Logiciel Libre
Cours 8 — Development

Stefano Zacchiroli
zack@pps.univ-paris-diderot.fr

Laboratoire PPS, Université Paris Diderot

2013–2014

URL http://upsilon.cc/zack/teaching/1314/freesoftware/
Copyright © 2014 Stefano Zacchiroli
© 2007-2013 Ralf Treinen
© 2004-2006 Roberto Di Cosmo
License Creative Commons Attribution-ShareAlike 4.0 International License
http://creativecommons.org/licenses/by-sa/4.0/deed.en_US


Rappel du cours 6

- Les quatre libertés fondamentales
- Les *Debian Free Software Guidelines*
- Exemples de licences libres : GPL, BSD, . . .
- Exemples de licences non libres
- Problématique de la notion de « travail dérivé »
- Problématique du champ d'application des licences

Contenu cours 8

- Comment fonctionne un projet de développement dans le monde du logiciel libre ?
- *La cathédrale et le bazar*
- Le cycle de vie d'un projet du logiciel libre
- L'infrastructure de développement
- Comment contribuer ?

Outline

Outline

Quelques *mythes* sur le Logiciel Libre

- développement anarchique sans leader
- connaissance qui émerge naturellement de la sagesse des masses

  > *The Wisdom of Crowds : Why the Many Are Smarter Than the Few and How Collective Wisdom Shapes Business, Economies, Societies and Nations*
  > — *James Surowiecki, 2004*

- le logiciel libre est *toujours* de meilleure qualité que le logiciel propriétaire
- l'écosystème du logiciel libre est basé sur des valeurs d'altruisme communautaire

Outline

La cathédrale et le bazar

Eric S. Raymond, 1997 : Publication de l'article *The Cathedral and the Bazaar*. http://www.catb.org/~esr/writings/homesteading/

http://commons.wikimedia.org/wiki/File:Beauvais_Cathedral_SE_exterior.jpg

Le modèle *cathédrale* de développement :

- Développement sous contrôle centrale.
- Importance d'un « maître architecte ».
- Code source disponible au publique seulement pour des versions officiellement publiées (*release*), les versions intermédiaires de développement ne sont pas publiques.
- Contributions de code pas encouragées.

La cathédrale et le bazar

Le modèle *bazar* de développement (le mot « bazar » n'a en anglais pas la même connotation négative qu'en français !) :

http://commons.wikimedia.org/wiki/File:Adjamemarche2.jpg

- Développeurs indépendants.
- Pas de plan rigide.
- Toutes les étapes du développement sont publiquement disponible. Mantra : *release early, release often*
- Les contributions de code venant de l'extérieur sont encouragées.

Exemples du modèle *cathédrale* . . .

. . . dans le mode du logiciel libre :

- *GNAT* : GNU NYU Ada Translator, un compilateur du langage ADA, développé par New York University sous contrat pour le militaire, publié sous licence GPL.
- *GCC* : GNU Compiler Collection.
- *GNU Emacs* : Éditeur de texte très puissant
- *Objective CAML* : Système de programmation avancé, développé par l'INRIA.

Problèmes avec le modèle cathédrale

- Complexité : contrôle central très inefficace pour des grands projets
- La « loi » de Brooks :

  Adding more programers to a late project will it make later

- Cloison entre développeurs (qui connaissent très bien le code) et testeur/utilisateurs qui ne peuvent pas le connaître.

Que sont devenu les projets « cathédrale » cités ?

- GNAT : développement terminé avec succès, le code a été intégré dans le compilateur *gcc* de GNU.
- GCC et GNU Emacs : tout les deux ont subis un *fork* (voir plus tard) et sont aujourd'hui en mode *bazar*.
- OCaml : Support de l'INRIA aujourd'hui réduit, le projet essaie de s'ouvrir et de passer en partie dans un mode *bazar*.

