM 60GB	FM Rec - Boot	FM Rec	FM Rec - Sim	M-Robe 100 - Boot	M-Robe 100	M-Robe 100 - Sim	M-Robe 500	M-Robe 500 - Sim	Ondio FM	Ondio FM - Sim	Ondio SP - Boot	Ondio SP	Philips HDD1630 -	Philips HDD1630	Philips HDD1630 -	Player - Boot	Player	Player - Sim	Recorder - Boot	Recorder	Recorder - 8MB	Recorder - Sim	Sansa Clip - Boot	Sansa Clip	Sansa Clip - Sim	Sansa Fuze - Boot	Sansa Fuze	Sansa Fuze - Sim	Sansa c200 - Boot	Sansa c200	Sansa c200 - Sim	Sansa e200 - Boot	Sansa e200	Sansa e200 - Sim	Sansa e200v2 - Bo	Sansa m200v4 - Bo	Sansa m200v4	Sansa m200v4 - Si	Gigabeat F - Boot	Gigabeat F	Gigabeat F - Sim	Gigabeat S - Boot	Gigabeat S	Gigabeat S - Sim	V2 Recorder	iAudio M3 - Boot	iAudio M3	iAudio M3 - Sim	iAudio M5 - Boot	iAudio M5	iAudio M5 - Sim	iAudio X5 - Boot	iAudio X5	iAudio X5 - Sim	iPod 1G/2G - Boot	iPod 1G/2G	iPod 3G - Boot	iPod 3G	iPod 4G Gray - Boo	iPod 4G Gray	iPod 4G Gray - Sim	iPod Color - Boot	iPod Color	iPod Color - Sim	iPod Mini 1G - Boo	iPod Mini 1G	iPod Mini 2G - Boo	iPod Mini 2G	iPod Mini 2G - Sim	iPod Nano - Boot	iPod Nano	iPod Nano - Sim	iPod Video - Boot	iPod Video	iPod Video - Sim	iPod Video 64MB	H10 - Boot	H10	H10 - Sim	H10 5GB	H100 - Boot
2009-02-23 23:00	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2009-02-23 22:50	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2009-02-23 21:09	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2009-02-23 16:21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2009-02-23 16:10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2009-02-23 08:46	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2009-02-23 04:34	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2009-02-22 23:35	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2009-02-22 19:26	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2009-02-22 18:25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2009-02-22 13:55	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2009-02-22 13:25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2009-02-22 10:13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2009-02-21 22:22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2009-02-21 17:57	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2009-02-21 16:37	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2009-02-21 11:11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2009-02-21 09:44	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2009-02-21 01:08	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2009-02-21 00:53	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

95 builds in 467 seconds (7mins 47secs) makes 4.9 seconds/build (the most recent build took 9mins 14secs)

File size deltas of the binary main Rockbox images during the most recent commits. Hover over the delta to get the exact file size in bytes.

