Création et gestion de projets «Logiciel Libre»

Thomas Petazzoni
thomas.petazzoni@enix.org

24/06/08

Sommaire

- Qu'est-ce qu'un Logiciel Libre ?
- Démarrer un projet
- Organisation du projet
  - Communication et infrastructure technique
  - Gestion des contributeurs
  - Développement au quotidien
  - Activités hors développement
  - Financement
- Aspects légaux

Intervenant

- Thomas Petazzoni
- Ingénieur en informatique diplômé de l'UTBM
- Ingénieur formateur et consultant en Linux et Logiciels Libres pour l'embarqué, notamment développement noyau
- Utilisateur et promoteur de Logiciels Libres depuis 2000
  - Participation à plusieurs LUGs et création de Toulibre à Toulouse
  - Membre du CA de l'April
  - Animateur de l'Agenda du Libre

« Producing Open Source Software »

- Cours très largement basé sur le livre « Producing Open Source Software »
- Écrit par Karl Fogel, un des principaux développeurs de Subversion
- Distribué sous licence Creative Commons BY-SA
- Disponible librement sur `http://producingoss.com`
- Traduction française en cours sur Framalang.org

Qu'est-ce qu'un Logiciel Libre ?

- Quatre libertés fondamentales
  - Liberté d'exécuter le code, pour n'importe quel usage ;
  - Liberté de copier le logiciel et d'en redistribuer les copies ;
  - Liberté d'étudier le logiciel, ce qui suppose la disponibilité du code source ;
  - Liberté de modifier et de redistribuer les modifications.

Intérêts du Logiciel Libre

- Valeurs éthiques et sociales
- Économies
- Contrôle de l'infrastructure informatique
- Adaptabilité
- Transparence, sécurité
- Standards ouverts et pérennité

Libre et communautaire

- Un projet au modèle de développement tout à fait fermé peut être publié sous licence libre
  - Pas de communauté d'utilisateurs ni de développeurs
- La plupart des projets libres fonctionnent en mode ouvert et transparent
  - Tout le monde peut participer
  - Création d'une communauté d'utilisateurs et de développeurs
  - Va bien au-delà de la simple apposition d'une licence libre
- Le cours se focalise sur ce second cas

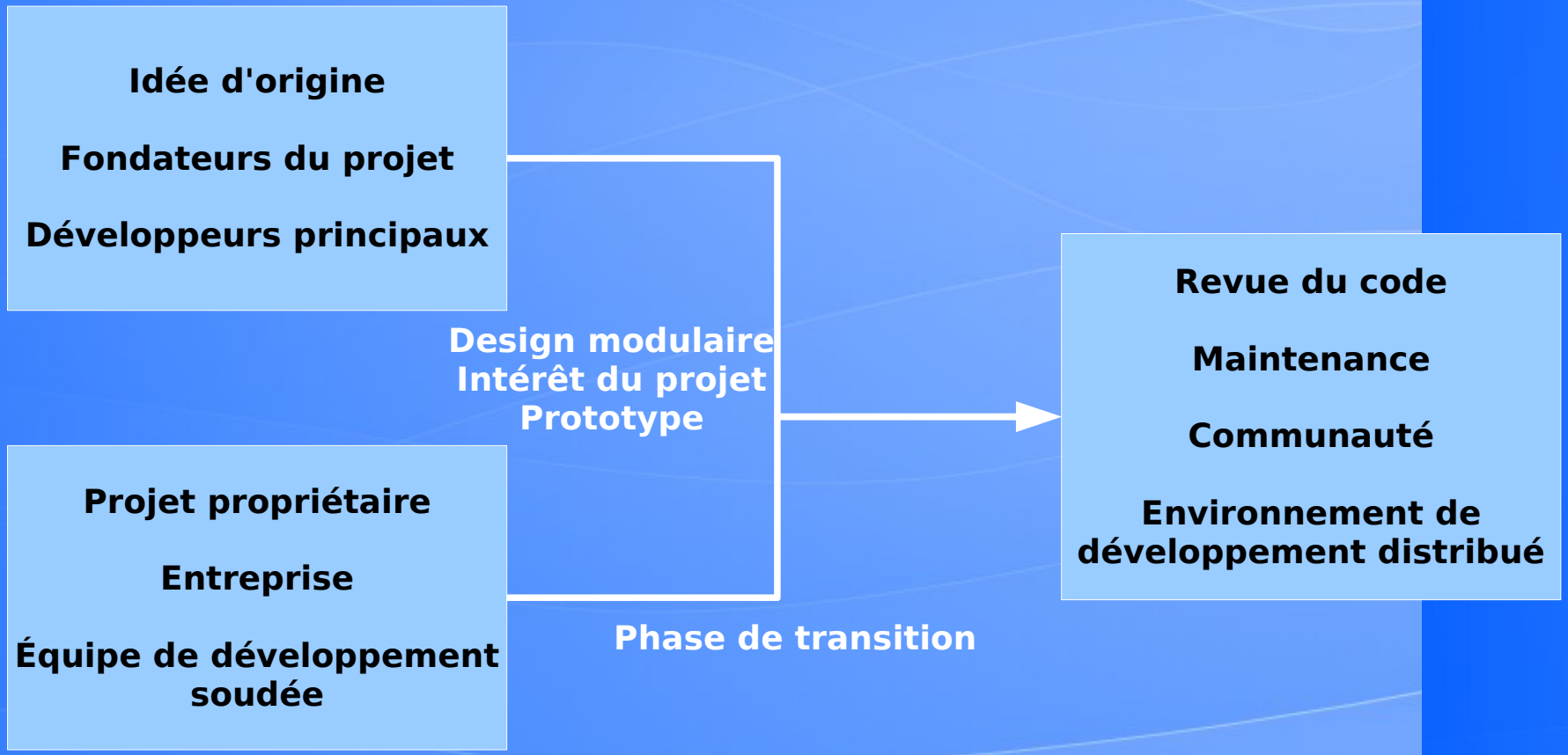
Pourquoi lancer un projet ?

- « Every good work of software starts by scratching a developer's personal itch »
- Eric Raymond

Démarrer un projet libre

- Le plus souvent, un projet est lancé par une petite équipe
  - Délimite le périmètre du logiciel, au moins pour les débuts
  - Réalise les premiers choix techniques (langages, outils, etc.)
  - Développe un premier prototype démontrant les fonctionnalités de base
- Puis ouverture réelle aux contributeurs
- Permet de positionner dès le départ le projet, sans commencer par de longues discussions

Démarrer un projet libre (2)



« How to have a successful Free Software Project », Anthony Senyard and Martin Michlmayr, Departement of Computer Science and Engineering, The University of Melbourne

Démarrer un projet libre (3) : Linux

From: Linus Benedict Torvalds (torvalds@klaava.Helsinki.FI)
Subject: Free minix-like kernel sources for 386-AT
Newsgroups: comp.os.minix
Date: 1991-10-05 08:53:28 PST

[...]

As I mentioned a month ago, I'm working on a free version of a minix-lookalike for AT-386 computers. It has finally reached the stage where it's even usable (though may not be depending on what you want), and I am willing to put out the sources for wider distribution. It is just version 0.02 (+1 (very small) patch already), but I've successfully run bash/gcc/gnu-make/gnu-sed/compress etc under it.

Sources for this pet project of mine can be found at nic.funet.fi (128.214.6.100) in the directory /pub/OS/Linux. The directory also contains some README-file and a couple of binaries to work under linux (bash, update and gcc, what more can you ask for . Full kernel source is provided, as no minix code has been used. Library sources are only partially free, so that cannot be distributed currently. The system is able to compile "as-is" and has been known to work. Heh. Sources to the binaries (bash and gcc) can be found at the same place in /pub/gnu.

[...]