Un avantage du mode *bazar*

En théorie, selon Eric Raymond :

> *Given enough eyeballs, all bugs are shallow (Linus' Law)*

autrement dit :

> *Étant donné un ensemble de bêta testeurs et de co-développeurs suffisamment grand, chaque problème sera rapidement isolé, et sa solution semblera évidente à quelqu'un.*

Il y a aussi des échecs du mode *bazar* . . .

- `openssl` : outil central pour des communications sécurisées (contre écoute, modifications de messages, . . .)
- Distribution *Debian* : le mainteneur du paquet a fait des modifications, pensant qu'elles étaient confirmées par l'auteur.
- Malheureusement :
  - Malentendu entre développeur et auteur
  - Introduction d'un bug catastrophique, mais très subtile (dégradation d'un générateur aléatoire)
- Le bug n'était pas détecté pendant 20 mois, malgré le fait que le code était disponible.
- Le bug était détecté par *Luciano Bello*, chercheur en cryptographie, et sa cause analysée grâce au fait que le code est libre.

Une analyse plus différenciée

Martin Michlmayr et Anthony Senyard, 2004

How to Have a Successful Free Software Project [1]

Trois phases dans le cycle de vie d'un projet du LL :

1. la phase « Cathédrale » : Le projet est initié par un développeur, ou un petit groupe soudé.
2. la phase de transition
3. la phase « Bazar »

1. http://ww.cyrius.com/publications/senyard_michlmayr-successful_project.pdf

La phase « cathédrale » (1)

- *Requirements* : Identifier les besoins
  - soit d'un client (exemple GNAT)
  - soit clarifier les idées quand initiateur = développeur.
    Selon Eric Raymond :

    Tout bon logiciel commence par gratter un développeur là où ça le démange

    En anglais : "scratching your own itch"

La phase « cathédrale » (2)

- *Investigate* : Est-ce qu'il y a déjà un logiciel qui répond aux besoins ?
- Réutilisation d'outils existants : la philosophie UNIX !
- Raisons (bonnes ou mauvaises) pour ne pas utiliser un logiciel existant :
  - ignorance (exemple : Linus Torvalds n'aurait pas commencé Linux s'il avait connu le projet BSD Unix pour PC),
  - licence du projet existent pas adaptée (voir KDE vs. Gnome)
  - désaccord avec les choix techniques : conception, langage de programmation, ...
  - *Not Invented Here* : méfiance de tout qui ne vient pas de la même maison (pas forcement une bonne raison...)
  - incompatibilité personnelles.

La phase « cathédrale » (3)

- *Initiate* : créer une équipe (très souvent : une seule personne !), mise en accord sur l'organisation si nécessaire.
- *Prototype* et *Implémentation* : Création d'une première version du logiciel, procédure similaire au génie logiciel traditionnel (mais avec attention particulière au découpage en modules, l'extensibilité, etc.).
- *Distribution* : Publication du code source.

La phase de transition

- *Distribution* : choisir le bon moment ! Le logiciel doit être suffisaient évolué pour être fonctionnel, mais le projet doit rester ouverts à des nouveaux collaborateurs.
- *Infrastructure* : choix d'une infrastructure adaptée à un développement décentralisé.

La phase « bazar »

- Développeurs avec des compétences et expériences différentes rejoignent le projet.
- *peer review* :
  - étude (critique, amélioration) du code source par des paires.
  - aujourd'hui aussi intégré dans le développement propriétaire (*code auditing*, *extreme programming*).
- Adhérence à des standards et bonnes pratiques (par exemple les *GNU coding standards*).

