Le logiciel libre

Réseau rdatadev @ OSUG

Gabriel Moreau

Laboratoire LEGI - CNRS / UGA / Grenoble-INP - France

13 décembre 2018

Au départ d'une discutions était

Compilateurs :

- `icc` (`ifort`)
- `clang` (LLVM)
- `gcc` (`gfortran`, `gnat`...)

Cela va nous amener à parler de logiciels libres...

Liminaire

- Je n'enfonce que des portes ouvertes
- Je ne vais pas révolutionner votre compréhension / philosophie
- Prière de m'interrompre et de poser des questions
- J'ai besoin de votre interaction pour tenir 1 h (quoique)...
- Je dis certainement quelques conneries. C'est une v0 !

On finit juste l'année avec un sujet bien facile !

Compilateurs :

- `icc` - licence propriétaire
- `clang` - licence libre NCSA (type BSD)
- `gcc` - licence libre GPL

Mais d'où viens ce concept de licence ?

À l'origine, le code source tombe sous les règles du droit d'auteur.

- Par défaut, pas de licence = droit d'auteur strict
- Tout appartient à votre employeur
- Donc logiciel à 100 % non libre !

Mettre une licence (libre ou non) est donc la première chose à faire dans un développement afin d'éviter le droit d'auteur.

Qui dans la salle pose le fichier de licence en premier dans un repository ?

- `LICENSE.txt`
- `AUTHORS.txt`
- `COPYRIGHT.txt`

Richard Stallman - 1980

- 1980 - La fameuse affaire de l'imprimante Xerox au MIT
- De TECO à Emacs - les utilisateurs étaient libres de modifier et de redistribuer le code, à la condition de redonner en retour à la communauté les extensions qu'ils écrivaient.
- 1983 - GNU (GNU's Not Unix)
- Copyleft - all rights reversed / Don Hopkins
- 1985 - FSF (Free Software Foundation)
- 1989 - GPLv1 avec Eben Moglen
- 1990 - Projet GNU quasi complet à l'exception du noyau $\longrightarrow$ projet Hurd
- Pris de douleurs aux poignées, Stallman ne peut quasiment plus programmer dans les années 90...

La norme POSIX doit son nom à Stallman

Linus Torvalds - 1991

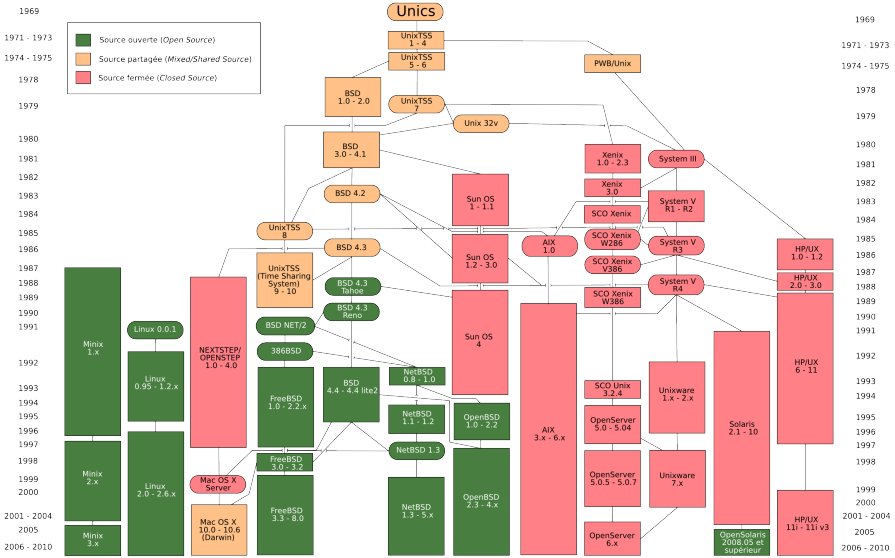
- 1991 - Annonce de Linux sur Usenet (mais non libre)
- *Ce n'est que quelques années plus tard que Linus Torvalds rendit Linux libre, quand il sentit que la popularité du noyau Linux le rendait indissociable de son créateur et qu'il ne courrait pas le risque de « perdre » son bébé* (Wikipedia)
- 2002 - Utilisation du logiciel BitKeeper !
- 2005 - Naissance de Git

