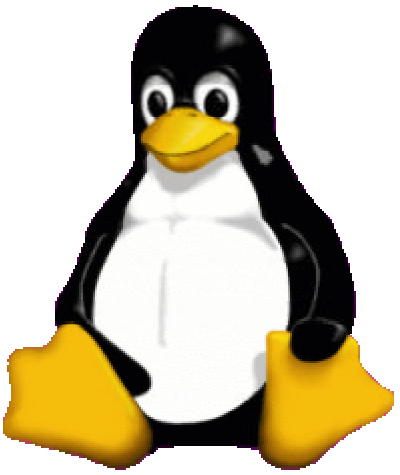
Université Robert Schuman – IECS Strasbourg
Formation Internationale à la Gestion



Mémoire de fin d'études :
Les logiciels libres - un modèle économique viable?

Réalisé par
Jean-Claude Héritier

sous la direction de
Catherine Ledig

Année Universitaire 2000-2001

Sommaire

Introduction :

L'industrie logicielle connaît depuis les années 60 un développement de plus en plus important, accompagnant d'abord la révolution informatique puis le développement d'Internet. Ce secteur s'est progressivement concentré autour de quelques acteurs majeurs. En plus de services, leurs stratégies se basent sur la vente de licences d'exploitation de leur produits, ces derniers étant protégés du mieux possible contre toute récupération, au nom de la propriété intellectuelle. Ce modèle ayant permis à certaines sociétés de réaliser des profits immenses et de contrôler en partie le secteur logiciel, se voit depuis quelques années opposé à un autre modèle de développement logiciel. Face aux logiciels propriétaires conçus en interne dans un but lucratif et extrêmement protégés, une communauté de programmeurs bénévoles dispersée dans le monde entier et uniquement reliée par l'Internet a développé des logiciels concurrents de façon radicalement différente. En effet, un idéal d'ouverture du code de programmation et de sa mise à la disposition gratuite pour tous et pour tous usages a engendré un modèle de développement logiciel totalement différent, lui-même à l'origine d'un nouveau modèle économique à la fois pour les éditeurs de logiciels, les entreprises et même pour certains Etats. Appelés logiciels « libres », leur croissance spectaculaire amène à se poser plusieurs questions tant ils vont à l'encontre des règles traditionnelles du développement logiciel mais surtout du modèle économique basé sur les logiciels propriétaires au code source fermé et protégé.

D'une part, comment expliquer le succès d'un mode de développement apparemment totalement désorganisé, sans hiérarchie ni contraintes d'aucune sorte sur ses membres, eux-mêmes totalement libres dans leurs contribution? Comment le travail collectif d'individus isolés, libérés de toute structure peut-il s'avérer performant au point de dépasser les modèles d'organisation extrêmement étudiés des industriels du logiciel?

D'autre part, comment ces logiciels peuvent-ils s'avérer performants et compétitifs dans le monde économique, pour lequel ils ne sont apparemment pas étudiés? Comment peuvent-ils présenter un quelconque avantage face à des produits soutenus par des structures commerciales très puissantes? Nous verrons qu'en plus d'offrir souvent un avantage technique, stratégique et financier, ces logiciels peuvent avoir des implications bien plus profondes dans la vie économique, tant au niveau des entreprises (éditrices ou utilisatrices) qu'au niveau des administrations ou même d'états entiers.

Ce mémoire a pour objectif d'expliquer comment les logiciels libres apportent un nouveau modèle d'organisation révolutionnaire, basé sur la mise en réseau du monde entier et sur une gestion particulière des rapports humains, et ensuite de montrer à quel point ils risquent de modifier considérablement l'économie logicielle et la stratégie de ses nombreux acteurs.

Pour cela, nous verrons d'abord en détail ce que sont les logiciels libres, d'où ils viennent et sur quels cadres juridiques ils s'appuient. Ensuite, dans une seconde partie,

comment l'on peut expliquer la réussite du modèle de développement libre au niveau technique, d'où provient la performance d'un tel type d'organisation autogéré et laissé suivre une sorte d'évolution « darwiniste ». Pour suivre le sujet de ce mémoire, la troisième partie expliquera comment le logiciel libre peut pourtant s'avérer transposable à la sphère économique, tant pour les éditeurs logiciels que leurs clients, et même constituer une opportunité politique au niveau d'Etats. La quatrième partie sera davantage une réflexion sur l'avenir envisageable à long terme pour ce modèle, avec les potentialités qu'il recèle, mais également les menaces qui peuvent peser sur lui avec son développement économique.

I) Qu'est-ce qu'un logiciel libre : cadre juridique et historique du logiciel libre

Ce mémoire s'articule autour du logiciel libre. Il est donc nécessaire de commencer par en préciser la définition, mais aussi les cadres juridiques et historiques dans lesquels il s'inscrit.

1- définition d'un logiciel libre

De façon formelle, un logiciel se définit comme *« l'ensemble des programmes, procédés et règles et éventuellement de la documentation, relatifs au fonctionnement d'un ensemble de traitement de données »* (http://www.ccip.fr/irpi/faq/logiciel/definition.htm). Plus généralement, on pourra réduire ici cette définition à l'ensemble des instructions logiques indispensable au fonctionnement d'un ordinateur. Avec la démocratisation de l'informatique et la démultiplication de ses effets au travers de l'Internet, l'importance des logiciels n'a fait que croître, depuis leur apparition dans les années 60 jusqu'à la société de l'information que nous connaissons aujourd'hui.

Si n'importe qui peut concevoir un logiciel avec peu de moyens (un ordinateur et du savoir faire), la conception logicielle est aujourd'hui devenue une industrie très importante et dominée par quelques acteurs majeurs dont les logiciels sont dits « propriétaires ». En effet ces sociétés conservent l'entière propriété intellectuelle de leurs logiciels, distribués sous forme compilée (binaire exécutable) et leur utilisation est soumise à des conditions très précises. Se définissant par opposition au logiciel « propriétaire », le "logiciel libre" se rapporte à la liberté des utilisateurs d'exécuter, de distribuer, d'étudier, de modifier et d'améliorer le logiciel au travers de son code source, l'ensemble des lignes de codes qui une fois compilées en langage machine, donneront le logiciel fini. Ces logiciels sont d'ailleurs généralement distribués sous forme de lignes de codes que l'utilisateur compilera au moment de l'installation. Plus précisément, on distingue trois niveaux de libertés pour le logiciel libre:

- La liberté d'exécuter le programme, pour tous les usages, ce qui est le degré minimum de liberté qu'on peut attendre d'un logiciel.
- la liberté d'étudier comment le programme fonctionne et de l'adapter à ses besoins.
- la liberté de redistribuer des copies sans aucune contrainte.
- la liberté d'améliorer le programme et de diffuser ses améliorations publiquement, de telle sorte que la communauté toute entière en bénéficie.

La liberté de modifier et de distribuer des logiciels libres sans payer de droits d'auteurs est fondée par un contrat de licence de type copyright. Les contrats de licence les plus connus sont le contrat GPL (GNU Public License), le contrat Berkeley et plus récemment le contrat NPL (Netscape Public License). Pour comprendre la légitimité du logiciel libre, il est nécessaire d'étudier les différentes licences sur lesquelles se base l'économie logicielle dans son ensemble.

2- les différents types de licences

A) Spécificités de l'économie logicielle :

L'économie logicielle présente des caractéristiques particulières par rapport à d'autres secteurs plus traditionnels, comme nous le verrons dans la partie III)1)B) p37. En effet, la valeur d'un logiciel reste essentiellement intellectuelle, vu que sa production et encore plus sa distribution ne demandent que des investissement réduits par rapport aux coûts liés au développement, qui emploie parfois des milliers de programmeurs simultanément. Le fort contenu intellectuel des logiciels est généralement protégé par de nombreuses licences, accordant différents droits selon le but visé par la version concernée. Un shareware qui aura pour but de faire découvrir le logiciel verra sa distribution et sa copie autorisée, mais pas son utilisation au delà d'une période d'essai. A l'opposé un logiciel propriétaire commercial, comme un système d'exploitation, un progiciel spécialisé ou un format de stockage de données, pourront se voir très contrôlés au niveau de la distribution et même de l'utilisation en entreprise. Par exemple utiliser certains logiciels propriétaires sur plus de machines que ne l'autorise sa licence, ou avec des périphériques particuliers peut entraîner l'annulation du contrat, en clair l'interdiction d'utiliser plus longtemps le logiciel.

Ces contrats de licence ont d'ailleurs de nombreux aspects très particuliers. Les éditeurs logiciels cèdent des licences d'exploitation propriétaires permettant à leur client d'utiliser sous certaines conditions leur produit. En aucun cas les clients ne possèdent le logiciel, seulement le droit de l'utiliser. De plus aucune garantie quant à son fonctionnement n'est présente, et en cas de perte de données ou de panne liée au logiciel, le client ne pourra se retourner contre l'éditeur, comme le montre un extrait de licence tout à fait courant ci-dessous. Ce qui pourrait sembler abusif dans d'autre secteurs d'activité est ici justifié juridiquement par le caractère « artistique » des logiciels, qui en tant qu'œuvres ne peuvent donc être associés à aucune garantie. La responsabilité de l'éditeur n'est donc jamais engagée, seule sa caution morale certifiant que le logiciel est fiable peut rassurer le client.

Extrait d'un contrat de license propriétaire (Netscape 4.7) :

> *"…is not responsible for any damages whatsoever, including loss of information, interruption of business, personal injury and/or any damage or consequential damage without limitation, incurred before, during or after the use of our products. Our entire liability, without exception, is limited to the customers' reimbursement of the purchase price of the software"*

Les éditeurs logiciels apportent donc une très grande attention aux droits accordés par leurs licences, qui garantissent leurs intérêts de la meilleure façon possible tout en dégageant leur responsabilité en cas de problème.