L'exemple Linux

- quelques dates :
  vers. 0.02 en 10/1991 ; 0.95 en 03/1992 ; 1.0 en 03/1994
- modèle du *dictateur bienveillant* :
  Évaluation des contributions par des « lieutenant »
  Linus Torvalds a le dernier mot sur tout
- élaboration des contributions sur la liste de diffusion (cela permet de discuter les propositions)
- distinction des versions publiées (aujourd'hui abandonnée) : *stable* (2.4.x, 2.6.x), et *expérimental* (1.3.xx or 2.1.x)
- gestion de version avancée (voir plus tard)

L'exemple KDE

- *K Desktop Environment*, première version en 1996
- Initié par un étudiant (Matthias Ettrich)
- Au début problèmes à cause de la licence de la libraire graphique utilise (Qt de l'entreprise Trolltech), aujourd'hui résolus.
- Projet concurrent : GNOME, aujourd'hui coopération des deux projets dans le cadre de `freedesktop.org`.
- Modèle de développement plus ouvert, les décisions sur des questions centrales sont prises par consensus (pas vote) par un groupe central (*core developers*).
- Gouvernance par *méritocratie*.

L'accident : un *fork* !

- Fork (fr : *fourchette*) : création d'un projet indépendant à partir d'un projet existant, sans détruire celui-ci.
- Toutes les licences libres permettent un *fork* (dans le cas contraire la licence n'est pas libre !)
- Gaspillage de ressources, à terme incompatibilités entre les logiciels issus des deux branches.
- Résultats possibles :
  - Une des deux branches est abandonnée.
  - Les deux branches sont abandonnées.
  - Les deux branches persistent.
  - Les deux branches sont fusionnées : meilleure solution, mais difficile à mettre en œuvre si les deux branches ont beaucoup divergé.
- une sorte de "darwinisme" appliqué au logiciel (libre)

Exemples de *fork*

- GNU Emacs : en 1991 divergence de *Lucid Emacs*, aujourd'hui XEmacs. Raison : désaccord avec la politique d'intégration de contributions, notamment concernant l'interfacage avec X.
  ⇒ Les deux branches existent toujours, mais XEmacs est quasiment abandonné
- GNU Compiler Collection : en 1997 divergence d'*Enhanced GNU Compiler System – EGCS*. Raison : comme pour GNU Emacs.
  ⇒ GNU a fini par abandonner sa branche, et d'adopter EGCS.
- *cdrtools* : outils pour graver des CD et DVD, à l'origine sous licence GPL.
  Changement de licence par l'auteur.
  ⇒ *fork* à partir de la dernière version qui était sous GPL, *cdrkit* en Debian.

La fin de vie d'un projet

- Quand les développeurs perdent l'intérêt : abandon du projet.
- Soit il y a des nouveaux développeurs qui prennent le relais, soit le projet tombe dans un état de suspension.
- Le code source reste disponible.
- Parfois continuation d'une maintenance minimale par des distributions.

Outline

Infrastructure : email

```
Date Fri, 16 Mar 2007 12:36:15 +0100
From Jarek Poplawski <>
Subject [PATCH] Re: Fwd: oprofile lockdep warning on rc1

On 28-02-2007 01:46, Dave Jones wrote:
> This happened on a 2.6.21rc1 kernel.
> ...

Here is my patch proposal for testing.
...
Regards,
Jarek P.
```