Timestamp	Player	Recorder	Recorder - 8MB	FM Rec	V2 Recorder	Ondio FM	Ondio SP	iAudio M3	iAudio M5	iAudio X5	H100	H120	H300	H10 5GB	H10	Philips HDD1630	iPod 1G/2G	iPod 3G	iPod 4G Gray	iPod Color	iPod Video	iPod Video 64MB	iPod Mini 1G	iPod Mini 2G	iPod Nano	Sansa e200	Sansa c200	Sansa Clip	Sansa Fuze	Sansa m200v4	Gigabeat F	Gigabeat S	M-Robe 500	M-Robe 100	Cowon D2	ZVM 30GB	ZVM 60GB	Zen Vision	Delta	
2009-02-23 23:00	0	0	0	0	0	0	-16	52	52	50	52	52	50	128	128	112	144	112	128	128	128	112	128	128	128	128	64	72	116	80	74	66	110	96	0	102	118	124	77	log
2009-02-23 22:50	0	0	0	0	0	0	0	12	12	14	12	12	14	-16	32	0	0	16	16	0	16	16	0	0	16	0	32	12	14	12	6	14	0	32	28	-4	6	0	8	log
2009-02-23 21:09	0	0	0	0	0	0	0	68	68	66	68	68	66	112	80	128	96	112	96	112	96	96	96	112	96	112	96	96	110	88	96	96	94	80	98	72	88	88	75	log
2009-02-23 16:21	0	0	0	0	0	0	0	32	32	32	32	32	32	16	32	16	32	0	16	32	32	32	32	32	32	32	32	0	20	0	14	28	0	0	20	0	0	6	17	log
2009-02-23 16:10	0	0	0	0	0	0	0	0	0	0	0	0	0	32	16	-16	16	0	0	16	0	0	0	-16	0	0	-16	0	0	0	14	-10	0	0	0	0	0	0	0	log
2009-02-23 08:46	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-16	0	0	0	16	0	-16	0	0	16	0	0	16	0	0	-16	0	10	0	0	-6	26	0	-6	0	log
2009-02-23 04:34	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	16	-16	0	0	-16	16	0	16	0	0	0	0	-8	12	0	0	0	-14	0	0	0	-26	6	0	log
2009-02-22 23:35	0	0	0	0	0	0	0	0	0	0	0	0	0	0	16	0	0	16	0	0	0	0	-16	-16	0	0	0	0	0	16	0	-10	0	0	6	-26	26	-6	0	log
2009-02-22 19:26	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-16	0	16	-16	0	0	-16	0	16	16	0	-16	-16	0	0	-16	0	10	14	0	0	26	0	0	0	log
2009-02-22 18:25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	16	-16	-16	16	0	0	16	0	0	-16	0	0	16	0	0	16	0	98	-14	0	-6	0	-26	6	2	log
2009-02-22 13:55	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-16	0	0	0	0	-16	16	0	16	0	0	-12	-16	0	0	14	0	6	-26	0	0	0	log
2009-02-22 13:25	32	32	32	32	32	0	0	32	32	32	32	32	32	32	48	48	48	64	48	32	32	32	48	32	32	0	0	8	0	16	30	0	32	32	0	26	26	0	26	log

Tests automatisés

- Cependant, ce n'est pas parce que le code compile qu'il est forcément bon.
- Et chercher les bugs prends du temps qui pourrait être utilisé pour développer…
- …mais si ce temps n'est pas pris, l'application n'est pas utilisable.
- On va donc automatiser nos tests.

Tests Unitaires et Tests Système

- Les tests unitaires testent uniquement une procédure ou un objet du code source.

- Les tests systèmes testent l'application dans son ensemble.

On met donc au point des cas typiques d'utilisation, que l'on retranscrit dans les tests.

JUnit

- Outil de test unitaire pour Java.
- Nos tests sont du code Java!
- On spécifie nos conditions de départ (paramètres, fichiers, etc.), les actions à effectuer et ce que l'on attend à l'arrivée.

- Il existe tout un ensemble de frameworks similaires pour les autres langages, fonctionnant sur le même principe:

- PHPUnit, CPPUnit, PyUnit, etc.

Cruise Control

- Outil de CI (*Continuous Integration*) interfaçant le gestionnaire de versions, l'outil de compilation automatisé, l'outil de test automatisé.
- Dès qu'une nouvelle version est codée, tout est effectué.
- Un rapport des tests peut par exemple être envoyé par mail aux développeurs.

Malheureusement…

- C’est toujours aux développeurs de trouver des tests pertinent
- Cela ne garantit pas une application intégralement sans bugs, même si ça facilite les choses.

Gestion d'équipe: le grand problème

- En développement « amateur », pas moyen d'imposer des horaires ou de baisser les salaires…
- C'est au responsable du projet de trouver le bon équilibre en liberté des développeurs et tâches à leur assigner (tout en gardant à l'esprit leurs compétences et préférences propres)
- Pas de solution miracle, si ce n'est s'adapter.

Outils

Communication

- Au sein de projets amateurs, il est dur voire impossible d'avoir des réunions « classiques »: distance, emploi du temps de chacun, etc.
- Il est malheureusement indispensable de communiquer perpétuellement.
- Heureusement, le net permet de faire cela très efficacement et de nombreuses manières.