B) Les différents types de logiciels et leurs licences :

On peut découper l'ensemble des logiciels en trois grandes familles de licences: les logiciels libres, les logiciels semi-libres, et les logiciels propriétaires.

* La famille des logiciels libres :

Un logiciel libre permet à tous de l'utiliser, de le copier, de le distribuer et de le modifier librement. Un logiciel est qualifié de «libre» parce que son accès est libre, sans relation avec le prix. Cela signifie que son code source est disponible, et que des sociétés commerciales peuvent éventuellement en tirer profit en le distribuant ou en offrants des services associés comme par exemple le soutien aux utilisateurs. En pratique, un logiciel libre peut se trouver gratuitement ou à très bas prix, par exemple en téléchargement. Il faut comprendre que la gratuité n'est qu'une conséquence de la liberté du code source, mais pas une obligation comme le rappelle la Free Software Foundation. De nombreuses sociétés commerciales sont d'ailleurs éditrices de logiciels libres, comme nous le verrons dans la partie III)1)A p36.

> *« L'anglais utilise le même mot « free » pour « libre » et « gratuit ». C'est pourquoi il y a souvent confusion sur la nature des termes "free software". Nous tenons à souligner qu'il ne s'agit pas du prix mais de la liberté d'utilisation. »*
>
> *(http://www.gnu.org/philosophy/categories.fr.html)*

Voyons en détails les différentes licences des logiciels libres (sources : http://www.opensource.org/licenses/) :

LOGICIEL COPYLEFTÉ (libre même si modifié par les distributeurs) :

Ce type de logiciel est un logiciel libre, dont les conditions de distribution interdisent aux distributeurs d'y ajouter des restrictions d'utilisation, même s'ils y ont apporté des modifications. Ceci veut dire que chaque copie du logiciel, même modifié, doit être un logiciel libre.

Par opposition au copyright, ce type de licence (la plus courante du «libre») vise à éviter l'apparition de restrictions quant au statut du logiciel au cours de son développement. Une fois qu'un logiciel est copylefté, ses évolutions ne pourront être revendiquées par qui que ce soit, et son code source restera toujours ouvert et librement modifiable. Le concept de copyleft est un concept général. Pour l'appliquer à un programme, il existe un ensemble de termes relatifs à la distribution, ce qui fait que concrètement il existe plusieurs façon d'écrire les conditions de distribution. Seul l'objectif reste le même.

Un cas particulier : LOGICIEL COUVERT PAR LA GPL :

La «GNU-GPL (Licence Publique Générale) (20 K caractères) » (appellation officielle) est un ensemble spécifique de conditions de distribution pour copylefter un programme. Le projet GNU (système d'exploitation complet, inspiré d'UNIX mais différent de Linux, et libre) l'utilise pour la distribution de la plupart des logiciels GNU, dont le célèbre Linux qui en est dérivé. **Une grande majorité des logiciels dits « libres » est en fait sous licence GPL.**

Quelques logiciels libres :
Les systèmes d'exploitation FreeBSD (licence BSD) et Linux (GPL),
le programme de manipulation d'images GIMP (GPL)

En résumé,

- un logiciel « libre » (plus précisément copylefté sous GPL) est librement distribuable, c'est-à-dire que n'importe qui a le droit de le distribuer avec son code source,
- logiciel libre ne signifie pas domaine public. Autrement dit, n'importe qui a le droit de faire payer le prix qu'il veut (si il trouve des personnes pour lui acheter),
- qu'il soit distribué gratuitement ou non, un logiciel libre doit être distribué avec son code source,
- un logiciel libre peut être modifié en toute liberté, le logiciel modifié étant également un logiciel libre.

LOGICIEL LIBRE SOUMIS A RESTRICTIONS (« non-copylefté ») :

Le logiciel libre soumis à restrictions, est défini par l'auteur avec la permission de le redistribuer, de le modifier, et d'y ajouter d'autres restrictions.

Si un programme est libre, mais soumis à restrictions, certaines versions modifiées peuvent ne plus être «libres» du tout. Une société informatique peut compiler un programme, avec ou sans modifications, et dstribuer le fichier exécutable, en tant que son propre produit payant et dont la redistribution, ou les modifications ne peuvent être réalisées sans son accord formel.

Le système X window (gestion de l'interface graphique : menus, fenêtres… pour Unix) en est un exemple. Le consortium X distribue le X11 avec des

conditions de distribution telles qu'il en fait un logiciel libre mais soumis à restrictions. L'accès à des copies d'X11 est libre. Toutefois, il existe aussi des copies «non libres», et il existe également des stations de travail, ainsi que des cartes graphiques PC pour lesquelles seules des versions non libres fonctionnent. Dans le cas d'utilisation de ce matériel spécifique, X11 n'est alors plus un logiciel libre.

(sources sur X window : http://www.urec.cnrs.fr/cours/Applis/x11/sld002.htm)

Cas particulier : LOGICIEL DU DOMAINE PUBLIC

Logiciel du domaine public veut dire logiciel non soumis aux droits d'auteurs. C'est un cas spécial du logiciel libre «non-copylefté», ce qui veut dire que certaines copies, ou certains versions modifiées, ne sont pas du tout gratuites. La différence avec un logiciel freeware est que le code source est disponible, généralement pour en permettre l'étude. Les seuls logiciels du domaine public sont ceux qui y ont été explicitement placés par leurs auteurs, en partie à cause du développement récent de l'informatique, et surtout du vieillissement rapide des logiciels utiles.

Parfois, le terme «du domaine public» est utilisé d'une façon large pour dire «libre» ou «disponible gratuitement». Cependant, «domaine public» est un terme légal qui signifie avant tout que le logiciel n'est pas soumis à des droits de copyright. Personne ne peut donc en demander des royalties, sauf en cas de modification significative.

On trouve des exemples de logiciels du domaine public sur
http://www.idris.fr/su/Produits_Publics/tab_Prod_public.html

LOGICIEL SEMI-LIBRE :

Le logiciel semi-libre n'est pas un logiciel entièrement libre, mais y sont autorisés : l'utilisation, la copie, la distribution, la modification, (y compris la distribution des versions modifiées), à condition que ce soit dans le cadre d'un usage privé, et à des fins non lucratives. P.G.P. est l'exemple d'un programme semi-libre. Son utilisation commerciale nécessite de payer une licence à Network Associates International, alors que le code source en est librement utilisable et vérifiable. Des versions « copyleftées » en ont d'ailleurs été dérivées telles que GnuPG.

PGP (semi-libre)
et sa version copyleftée **GnuPG**
(www.pgp.com)
(www.gnupg.org)

⇔

*** En face, la famille des logiciels propriétaires :**

La majorité des logiciels reste encore propriétaire. Le logiciel propriétaire n'est par définition ni libre, ni semi-libre. Son utilisation, redistribution ou modification sont interdites, ou exigent une autorisation spécifique. Ses conditions d'utilisation sont généralement tellement restrictives, qu'une utilisation libre est impossible. Chaque usage possible du logiciel est détaillé dans sa licence d'exploitation. Avant tout, l'auteur conserve la plupart des droits associés au logiciel.

On distingue là aussi plusieurs types de licences propriétaires :

FREEWARE (ou *graticiel* en français) :

Un logiciel Freeware n'est pas soumis au paiement d'une licence ou de quoi que ce soit (sauf parfois pour une utilisation professionnelle), son utilisation et sa distribution sont encouragées. Le but d'un logiciel mis sous une licence de type Freeware est évidemment de gagner un grand nombre d'utilisateurs, ce qu'une version payante ne pourrait faire. Par contre l'auteur conserve les sources de son programme et tous les droits sur ce dernier. Il est donc le seul à pouvoir le modifier, voire en exploiter des développements ultérieurs. C'est le cas de logiciels très connus comme Winamp ou Napster, dont le succès se base avant tout sur leur part de marché et la base d'utilisateurs actifs. Les sociétés les exploitant les offrent gratuitement en se rétribuant sur d'autres services associés qui sont alors payant, comme la publicité sur le site de téléchargement ou l'intégration de fonctionnalités préférentielles dans le freeware.

Quelques Freewares : Netscape, Napster, Winamp

En résumé pour les freewares :

- Aucun accès au code source
- Droit d'utiliser gratuitement
- Droit de distribuer gratuitement
- **L'auteur original conserve tous les autres droits**

SHAREWARE (ou *partagiciel*) :

Le *shareware* est un logiciel dont l'utilisation est soumise au paiement de royalties à l'éditeur. Les Shareware ne sont pas des logiciels libres ou même semi-libres pour deux raisons:

- Pour les *shareware*, le code source n'est pratiquement jamais fourni et le programme n'est donc pas modifiable, sauf par son auteur.

- Avec le *shareware*, il n'est pas permis d'effectuer de copie du logiciel et pour conserver le logiciel installé, une licence doit être achetée pour rester dans la légalité, y compris pour des activités non lucratives (en réalité de nombreuses personnes utilisent ces logiciels sans payer, mais ce n'est pas permis, seulement toléré de manière officieuse).

La license Shareware a pour but de permettre aux utilisateurs potentiels de tester le logiciel, avec ou sans restrictions, afin que ceux-ci puissent en apprécier les fonctionnalité et l'achètent ensuite. Parmi les célèbres Sharewares, on compte par exemple l'utilitaire de compression Winzip, l'éditeur HTML Webexpert, des clients FTP, des logiciels de gravure CD… Tous les domaines sont en fait concernés par le Shareware, qui reprend l'idée du Freeware de conquête de marché, tout en devenant payant en cas d'utilisation prolongée.