Les différences entre fichiers

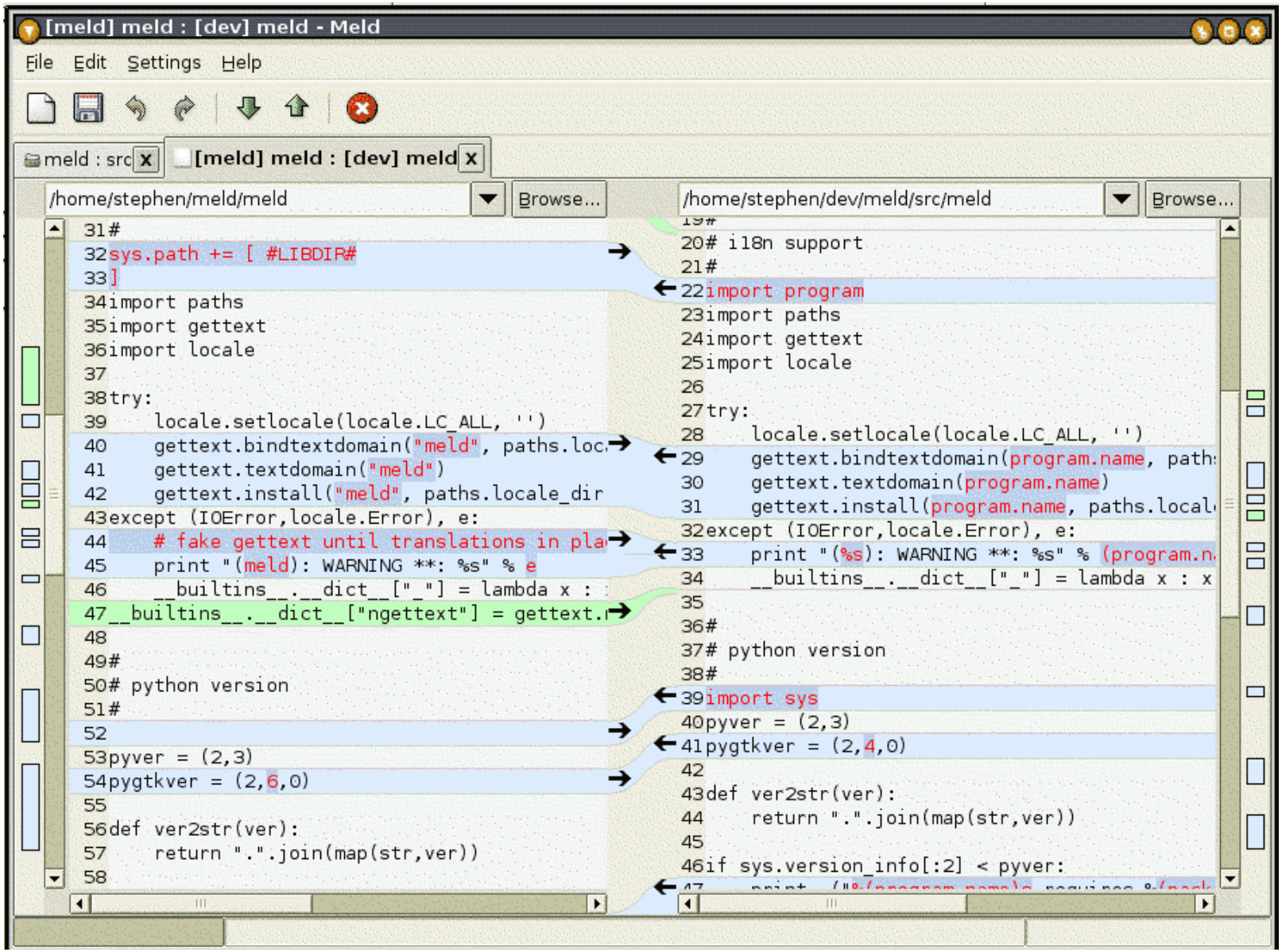
Il existe un outil (`diff`) qui permet d'obtenir la *différence* entre deux fichiers :

Fichier `old`	Fichier `new`
`Longtemps, je me suis couché de bon heure.`	`Longtemps, je me suis couché de bonne heure.`

Afficher la différence entre les deux fichiers :

```
 Longtemps, je me
 suis couché de
-bon heure.
+bonne heure.
```

Interface graphique pour voir des différences



Les *patchs*

- Cet outil permet d'obtenir facilement la différence entre deux copies d'un répertoire, même si très grands (par exemple du code source du noyau Linux).
- Il y a un deuxième outil (`patch`) qui permet d'*appliquer* la différence à un des fichiers pour transformer un fichier dans l'autre.
- `diff`+`patch` sont des outils idéal pour communiquer les modifications des fichiers.
- On parle d'un *patch* (rarement en français : *rustine*, ou *correctif*).