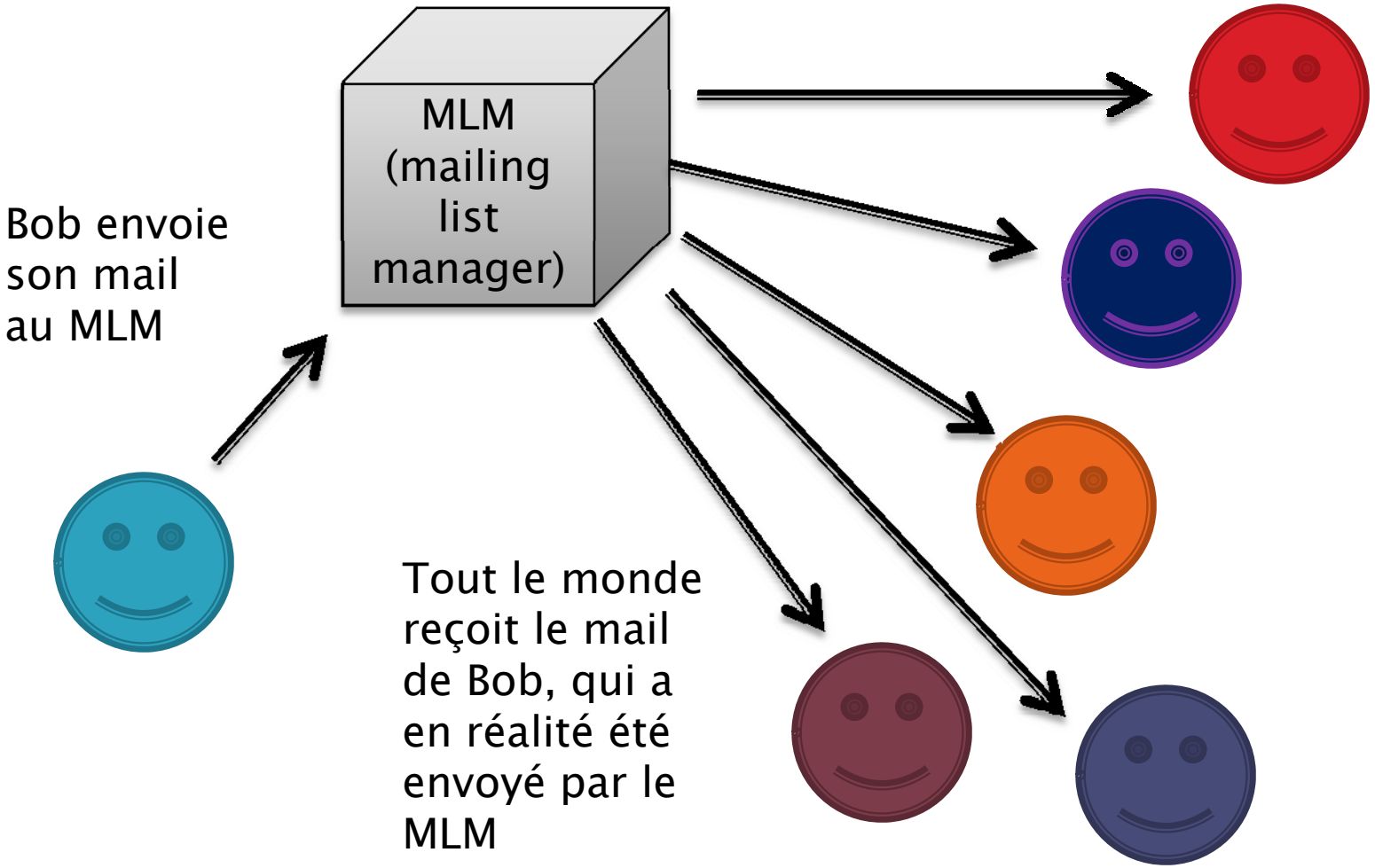
Mailing-lists

- Le mail est le moyen de communication privilégié sur internet
- Malheureusement limité quand il s'agit d'envoyer un même mail à plusieurs dizaines voire centaines de personnes.

- Il est donc intéressant de mettre en place une mailing list.

Utilisé par:

- L'équipe de développement du noyau Linux
- Toutes les équipes Gnome/KDE
- Et une écrasante majorité des projets libres.


MLM
(mailing
list
manager)
Bob envoie
son mail
au MLM
Tout le monde
reçoit le mail
de Bob, qui a
en réalité été
envoyé par le
MLM

Exemples de MLM

- SYMPA (SYsteme de Multi-Postage Automatique)
- Majordomo
- GNU Mailman

- Etc, etc.

Avantages des MLM

- Les messages sont archivés
- Tout passe par le MLM: pas « d'aparté » entre deux personnes.
- Cependant, on se retrouve au fil du temps avec des centaines, voire milliers de messages très grossièrement classés, et ce n'est pas idéal pour s'y retrouver.

Wikis

- Terme englobant les plates formes web permettant un accès collaboratif public sur des documents
- Premier wiki installé en 1995

- Aujourd'hui, les wikis sont surtout utilisés pour de la documentation et à des fins encyclopédiques.