Linus n'est clairement pas un apôtre de la philosophie du logiciel libre mais l'utilise à bon escient

La bataille BSD / AT&T

- UNIX appartient à AT&T
- Mais les universités (Berkeley, MIT...) avaient le code source et l'amélioraient
- 1983 - Guerre System V (flux, signal) versus BSD (tcp/ip, tty)
- 1988 - Norme POSIX
- 1992 - Attaque en justice $\longrightarrow$ accord et quelques modifications
- 1995 - FreeBSD2.0 basé sur 4.4BSD-Lite et des éléments de 386BSD
- FreeBSD, NetBSD, OpenBSD - Licence BSD permissive

Linux est passé dans une brèche étroite entre les BSD et le Hurd !


Source ouverte (Open Source)
Source partagée (Mixed/Shared Source)
Source fermée (Closed Source)
1969
1971 - 1973
1974 - 1975
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001 - 2004
2005
2006 - 2010
Unics
UnixTSS 1 - 4
UnixTSS 5 - 6
UnixTSS 7
PWB/Unix
BSD 1.0 - 2.0
Unix 32v
BSD 3.0 - 4.1
Xenix 1.0 - 2.3
System III
Xenix 3.0
BSD 4.2
Sun OS 1 - 1.1
System V R1 - R2
SCO Xenix
UnixTSS 8
BSD 4.3
Sun OS 1.2 - 3.0
AIX 1.0
SCO Xenix W286
System V R3
SCO Xenix V386
HP/UX 1.0 - 1.2
UnixTSS (Time Sharing System) 9 - 10
BSD 4.3 Tahoe
BSD 4.3 Reno
SCO Xenix W386
System V R4
HP/UX 2.0 - 3.0
BSD NET/2
Sun OS 4
386BSD
NetBSD 0.9 - 1.0
AIX 3.x - 6.x
HP/UX 6 - 11
Minix 1.x
Linux 0.0.1
NEXTSTEP/ OPENSTEP 1.0 - 4.0
FreeBSD 1.0 - 2.2.x
BSD 4.4 - 4.4 lite2
NetBSD 1.1 - 1.2
OpenBSD 1.0 - 2.2
SCO Unix 3.2.4
Unixware 1.x - 2.x
OpenServer 5.0 - 5.04
Solaris 2.1 - 10
Linux 0.95 - 1.2.x
Minix 2.x
FreeBSD 3.0 - 3.2
NetBSD 1.3
OpenServer 5.0.5 - 5.0.7
Unixware 7.x
Linux 2.0 - 2.6.x
Mac OS X Server
OpenBSD 2.3 - 4.x
FreeBSD 3.3 - 8.0
NetBSD 1.3 - 5.x
OpenServer 6.x
OpenSolaris 2008.05 et supérieur
HP/UX 11i - 11i v3
Minix 3.x
Mac OS X 10.0 - 10.6 (Darwin)

Open source - 1998 - Eric Raymond

- L'open source se focalise sur des considérations techniques plus que philosophiques
- Source ouvert ne veut pas forcément dire logiciel libre !
- Ambiguïté de l'expression anglaise « free software » (logiciel libre).
  « free » possède deux significations : « libre » (au sens de « liberté ») et « gratuit ».

Personnellement, je trouve que ce concept *Open Source* n'est pas super clair et je préfère l'éviter !
Le terme *logiciel libre* en français est lui très clair.

- Le logiciel libre, selon Stallman, est un mouvement social qui repose sur les principes de Liberté, Égalité, Fraternité
- L'open source décrit pour la première fois dans La Cathédrale et le Bazar (Raymond), s'attache aux avantages d'une méthode de développement au travers de la réutilisation du code source

Avec le temps, les deux mouvements se sont rapprochés sur les définitions de ce qu'est un logiciel libre.

Les 4 libertés

- 0 - La liberté d'exécuter le programme, pour tous les usages
- 1 - La liberté d'étudier le fonctionnement du programme et de l'adapter à ses besoins
- 2 - La liberté de redistribuer des copies du programme (ce qui implique la possibilité aussi bien de donner que de vendre des copies)
- 3 - La liberté d'améliorer le programme et de distribuer ces améliorations au public, pour en faire profiter toute la communauté

L'accès au code source est une condition d'exercice des libertés 1 et 3.
Ces libertés sont irrévocables.