Exemple de patch dans la mailing list

```
lockdep found oprofilefs_lock is taken both in process
context (oprofilefs_ulong_from_user()) and from hardirq
(nmi_cpu_setup()), so the lockup is possible.

Reported-by: Richard Hughes <hughsient@gmail.com>
Signed-off-by: Jarek Poplawski <jarkao2@o2.pl>

diff -Nurp linux-2.6.21-rc1-/drivers/oprofile/oprofilefs.c linux-2.6.21-rc1/drivers/oprofile/opr
--- linux-2.6.21-rc1-/drivers/oprofile/oprofilefs.c 2007-02-27 10:47:38.000000000 +0100
+++ linux-2.6.21-rc1/drivers/oprofile/oprofilefs.c 2007-03-16 11:36:30.000000000 +0100
@@ -65,6 +65,7 @@ ssize_t oprofilefs_ulong_to_user(unsigned
 int oprofilefs_ulong_from_user(unsigned long * val, char const __user * buf, size_t count)

   char tmpbuf[TMPBUFSIZE];
+ unsigned long flags;

   if (!count)
   return 0;
@@ -77,9 +78,9 @@ int oprofilefs_ulong_from_user(unsigned
   if (copy_from_user(tmpbuf, buf, count))
   return -EFAULT;

- spin_lock(&oprofilefs_lock);
+ spin_lock_irqsave(&oprofilefs_lock, flags);
   *val = simple_strtoul(tmpbuf, NULL, 0);
- spin_unlock(&oprofilefs_lock);
+ spin_unlock_irqrestore(&oprofilefs_lock, flags);
   return 0;
```

La gestion de version

- Les *différences* sont aussi idéales pour archiver les versions successives d'un projet.
- Systèmes de gestion de versions (*Version Control Systems* — VCS)) : permettent
  - d'archiver les versions différentes d'un projet,
  - récupérer des versions anciennes,
  - obtenir les différences entre des versions quelconques archivées (calcul de l'effet combiné de plusieurs différences).
- Exemples : CVS (un peu daté), Subversion (encore populaire mais à la baisse), Git (le vrais gagnant de la *guerre de VCS*, par Linus Torvalds), arch, mercurial, ...

Bug Tracking System

- Suivi des rapport d'erreurs, de la soumission à la résolution.
- Soumission de rapports d'erreur par email, interface web, applications utilisateurs, . . .
- Tout le monde peut suivre l'état d'avancement.
- Classification, par exemple par sévérité.

Exemple : Bugzilla

GNOME Bugzilla – Bug List

Home | New | Browse | Search | Find | Reports | My Requests | Preferences | Help | Log out zack@upsilon.cc

Wed Feb 26 2014 08:18:06 UTC

Status: REOPENED, NEW, ASSIGNED, UNCONFIRMED, NEEDINFO **Product:** gnome-calculator

83 bugs found.