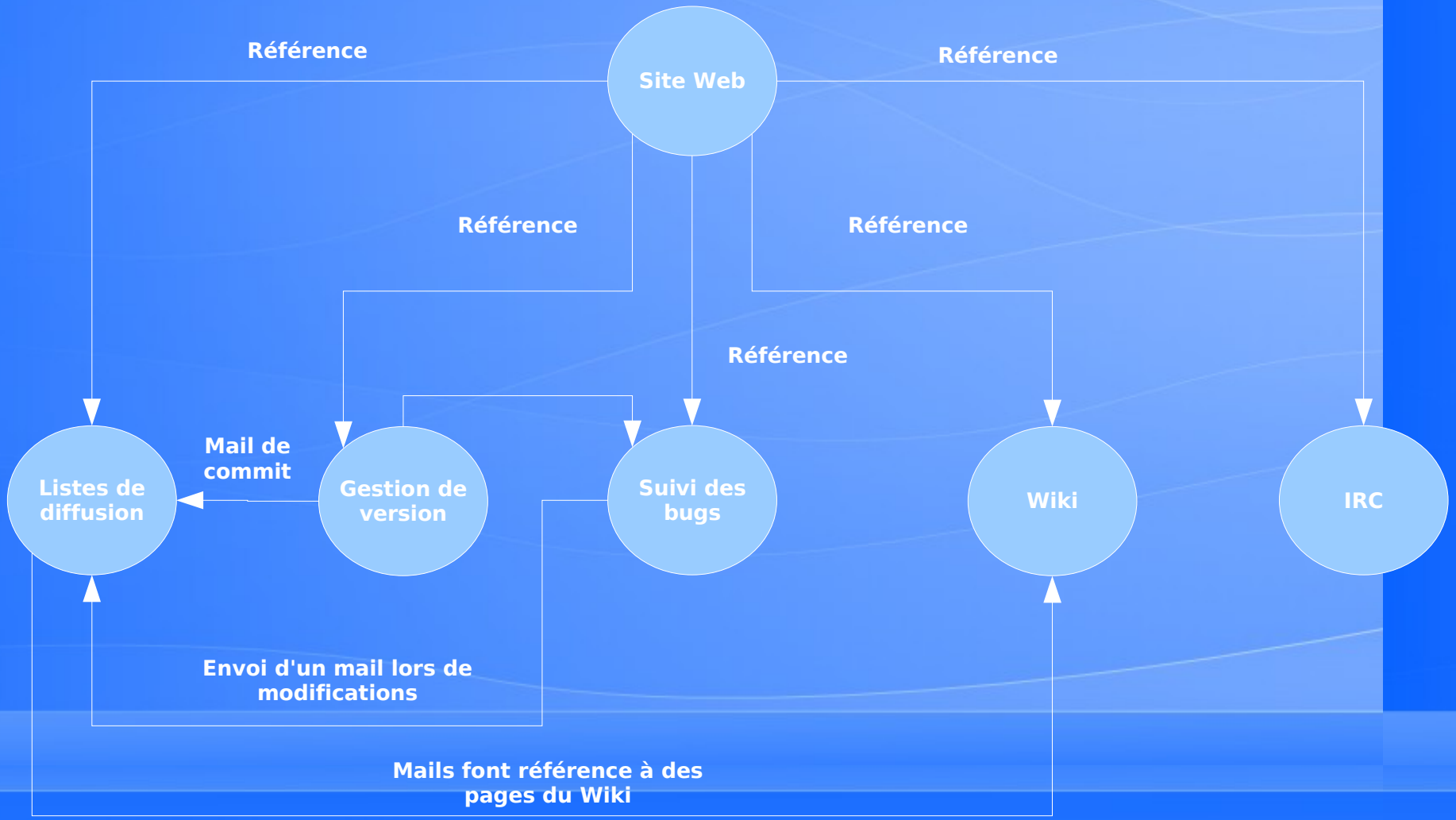
Communication

- Modèle de développement ouvert
- Participation de contributeurs variés
  - Répartis géographiquement
  - De compétences diverses
  - De disponibilités différentes
- Besoins de communication importants

Infrastructure technique

- Pour fonctionner, besoin d'une infrastructure technique avec de nombreux outils collaboratifs
- La plupart sont communs à beaucoup de projets Logiciels Libres et utiliser des outils classiques permet l'intégration facile et rapide de nouveaux contributeurs

Infrastructure technique



Liste de diffusion

- Le principal médium de communication dans tout projet
- Permet des discussions
  - Structurées
  - Asynchrones, qui laissent le temps de la réflexion
  - Archivées
- Des archives publiques doivent être disponibles
- Les forums Web ne sont pas utilisés, en tout cas pour la partie développement

Liste de diffusion (2)

- Tout projet doit démarrer avec deux listes de discussion
  - Discussions générales
  - Logs de commits dans le système de gestion de version
- Ensuite, possibilité de séparer
  - Discussions utilisateurs et discussions développeurs
  - Discussions sur des sous-projets
- Trop de listes tue l'activité et le dynamisme du projet en le fragmentant

Liste de diffusion (3)

- Outils principaux
  - Mailman, the GNU mailing list manager
    http://www.list.org/
  - Sympa mailing list server
    http://www.sympa.org/
  - Autres: SmartList, Ecartis, ListProc, Ezmlm, Dada

Gestion de version (1)

- Deuxième outil central et indispensable après les listes de diffusion
- Utile pour le code, mais aussi la documentation, les traductions, le site Web, etc.
- Permet de
  - Mettre à disposition la dernière version
  - Versionner, et donc de revenir en arrière
  - Contrôler et valider les modifications
  - Lancer des développements parallèles

Gestion de version (2)

- Mettre à disposition la dernière version
  - Doit être accessible en lecture seule pour tous, réservé à quelques-uns pour la modification
  - Doit également être accessible via une interface Web
- Contrôler et valider les modifications
  - Tout « commit » fait l'objet de l'envoi d'un mail sur la liste de diffusion appropriée, accompagné du *diff*
  - Tout le monde sait ce qui se passe, peut relire et commenter la modification

Obtenir une copie du dépôt SVN

Not familiar with Subversion?

If you've never used Subversion (SVN for short), you should read some documentation. A good starting point is the official Subversion website.

Note that there are many graphical clients to access a Subversion repository out there. This page will give instructions on accessing your project on the Gna! servers with the official command line client. Your mileage may vary.

Anonymous SVN Access

This project's Subversion repository can be checked out through anonymous access over the SVN protocol (TCP 3690), or over http. Depending on your network configuration you may prefer to use one or the other (for instance, if your internet access goes through a web proxy, it's unlikely that access over http will work).

Checkout over SVN protocol (TCP 3690):

```
svn co svn://svn.gna.org/svn/gdtc/trunk gdtc
```

Checkout over http:

```
svn co http://svn.gna.org/svn/gdtc/trunk gdtc
```

Note that these two commands assume that the repository is using the recommended layout *tags/ branches/ trunk/* at the toplevel. The example will download the tree under *trunk/* in the repository and place it in a directory named after the project.

Project Member SVN Access via SSH

Only project members can access the SVN tree via this method. SSH must be installed on your client machine.

You have to register a SSH key! Check the Cookbook for more details.

Sourcecode repository :

```
svn co svn+ssh://tpetazzo@svn.gna.org/svn/gdtc/trunk gdtc
```

Webpages repository (if relevant) :

```
svn co svn+ssh://tpetazzo@svn.gna.org/svn/gdtc/website gdtc
```

[gdtc]

Project Root: gdtc Go

gdtc

Current directory: **[gdtc]**
Current revision: 924
Jump to directory revision: 924 Go

Files shown: **10**

File	Rev.	Age	Author	Last Log Entry
admin/	924	12 days	benj	Improve benevalo so that we can create more from admin page
cgi-bin/	908	8 weeks	mad	suppression du CC qui pollue la liste ca@
config/	819	6 months	benj	Ajout exemple config
css/	919	5 weeks	benj	Improve styles
hackergotchis/	621	9 months	benj	Handle hackergotchis as well
html/	650	9 months	benj	Fix data
images/	912	7 weeks	benj	Update image
include/	924	12 days	benj	Improve benevalo so that we can create more from admin page
javascript/	838	6 months	benj	Add utility functions, that is sooooo web 2.0
my/	924	12 days	benj	Improve benevalo so that we can create more from admin page
photos/	402	9 months	benj	Set SVN properties
scripts/	915	5 weeks	benj	Don't spam the CA
sql/	910	7 weeks	benj	Ajout benevalo
templates/	924	12 days	benj	Improve benevalo so that we can create more from admin page
website/	376	9 months	root	"website at home.gna.org"
AUTHORS	563	9 months	benj	Add icon
COPYING	294	10 months	benj	Initial import
Makefile	97	14 months	benj	initial import
README	134	13 months	tpetazzoni	Test

Gestion de version (4)

```
Revision: 4726
          http://svn.sv.gnu.org/viewvc/?
view=rev&root=qemu&revision=4726
Author:   ths
Date:     2008-06-10 15:29:15 +0000 (Tue, 10 Jun 2008)

Log Message:
-----------
Fix typo, by Laurent Desnogues.

Modified Paths:
--------------
    trunk/linux-user/syscall.c

Modified: trunk/linux-user/syscall.c
===================================================================
--- trunk/linux-user/syscall.c     2008-06-10 01:47:17 UTC (rev 4725)
+++ trunk/linux-user/syscall.c     2008-06-10 15:29:15 UTC (rev 4726)
@@ -3070,7 +3070,7 @@
 #if TARGET_ABI_BITS == 32
 static inline uint64_t target_offset64(uint32_t word0, uint32_t
word1)
 {
-#ifdef TARGET_WORDS_BIG_ENDIAN
+#ifdef TARGET_WORDS_BIGENDIAN
     return ((uint64_t)word0 << 32) | word1;
 #else
     return ((uint64_t)word1 << 32) | word0;
```

Gestion de version (3)

- Revenir en arrière
  - En cas de désaccord, on peut retirer une modification
  - On peut donc être souple sur l'autorisation d'accès en écriture

Gestion de version (5)

- Le mécanisme de branches permet de travailler en parallèle
  - Stabilisation, maintenance d'une ancienne version
  - Expérimentation de nouvelles fonctionnalités
- Deux approches
  - Centralisée, un seul dépôt, toutes les branches doivent y être créées
  - Distribuée, pas de dépôt central, chacun possède le sien et créé les branches qu'il souhaite en local

Gestion de version (6)

- Outils centralisés
  - Le classique mais dépassé CVS
    http://www.cvshome.org
  - Le moderne Subversion
    http://subversion.tigris.org
- Outils décentralisés
  - Git
    http://git.or.cz/
  - Mercurial
    http://www.selenic.com/mercurial/
  - Monotone, Arch, Bazaar, Darcs, etc.