...

Open Source Definition / Debian Free Software Guidelines (DFSG)

1. Redistribution gratuite
2. Code source
3. Travaux dérivés
4. Intégrité du code source de l'auteur
5. Pas de discrimination contre les personnes ou les groupes
6. Pas de discrimination contre les champs d'utilisation
7. Distribution de la licence
8. La licence ne doit pas être spécifique à un produit
9. La licence ne doit pas restreindre d'autres logiciels
10. La licence doit être neutre sur le plan technologique

Avantages

- Le recyclage de fonctionnalités
- L'efficacité et la fiabilité
- Le respect des standards
- Une garantie pour la sécurité et les libertés
- L'indépendance et la pérennité
- Un avantage économique

Inconvénients

- Une offre dispersée
- Des modèles économiques complexes

Croissance et prévision de croissance du marché du logiciel libre en France

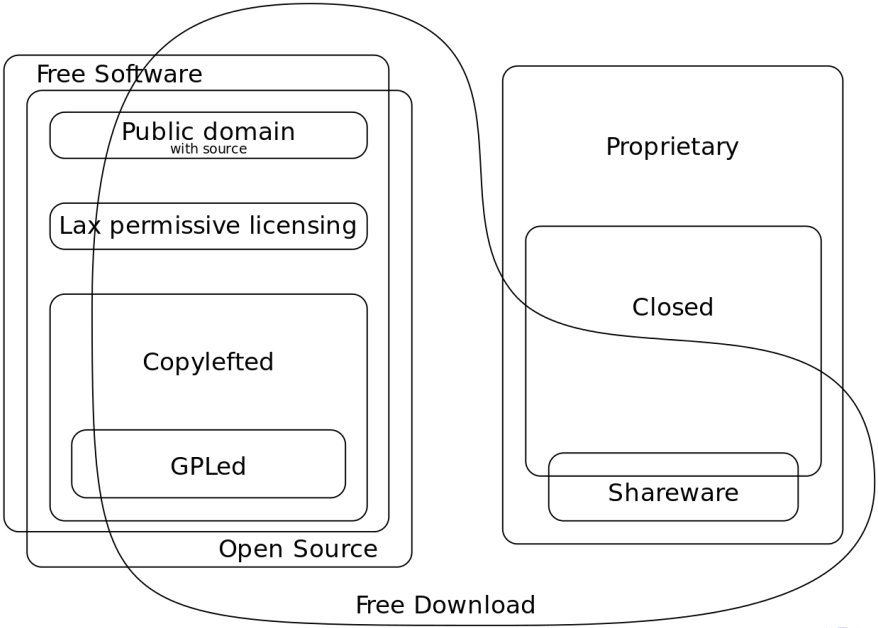
	2002	2003	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017
Chiffre d'affaires du logiciel libre (millions d'€)	60	100	140	250	440[49]	733	1 100[50]	1 500[réf. nécessaire] (Europe : 5 100)	2 200	2 501	3 000[51]			4 100[52]	4 500[53]	
Part de marché du logiciel libre (dans l'industrie du logiciel)	0,2 %	0,4 %	0,5 %	0,9 %	1,4 %	2,1 %				6 %[54]			10 %[55]	13 %[56]		
Croissance du marché du logiciel libre		67 %	40 %	79 %	72 %	66 %[57]		47 %[réf. nécessaire]	30 %[58]		20 %[49]		33 % depuis 2012[55]		15 %	
Croissance du reste du marché		-4,2 %	3,8 %	6,3 %	6,6 %	7,1 %							17 %[59]			

Attention : Libre différent de Gratuit

- Voir les services Google et des autres GAFAM !
- À l'origine les outils libres Cygnus du GNU était payant
- Les développeurs libres ont besoin de manger le soir
- Financement des nouvelles contributions
- Système participatif
- La double licence GPL + Propriétaire $\longrightarrow$ transfert des droits pour les contributions (Qt de Trolltech)
- ...

Les trois types de licences

- Licences propriétaires (*privateur*)
- Licences permissives (*encore plus libre que libre*)
- Licences copylefts - gauche d'auteur - héréditaire (*virale, cancer...*)
- *Domaine publique*


Free Software
Public domain
with source
Lax permissive licensing
Copylefted
GPLed
Open Source
Proprietary
Closed
Shareware
Free Download

La bataille BSD / GPL - Permissive versus Copyleft

- Celui qui lave plus blanc que blanc, plus libre que libre...