ID	Sev	Pri	OS	Product	Status	Resolution	Summary
631688	nor	Nor	Linu	gnome-calcul	UNCO		horizontal scrolling in gcalctool
634664	nor	Nor	All	gnome-calcul	UNCO		Entry field does not scroll/update when entering long text (cropped input display)
643589	enh	Nor	Linu	gnome-calcul	UNCO		Add distinction between units from the imperial system and the United States customary system in the units display names.
644052	maj	Nor	Linu	gnome-calcul	UNCO		inverse trigonometric functions not usable when setting non-C locales
650949	nor	Nor	Linu	gnome-calcul	UNCO		Wrong calculus for e^x
652362	nor	Nor	Linu	gnome-calcul	UNCO		gcalctool needs to interpret comma (",") as a decimal dot (".")
654202	nor	Nor	Linu	gnome-calcul	UNCO		Translated units conversion does not work with non-ASCII translations
655104	nor	Nor	Linu	gnome-calcul	UNCO		unsure use of cable
661463	enh	Nor	Linu	gnome-calcul	UNCO		[Programming mode] No conversion of fractions
664366	nor	Nor	Linu	gnome-calcul	UNCO		Rename "Pound Sterling" to "British Pound"
666655	nor	Nor	Linu	gnome-calcul	UNCO		Can't insert dot from keypad when numlock is off
673728	tri	Nor	Linu	gnome-calcul	UNCO		Subscripted base digits of entered number are displayed unevenly
675940	maj	Nor	Linu	gnome-calcul	UNCO		Display sometimes fails to repaint upon calculation
681775	maj	Nor	Linu	gnome-calcul	UNCO		error in division using scientific notation, negative exponent
682971	nor	Nor	Linu	gnome-calcul	UNCO		Invalid text buffer iterator pressing dot keyboard key
683731	nor	Nor	Linu	gnome-calcul	UNCO		buttons in gcalctool packed together
698400	maj	Nor	Linu	gnome-calcul	UNCO		Improve gnome-calculator back-end
699155	enh	Nor	Linu	gnome-calcul	UNCO		Provide unit conversion to feet and inches
699204	nor	Nor	Linu	gnome-calcul	UNCO		calculator does not respect formats settings
700617	nor	Nor	Linu	gnome-calcul	UNCO		doesn't handle expressions with parentheses well

Exemple : Bugzilla (cont.)

GNOME Bugzilla – Bug 634664 Entry field does not scroll/update when entering long text (cropped input display) Last modified: 2012-06-11 06:59:52 UTC

Home | New | Browse | Search | Find | Reports | My Requests | Preferences | Help | Log out zack@upsilon.cc

Bug List: (2 of 83) First Last Prev Next Show last search results

Bug 634664 - Entry field does not scroll/update when entering long text (cropped input display) Commit

Collapse All Comments - Expand All Comments

Product: gnome-calculator
Component: general
Version: 6.2.x
Status: UNCONFIRMED
Priority: Normal
Severity: normal

Robert Roth [reporter] [developer] 2010-11-12 08:24:52 UTC Description [reply] [-]

Created an attachment (id=174311) [details]
Cropped entry field.

I need to be able to resize the window as adding or subtracting many entries
the entries get lost from sight so it is impossible to check before totalling
after version 5.30.0. They just get lost from view. See the screenshot.

Forwarded from downstream Ubuntu, see the report at
https://bugs.launchpad.net/ubuntu/+source/gcalctool/+bug/671674

Bob 2010-11-12 15:44:50 UTC Comment 1 [reply] [-]

I have this same problem and believe it to be a bug. It renders the application
unusable.

Clean install if Ubuntu 10.10 and gcalctool 5.32.0.

Thanks, Bob brazie.

Kamil Páral 2011-11-15 15:27:27 UTC Comment 2 [reply] [-]

Confirmed in gcalctool-6.2.0-1.fc16.x86_64, Fedora 16.