- On peut cependant en faire usage comme d'un « tableau blanc ».
- Mieux structuré qu'une mailing list
- Accès public ou privé
- Se prête très bien aux « brainstorms »
- A réserver cependant aux phases de réflexion: au-delà, ils s'avèrent limités.

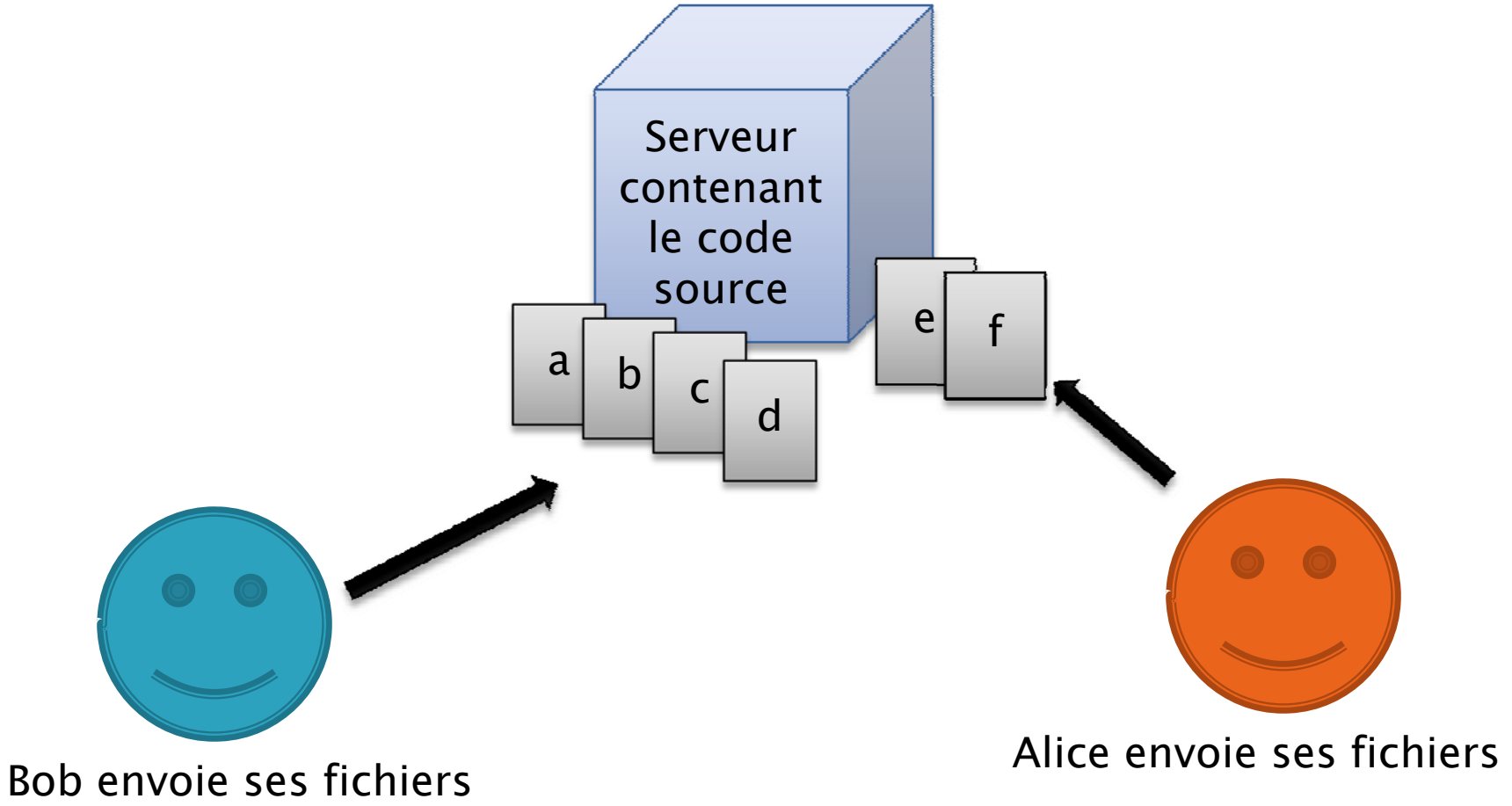
Logiciels de wiki

- Mediawiki
- Dokuwiki
- TikiWiki