Suivi des bugs

- Outil permettant l'enregistrement des bugs et le suivi de leur correction
  - Interagir avec le rapporteur du bug
  - Assignation à un développeur et à une version
  - Association au correctif
  - Conservation d'une trace
- Également utilisé pour le suivi du développement de fonctionnalités

Suivi des bugs

- Doit être connecté aux autres outils
  - Envoi d'un message sur la liste de diffusion des «commits» à chaque modification d'un bug
  - Lors d'un commit, si un numéro de bug est associé dans le message, alors ce commit doit être automatiquement inscrit dans l'historique du bug
- Objectifs
  - Rendre visible les modifications apportées dans l'outil de suivi
  - Garder une trace des modifications apportées au code correspondant à une anomalie

gDTC - Anomalies : Consulter les items

Groupe | Accueil | Site web | Téléchargement | Documentation | Assistance | Listes de discussion | Code source | Anomalies | Tâches | Dépêches

(+) Critères d'affichage

Début | Résultats précédents | 112 items correspondants - Items de 1 à 50 | Résultats suivants | Fin

Item ID	Summary	Status	Assigned To	Submitted On
#11812	Envoyer mail de relance aux associations	None	None	mar 10.06.2008, 16:02
#11756	Statut non mis à jour lors de la création manuelle d'une nouvelle adhésion	None	None	mer 28.05.2008, 21:38
#11724	stats "Indice de satisfaction"	None	None	ven 23.05.2008, 13:14
#11722	Création de deux cotisations lors de l'enregistrement du paiement cbweb	None	None	ven 23.05.2008, 13:10
#11632	Créer une vue synthétique des réponses « Participation des membres »	None	None	lun 05.05.2008, 12:45
#11548	pb affichage dans l'historique des cotisations	None	None	mar 22.04.2008, 13:52
#11543	Transmission d'identifiant de connexion en clair	None	None	lun 21.04.2008, 20:09
#11539	Action Supprimer une photo/logo postée dans l'Annuaire	None	None	lun 21.04.2008, 12:34
#11457	Support du microformat hCard dans l'annuaire	Fixed	benj	mar 08.04.2008, 16:29

Suivi des bugs

- Outils
  - Bugzilla
    http://www.bugzilla.org/
  - Trac, associé à Subversion
    http://trac.edgewall.org/
  - Mantis
    http://www.mantisbt.org/
  - etc.

IRC

- Outil optionnel
- Permet la discussion instantanée entre utilisateurs et développeurs ou entre développeurs
- Peut être pratique pour faire du déboguage en temps réel
- Inconvénients
  - Très chronophage
  - Risque de conversations privées qui excluent des membres du projet

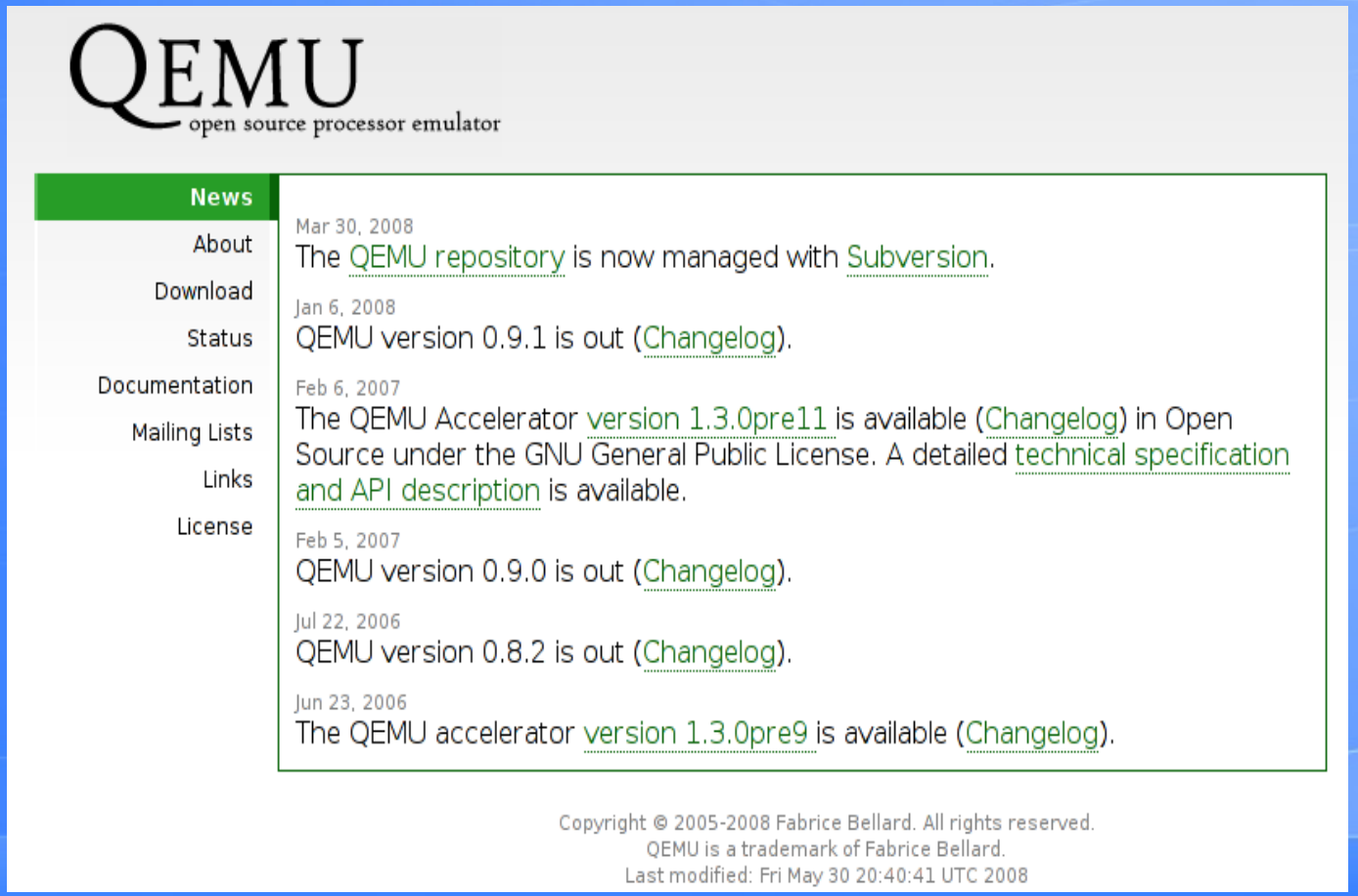
Wiki

- Site Web que n'importe quel visiteur peut éditer
- Principe utilisé par l'encyclopédie Wikipédia
- Permet de construire du contenu de manière collaborative
- Dans le cadre d'un projet libre, travail sur les spécifications et l'implémentation
  - Brainstorming
  - Tableau blanc
- Pas utilisé à l'origine pour les projets libres, nouvelle tendance