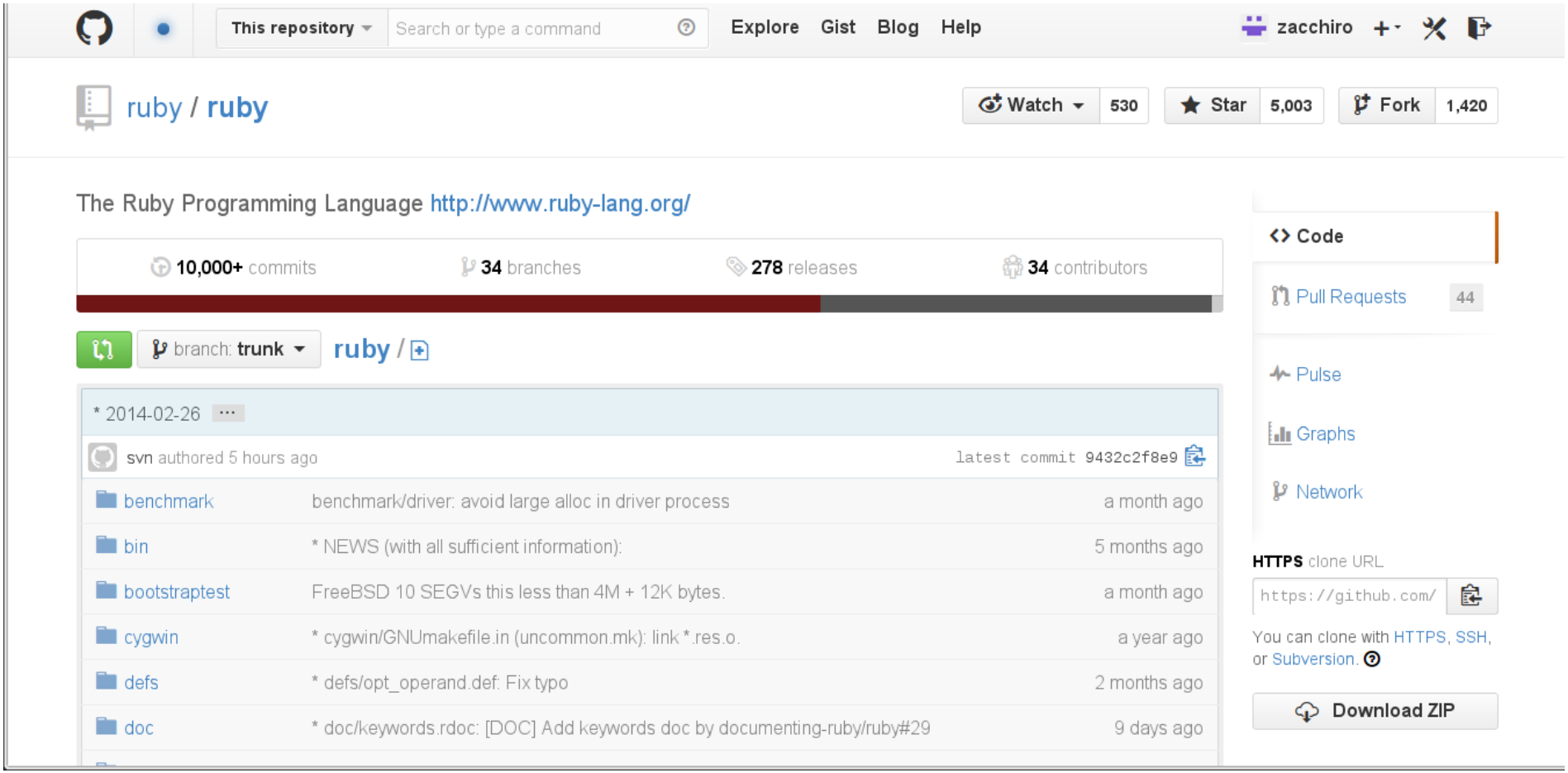
CoudCoud 2012-05-06 21:18:55 UTC Comment 3 [reply] [-]

Les forges

- Solution intégrée : gestion d'utilisateurs, création de nouveaux projets, système de contrôle de version, *bug tracking system*, gestion de listes de diffusion, ...
- Pionnier *sourceforge*, il existent aujourd'hui des solutions (par exemple FusionForge) qui peuvent être installées facilement sur un serveur.

Exemple : FusionForge

Exemple : GitHub



Contributions

Il y a plein de possibilités de contribuer, aussi pour des non programmeurs :

- Écrire des bons rapports de bugs qui permettent à un développeur de reproduire le problème.
- Rédiger des documentations.
- Créer des graphismes.
- Traduire des documentations, ou les dialogues des logiciels.
- Participer à de forums et répondre aux questions.