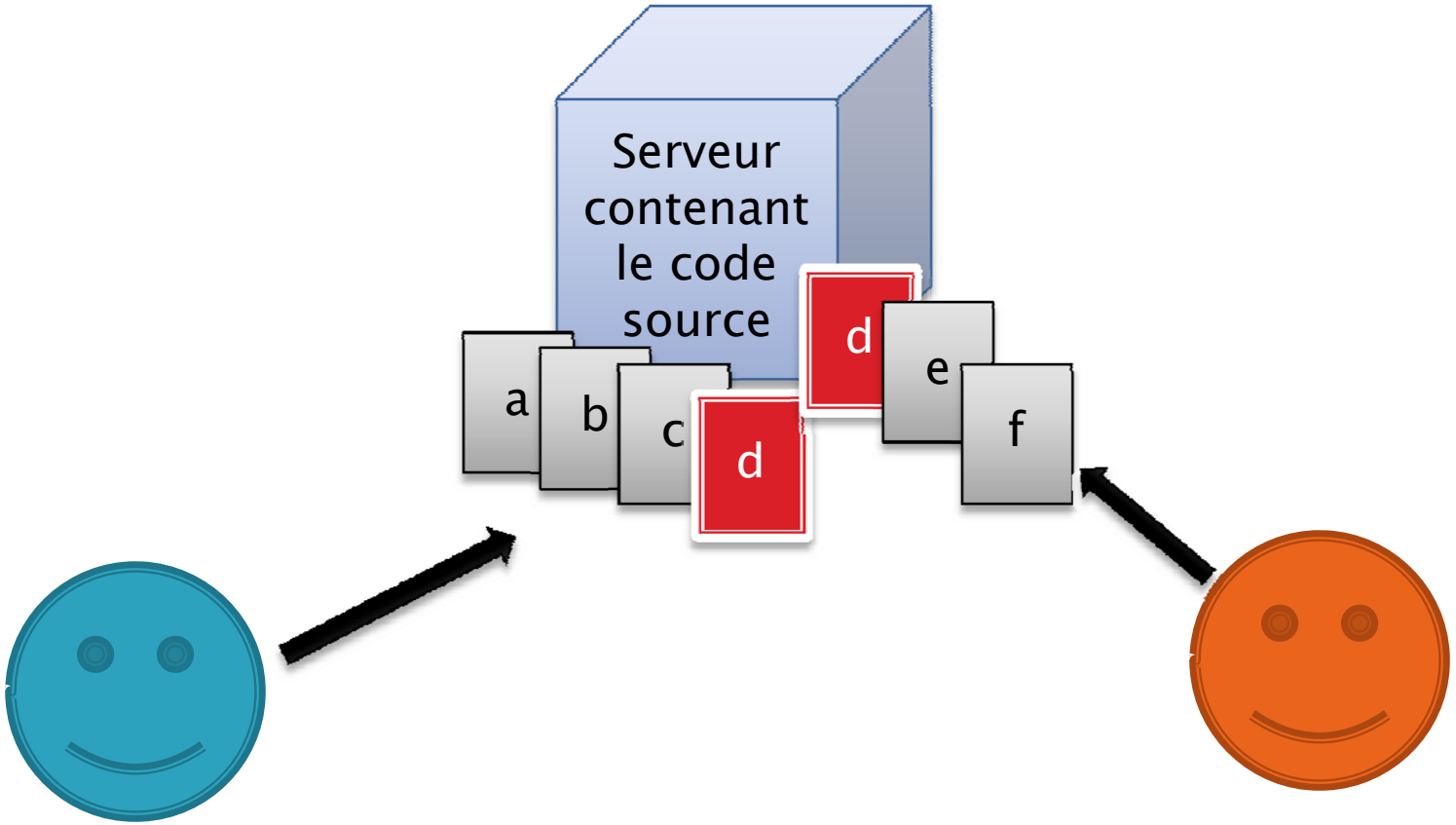
- etc., etc.

- Utilisé pour de la documentation technique.

Comment collaborer sur un même code source?



Mais que se passe t-il si le même fichier est copié?



…ici, seule la version d'Alice est conservée étant donné qu'elle a copié le fichier après: les changements de Bob sont perdus.