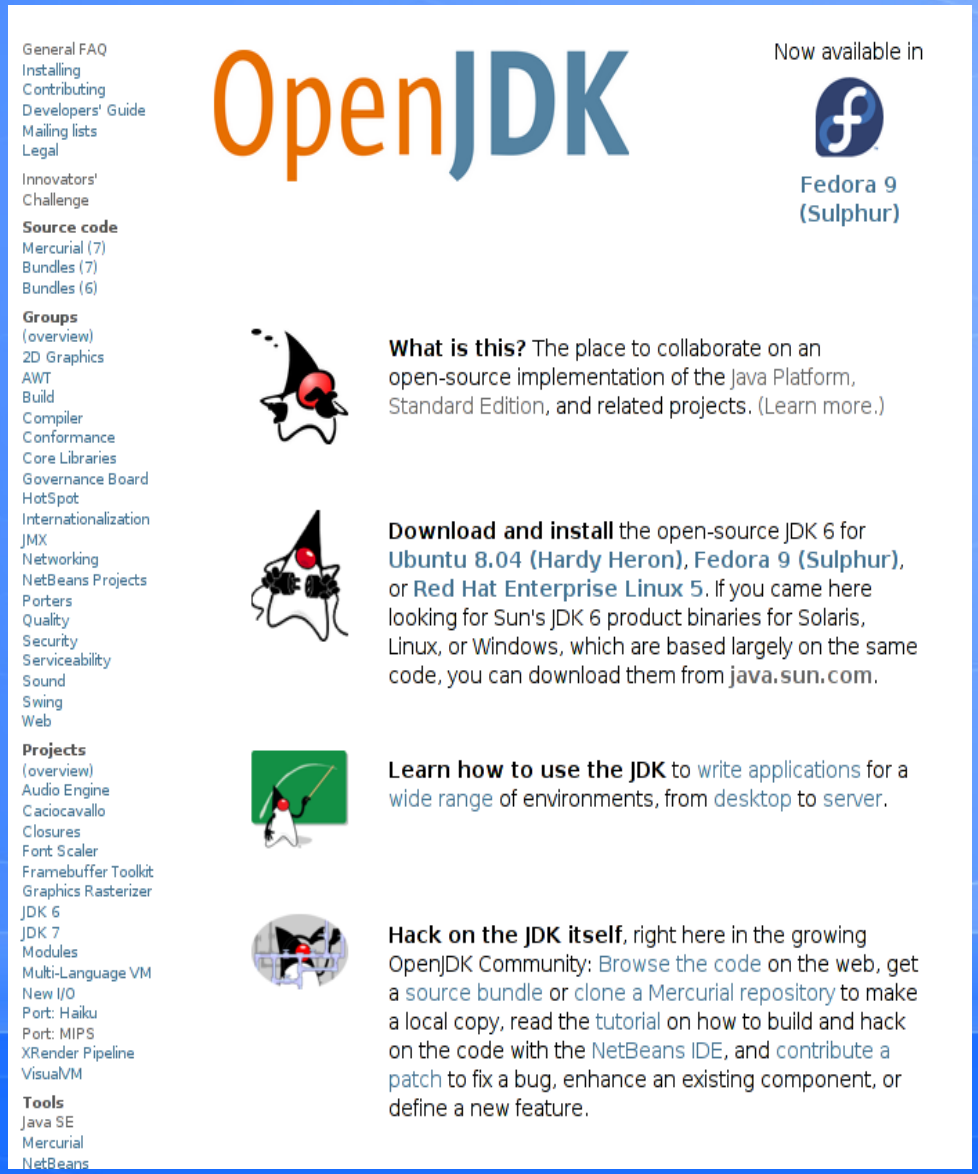
Site Web

- De préférence simple, sans jargon marketing, mais avec des informations techniques
- Informations indispensables accessibles de manière simple et rapide
  - Nom, licence et courte description
  - Fonctionnalités actuelles et prévues
  - Téléchargement du code source et du binaire
  - Accès aux outils de développement: liste de diffusion, gestion de version, suivi des bugs, IRC
  - Documentation utilisateur et développeur

Site Web


QEMU
open source processor emulator
News
About
Download
Status
Documentation
Mailing Lists
Links
License
Mar 30, 2008
The QEMU repository is now managed with Subversion.
Jan 6, 2008
QEMU version 0.9.1 is out (Changelog).
Feb 6, 2007
The QEMU Accelerator version 1.3.0pre11 is available (Changelog) in Open Source under the GNU General Public License. A detailed technical specification and API description is available.
Feb 5, 2007
QEMU version 0.9.0 is out (Changelog).
Jul 22, 2006
QEMU version 0.8.2 is out (Changelog).
Jun 23, 2006
The QEMU accelerator version 1.3.0pre9 is available (Changelog).
Copyright © 2005-2008 Fabrice Bellard. All rights reserved.
QEMU is a trademark of Fabrice Bellard.
Last modified: Fri May 30 20:40:41 UTC 2008

Site Web


General FAQ
Installing
Contributing
Developers' Guide
Mailing lists
Legal
Innovators' Challenge
Source code
Mercurial (7)
Bundles (7)
Bundles (6)
Groups
(overview)
2D Graphics
AWT
Build
Compiler
Conformance
Core Libraries
Governance Board
HotSpot
Internationalization
JMX
Networking
NetBeans Projects
Porters
Quality
Security
Serviceability
Sound
Swing
Web
Projects
(overview)
Audio Engine
Caciocavallo
Closures
Font Scaler
Framebuffer Toolkit
Graphics Rasterizer
JDK 6
JDK 7
Modules
Multi-Language VM
New I/O
Port: Haiku
Port: MIPS
XRender Pipeline
VisualVM
Tools
Java SE
Mercurial
NetBeans
OpenJDK
Now available in
Fedora 9 (Sulphur)
What is this? The place to collaborate on an open-source implementation of the Java Platform, Standard Edition, and related projects. (Learn more.)
Download and install the open-source JDK 6 for Ubuntu 8.04 (Hardy Heron), Fedora 9 (Sulphur), or Red Hat Enterprise Linux 5. If you came here looking for Sun's JDK 6 product binaries for Solaris, Linux, or Windows, which are based largely on the same code, you can download them from java.sun.com.
Learn how to use the JDK to write applications for a wide range of environments, from desktop to server.
Hack on the JDK itself, right here in the growing OpenJDK Community: Browse the code on the web, get a source bundle or clone a Mercurial repository to make a local copy, read the tutorial on how to build and hack on the code with the NetBeans IDE, and contribute a patch to fix a bug, enhance an existing component, or define a new feature.

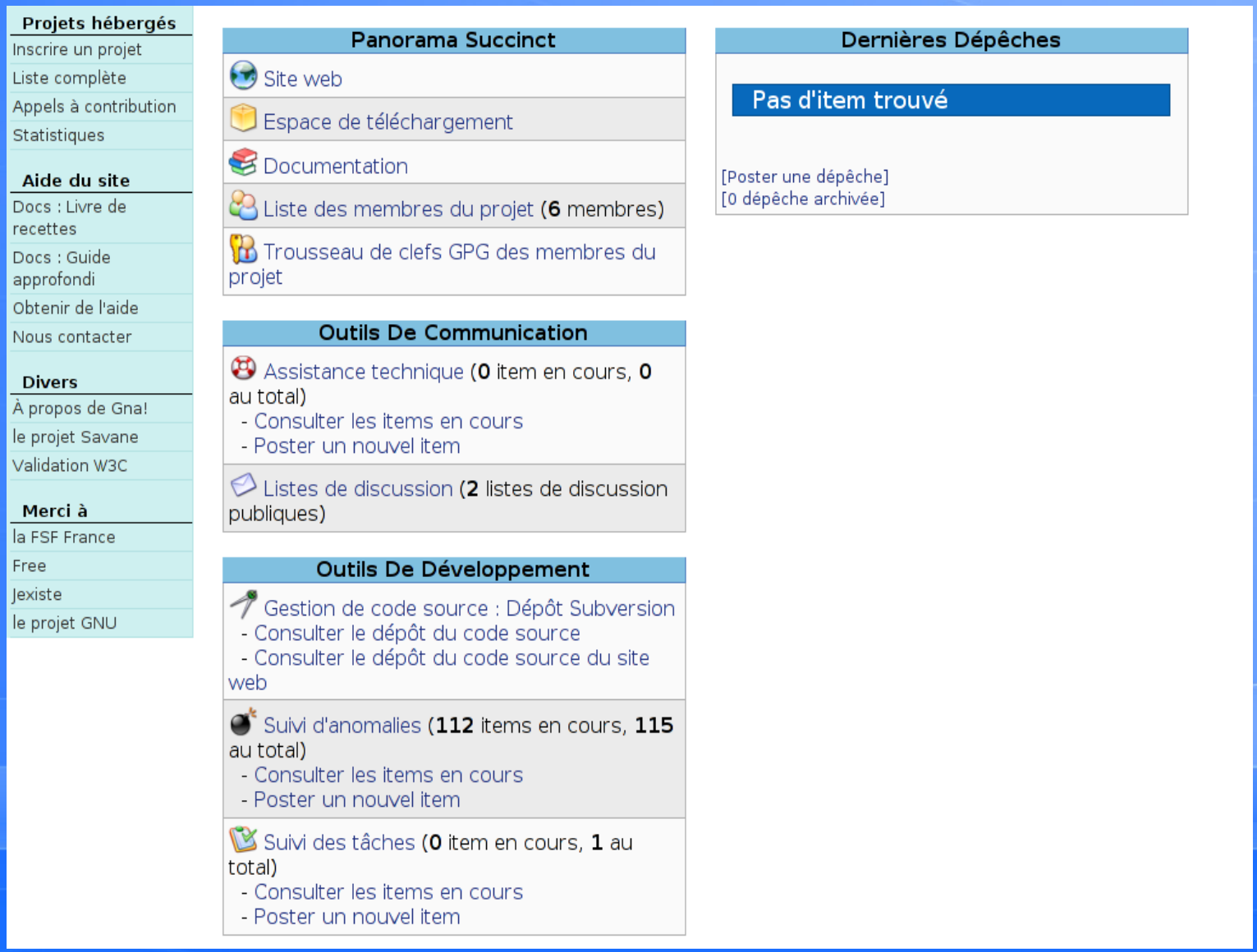
Mise en place des outils

- Plateformes prêtes à l'emploi
  - Très intéressant au début du projet, permet de disposer de tous les outils en quelques minutes
  - Sourceforge, http://sourceforge.net
  - Savannah, http://savannah.gnu.org
  - Gna!, http://gna.org
  - Berlios, http://berlios.de
- Hébergement personnalisé
  - Nécessite plus de travail, à éviter au début
  - Permet de répondre aux besoins plus spécifiques du projet