Permissive

- Intégration possible dans du code propriétaire - Coup de pousse aux startups
- Pas de demande de retour
- Bibliothèque de base (réseau, image, hdf5...)
- Diffusion d'un standard sur tous les OS

Copyleft

- Oblige le retour du code source des contributions (en cas de diffusion)
- Partiellement incompatible avec le propriétaire
- Logiciel utilisateur souhaitant rester libre
- Limite le nombre de forks

$\longrightarrow$ Dans tous les cas, le logiciel libre pousse à l'usage de **standards et formats ouverts**

Le cas du domaine publique

- 70 ans après la mort de l'auteur $\longrightarrow$ on ne le verra pas de notre vivant !
- Impossible en droit français de mettre son oeuvre dans le domaine publique

Palliatif :

- CC0 - Creative Common Zero
- WTFPL - Do What the Fuck You Want To Public License (exemple de code : `https://huit.re/`)

L'objectif de ces licences est de coller au domaine publique et de n'imposer aucune contrainte.

La galaxie GNU : LGPL, GPL, AGPL

Toute diffusion d'une version modifiée impose de donner les sources à ≪l'acheteur≫.

- LGPL - Bibliothèque - Liens dynamique
- GPL - Programme - Appels systèmes
- AGPL - Service (daemon) - Sockets réseaux

Ce qui les distingue est la frontière sur laquelle s'arrête de fonctionner l'hérédité de la licence.

Il y a des exceptions avec les bibliothèques proches du noyau...

Les «putains» de contournements

Ou comment rendre le libre non libre...

Contournement - La Tivoïsation

La tivoïsation est la création d'un système qui inclut des logiciels libres, mais utilise le matériel électronique pour interdire aux utilisateurs d'y exécuter des versions modifiées.

Exemple : enregistreurs vidéo numériques de la marque TiVo vidéo numériques

$\longrightarrow$ GPLv3 (2007) a des clauses anti tivoïsation.

Contournement - Les DRM - Digital Rights Management

Les DRM sont des systèmes qui restreignent les libertés de l'utilisateur, en ne lui donnant pas librement accès au code source, et ainsi il ne peut copier ou modifier le code source comme il l'entend et comme lui permet normalement la licence.

Exemple : diffusions vidéos des grands groupes

$\longrightarrow$ GPLv3 (2007) a des clauses limitant la portée des DRM (l'utilisateur doit pouvoir les enlever librement).

Contournement - La location

Dans le cas d'une location d'un équipement, le propriétaire n'est pas obligé de donner le code source (même libre) faisant tourner celui-ci.

Exemple : Free ne voulait pas donner le code source de sa première box en argumentant que celle-ci était partie intégrante de son réseau (et donc lui appartenait)

$\longrightarrow$ Au procès, Free a préféré jeter l'éponge avant la fin.

Contournement - La daube des brevets logiciels

Il est possible, via des brevets, de limiter l'usage des logiciels libres. Heureusement, les brevets logiciels n'ont pas cours en Europe mais existe aux USA et au Japon.

Exemple : en décembre 1994, Unisys, détenteur de deux brevets sur la compression LZW, a soudainement annoncé que les auteurs de logiciel produisant des images GIF devaient payer des royalties (brevet tombé en 2004 environ). Idem avec le MP3, le H264, H265...

$\longrightarrow$ Apache et GPLv3 ont ajoutés des clauses limitant l'usage des brevets suite à une faille dans les accords de brevets du pacte Novell-Microsoft. L'objectif à terme de la GPLv3 est l'élimination du système de brevet !

Contournement - L'auto-compilation

La licence GPL oblige à donner le code en cas de diffusion. L'idée est donc de diffuser du code propriétaire et c'est l'utilisateur lui même, sur son poste qui fait la dernière étape de compilation avec le linkage. Il n'y a donc pas violation de licence.

Exemple : NVidia avec son driver Linux. Idem ZFS on Linux (licence CDDL).

$\longrightarrow$ Éviter si possible ce genre de produit et de démarche...

Contournement - La mode Saas

On connecte un programme libre modifié sur une interface réseau. Le service est rendu à distance. Dans de nombreux cas, il n'y a pas d'obligation de délivrer le code source.