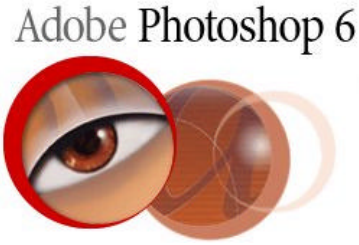
Quelques sharewares : Nero Burning Rom, Winzip, Webexpert 2000

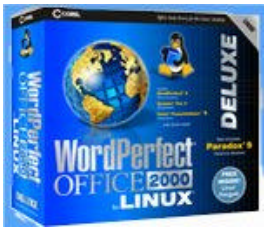
En résumé pour les sharewares :

- Aucun accès au code source
- Droit de distribuer gratuitement
- Droit d'utilisation limité (durée, nombre, prix…)
- **L'auteur original conserve tous les autres droits**

LOGICIEL COMMERCIAL :

Le logiciel commercial est un logiciel développé par une entreprise dont le but est de se faire payer par l'utilisateur. Les logiciels commerciaux traditionnels sont placés sous le contrôle de licences d'utilisation variées dont l'objectif est d'en limiter la reproduction et les conditions d'utilisation. Légalement, l'utilisation d'un partagiciel sans paiement de la licence est souvent considéré comme assimilable à une violation du droit d'auteur.





Quelques logiciels propriétaires :
Adobe Photoshop 6
Microsoft Windows et Office
Corel Wordperfect Office 2000 **(pour Linux)**

. **En résumé pour les logiciels propriétaires :**

- Aucun accès au code source
- Aucun droit de copie, excepté comme sauvegarde (et non utilisée simultanément)
- Aucun droit de redistribution
- Droit d'utilisation très limité (nombre, prix, lieu…)
- **L'auteur original conserve tous les autres droits**

NB : La plupart des logiciels commerciaux sont propriétaires, et de nombreuses personnes considèrent les deux termes comme synonymes. Mais c'est une erreur car le logiciel propriétaire n'est pas toujours commercial, et le logiciel libre peut être commercial.

Par exemple, le compilateur **GNU Ada** est toujours distribué sous les termes de la GNU GPL (voir I) 3 B p.15), et chaque copie est libre; mais ses développeurs vendent des contrats de support. Concrètement, si certains clients peuvent préférer un compilateur commercial qui leur semble plus sûr qu'un logiciel libre, le vendeur du logiciel pourra répondre que GNU Ada est un compilateur commercial mais que c'est également un logiciel libre.

Ce statut particulier des logiciels libres distribués de façon commerciale ne pose aucun problème à la communauté des développeurs « libres », généralement

pointilleuse au niveau des licences, au point de développer à partir de rien des versions entièrement libres d'applications, comme le gestionnaire de fenêtres Gnome pour Linux (car KDE utilisait des librairies Qt alors non libres). En effet, vu que la vente de ce produit reste secondaire au fait qu'il soit librement disponible et modifiable, l'élargissement de l'audience du logiciel libre par ce développement commercial est plutôt considérée comme bénéfique, comme nous le verrons en détail plus loin

NB : D'autres licences propriétaires, parfois étonnantes, existent également. Un *Cardware* ne demande à son utilisateur que d'envoyer une carte postale à son auteur, comme STUDIOS qui est un éditeur de diaporamas (http://www.multimania.com/studios)… Mais ces autres catégories restent anecdotiques.

3- le cadre juridique des logiciels libres :

Comme nous l'avons vu, les logiciels libres ou non obéissent à des licences plus ou moins ouvertes. Dans leur cas, le système des droits d'auteurs a été "détourné", inversé, pour favoriser leur développement au travers de leur ouverture et de sa garantie. Ces logiciels sont munis d'une licence qui, au lieu de limiter la diffusion ou l'usage du logiciel, protège au contraire ces droits au bénéfice de l'ensemble du public (les restreignant parfois uniquement pour certains types d'utilisations commerciales). Un inconvénient de cette très grande liberté, presque équivalente à celle du domaine public, est que ces logiciels pourraient se voir détournés en version propriétaire commerciale, en abusant de la popularité de la version libre et du travail des auteurs originaux, qui seraient alors eux-mêmes privés des améliorations apportées à leur œuvre. Pour garantir qu'un logiciel placé sous licence « libre » ne puisse d'une part être récupéré et devenir propriétaire, et d'autre part que son code source ne soit éclaté en parties libres et non-libres selon le bon vouloir des contributeurs, un cadre juridique très clair a été mis en place dès le début du développement en Open Source.

Pour comprendre sur quelles règles juridiques se base le logiciel « libre », il est nécessaire de décrire le rôle de la **Free Software Foundation**, de la **licence GNU-GPL**, ainsi que l'effet fédérateur du **projet GNU** que ces deux dernières ont amorcé. Il est à noter que si d'autres bases juridiques existent pour d'autres catégories de logiciels libres, la licence GPL est devenue la référence pour l'immense majorité des programmes en Open Source.

a) la Free Software Foundation

La FSF (Free Software Foundation) a été fondée au début des années 80 par Richard M. Stallman, alors chercheur au laboratoire d'Intelligence Artificielle du MIT. Comme son nom l'indique, le but de cette fondation est de développer des logiciels libres, pouvant être légalement copiés, utilisés, modifiés et redistribués sans la moindre contrainte, les sources de ces logiciels devant rester disponibles gratuitement pour quiconque en fait la demande.

La tête de Gnou,
symbole de la Free Software Foundation et du projet GNU (Gnu is Not Unix)

Il est important de comprendre que le mot Free dans Free Software Foundation ne doit pas être traduit comme gratuit mais bien comme libre. Ces logiciels peuvent tout à fait être vendus et exploités à but commercial, mais il existe toujours un moyen légal de se les procurer gratuitement.

> « Lorsque nous parlons de free software, nous entendons free dans le sens de liberté, et non pas de gratuité. Notre licence est conçue pour s'assurer que vous avez la liberté de distribuer des copies des programmes, gratuitement ou non, et que vous recevez ou pouvez obtenir le code source, que vous pouvez modifier les programmes ou en utiliser des parties dans d'autres programmes libres, en sachant que vous pouvez le faire. » (http://www.gnu.org/philosophy/selling.fr.html)

La Free Software Foundation est à l'origine de la licence GPL.

b) **La license GPL** (General Public License)

La licence GPL (General Public License) est une licence qui spécifie les conditions de distribution de tous les logiciels GNU. La LGPL (Library General Public License) est son équivalent pour les bibliothèques de sous-programmes.

Ces licences spécifient que les logiciels GNU peuvent être copiés, modifiés et redistribués de quelque manière que ce soit, aussi longtemps que les sources sont disponibles gratuitement. Une traduction française de la GPL est disponible sur http://www.cam.org/~trot/cpa/gpl_fra.html et en annexe p64 tandis que l'originale en anglais se trouve sur http://www.gnu.org/copyleft/gpl.txt .

Le gros avantage des logiciels distribués selon ces conditions est que si quelqu'un désire les améliorer, tout d'abord il pourra le faire, mais également distribuer (gratuitement ou non) la nouvelle version. De ce fait, tout le monde en profitera et pourra à son tour l'améliorer. Cette méthode de développement conduit à d'excellents programmes écrits par des dizaines de personnes différentes et crée ainsi un cercle vertueux dans le développement du logiciel. Personne ne peut s'approprier un programme placé sous GPL.

Extrait de la GPL :

> « Afin de protéger vos droits, nous devons faire des restrictions qui interdisent à quiconque de vous refuser ces droits ou de vous demander d'y renoncer. Ces restrictions vous imposent par conséquent certaines responsabilités si vous distribuez des copies des programmes protégés par la Licence Publique Générale ou si vous les modifiez.

> Par exemple, si vous distribuez des copies d'un tel programme, gratuitement ou non, vous devez transmettre aux utilisateurs tous les droits que vous possédez. Vous devez vous assurer qu'ils reçoivent ou qu'il peuvent se procurer le code source. Vous devez leur montrer cette licence afin qu'ils soient eux aussi au courant de leurs droits. »

Le cadre posé par la GPL l'a été pour permettre d'entamer le projet GNU et de garantir son maintien en tant que système basé sur le logiciel libre. Ainsi on ne peut dissocier la Free Software Foundation, la licence GPL et le projet GNU.

c) Le projet GNU

Le projet GNU (nommé ainsi par clin d'œil aux programmeurs qui apprécient la récursivité de l'acronyme : GNU = GNU is Not Unix) est un projet de la Free Software Foundation dont le but est de développer un système d'exploitation complet et entièrement libre, comprenant donc également tous les outils associés et nécessaires à son élaboration, distribué selon les conditions de la GPL. Ce système d'exploitation reprend un certain nombre de concepts de UNIX mais ce n'est pas UNIX. S'il ressemble à UNIX, c'est principalement du fait que dès ses débuts ATT a rendu le code source de son système UNIX accessible dans les grandes université américaines, mais sans en permettre l'utilisation (voir p.16). Richard Stallman a commencé ce projet lui-même, juste après avoir créé la FSF.

Richard Stallman
créateur de la Free Software Foundation, de la licence GPL, et des outils GCC et EMACS **(« le père du logiciel libre »)**.

Pour le mener à bien, il a d'abord mis au point les outils indispensables à la création logicielle et les a placés sous licence GPL. La première partie a consisté à écrire un éditeur avec lequel il puisse éditer ses programmes, le célèbre GNU EMACS (« Editor MACroS »). Ensuite, il a écrit un compilateur C (le logiciel GCC) pour pouvoir compiler son système d'exploitation, c'est à dire le transformer en langage binaire adapté à la machine. Depuis lors, un certain nombre de personnes se sont jointes à lui pour écrire toutes sortes de programmes. Le système d'exploitation du projet GNU en lui-même, nommé HURD, est disponible depuis peu. Le projet GNU est aussi associé à d'autres systèmes d'exploitation, clones d'Unix et soumis aux conditions de la GPL, dont le plus médiatisé est Linux. Cependant, bien que proches techniquement et philosophiquement, ainsi que compatibles grâce aux bibliothèques GNU C *«conformes aux standards ANSI/ISO, BSD, POSIX, Single Unix, SVID et X/Open»*, GNU-Hurd et GNU-Linux sont deux systèmes distincts.