Gna (1)


Gna!
Authentifié(e) en tant que tpetazzo
Mes nouveaux items
Mes items
Mes groupes
Mes préférences
Retour à l'anonymat
Cette page
Recharger proprement
Version imprimable
Rechercher
in
Projets
Rechercher
gDTC - Accueil
Groupe
Accueil
Site web
Téléchargement
Documentation
Assistance
Listes de discussion
Code source
Anomalies
Tâches
Dépêches
L'administrateur de ce projet n'a pas encore renseigné de description courte. Si vous êtes l'administrateur du projet, vous pouvez l'enregistrer maintenant.
Date d'enregistrement : lundi 23.07.2007 à 20:26
Licence : GNU General Public License V2 or later
Stade de développement : 0 - Undefined
Membres
Administrateur :
- Benjamin Drieu
6 membres
[Lister les membres]
Identifiants Du Groupe
Id : #2313
Nom système : gdtc
Nom : gDTC
Type de groupe : Programs
Rechercher Dans Ce Groupe
in
Livre de recettes
Rechercher

Gna (2)


Projets hébergés
Inscrire un projet
Liste complète
Appels à contribution
Statistiques
Aide du site
Docs : Livre de recettes
Docs : Guide approfondi
Obtenir de l'aide
Nous contacter
Divers
À propos de Gna!
le projet Savane
Validation W3C
Merci à
la FSF France
Free
Jexiste
le projet GNU
Panorama Succinct
Site web
Espace de téléchargement
Documentation
Liste des membres du projet (6 membres)
Trousseau de clefs GPG des membres du projet
Outils De Communication
Assistance technique (0 item en cours, 0 au total)
- Consulter les items en cours
- Poster un nouvel item
Listes de discussion (2 listes de discussion publiques)
Outils De Développement
Gestion de code source : Dépôt Subversion
- Consulter le dépôt du code source
- Consulter le dépôt du code source du site web
Suivi d'anomalies (112 items en cours, 115 au total)
- Consulter les items en cours
- Poster un nouvel item
Suivi des tâches (0 item en cours, 1 au total)
- Consulter les items en cours
- Poster un nouvel item
Dernières Dépêches
Pas d'item trouvé
[Poster une dépêche]
[0 dépêche archivée]

Organisation du projet

- « La cathédrale et le bazar », Eric Raymond, 1997
- Il compare l'organisation classique hierarchique des entreprises à des cathédrales
- Et l'organisation du développement du Logiciel Libre à un bazar, basé sur des interactions et des relations implicites
- En réalité
  - Chaque projet a une structure, avec un ou plusieurs mainteneurs, des contributeurs
  - Basée sur la méritocratie

Consensus ou fork

- Un aspect important de tout projet libre est qu'il est possible de « forker » à tout moment
- Personne ne peut contrôler indéfiniment le projet
- Deux forces sont en action
  - D'un coté l'union faisant la force, les développeurs sont incités à rester ensemble
  - De l'autre, les avis divergents, les objectifs à long terme différents, les désaccords sur l'organisation incitent à « forker »
- Un équilibre est nécessaire

Consensus ou fork (2)

- Exemples
  - Sodipodi et Inkscape, désaccord sur les fonctionnalités
  - Xfree86 et X.org, désaccord sur la gestion du projet et la licence
  - Mambo et Joomla, trop forte emprise d'une entreprise

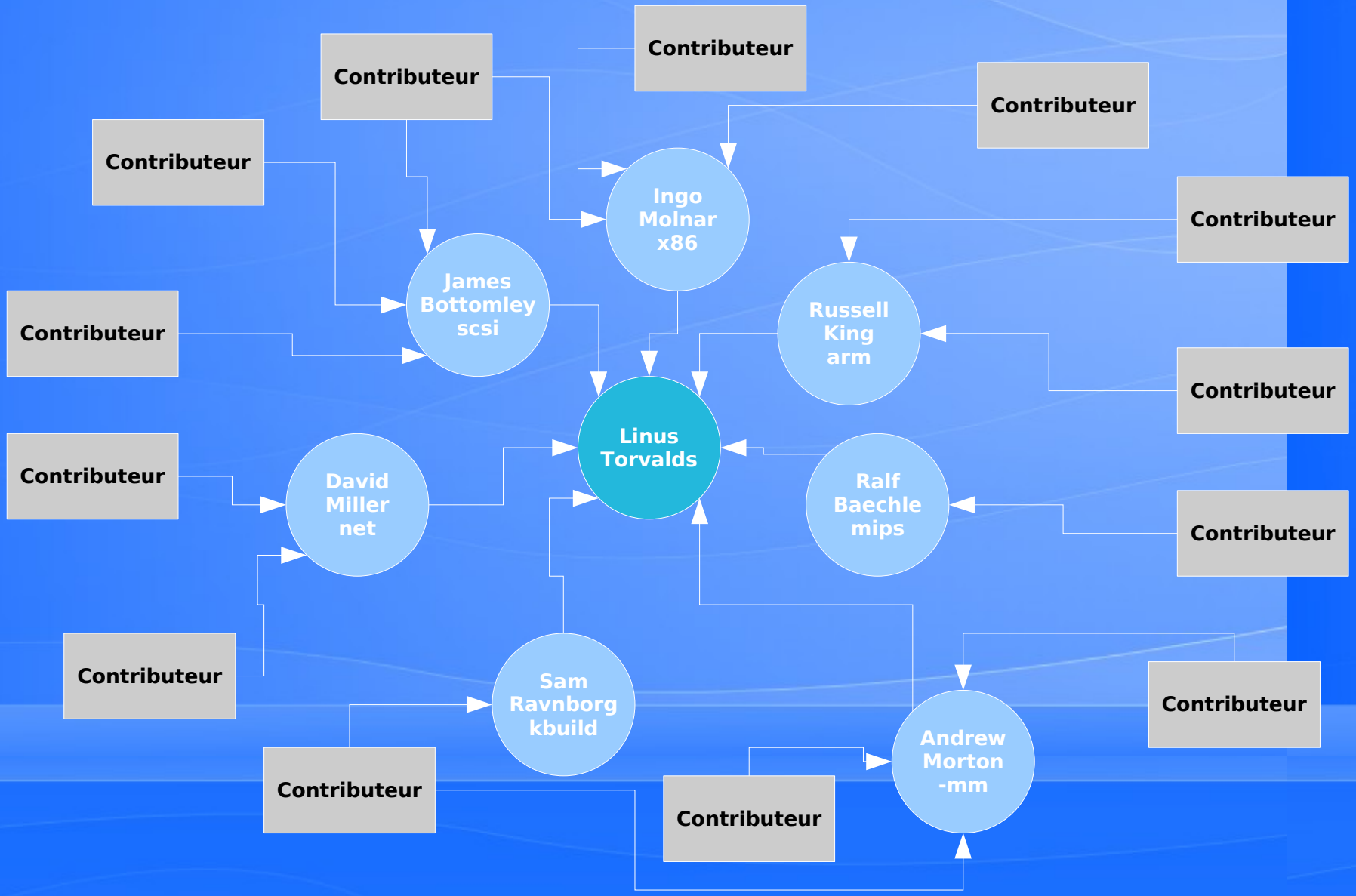
Dictateur bénévole et bienveillant

- Forme la plus classique pour les projets de petite taille, qui sont les plus nombreux
- Une personne, souvent le créateur du projet, guide les développements
- Il doit s'assurer que les décisions qu'il prend font déjà l'objet d'un consensus assez fort
- Sinon
  - Risque de fork
  - Pas de contributeurs
- Nécessite plus que des talents techniques : réactivité, disponibilité, diplomatie, etc.