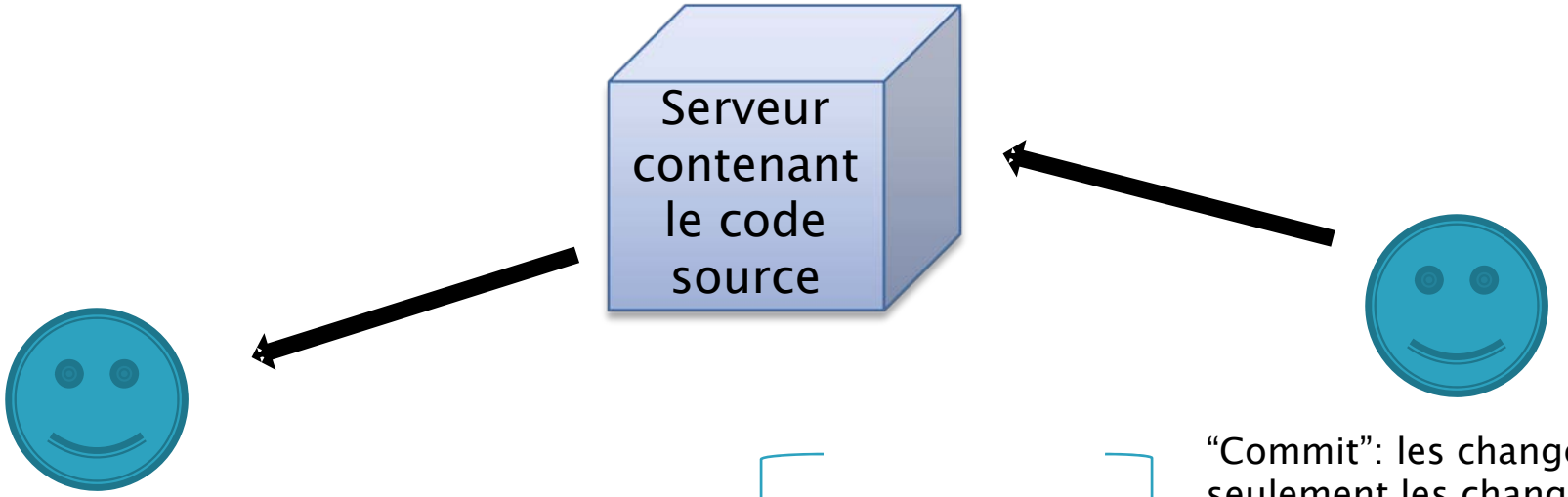
La solution: un système de contrôle de version

- Connu sous le nom de VCS (Version Control System) ou SCM (Source Code Management)
- Permet de gérer les versions concurrentes d'un même programme.

- Le plus connu est CVS (Concurrent Versin System), développé dans les années 80.
- Remplacé par SVN au début des années 2000, moins limité

Utilisé par tous les projets open source de taille considérable.

Comment ça marche?



"Checkout": le développeur obtient les fichiers sources de la version actuelle sur son poste

développement

"Commit": les changements (et seulement les changements) sont envoyés sur le serveur.

- En cas de conflit de fichiers, la stratégie abordée dépend du client: généralement, on demande à l'utilisateur si il veut fusinner les fichiers, le remplacer, etc.

- Le serveur a d'autres avantages: notamment il sauvegarde toutes les "révisions" du code et permet de revenir à une version plus ancienne n'importe quand ("rollback"): utile en cas d'apparition de bug critique.

SVN

- Débuté en 2000, sous licence Apache
- Utilisé pour les projets Apache, KDE, GNOME, FreeBSD, GCC, Python,…
- Aussi très utilisé dans le monde de l'entreprise: leader du marché des SCM

- Utilisable sous Linux/OS X en ligne de commande:

  svn checkout http://svn.serveur.com dossier/
  svn update
  svn commit –m “Message de commit”

- Intégration à l’explorateur Windows avec “Tortoise SVN”

Mettre en place un serveur SVN

- Le serveur SVN est critique: en cas de dysfonctionnement, on perd tout.
- Pour un projet personnel, on a pas forcément les moyens pour mettre en place un serveur dédié.

- Heureusement, de nombreuses solutions performantes et gratuites existent.

Google Code

- Comme tous les produits Google, interface épurée et fonctionnelle
- Limité à 10 Go de données: largement suffisant
- Accès en lecture public, accès en écriture donné par le propriétaire du projet.
- Seule conditions: avoir un compte Google Code, et le projet doit être sous licence libre.

Google Code http://code.google.com

Google Code **vortex-ee**

A multi-user 3D entertainment environment

Search Projects

Project Home | Downloads | Wiki | Issues | Source | Administer

Summary | Updates

Vortex allows you to share and view multimedia content (photos, videos, music,...) with your friends in a realtime 3D environment, as well as to chat with them, send messages, etc.