GNU-Hurd :
le système d'exploitation du projet GNU

En plus des principaux logiciels GNU, il existe des versions GNU de la plupart des utilitaires UNIX. Ces versions n'ont souvent rien à envier à leurs équivalents propriétaires.

4- Historique du logiciel libre et principaux logiciels « libres »

Bien avant que l'Internet et la mise en réseau progressive des ordinateurs du monde entier ne démultiplie l'importance des logiciels et permette le travail collaboratif à la base du logiciel libre, il existait un besoin de faire fonctionner les ordinateurs sur la base de programmes librement disponibles pour tout le monde. Ce désir de liberté provenait d'une réaction contre les logiciels propriétaires issus des supercalculateurs, et demande un petit rappel historique.

a) Le modèle Unix :

A la fin des années 60, bien qu'IBM soit alors le plus grand vendeur d'ordinateurs généralistes, la compagnie American Telephone & Telegraph, (AT&T) détenant le monopole du téléphone aux Etats-Unis, était de taille encore plus grande et utilisait ses propres outils informatiques (matériels et logiciels) en interne. Les Bell Labs, département de la recherche d'AT&T ont donné naissance à un système d'exploitation appelé Unix. L'idée d'Unix était de créer un système d'exploitation simple, s'adaptant à toutes les échelles, pour tous les ordinateurs, des petits aux grands supercalculateurs qu'AT&T construisait pour son propre usage. Pour atteindre ce but, il fallait écrire un système d'exploitation d'un nouveau type, c'est à dire ni dans un langage machine ni dans un langage assembleur dont la forme reste liée au matériel utilisé, mais dans un langage plus expressif et généraliste. Le langage retenu était aussi une invention des Bell Labs, appelé «C » . Le langage C, alors révolutionnaire, s'est depuis généralisé et est même devenu dominant pour de nombreuses types de programmation. A la fin des années 1970, le système d'exploitation Unix écrit en C a été porté sur des ordinateurs de nombreux constructeurs et de conceptions variées.

En effet, AT&T a largement distribué Unix, et en raison de la conception même du système d'exploitation (multi-plateformes), la compagnie devait effectuer cette distribution sous forme de code source C. Si le client avait des problèmes avec Unix (bug, fonctionnalité manquante), c'était à lui de plonger dans le code source afin d'y apporter les modifications nécessaires. Mais AT&T a conservé la propriété du code source et a contraint les utilisateurs à acheter des licences qui ont interdit la redistribution et la création de travaux dérivés. Les gros centres informatiques, industriels ou académiques, pouvaient se permettre d'acheter de telles licences, mais pas les individus qui en appréciaient pourtant les nombreux avantages. En même temps, les restrictions des licences interdisaient aussi à la communauté des utilisateurs

qui utilisaient Unix de l'améliorer autrement que de façon épisodique. Et comme les programmeurs à travers le monde commençaient à aspirer à une révolution de l'ordinateur personnel (et même à l'attendre), le statut « non libre » d'Unix a commencé à devenir une source de problèmes.

b) Les premiers systèmes libres :

L'apparition des logiciels libres remonte apparemment à l'habitude universitaire de mettre à la disposition de toute la communauté les résultats théoriques ou expérimentaux des recherches. Cette habitude s'est logiquement étendue aux logiciels, produits très tôt en milieu universitaire vu leur fort contenu théorique. Depuis longtemps, les résultats des recherches universitaires sont utilisés en dehors de cet environnement et généralement sous forme propriétaire, en particulier dans l'industrie.

Cependant, certains auteurs universitaires de logiciels se sont progressivement convaincus que, même sans les importantes structures associées à la production industrielle et commerciale, ils étaient capables de produire des logiciels de qualité comparable et pouvant rivaliser avec leurs concurrents "professionnels".

Richard Stallman, alors employé au laboratoire d'intelligence du MIT, fut le premier à imaginer un projet de reconception et d'amélioration d'un système d'exploitation composé de vrais logiciels libres, basé sur le partage de connaissances et la coopération entre programmeurs. La raison d'être du logiciel libre serait la liberté pour tous de modifier et redistribuer de tels logiciels, avec pour seule restriction de ne pas réduire les droits de ceux à qui ils seraient redistribués. La gratuité ne faisait pas partie de ses objectifs, bien qu'elle soit une conséquence pratique de la liberté du code source. Par ce système d'ouverture, Stallman voulait que le logiciel libre puisse devenir un projet auto-organisé, s'enrichissant de lui-même, dans lequel aucune amélioration ne serait perdue pour d'autres à travers les copyrights. Ce système a été nommé GNU, signifiant *« GNU is Not Unix »* pour marquer la différence fondamentale entre ces deux systèmes pourtant similaires sur de nombreux points. Malgré des doutes sur la conception fondamentale d'Unix (système *« pas trop mauvais »* d'après Stallman) et sur ses conditions de distribution restrictives, GNU était conçu pour bénéficier de la large (bien que non libre) distribution de sources d'Unix, connues par un grand nombre de programmeurs. Richard Stallman a commencé le projet GNU en écrivant lui-même des composants du système final qui étaient aussi conçus pour fonctionner sans modification sur les systèmes Unix existants. Le développement des outils GNU pouvait ainsi se faire directement dans l'environnement des universités et des autres centres de calcul avancé à travers le monde.

Pour mener a bien un projet d'une telle ampleur, il fallait parvenir à recruter, organiser et motiver un grand nombre de programmeurs volontaires qui construiraient tous les outils nécessaire au système d'exploitation. Pour cela, des cadres juridiques clairs ont dès le début été mis en place comme nous l'avons vu avec la Free Software Foundation et la licence GPL. Cependant ce mode de production ne s'est

véritablement exprimé qu'à partir de 1991 avec le projet de Linus Torvalds, un étudiant en informatique de l'université d'Helsinki, qui a commencé le projet Linux et a vraiment dynamisé la vision et l'énergie du logiciel libre.

c) l'arrivée de Linux

Linus Torvalds a en fait commencé à adapter un outil informatique pédagogique à un usage réel. Le noyau MINIX d'Andrew Tannenbaum était une base de cours sur les systèmes d'exploitation, fournissant des exemples de solutions de base à des problèmes de base. Lentement, et d'abord sans le reconnaître, Linus Torvalds a commencé à transformer le noyau MINIX en un vrai noyau Unix pour les processeurs Intel x86, qui fonctionnent toujours sur les ordinateurs personnels de base du monde entier. Au fur et à mesure du développement personnel de son noyau, qu'il appela Linux, il réalisa que la meilleure manière de faire fonctionner le projet était d'ajuster ses décisions liés à la conception du système afin que les composants GNU (donc libres) soient compatibles avec son noyau.

Linus Torvalds : l'initiateur de Linux

Le résultat du travail de Torvalds aboutit, en 1991, à la distribution sur l'Internet d'une esquisse de modèle fonctionnel d'un noyau libre pour un système d'exploitation sur PC semblable à Unix, entièrement compatible, et conçu de manière à pouvoir reprendre l'énorme ensemble de composants systèmes de haute qualité créés par le projet GNU de Stallman et distribués par la Free Software Foundation. Comme Torvalds avait décidé de distribuer le noyau Linux sous la Licence Générale Publique (GPL) de la Free Software Foundation, les centaines et finalement milliers de programmeurs à travers le monde qui ont décidé de contribuer par leurs efforts au développement futur du noyau étaient assurés que leurs efforts auraient pour résultat un logiciel perpétuellement libre, que personne ne pourrait transformer en produit propriétaire. Les contributeurs agissaient en sachant que leur travail serait accessible, améliorable et redistribuable. Torvalds acceptait les contributions volontiers, et grâce à son style de direction efficace (analysé en détail dans la partie II)1) le projet a maintenu son unité et l'enthousiasme des développeurs. Le développement du noyau Linux a prouvé que l'Internet rend possible le travail collaboratif d'ensembles de programmeurs bien plus grands que n'importe quel éditeur commercial ne pourrait se le permettre, et de plus rassemblés de manière pratiquement non hiérarchique dans un projet à grande échelle (plus d'un million de lignes de code source pour le seul noyau Linux). Comme le

rappellent de nombreuses analyses sur le mouvement Open Source, une telle échelle de collaboration entre des volontaires non payés et dispersés géographiquement, n'avait auparavant jamais eu lieu dans l'histoire humaine.