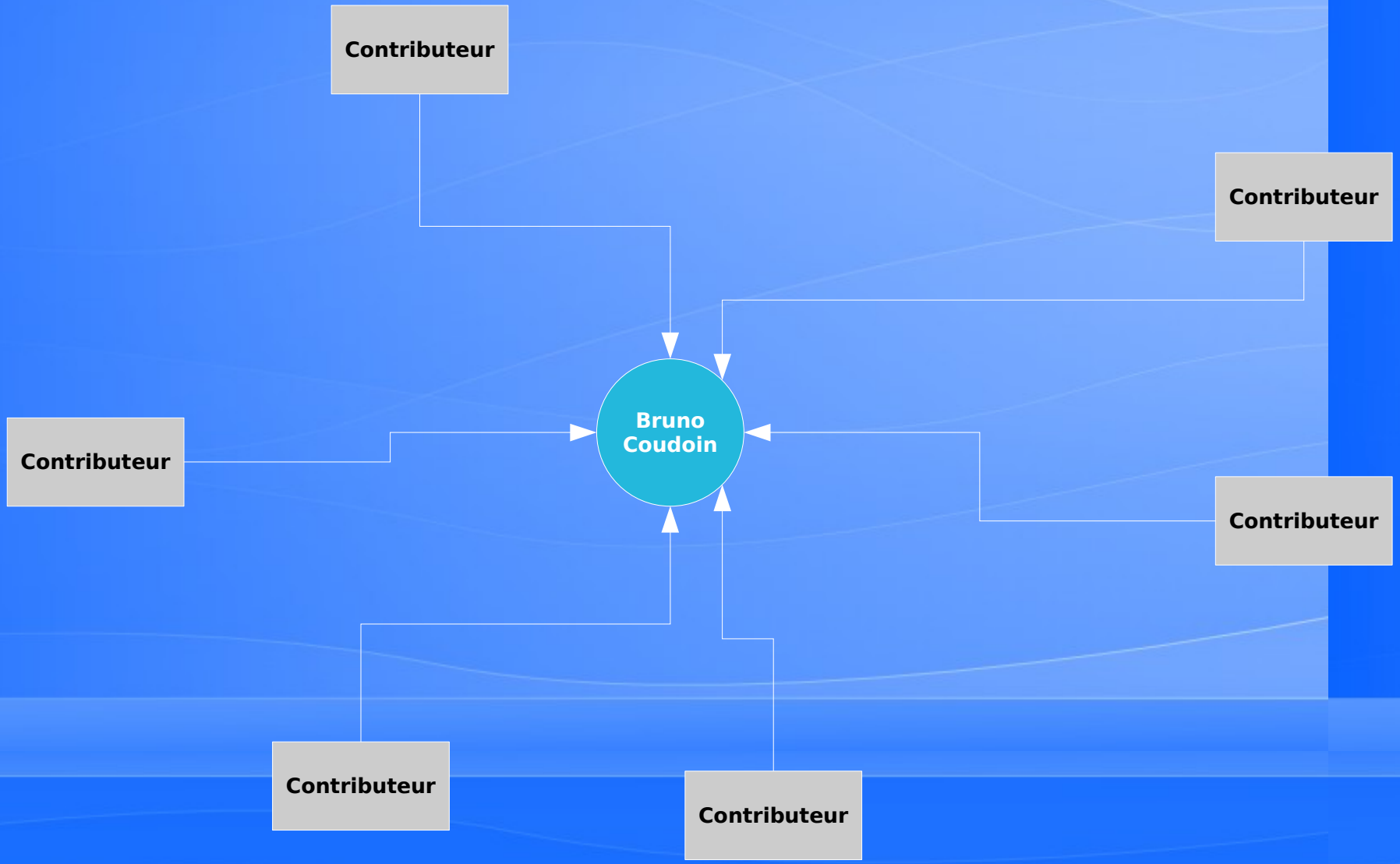
Dictateur bénévole et bienveillant

- « Provided the development coordinator has a communications medium at least as good as the Internet, and knows how to lead without coercion, many heads are inevitably better than one. » Eric Raymond

Exemple de Linux



Exemple de GCompris



Projets structurés

- Les membres se mettent d'accord sur un fonctionnement, avec en général élection d'un coordinateur ou d'une équipe de coordination
- La plupart des décisions continuent à se prendre par consensus, un processus de vote est prévu pour les décisions litigieuses
- Beaucoup de grands projets se structurent de cette manière
  - Gnome, Debian, KDE, etc.
- Concerne tous les projets fortement liés à une entreprise, pour leur donner une autonomie

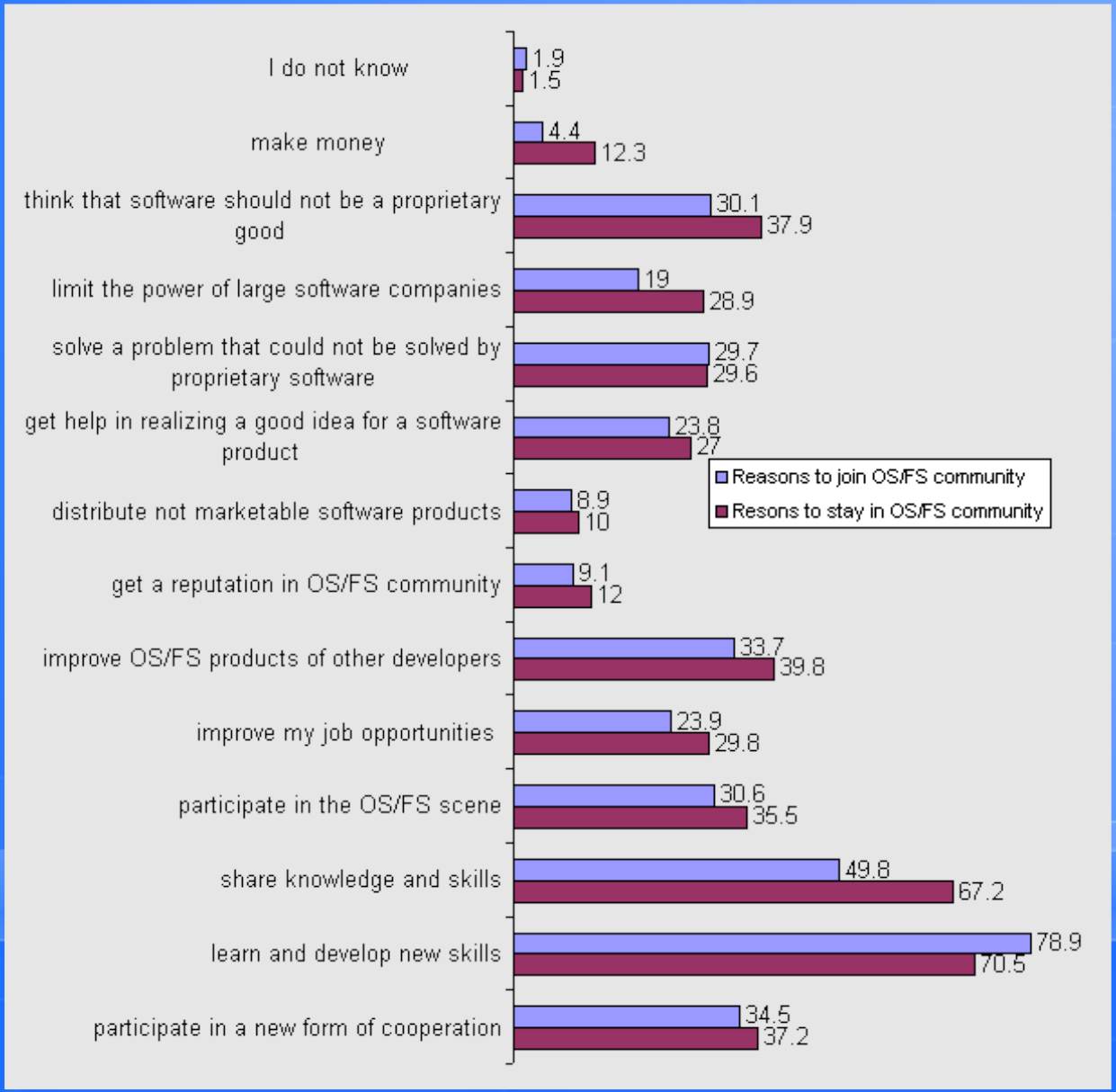
Debian

- Distribution GNU/Linux conçue et développée entièrement par des bénévoles
- Organisation dictée par une Constitution
- Debian Project Leader élu chaque année
- Plus de mille développeurs officiellement reconnus ayant le droit de vote
- Le développement au jour le jour continue à se faire par consensus
- Des contributeurs extérieurs participent également