It is coded in C++, and based upon the Ogre3D library.

Are you on Facebook ? If so, you can join our Vortex group ! http://www.facebook.com/pages/Vortex-Entertainment-Environment/44296249702?ref=s

Star this project

Code License: Mozilla Public License 1.1

Labels: multimedia, ogre, 3D, share, multiuser, environment, music, video, photo, virtual

Featured Wiki Pages: Show all

ProtocolSpecification

Links: Official webpage

Feeds: Project Feeds

Project owners:

guillaume.ardaud

Project members:

twk.theainur, seb.owk, alexandre.zahm, saussage.warrior

©2008 Google - Code Home - Terms of Service - Privacy Policy - Site Directory

Le choix ne manque pas:

- Sourceforge, utilisé par un grand nombre de projets libres
- Tuxfamily
- D'autres solutions payantes, offrant des solutions de backup, etc.

- http://www.svnhostingcomparison.com/

GIT

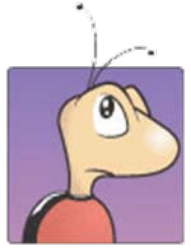
- Développé par Linus Torvalds, utilisé pour le développement du kernel Linux
- Préféré par certains à SVN et aux SCM classiques

- Approche différente, plus adaptée pour des équipes de développeurs hautement compétents et indépendants
- Utilisé pour Wine, VLC, DragonflyBSD,…

Bugs: comment s'en occuper efficacement?

- Comme nous l'avons vu, le debug est une phase critique du modèle TDD.
- Il est donc indispensable d'avoir un système de suivi de bugs performant.
- « Bugtracker »: nombreuses solutions, libres et commerciales, sous diverses formes