En 1994, Linux a atteint la version 1.0, représentant un noyau utilisable en production et le niveau 2.0 a été atteint en 1996. En 1998, avec le noyau à la version 2.2.0 et disponible non seulement pour les machines à base de x86 (= PC équipé d'un processeur Intel) mais pour toute une variété d'autres architectures de machines, GNU/Linux (soit la combinaison du noyau Linux et des très nombreux composants du projet GNU) et Windows NT étaient les deux seuls systèmes d'exploitation du monde à gagner des parts de marché. Une évaluation interne à Microsoft ayant filtré en octobre 1998 (et ensuite reconnue comme authentique par l'entreprise…mais peut-être délibérément dans le cadre du procès anti-trust en cours) concluait que *« Linux représente un UNIX qui sort du rang, en qui on fait confiance pour des missions d'applications critiques et (ce qui est en partie dû au code source [sic] ouvert) et qui a une crédibilité sur le long terme qui excède celle de nombreux systèmes d'exploitation compétitifs. »* (http://www.opensource.org/halloween/halloween1.html). Les systèmes GNU/Linux sont maintenant utilisés à travers le monde, fonctionnant partout, faisant office de serveurs web dans des sites de commerce électronique majeurs ou en tant que clusters dédiés à l'infrastructure réseau de centres de paiement de banques. On trouve aussi GNU/Linux à des endroits surprenants, comme dans la navette de l'espace ou jusqu'à récemment encore sur les serveurs de Microsoft. Les évaluations de l'industrie sur la fiabilité des systèmes Unix ont montré de manière répétée que Linux est de loin le noyau Unix le plus stable et le plus fiable, dont la fiabilité est seulement dépassée par les outils GNU eux-mêmes. GNU/Linux ne dépasse pas seulement les versions propriétaires d'Unix pour les PC dans les tests de performances, mais est renommé pour sa capacité à fonctionner, sans perturbation et sans plainte, pendant des mois sur des environnements de haut volume et de haute sollicitation sans se planter.

d) Les principaux logiciels libres :

Il existe de très nombreux logiciels libres de toutes tailles, et pour toutes les plates-formes, bien que les plates-formes de type Unix, incluant Linux et FreeBSD, restent les plus nombreuses. Nombre de logiciels libres à succès (Emacs, TeX, GCC, PERL, Gimp, par exemple), développés initialement dans un environnement Unix, ont été ultérieurement portés sur les plates-formes Macintosh et Windows, ainsi que nombre de petits utilitaires classiques de l'environnement Unix. La variété de ces logiciels est immense, traitement de texte ou d'images, télécommunications, courrier, logiciels scientifiques, compilateurs... Sans entrer dans une étude exhaustive, il faut savoir que de nombreuses collections organisées sont disponibles dans les archives publiques de l'Internet, ainsi que dans des collections de CDROM à bas prix, mais au contenu très riche. La Free Software Foundation recense d'ailleurs pour chaque pays les moyens de se procurer les principaux logiciels libres, ainsi que leur sources, pour le coût du support et de l'envoi. En France par exemple, on trouve des CD-rom pressés de la plupart des distributions Linux et BSD sur www.ikarios.com pour moins de 20F.

Plus en détail, on peut donc distinguer quelques catégories de logiciels libres:

- les systèmes d'exploitation : Linux, FreeBSD, NetBSD, OpenBSD et Hurd.
- Les environnements de composition de texte avec les outils TeX, LaTeX, Lyx, ainsi que groff.
- Les environnements de traitement d'images Gimp (très proche de Photoshop), Gyve ou Blender.
- Les éditeurs GNU Emacs, XEmacs. L'environnement graphique multi-fenêtres XFree86.
- Les environnements de programmation Gcc, G++, Perl, Php, Phyton, Scheme, Caml, Tcl/Tk, MesaGL…
- Les bases de données relationnelles MSQL, POSTGRES.
- L'ensemble des outils GNU tels que Gawk, Gmake, et bien d'autres.
- Le serveur Web Apache, le serveur de newsgroups Inn. Samba qui permet d'utiliser une machine Unix comme serveur de fichiers et d'imprimantes pour des clients sous Macintosh ou Windows, ainsi que daccéder aux ressources partagées de ces machines.

Plusieurs composants du mouvement du logiciel libre ont également été couronnés de succès. Apache, le serveur web le plus répandu (taux d'occupation des serveurs web : 55% seul et 7O% avec ses dérivés, selon les statistiques de l'organisme Netcraft), est libre, de même que Perl et plus récemment PHP, deux langages de programmation très largement utilisés dans les sites web sophistiqués. Netscape Communications distribue désormais son navigateur Netscape Communicator 6.0 en tant que logiciel libre, sous une licence proche de la Licence Générale Publique de la Free Software Foundation. De grands constructeurs de PC, comme IBM ou Compaq, ont annoncé des projets de distribution de GNU/Linux ou sont déjà en train en train de le distribuer en tant qu'option pour le client sur leurs PC de haut de gamme destinés à une utilisation en tant que serveurs web ou de partage de fichiers. Samba, un programme qui permet aux ordinateurs sous système libre de se comporter en tant que serveurs de fichiers Windows NT, est mondialement utilisé comme une alternative à Windows NT Server. De très nombreuses petites applications pour sites web connaissent aussi un franc succès, comme par exemple PHPNuke (script PHP permettant de créer et gérer un site portail complet), ou Phorum (forum web).

Quelques composants du logiciel libre :
le langage de scripts **PHP**, le serveur web **Apache** et le logiciel de partage réseau **Samba**

Pour mieux comprendre le succès des logiciels libres, il est important de revenir sur leur mode de production original et sur ce qui en fait l'efficacité.

II) Comment expliquer la réussite du modèle de développement « libre » ?

1) Un mode de développement coopératif et volontaire (ou comment rendre performant le travail collectif d'individus isolés)

Le mode de développement des logiciels libres est particulièrement intéressant car il s'inscrit dans une logique totalement différente des logiciels propriétaires. Là où les grands éditeurs de logiciels mettent en place des cahiers des charges et « roadmaps » très détaillés, où le partage des tâches très étudié et l'organisation doivent être sans défauts, les logiciels libres semblent justement s'affranchir d'une bonne partie de ces contraintes. A un modèle très centralisé et hiérarchisé s'oppose une structure totalement éclatée, basée sur le volontariat et unie seulement par les moyens de communications de l'Internet De plus la recherche de profit reste initialement très secondaire dans le logiciel libre, bien que ceci soit à nuancer comme nous le verrons dans la partie III, alors qu'elle reste la raison d'être des éditeurs propriétaires.

Il est intéressant d'opposer ces deux mondes, qu'Eric S. Raymond illustre en comparant le logiciel propriétaire à une cathédrale et le logiciel libre à un bazar :

> *« Je pensais que les logiciels les plus importants (comme les systèmes d'exploitation et les très gros outils comme Emacs) devaient être conçus comme des cathédrales, soigneusement élaborés par des sorciers isolés ou des petits groupes de mages travaillant à l'écart du monde, sans qu'aucune version bêta ne voie le jour avant que son heure ne soit venue.*
>
> *Le style de développement de Linus Torvalds - distribuez vite et souvent, déléguez tout ce que vous pouvez déléguer, soyez ouvert jusqu'à la promiscuité - est venu comme une surprise. À l'opposé de la construction de cathédrales, silencieuse et pleine de vénération, la communauté Linux paraissait plutôt ressembler à un bazar, grouillant de rituels et d'approches différentes (très justement symbolisé par les sites d'archives de Linux, qui acceptaient des contributions de* n'importe qui*) à partir duquel un système stable et cohérent ne pourrait apparemment émerger que par une succession de miracles. »*

(*http://aemiaif.lip6.fr/wlfo/article/these/cathedrale-bazar/cathedrale-bazar_monoblock.html*)

Ces quelques lignes posent bien la question du mode de développement des logiciels libres : comment arriver à un système ordonné et développé de façon cohérente, en l'absence apparente de toute structure et partage des tâches défini, comme avec les logiciels propriétaires?

A) un style de direction intelligent et fédérateur

Dans cette partie, nous nous limiterons à l'exemple du système d'exploitation Linux en tant que modèle de développement, d'une part vu sa place majeure dans le logiciel libre et sa popularité actuelle, d'autre part parce que c'est un des exemples les plus significatifs en termes de modèle de développement d'un logiciel libre.

Linux fut en effet le premier projet à faire un effort conscient et déterminé pour utiliser le monde entier comme réservoir de talent. Linux reste très lié au développement de l'Internet : ses premières ébauches datent de l'apparition du World Wide Web (1991), et son développement massif de 1994, lorsque l'intérêt général accordé à l'Internet et l'industrie des fournisseurs d'accès ont décollé. Linus Torvalds fut un des premiers à comprendre comment se servir des nouvelles règles rendues possibles par la démocratisation d'Internet, et à les utiliser dans le développement d'un logiciel complexe. Même si le faible coût d'utilisation du net a joué pour que le modèle Linux se développe, il y a surtout un autre facteur vital: le développement d'un style de direction de projet et d'un ensemble de coutumes de coopération qui permettaient aux développeurs d'attirer des co-développeurs et de rentabiliser au maximum ce nouveau média.

Voyons donc quels sont ce style de direction et ces coutumes

- l'absence de rapports de force

Basés sur le volontariat, les rapports entre développeurs se caractérisent tout d'abord par l'absence de rapports de force. Même si c'était le cas, la direction par coercition ne produirait pas les résultats très significatifs qu'on peut observer. Pour l'illustrer, Gerald Weinberg (*The Psychology Of Computer Programming (la psychologie de la programmation des ordinateurs)* (New York, Van Nostrand Reinhold 1971)) cite l'autobiographie de Pyotr Alexeyvitch Kropotkine, anarchiste russe du 19ème siècle, ``Mémoires d'un révolutionnaire":

> ``*Élevé dans une famille possédant des serfs, j'entrai dans la vie active, comme tous les jeunes gens de mon époque, avec une confiance aveugle dans la nécessité de commander, d'ordonner, de brimer, de punir et ainsi de suite. Mais quand, assez tôt, je dus diriger d'importantes affaires et côtoyer des hommes libres, et quand chaque erreur pouvait être immédiatement lourde de conséquences, je commençai à apprécier la différence entre agir selon les principes du commandement et de la discipline et agir selon le principe de la bonne intelligence. Le premier fonctionne admirablement dans un défilé militaire, mais ne vaut rien dans la vie courante, où on ne peut atteindre son but que grâce à l'effort soutenu de nombreuses volontés travaillant dans le même sens."*

« L'effort soutenu de nombreuses volontés travaillant dans le même sens » est exactement ce qu'un projet comme Linux demande, et le *« principe de commandement et la discipline »* sont en effet impossible à appliquer aux volontaires (« hommes libres ») liés par les structures horizontales (et d'ailleurs anarchiques) de l'Internet. Pour avancer et se confronter les uns aux autres de manière efficace, les bidouilleurs qui cherchent à prendre la tête d'un projet de collaboration commune doivent apprendre à recruter et à insuffler de l'énergie dans les communautés d'intérêts convergents à la manière suggérée par le *«principe de bonne intelligence»* de Kropotkine. Ils doivent s'inspirer de l'exemple de Linus Torvalds.