Exemple : les services en lignes des GAFAM

$\longrightarrow$ Mettre une licence AGPL dès qu'on imagine une utilisation depuis le réseau de son code.

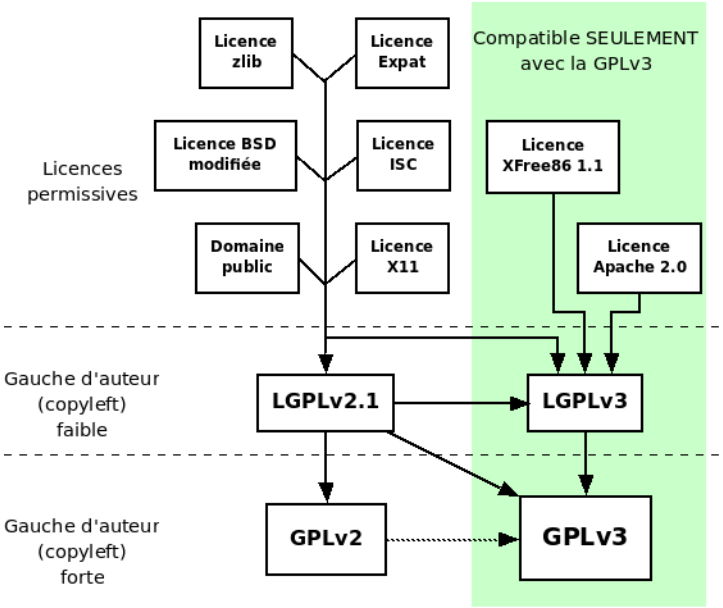
Contournement - Le système des greffons

La société maitresse développe du code libre (GPL) étentu par un système de greffon propriétaire payant. En général, il y a un transfert des droits pour les contributions.

Exemple : GitLab

$\longrightarrow$ Ne pas demander le transfert des droits lors des contributions
$\longrightarrow$ Fige la licence (cf Linux versus projet GNU) s'il y a beaucoup de contributeurs sauf à ajouter dès le début la clause *or later* (confiance envers la FSF).

Compatibilité des licences entre elles ?



C'est bien beau mais à un moment donné, faut choisir sa licence !

C'est pas simple simple...

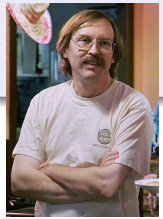
Le pseudo schisme de la GPLv3 (ou pourquoi mettre GPLv2 or later)

Le cas des langages de script

Le cas Perl5

- Licence Perl Artistic (permissive) et GPLv2 or later (copyleft)
- 95% des modules sur CPAN

$\longrightarrow$ Faire pareil !
Prendre la même double licence pour éviter toute question de compatibilité.



Le cas Python

- PSFL (Python Software Foundation License) utilisé pour le langage lui même
- Pour les modules, chacun fait un peu comme il veut !

  `https://docs.python-guide.org/writing/license/`

$\longrightarrow$ Avec Pypy (`pip`), attention aux risques d'auto-compilation.
Rester sur des licences simples et connues (PSFL, Apache, BSD, GPL*).

Le cas Javascript

- `npm` - Artistic License 2.0
- `Node.js` - Licence X11
- Pour les modules, chacun fait un peu comme il veut !

Attention, dans le monde `npm`, on écrit des micro modules parfois d'une ligne.
Il y a souvent des milliers de dépendances.
On est clairement tombé dans le piège de l'auto-compilation.

- Bien pour l'utilisateur $\longrightarrow$ Rien à foutre des licences
- Attention aux pièges de la sécurité et des portes dérobées (arrivent régulièrement)

La sécurité

Packaging / Diffusion

- checksum `md5sum` / `sha512sum`
- signature

Gestionnaire de paquets des distributions $\longrightarrow$ signé
Gestionnaire d'extension de Firefox $\longrightarrow$ non signé !
Gestionnaire de modules des langages $\longrightarrow$ non signé en général

La confiance en prends quand même un coup. Pas d'équipe de ftpmasters pour valider les dépôts...