Gestion des contributeurs

- Trois types principaux de contributeurs
  - Bénévoles
  - Salariés fixes
  - Salariés intermitents
- Comprendre leurs motivations pour les attirer
  - La communauté du libre n'est pas un vivier de développeurs qui vont travailler gratuitement pour vous
- Les faire collaborer ensemble
  - Objectifs différents
  - Disponibilités différentes
  - Motivations différentes

Motivation des bénévoles



Source:
Free/Libre and Open Source Software: Survey and Study

Part IV : Survey of Developers

University of Maastricht

Salariés fixes

- Coopération dans le monde du libre se fait entre individus, pas entre entreprises
- Risque d'avoir un cercle fermé qui prend les décisions, en laissant la communauté à l'écart
- Risque d'avoir l'entreprise qui veut trop «tirer» le logiciel vers son activité économique et refuse quasiment toutes les contributions
- Exemples
  - Mozilla Firefox
  - OpenOffice.org
  - Linux

Salariés intermitents

- Prestataires payés par un client pour développer une fonctionnalité donnée
  - La fonctionnalité correspond-elle aux orientations du projet ? Sera-t-il possible de l'intégrer ? Conflit entre les intérêts des prestataires et du projet ?
- Financement par des opérations type « Google Summer of Code »
  - Sujet défini par les membres du projet, pas de conflit d'objectifs

L'arrivée de nouveaux contributeurs

- Les projets libres ont des similitudes, mais aussi des cultures et des pratiques différentes
- L'accueil des nouveaux contributeurs est important, ils sont peut-être les futurs développeurs du projet
- Les nouveaux contributeurs ne sont pas choisis, des bonnes et des mauvaises surprises
- Faciliter leur arrivée par un accès facile aux informations de développement
  - Outils de développement
  - Guidelines (style de code, design, processus)

Difficultés

- Forte variabilité de la force de travail, surtout dans les projets constitués uniquement de bénévoles
- Les bénévoles travaillent uniquement sur ce qui les intéresse
  - Sur les fonctionnalités motivantes
  - Moins sur la correction de bugs, la documentation

Types de contribution

- Utilisateur qui rapporte une anomalie sur la liste de discussions
  - L'inciter à la soumettre dans l'outil de suivi de bug
  - Entamer les discussions pour cerner le bug, en s'adaptant au niveau technique de l'utilisateur
  - En général, faire développer le correctif par l'auteur du code incriminé
  - Correct proposé sur la liste de discussion pour relecture, puis intégration dans le logiciel
  - « Treating your users as co-developers is your least-hassle route to rapid code improvement and effective debugging »

Types de contribution

- Utilisateur qui suggère une nouvelle fonctionnalité
  - Lancer une discussion sur la liste, éventuellement une page Wiki pour collaborer sur la définition de cette fonctionnalité
  - Après définition plus précise, noter la fonctionnalité pour la développer plus tard (Wiki, suivi de bug)
  - Étape cruciale pour trouver une adéquation entre la demande de l'utilisateur et les motivations des contributeurs
  - Rester ouvert aux nouvelles fonctionnalités dont en conservant un cap pour l'objectif du logiciel
  - « The next best thing to having good ideas is recognizing good ideas from your users. Sometimes the latter is better » Eric Raymond

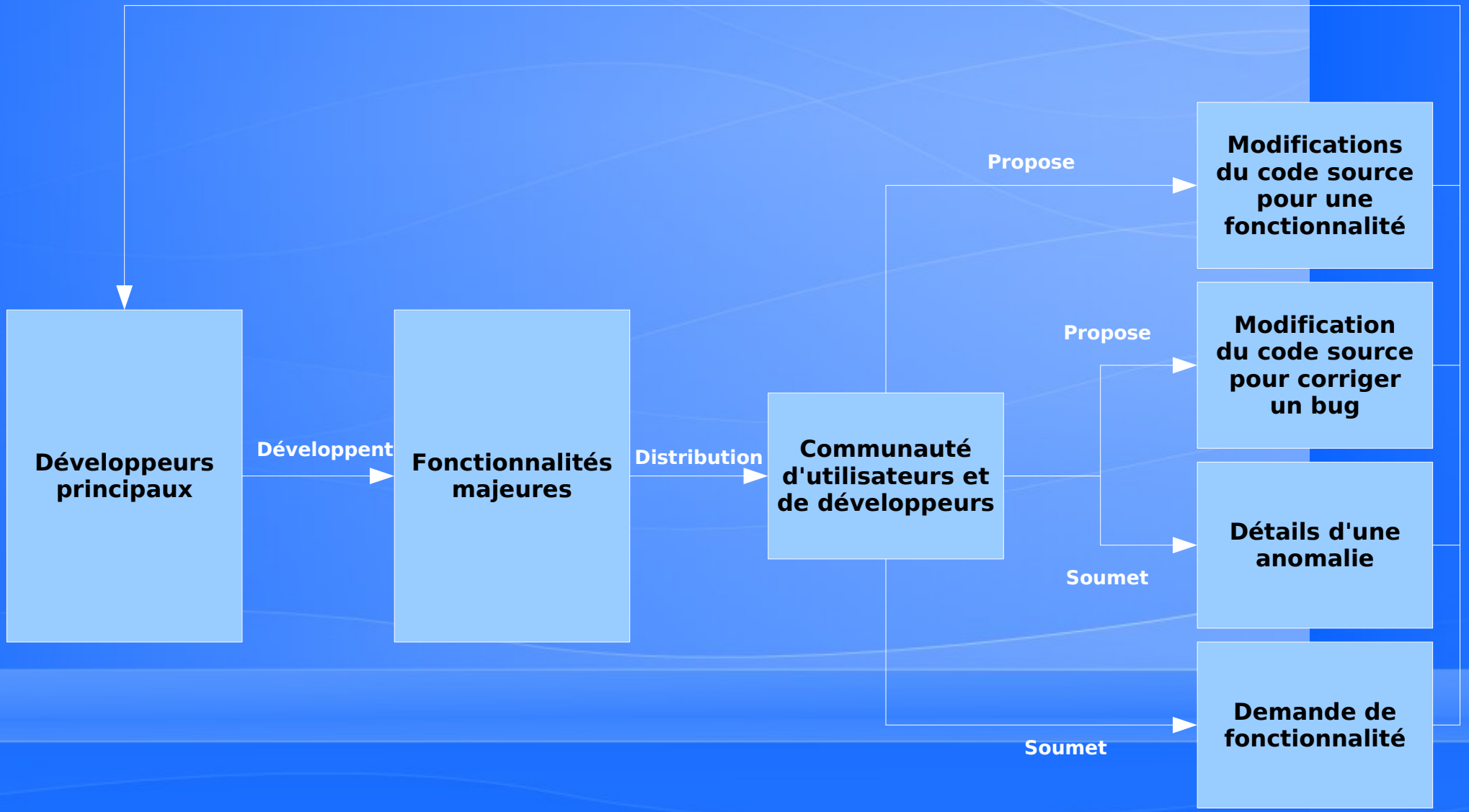
Types de contribution

- Développeur propose une modification mineure
  - Envoi d'un patch sur la liste de discussion, avec une description claire et concise de la modification
  - Période de discussion permettant aux autres développeurs de commenter, de suggérer des améliorations, de proposer des solutions alternatives
  - À l'issue de cette phase de discussion, un consensus apparaît en général naturellement
    - Rejet de la modification, il faut retravailler sur la proposition
    - Acceptation et intégration

Types de contribution

- Développeur propose une modification majeure
  - Souvent une bonne idée d'en discuter un peu avant l'implémentation, pour trouver le bon design, comment rendre la fonctionnalité plus générique
  - Certains projets ne discutent que sur du code
  - Découper la fonctionnalité en plusieurs patches pour rendre la relecture plus aisée
  - Phase de discussion avec la communauté, itérations pour l'amélioration de l'implémentation
  - Intégration dans le logiciel ou rejet

Déroulement



Gestion des releases

- Un principe fondateur « Release early. Release often. » (Linus Torvalds)
  - Créé de l'activité dans le projet
  - Permet aux utilisateurs de tester rapidement les nouvelles fonctionnalités
  - Réduit le temps entre le développement et la distribution aux utilisateurs, ce qui motive les développeurs
- Version de développement toujours disponible grâce au système de gestion de version
  - N'importe qui peut y accéder pour proposer une modification

Gestion des releases

- Une fois les principaux développements réalisés, publier des versions de test (alpha, beta, release candidate)
  - « Étant donné une base de testeurs et de co-développeurs suffisamment large, presque tous les problèmes seront identifiés rapidement et le correctif sera évident pour quelqu'un » Eric Raymond
- Puis publier la version majeure