- fédérer les égoïsmes individuels hors de toute structure coercitive

En effet, Linus Torvalds a su se positionner comme le leader incontestable du projet Linux où le développement est essentiellement fait par d'autres. S'il a acquis sa légitimité en partie du fait d'avoir écrit les bases du noyau Linux, son leadership provient surtout de sa capacité a avoir su fédérer les égoïsmes individuels des programmeurs volontaires pour réaliser des tâches impossibles sans une coopération soutenue. On peut rapprocher le monde Linux des systèmes adaptatifs en biologie et en économie, vu que sous de nombreux aspects il se comporte comme un marché libre ou un écosystème, avec des ensembles d'agents égoïstes qui tentent de maximiser une utilité (celle du logiciel). Cela produit un ordre spontané, se corrigeant de lui-même, plus élaboré et plus efficace que toute planification centralisée n'aurait pu l'être, en accord avec le principe de bonne intelligence. L'utilité que les développeurs de logiciels libres maximisent n'est pas économique, c'est en fait leur propre satisfaction et leur réputation au sein des autres bidouilleurs. Si leur motivation peut sembler altruiste, il ne faut pas oublier que l'altruisme est en soi une forme d'égoïsme pour l'altruiste. C'est une façon de prouver que ses capacités individuelles dépassent d'une certaine manière celles des autres dans un domaine qui leur tient à cœur et où les performances sont réellement mesurables et comparables. Les cultures volontaires qui fonctionnent sur ce principes sont fréquentes, on les retrouve par exemple sur les forums de discussions techniques ou d'une certaine façon à l'origine de tout acte de don (car donner c'est aussi s'enrichir de ce qu'on donne!).

Le projet Linux montre qu'en flattant à bon escient l'ego de beaucoup d'autres bidouilleurs, un coordinateur-développeur fort peut utiliser l'Internet pour tirer parti du fait d'avoir énormément de co-développeurs sans que le projet ne s'effondre dans le chaos.

Le chef ou coordinateur d'un projet décentralisé et non hiérarchisé doit également être bon en relations humaines et avoir un bon contact. Pour construire une communauté de développement, il est essentiel séduire les gens, de susciter leur intérêt, et de les encourager constamment pour leurs efforts. La personnalité projetée compte énormément, et ce n'est pas surprenant que les principaux « gourous » comme Torvalds soient « *de chics types qu'on apprécie volontiers et qu'on a envie d'aider* » (d'après Eric S. Raymond). Ainsi une communauté de contributeurs animée par une forte motivation a tendance à se créer d'elle-même, hors de toute structure coercitive.

Linus Torvalds *"And no, I don't select the people I work with. People tend to select themselves."*

(http://www.crn.com/Components/Search/Article.asp?ArticleID=23368)

B) L'importance d'une intelligence conceptuelle

Pour qu'un projet rallie les développeurs, il est essentiel que deux points soient respectés au niveau de la conception : d'une part que le projet suscite l'adhésion d'un grand nombre de développeurs, et d'autre part que son développement soit cohérent avec ses objectifs, sans se fourvoyer dans des impasses de développement.

- Un idéal partagé :

Le développement d'un logiciel libre correspond à la recherche d'un idéal pour ses multiples contributeurs. Tous ces efforts désintéressés financièrement ne peuvent se faire qu'en étant motivés par un but ultime, au delà de l'ajout d'une fonctionnalité particulière que recherchaient les nouveaux contributeurs et qui a décidé de leur engagement. Richard Stallman a par exemple d'abord réagi au fait qu'il ne pouvait obtenir d'Hewlett Packard les spécifications matérielles nécessaires pour écrire un pilote d'imprimante qui lui manquait, puis ceci a alimenté le rêve d'un système totalement ouvert et libre, pour lequel il s'investit depuis vingt ans. Comme pour la plupart des développeurs, la motivation n'est pas uniquement de combler ce qui leur semble une lacune dans un logiciel libre ou propriétaire, mais bien plus de contribuer à un projet qui leur tient à cœur, une sorte de « chef d'œuvre », de perfection logicielle, dont ils seraient totalement propriétaires au même titre que n'importe qui d'autre. Pour comprendre cela, il faut se rappeler que les bidouilleurs bénévoles ont quasiment tous une culture bien plus scientifique qu'économique. Pour eux seule la qualité technique ou intellectuelle fonde la valeur d'un logiciel, tout comme la recherche de la « Vérité » fait partie de la culture scientifique.

Un logiciel libre se développera d'autant plus vite qu'il motive un grand nombre de contributeurs actifs - pas forcément d'utilisateurs finaux même si les deux sont souvent liés. Pour initier un travail de développement en communauté, le projet doit présenter une promesse plausible. Le programme ne doit pas nécessairement fonctionner très bien. Il peut être grossier, bogué, incomplet, et mal documenté, mais il ne doit pas manquer de convaincre des co-développeurs potentiels qu'il peut évoluer en quelque chose de vraiment bien dans un futur pas trop lointain.

- une orientation intelligente

Pour être performante, une communauté dispersée ne doit pas seulement être fédérée autour d'un idéal commun, il faut aussi que ses efforts soient toujours coordonnés dans la meilleure direction afin d'éviter les impasses conceptuelles et de décourager les contributeurs. Si le style de direction doit savoir concentrer des égoïsmes particuliers, il faut ensuite qu'il maintienne cette cohésion en sachant toujours choisir les meilleures orientations pour la réussite du projet. Il ne peut se contenter d'exalter les talents d'autres programmeurs, mais doit aussi faire preuve d'un talent exceptionnel au niveau de la conception. Le coordinateur ne doit pas spécialement être un programmeur particulièrement brillant, mais doit absolument savoir reconnaître les bonnes idées de conception des autres. A ce titre, Linus Torvalds peut être considéré comme une personne extrêmement compétente pour savoir choisir les meilleures voies de développement qu'on lui soumet, pour trouver le chemin de moindre effort reliant A à B, et pour éviter les bogues ou autres impasses de développement. La conception de Linux est entièrement imprégnée de ces qualités et reflète l'approche de Linus Torvalds : conservatrice et simplificatrice. Comme il le rappelle lui-même, une entité reconnue de tous doit s'occuper de faire les bons choix parmi les multiples contributions pour garantir un développement sûr. Concrètement, un petit nombre de personnes très spécialisées se partagent la gestion des différentes parties de Linux, et

directement avec Linus Torvalds, décident des orientations stratégiques du développement.

Extrait du Manifeste de Linux (interview de Linus Torvalds)

« Cette idée d' «absence de propriétaire » signifie que seule une entité peut vraiment avoir du succès en développant Linux --- l'entité à qui on fait confiance pour faire les bons choix. Et telles que les choses se présentent à l'heure actuelle, je suis l'unique personne/entité qui jouit d'un tel degré de confiance. Et même si quelqu'un pensait que je travaille mal (ce qui est assez rare) et que cette personne décide « il faut vraiment que je corrige cette fonctionnalité », il lui sera très difficile de convaincre tout le monde qu'elle en est ***capable****.*

Le chaos peut résulter de tout ceci, mais en même temps, ce système intègre certains mécanismes qui le rendent très stable. Mais il faut vraiment être très bon pour prendre en charge le développement. Savoir que la meilleure personne sera là pour s'en occuper est exactement le genre de sécurité dont on a besoin dans un réseau de développement. »

http://www.linux-france.org/article/these/manifesto

2) Une haute qualité technique

A) La « loi de Linus »

Ces principes opposés au mode de production des logiciels propriétaires sont aussi directement responsable de la stabilité reconnue et de la robustesse du logiciel libre, qui découle de ce qu'Éric Raymond appelle «la loi de Linux » : avec suffisamment d'yeux, tous les bugs disparaissent. En pratique, l'accès au code source signifie que tout le monde peut théoriquement identifier et résoudre soi-même les problèmes rencontrés. Et par l'effet démultiplicateur du nombre d'utilisateurs actifs, l'utilisateur final n'est quasiment jamais confronté à un problème non résolu par d'autres. L'effet de masse peut s'exprimer de façon très positive en ce qui concerne la qualité des logiciels dans un modèle de développement en open source.

``Étant donnés suffisamment d'observateurs, tous les bogues sautent aux yeux.'' C'est ce que j'appelle: ``La Loi de Linus''. »

(``Given enough eyeballs, all bugs are shallow.'' I dub this: ``Linus's Law''.)

(Eric S. Raymond http://www.tuxedo.org/~esr/writings/cathedral-bazaar/cathedral-bazaar/x147.html)

Un nombre plus élevé d'utilisateurs augmente directement la qualité du logiciel car l'ajout de nouveaux utilisateurs introduit de nouvelles manières de pousser le programme dans ses derniers retranchements par la grande diversité des contraintes auxquelles il doit répondre. Ce système est également utilisé par l'industrie logicielle propriétaire, mais à échelle bien plus réduite, les quelques bêta testeurs n'ayant aucun accès au code source et se contentant de rapporter les bugs constatés. Avec les logiciels libres, tout utilisateur peut devenir bêta testeur et cet effet est encore amplifié

quand les utilisateurs se trouvent être des co-développeurs. Chacun d'entre eux aura une manière personnelle de déboguer le logiciel et d'en rechercher les faiblesses. Par l'ouverture du code source et la liberté de le modifier, toute personne compétente en informatique pourra elle-même corriger le bogue à sa façon et soumettre son travail au responsable du développement. L'absence de hiérarchie dans ce modèle d'optimisation est plutôt un avantage en supprimant les barrières entre utilisateurs et concepteurs. Tout dépend là aussi de la personnalité du responsable du logiciel et de sa capacité à savoir fédérer et canaliser les multiples contributions qui lui parviennent. Chacune d'entre-elles est un pas supplémentaire vers la qualité du logiciel. Cette caractéristique particulière des utilisateurs développeurs rappelle qu'à la base, le principe du logiciel libre a été conçu par des informaticiens pour des informaticiens, d'où l'important taux d'utilisateurs actifs et impliqués dans le développement et le débogage.