$\longrightarrow$ Limiter le nombre de dépendances et choisir de préférence des modules connus et très utilisés (donc souvent mieux maintenus)
En Java, il est possible de signer ses programmes

Le problème du linkage statique

- Sécurité des mises à jour ?
- Vérification de la compatibilité des licences ?
  (Oui, on sais que ce n'est pas votre tasse de thé)

$\longrightarrow$ *A priori*, c'est à éviter.

Reproductible Build

- Être capable à partir des sources de produire les mêmes programmes binaires
- Les archives (paquets...) doivent être identiques au bit près

Seule méthode pour avoir confiance dans des paquets binaires
Le Reproductible Build est le complément idéal de la signature et des sommes de contrôles
Gros travail dans la distribution Debian pour avoir un ensemble complet reproductible
C'est impossible à faire si vous n'avez pas accès au code source

$\longrightarrow$ Très bon argument en faveur du logiciel libre
$\longrightarrow$ **Le point à travailler pour les développeurs en 2019**

Confiance dans les plateformes en ligne

- Quid des codes tournant sur un Jupyter distant ?
- Quid des codes tournant AWS ?
- Quid des codes des plateformes comme Parcoursup (même si le source était diffusé) ?

$\longrightarrow$ Il est prouvé mathématiquement qu'il n'y a pas de système de vote en ligne sur !
$\longrightarrow$ Comment le logiciel libre pourrait amener sa brique dans ce débat ?

Les outils du dev

- Faire du libre sur du libre ?
- Utiliser des gestionnaires de codes libres (BitKeeper vs Git) ?
- Utiliser des plateformes libres (GitHub vs GitLab) ?
- Utiliser des compilateurs libres (`ifort` vs `gfortran`)?
- Développer sur un OS libre avec un éditeur libre (`vscode` vs `vscodium`) ?
- ...

$\longrightarrow$ Le logiciel est-il plus libre si l'intégralité de la chaîne est libre ?
$\longrightarrow$ Quid du matériel ?

Autour du développement

Quelques points à creuser un peu plus pour le prochain séminaire

- Les licences libres pour la documentation (Problème de la GFPL)
- Les licences libres pour les schémas, dessins... (Art Libre - copyleft)
- Les licences libres pour l'open data (CC0, Creative-Common-Attribution, Community-Data-License-Agreement permissive ou sharing, Licence-Ouverte...)
- Les licences libres pour les bases de données (Open-Database-License)
- Les licences libres pour les architectures matérielles (carte, montage, plan...) ?

$\longrightarrow$ Autre droit juridique, autres licences... Il n'y a pas une licence qui fonctionne partout !
$\longrightarrow$ Attention à l'anonymisation des données (RGPD)

La position de l'état français

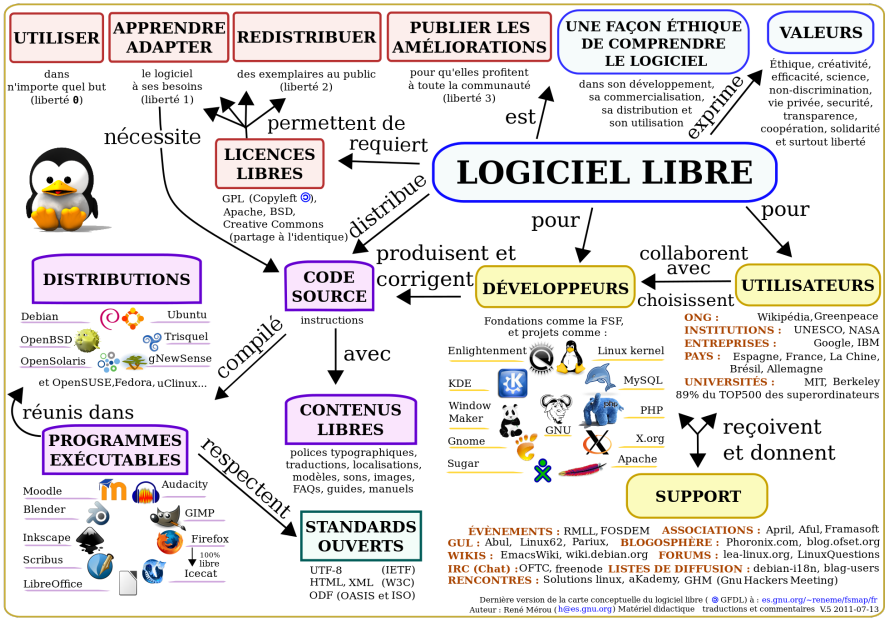
- Soutien aux logiciels libres (par défaut, on doit développer avec des licences libres)
- Forte pression pour de l'Open Data (archive Zenodo...) pour tous les projets Européens
- Fort coup de pousse aux formats libres via le RGI 2.0

$\longrightarrow$ Peut être éviter de se perdre avec les services valorisations !
$\longrightarrow$ Qui a réussi à gagner et faire gagner de l'argent à son unité avec ses logiciels ?