Gestion des releases

- Similaire au monde du logiciel propriétaire
  - Sauf que l'échelle de temps est beaucoup plus réduite
    - Plusieurs mois / années pour les logiciels propriétaires, faible diffusion des versions de test
    - Quelques semaines / mois pour les logiciels libres, large diffusion de toutes les versions
  - La version en développement est accessible à tous

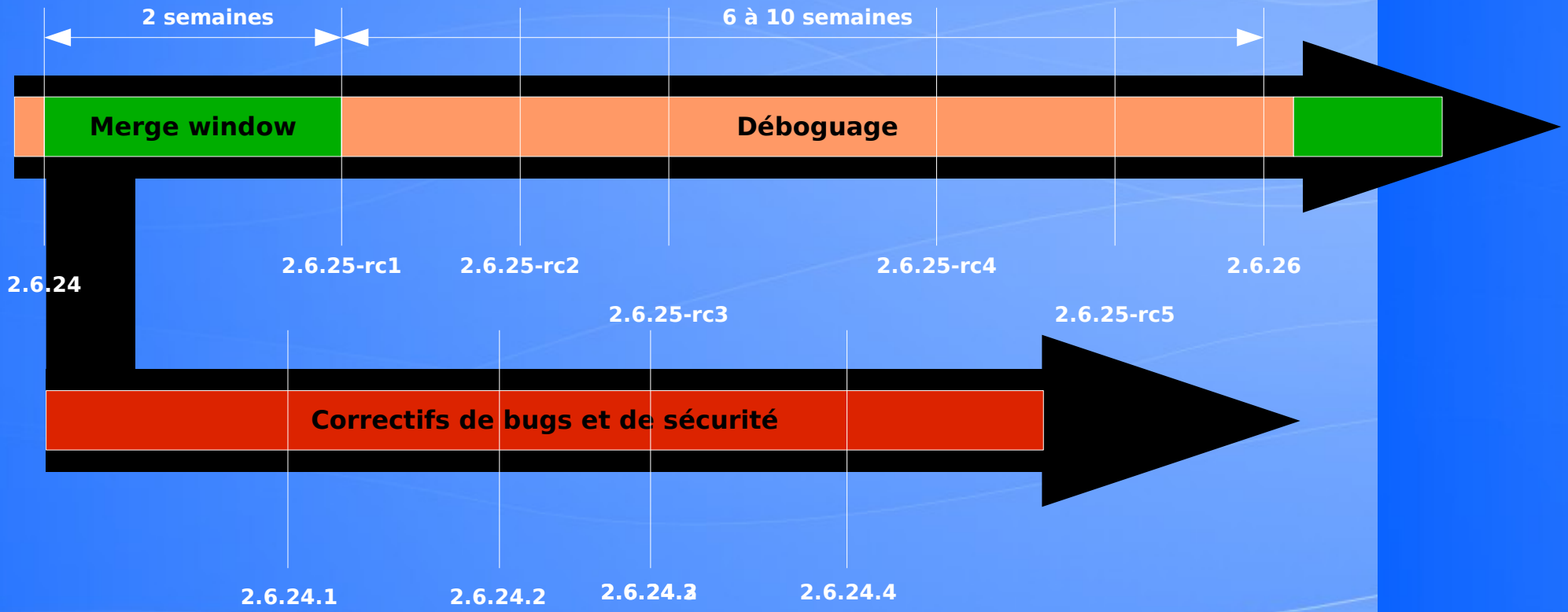
Gestion des releases

- Certains projets fonctionnent sur un mode temporel
  - Ils fixent une date pour la prochaine version, et intègrent ce qui est prêt pour cette version
  - Le reste sera pour les prochaines versions
  - GNOME, Linux
- D'autres fonctionnement sur un mode fonctionnel
  - Ils fixent les fonctionnalités, et attendent qu'elles soient implémentées pour publier la nouvelle version

Exemple de Linux, modèle de dev

- Après la sortie d'une version stable, une «merge window» s'ouvre
  - Pendant deux semaines
  - Linus intègre les nouvelles fonctionnalités, les changements majeurs
  - Cloturé par la sortie d'une «release candidate»
- À la fin de la «merge window», la période de déboguage s'ouvre
  - Pendant six à dix semaines
  - Seuls les correctifs de bugs ou améliorations mineures sont intégrées
  - Publication régulière de versions de test

Exemple de Linux, modèle de dev



Orientations du projet

- Certains projets n'ont pas de *roadmap,* les nouvelles fonctionnalités sont celles que les contributeurs développent, en fonction de leurs besoins. Ex: Linux
  - Discussions nécessaires pour faire converger des solutions concurrentes
- D'autres se structurent autour d'objectifs pour la ou les prochaines versions du logiciel
  - Définition par consensus des objectifs
  - Les contributions restent le moteur du projet, les objectifs doivent correspondre aux aspirations des développeurs

Communiquer par le réseau

- L'infrastructure technique permet la communication, mais uniquement de manière électronique
- Pose un certain nombre de challenges par rapport à la communication orale
  - Différences de culture et de langue impliquant des difficultés de compréhension
  - Discussions pouvant s'éterniser et devenir tendues
  - Besoin de respecter certaines règles (forme, fond, attitude)
  - Parasitage

Langue du projet

- La quasi-totalité des projets Logiciels Libres utilisent la langue anglaise
  - Communication sur les listes, gestionnaire de bugs
  - Commentaires et noms de variables/fonctions dans le code source
- Permet de toucher le plus grand nombre de développeurs potentiels
- N'empêche pas la création de listes d'utilisateurs par langues quand le projet atteint une masse critique

Communiquer hors réseau

- Nombreuses conférences autour du Logiciel Libre, généralistes ou spécialisées
- Bonnes occasions pour rencontrer les autres développeurs et présenter les nouveautés et idées
- Moyens techniques à disposition: connexion Internet, salles de travail
- Généralistes: FOSDEM, RMLL
- Spécialisées: GUADEC, aKademy, Linux Symposium

Faire connaître son projet

- Annoncer sa disponibilité sur les sites de nouvelles du Logiciel Libre
  - http://slashdot.org, http://linuxfr.org
  - Sites spécialisés dans le domaine
- Le référencer dans les annuaires comme Freshmeat
  - http://www.freshmeat.net
- Écrire des articles dans les magazines
- Soumettre des propositions d'interventions dans les conférences

Activités hors développement

- Documentation
  - Utilisateur
  - Développeur, pour intégrer facilement de nouveaux développeurs
- Traduction
  - Utilisation d'outils spécialisés facilitant la traduction et sa mise à jour (gettext, kbabel)
- Marketing
  - Site Web
  - Conférences

Financement

- Pourquoi des entreprises participent-elles à des projets logiciel libre ?
  - Répartir la charge de travail, quand le logiciel n'est pas l'activité principale de l'entreprise
  - Vendre des services
  - Supporter du matériel
  - Concurrencer une autre entreprise
  - Image
  - Double-licence

Financement

- Comment financent-elles les projets ?
  - Mise à disposition de développeurs et contributeurs qui vont travailler au développement de fonctionnalités pertinentes pour l'entreprise
  - Paiement d'un prestataire, souvent un développeur principal existant du projet
  - Mise à dispositions de ressources techniques (serveur, bande passante)
  - Donations

Avec une entreprise

- Certains projets libres sont issus d'un code fermé à l'origine ou sont lancés par une entreprise
- Gestion du projet pas toujours facile
  - Risque d'avoir l'entreprise qui veut trop «tirer» le logiciel vers son activité économique et refuse quasiment toutes les contributions
  - Nécessite une réelle ouverture, dans les outils mais aussi et surtout dans les pratiques

Aspects légaux

- Les logiciels libres sont soumis au droit d'auteur, comme toute oeuvre artistique ou logiciel
- Les licences libres donnent des libertés à tous les utilisateurs, sans distinction
- Certaines posent certaines limitations à ces libertés
- Principe du copyleft
  - Ce qui est libre doit le rester
  - Impose que les versions modifiées soient redistribuées sous la même licence
- Choisir une licence connue rassure et permet l'échange de code source

Licences

- Licence GPL
  - Principe du copyleft pour toutes les oeuvres dérivées
  - Pas d'intégration dans un logiciel propriétaire
- Licence LGPL
  - Possibilité d'intégration dans un logiciel propriétaire en tant que bibliothèque
- Licence type BSD
  - Pas de copyleft
- Choix à faire en fonction
  - Du type du logiciel, du public souhaité, du modèle économique

Questions ?

Évaluation

1)En dehors de sa licence, qu'est-ce qui différencie un projet logiciel libre d'un projet de logiciel propriétaire ? Présentez en détail ces différences.

2)Vous démarrez un projet de Logiciel Libre, quelles sont les étapes cruciales du lancement du projet à la construction d'une communauté ?

3)Selon vous, quelles sont les qualités que doit avoir un gestionnaire de projet Logiciel Libre ?

http://thomas.enix.org/pub/conf/ups2008/