Cette absence d'organisation dans l'optimisation du logiciel pourrait apparaître comme une faiblesse, tout comme l'absence apparente de structure et partage des tâches bien défini l'était en théorie au niveau de la conception. En pratique, le monde Linux n'est quasiment pas affecté par la perte théorique d'efficacité qui découle du fait que plusieurs débogueurs travaillent sur la même chose au même moment. L'une des conséquences de la politique du *«distribuez tôt, mettez à jour souvent»* est de minimiser les pertes de ce type en propageant au plus vite les corrections qui sont revenues au coordinateur. Ainsi, bénéficier de nouveaux bêta-testeurs ne réduit pas la complexité du bogue le plus profond pour le développeur, mais cela augmente la probabilité que l'approche et les compétences d'un bêta-testeur seront adaptées au problème de telle sorte que ce bogue lui saute aux yeux.

De façon empirique, on peut soutenir que si la «Loi de Linus» était fausse, alors tout système aussi complexe que le noyau Linux et soumis aux bidouilles simultanées d'autant de personnes, aurait dû finir par s'effondrer sous le poids des interactions néfastes et imprévues, de bogues profonds non découverts. D'un autre côté, l'absence relative de bogues dans Linux donne un argument de poids à cette « loi ».

Mais un minimum d'organisation existe pourtant dans le développement de Linux. Au cas où il y aurait des bogues sérieux, les versions du noyau Linux sont numérotées de telle sorte que des utilisateurs potentiels peuvent faire le choix d'utiliser la dernière version désignée comme étant stable, ou de profiter des dernières innovations en courant le risque que quelques bogues accompagnent les nouvelles fonctionnalités. La dernière version stable du noyau de Linux est actuellement la 2.4.3, et la version de développement la 2.4.4. Même si cette tactique n'est pas encore formellement imitée par la plupart des bidouilleurs Linux qui ne classent pas forcément leurs versions comme stable ou non, le fait d'avoir une alternative rend les deux choix séduisants.

Dans la programmation du point de vue propriétaire, les bogues et les problèmes de développement représentent des phénomènes difficiles, fastidieux et extrêmement longs à identifier et résoudre. Il faut à quelques bêta testeurs triés sur le volet des mois d'observations minutieuses avant de bien vouloir se laisser convaincre que tous les bogues ont été éliminés, d'où les longs intervalles séparant les mises à jour, et l'inévitable présence de défauts plus ou moins contraignants.

B) La qualité comme seul objectif :

Un facteur marquant de l'opposition entre les univers du logiciel libre et du logiciel propriétaire est l'absence de marketing et de toute promotion économique. Ceci rend le succès d'un logiciel libre directement dépendant de ses qualités techniques, ignorant d'autres facteurs très présent dans le monde propriétaire comme la publicité, la promotion commerciale ou encore le lobbying qui peuvent faire passer la qualité objective derrière des facteurs plus subjectifs.

Ceci permet d'expliquer pourquoi, à l'inverse d'un grand nombre de sociétés logicielles aux produits pourtant innovateurs tués par la concurrence commerciale, un logiciel libre n'a aucune contrainte en terme de développement si ce n'est la qualité. Peu importe le respect des échéances, ou la conquête de nouveaux marchés, le seul intérêt du logiciel est de remplir son rôle technique et de satisfaire des besoins exprimés par ses propres utilisateurs. Les sorties de nouveaux noyaux pour Linux sont souvent repoussées jusqu'à ce que le nouveau donne satisfaction (cependant certaines distributions commerciales de Linux peuvent avoir tendance à sortir de nouvelles versions pas forcément éprouvées pour répondre à la pression concurrentielle, ce qui souligne encore plus les effets pervers de la pression commerciale dans le domaine logiciel). Certaines fonctionnalités plutôt marketing sont absentes, alors que des logiciels propriétaires similaires s'en servent abondamment (par exemple aucune publicité liée n'apparaît lors de l'installation)… L'apparence cède la place au coté technique pur et dur, particulièrement dans les versions non commerciales de logiciels libres (comme la distribution Debian ou FreeBSD). De plus les logiciels libres n'ont pas à craindre la concurrence commerciale, s'étant volontairement placé hors de cette sphère où la valeur se base sur la propriété intellectuelle du code source.

Cependant, ce n'est pas parce qu'un logiciel est libre qu'il est forcément de bonne qualité. Son code source aura beau être disponible, si en dehors des auteurs personne d'autre ne va l'examiner de près, la différence avec les logiciels du domaine public se réduit fortement. Si des projets tels que Linux ou Apache ont eu le succès qu'on leur connaît, le fait d'avoir été libres a été indispensable mais pas suffisant. Dans les deux cas, ces logiciels ont dû leur succès à une communauté fédérée à la fois par un leader et d'un rêve commun. Dans le cas de Linux, ce rêve a été de se créer son propre système d'exploitation.

On peut y opposer le fait que les milliers de développeurs professionnels des grands éditeurs commerciaux constituent une force très importante avec le soutien d'instituts de recherche et d'investissements très lourds. Cependant l'effet de masse des développeurs libres et l'amélioration permanente des logiciels ouverts reste bien plus efficace, du fait de la collaboration sur une échelle bien plus grande et sans doute de l'absence d'autre objectif que la qualité. Une comparaison a été faite, disant que quels que soient ses moyens, aucune entreprise logicielle ne pouvait lutter contre les talents combinés des meilleurs programmeurs de la planète. En effet, les communautés de logiciel libre peuvent mettre sur le problème un temps humain cumulé bien plus important que toute équipe de programmeurs.

Une analyse indépendante de la qualité des logiciels libres a été produite par le Computer Sciences Department de l'University of Wisconsin, illustrant l'importance de la culture du développement sur la qualité des logiciels :

> *"Fourth, the reliability of the freely-distributed GNU and Linux software was surprisingly good, and noticeably better than the commercially produced software. It is difficult to tell how much of this is a result of programmer quality, the culture of the programming environment, or the general burden supported by the software developers. Large companies will need to make some concrete changes in their software development environments and culture if they hope to produce higher quality software."*
>
> (ftp://grilled.cs.wisc.edu/technical_papers/fuzz-revisited.pdf , page 22)

3) Les autres facteurs de succès des logiciels libres

Comment savoir si le succès d'un logiciel libre est possible ? On peut isoler quelques points déterminants pour qu'un logiciel libre se développe avec succès

La qualité technique est un élément déterminant dans le succès des logiciels libres, dans la mesure où le facteur publicité/marketing des logiciels commerciaux est éliminé. Les autres facteurs de succès que l'on peut relever, sont finalement assez logiques et sont directement liés aux signes indicateurs de popularité, de bonne maintenance, d'évolutivité, et de convivialité (installation, documentation) :

- **Une structure de gestion et de production bien identifiée.** Les auteurs des logiciels libres sont d'origines très diverses: universitaires étudiants, enseignants ou chercheurs, mais aussi personnes privées, associations ou sociétés commerciales. Que le développement soit centralisé dans un petit groupe, ou réparti dans le monde entier, il doit être bien géré pour éviter une prolifération de modifications et de versions de qualité médiocre, et pour bénéficier au mieux de toutes les contributions de valeur.

- **Une communauté d'utilisateurs large et vivante :** c'est la principale motivation pour que les développeurs continuent à faire vivre et évoluer le logiciel et attirent de nouvelles contributions. De plus, une large communauté est plus susceptible d'apporter de l'aide aux nouveaux utilisateurs par l'intermédiaire des forums et des mailing listes, voire d'avoir ses propres testeurs spécialisés pour le logiciel concerné. Les utilisateurs contribuent également à la mise au point par leurs remarques et leurs expériences. Il existe assez peu de statistiques d'usage pour les ressources libres, mais le nombre de discussions sur les forums est révélateur. Un grand nombre d'utilisateurs est d'ailleurs un moyen de devenir un standard, comme le montre bien l'exemple des logiciels les plus piratés (Microsoft Windows et Word, ou Adobe

Photoshop) qui deviennent aussi une norme en bénéficiant de la loi de rentabilité croissante fondée sur la familiarité, en plus de leurs qualités propres.

- **Une communauté active de développeurs et de contributeurs.** Cette communauté est le principal gage de la pérennité et de la qualité du logiciel, par la diversité des contributions. Elle assure une grande qualité technique par la mise en concurrence des diverses propositions d'implémentation ou d'évolution. Par exemple, les processus légers du système d'exploitation Linux, essentiels pour certaines applications, ont vu le développement en parallèle de douze propositions indépendantes, dont seule la meilleure fut finalement retenue. Ceci n'empêche pas ces propositions de continuer à évoluer dans une concurrence continue, permise par un consensus sur les interfaces. Par exemple, on peut noter l'évolution de la famille des BSD en trois versions suite au divergences entre les besoins des développeurs, FreeBSD privilégiant la performance sur plateformes x86 (Intel), OpenBSD orienté vers une sécurité maximale et NetBSD cherchant le support du plus grand nombre d'environnements matériels. Sans se faire concurrence, ces versions parallèles répondent au contraire à tous les besoins exprimés.