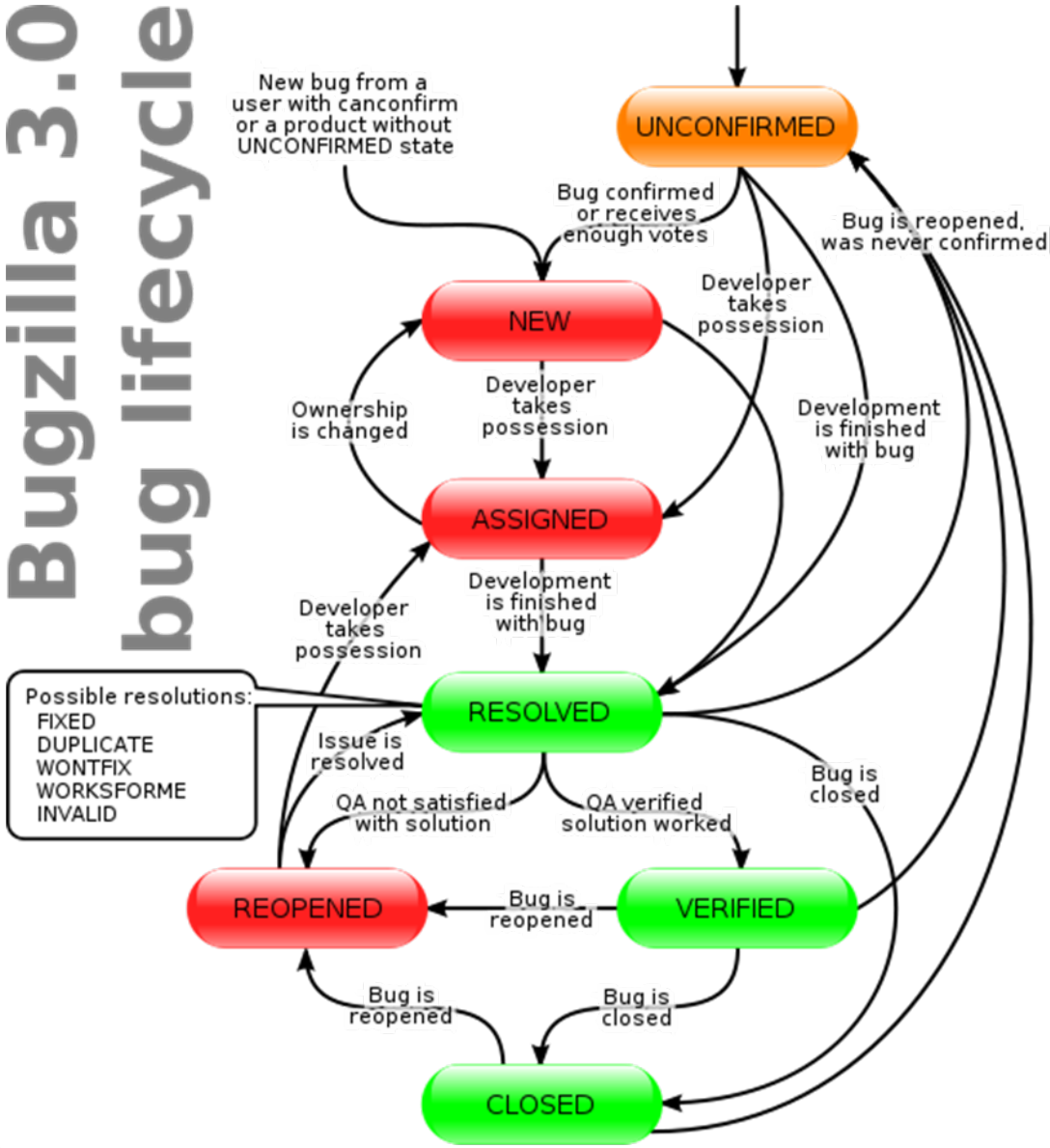
Bugzilla



- Ecrit en 1998, en Tcl, pour le projet Mozilla afin de remplacer la solution de Netscape.
- Version 2 en Perl sortie en 2001.
- Version 3 sortie en 2007.

- Utilisé par Mozilla, le noyau Linux, Gnome, KDE…

- Interface web: nécessite donc un serveur web, un serveur SMTP, un SGBD.



Cycle de vie d'un bug dans Bugzilla 3

- Les bugtrackers sont donc une solution très intéressantes: ils centralisent les bugs recensés, permettent aux développeurs d'avoir une vue claire sur qui s'occupe de quoi, et peuvent même permettre au public de soumettre des bugs.

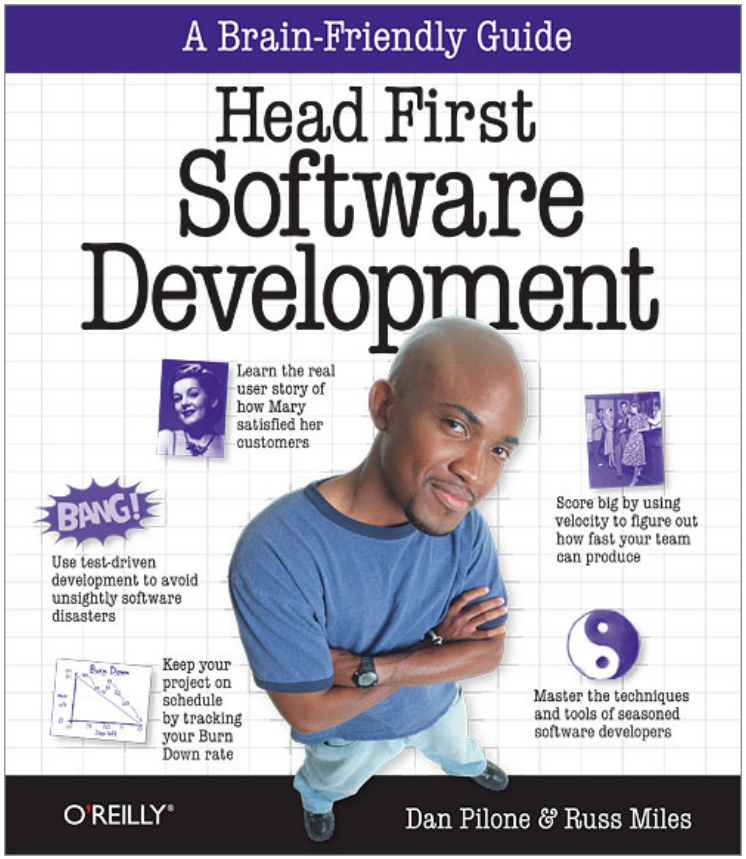
- Certaines solutions SVN (Google Code) offrent leur propre système de tracking de bugs, suffisant largement pour des projets de petite ampleur.

- Ce dont on n'a pas parlé:

- IRC
- Forums/Newsgroup
- Outil de travail collaboratif
- Etc, etc.

- L'essentiel, c'est de trouver l'outil qui correspond à votre besoin…
- (ou de le développer…)

Pour aller plus loin:



"Head First Software Development"

Pilone & Miles

Publié par O'reilly

Merci!

- Questions? (trolls?)