Au départ étaient les compilateurs

- `icc` (`ifort`) - Licence propriétaire
- `clang` (LLVM) - Licence permissive
- `gcc` (`gfortran`, `gnat...`) - Licence copyleft

- Pourquoi pensez-vous qu'Apple et Google pousse très fort sur LLVM ?
- Pourquoi hrefhttps://www.apple.com/Apple évite la GPL comme la peste ?
- En tant qu'établissement public (chercheur, ITA), en cas de choix, vaut-il mieux contribuer à un logiciel propriétaire, permissif ou copyleft ?
- Où le contribuable citoyen s'y retrouve le plus ?


UTILISER
dans n'importe quel but (liberté 0)
APPRENDRE ADAPTER
le logiciel à ses besoins (liberté 1)
REDISTRIBUER
des exemplaires au public (liberté 2)
PUBLIER LES AMÉLIORATIONS
pour qu'elles profitent à toute la communauté (liberté 3)
UNE FAÇON ÉTHIQUE DE COMPRENDRE LE LOGICIEL
dans son développement, sa commercialisation, sa distribution et son utilisation
VALEURS
Éthique, créativité, efficacité, science, non-discrimination, vie privée, securité, transparence, coopération, solidarité et surtout liberté
permettent de
requiert
est
exprime
nécessite
LICENCES LIBRES
GPL, (Copyleft), Apache, BSD, Creative Commons (partage à l'identique)
LOGICIEL LIBRE
distribue
pour
pour
DISTRIBUTIONS
Debian
Ubuntu
OpenBSD
Trisquel
OpenSolaris
gNewSense
et OpenSUSE,Fedora, uClinux...
produisent et corrigent
CODE SOURCE
instructions
DÉVELOPPEURS
collaborent avec
choisissent
UTILISATEURS
compilé
avec
Fondations comme la FSF, et projets comme :
Enlightenment
Linux kernel
KDE
MySQL
Window Maker
GNU
PHP
Gnome
X.org
Sugar
Apache
ONG : Wikipédia, Greenpeace
INSTITUTIONS : UNESCO, NASA
ENTREPRISES : Google, IBM
PAYS : Espagne, France, La Chine, Brésil, Allemagne
UNIVERSITÉS : MIT, Berkeley
89% du TOP500 des superordinateurs
réunis dans
PROGRAMMES EXÉCUTABLES
CONTENUS LIBRES
polices typographiques, traductions, localisations, modèles, sons, images, FAQs, guides, manuels
reçoivent et donnent
respectent
Moodle
Audacity
Blender
GIMP
Inkscape
Firefox
Scribus
100% libre
Icecat
LibreOffice
STANDARDS OUVERTS
UTF-8 (IETF)
HTML, XML (W3C)
ODF (OASIS et ISO)
SUPPORT
ÉVÈNEMENTS : RMLL, FOSDEM
ASSOCIATIONS : April, Aful, Framasoft
GUL : Abul, Linux62, Pariux,
BLOGOSPHÈRE : Phoronix.com, blog.ofset.org
WIKIS : EmacsWiki, wiki.debian.org
FORUMS : lea-linux.org, LinuxQuestions
IRC (Chat) : OFTC, freenode
LISTES DE DIFFUSION : debian-i18n, blag-users
RENCONTRES : Solutions linux, aKademy, GHM (Gnu Hackers Meeting)
Dernière version de la carte conceptuelle du logiciel libre (GFDL) à : es.gnu.org/~reneme/fsmap/fr
Auteur : René Mérou (h@es.gnu.org) Matériel didactique traductions et commentaires V.5 2011-07-13

Licence

Merci à Guillaume Mella pour m'avoir entraîné dans cette histoire

Cette présentation est sous : Licence Art Libre

http://artlibre.org/