- **Une architecture ouverte, documentée et modulaire.** La modularité et la documentation sont des facteurs essentiels de réussite d'un projet logiciel. Ceci se vérifie encore plus dans le contexte des gros logiciels libres, compte tenu de la décentralisation du développement. En fait, on peut même conclure qu'un gros logiciel libre qui réussit à se développer doit être a priori particulièrement bien structuré, et donc probablement de qualité. Une plate forme commune à l'interface standardisée et documentée doit aussi permettre une modularité verticale, afin d'avoir de nombreux développements et davantage de créativité autour d'une base standard. La séparation entre la réalisation et la documentation du projet est un moyen de répartir la charge de travail utilisé également dans l'industrie. Le Linux Documentation Project qui vise à écrire une documentation complète sur Linux dans toutes les langues est un projet séparé du développement en lui-même.

- **Un copyright ouvert et donnant une bonne protection.** Il faut que la licence permette les mécanismes de développement décrits ci-dessus en les encourageant par une reconnaissance au moins sociale des contributions et en évitant de les gêner ou de les décourager par une protection inadaptée du travail bénévole.

4) Les avantages intrinsèques des logiciels libres par rapport aux logiciels commerciaux :

A) De multiples avantages…

Le mode de développement des logiciels libres semble donc être particulièrement compétitif, malgré son apparente anarchie. Vis à vis des logiciels au développement propriétaire, on peut retenir les caractéristiques techniques suivantes dans la majeure partie des cas : des fonctionnalités avancées, une rentabilité maximum des efforts

fournis et du temps investi, une amélioration « darwinienne » aboutissant à une efficacité maximum, une très grande fiabilité et un minimum de bugs, un respect des standards supérieur, une garantie de fonctionnement optimale si le suivi est maîtrisé, une liberté de choix supérieure pour les utilisateurs, et finalement la pérennité des solutions choisies.

Voyons ces différents points plus en détail :

Fonctionnalité

Les logiciels libres sont réalisés par des personnes passionnées par un sujet donné ou par des fonctions particulières. Par conséquent, ils disposent naturellement des fonctionnalités les plus avancées dans leurs domaines respectifs, tandis que les logiciels propriétaires ont davantage tendance à faire évoluer des techniques plus anciennes. Les dernières versions de Windows 98 sont par exemple encore basées sur Ms-Dos qui date de 1981.

D'autre part, les logiciels libres existants peuvent servir de base fiable pour les personnes et les sociétés qui désirent ajouter des fonctionnalités spécifiques non encore couvertes. Ils peuvent recevoir de nouvelles fonctionnalités qui n'auraient jamais été introduites dans un logiciel fermé ou propriétaire vu la spécificité de la demande.

Rentabilité

Ce phénomène permet de gagner énormément de temps et de ressources, puisque ceux qui veulent ajouter une nouvelle fonctionnalité peuvent réutiliser tous les codes sources existants. La déperdition d'énergie dans le développement est minimisée par les mises à jour fréquentes des améliorations. Il est en effet plus simple de repartir d'un logiciel libre existant et de lui ajouter la fonctionnalité recherchée que de repartir à zéro et d'écrire un logiciel complet. Ceci a une incidence importante sur leur rentabilité par rapport aux produits commerciaux au code source protégé, comme nous le verrons dans toute la partie III.

Efficacité

Par l'ouverture des sources et la possibilité de les modifier, les logiciels libres permettent la contribution de tout volontaire. Ces contributions portent parfois sur des petites parties du logiciel, et se font par des personnes différentes dans le monde entier et sans autre rapport que la base de source commune. Ceci permet l'exploration de différentes solutions techniques et la meilleure est généralement retenue, tandis que l'industrie logicielle ne peut pas se permettre une telle recherche. Sur le long terme, grâce à la sélection naturelle (darwinienne pour certains) des solutions techniques, les logiciels libres se révèlent être les plus performants et les plus efficaces en permettant aux utilisateurs de faire du programme ce qu'ils en attendent réellement.

Fiabilité

Toujours grâce à l'ouverture des sources, tout utilisateur sachant le faire a la possibilité de corriger les erreurs éventuelles qu'il peut détecter. Quasiment aucune erreur ne peut

donc passer au travers de ce filtrage continu, à laquelle tous les utilisateurs actifs participent. Les logiciels libres atteignent donc une très grande fiabilité en peu de temps.

Compatibilité avec les standards

Les développeurs de logiciels libres favorisent le respect des standards. En effet, seuls les standards garantissent une interopérabilité parfaite avec les autres logiciels. Personne n'a intérêt, dans le monde du libre, à utiliser des protocoles incompatibles ou des formats de fichiers non standards, puisque les sources sont ouvertes et qu'il est impossible d'utiliser les techniques de rétention d'informations classiques dans le but de gagner des parts de marché. Les logiciels libres manipulent donc leurs données sous des formats standards, qui permettent de les récupérer et de les traiter avec d'autres logiciels de manière fiable et à moindre coût.

Garantie de fonctionnement

Professionnellement, la possibilité de modifier les sources garantit que les logiciels vont fonctionner d'une manière ou d'une autre. En effet, avec les logiciels propriétaires, les utilisateurs sont absolument dépendants des sociétés éditrices en cas de problème. Les contrats de service classiques sont non seulement chers, mais souvent inefficaces, car la correction d'un bogue passe souvent par l'attente de la version suivante (et de son achat).

Au contraire, les logiciels libres sont plus réactifs et permettent une correction immédiate, mais ils permettent également aux utilisateurs de choisir la solution à leur problème. Ils peuvent le résoudre eux-mêmes s'ils en ont les moyens, ou sinon louer les services d'une société spécialisée qui assure alors un fonctionnement optimal. Dans les deux cas, les utilisateurs ont l'assurance du bon fonctionnement de leurs logiciels.

Garantie de la liberté

La disponibilité des sources garantit en permanence la liberté de tout utilisateur. Il n'est pas possible, dans un logiciel en open source, d'inclure des fonctionnalités cachées dans le but de restreindre les libertés individuelles ou de collecter des informations sur les utilisateurs (ce que l'on appelle le « spyware »).

Du fait qu'ils respectent les standards, les logiciels libres n'utilisent pas des formats de fichiers non documentés (comme par exemple les différentes versions de Microsoft Word) ou des protocoles de communication propriétaires. Ils garantissent donc la libre circulation des informations et la liberté d'expression de chacun quel que soit son équipement.

De plus, les logiciels libres proposent souvent une ou plusieurs alternatives aux autres logiciels, garantissant ainsi la liberté de choix à leurs utilisateurs.

Pérennité

La disponibilité des sources garantit aux utilisateurs la pérennité des logiciels qu'ils

utilisent. L'abandon du support du logiciel par la société éditrice n'est donc pas à craindre.

B) …et quelques limites :

Comme on l'a vu, l'ouverture des sources, l'importance de la base d'utilisateurs et leur motivation est un élément fondamental dans la rapidité de correction et d'adaptation d'un logiciel libre. Cependant ce facteur constitue une limite dans le cas de logiciels à diffusion réduite ou hautement spécialisés, dont la faible base d'utilisateurs ne permet pas toujours une amélioration coordonnée et efficace. Si l'ouverture des sources permet un développement personnalisé, la base de départ n'existera pas nécessairement dans le cas de logiciels destinés à un public très restreint. Cela n'empêchera pas le développement d'une version au code source ouvert, par exemple par un prestataire de services spécialisé, mais les avantages du modèle coopératif de développement resteront bien moins importants (bien que non nuls).

Un exemple typique est fourni par les logiciels utilisés dans le monde médical ou bancaire, pour lesquels les offres commerciales propriétaires semblent plus adaptées vu les exigences spécifiques de ces secteurs. De même, certains types de logiciels restent encore négligés par les programmeurs open source vu leur relatif manque d'intérêt personnel, comme les logiciels de comptabilité, d'édition sonore ou vidéo, ou encore de modélisation 3D par exemple, bien que certaines ébauches existent (Blender). Le succès d'Apache s'explique aussi par l'intérêt que lui a porté la communauté des développeurs pour laquelle il constituait un modèle de performance, de sécurité et de configurabilité optimale recherchée par le plus grand nombre dans le cas d'un serveur web.

Si le mode de développement des logiciels libres a donc su produire des résultats étonnamment concluants au point de vue technique, comment peut-on accorder une viabilité économique à ces produits totalement étrangers à ces exigences ? Où ces deux mondes opposés peuvent-ils se rejoindre?

III) Quelle viabilité pour le modèle économique basé sur les logiciels libres ?

1 – Au niveau des éditeurs de logiciels

A) Les acteurs et leurs stratégies :

Si la majorité des logiciels libres n'est pas développée dans un objectif commercial mais simplement pour répondre aux besoins des utilisateurs, les logiciels libres constituent depuis 1994 un modèle économique pour un certain nombre de sociétés éditrices de logiciels. Ceci peut apparaître paradoxal, vu l'ouverture et la gratuité de fait des logiciels libres, cependant la stratégie de ces acteurs ne se base pas sur la vente de licences contrairement aux éditeurs de logiciels propriétaires.

- la vente de services avant tout

Ces acteurs majeurs du logiciels libres et de sa promotion commerciale existent depuis 1994 lorsque la société américaine Red Hat Software a adapté et distribué une compilation des sources de Linux disponibles sur l'Internet. Depuis Red Hat a été rejoint par de nombreuses autres sociétés, dont les principales sont SuSE (Allemagne), Mandrake (France), Corel, Caldera, Turbolinux ou encore RedFlag Software pour la Chine. En fait il existe une infinité de distributions plus ou moins dérivées des grands noms comme RedHat et adaptées à un usage ou pays spécifique. Ces distributions respectent la GPL et se trouvent donc librement adaptables, modifiables et même revendables par tous. Il est tout à fait possible de récupérer une distribution existante, de lui ajouter quelques packages logiciels et quelques logos personnels, et de la revendre sous son propre nom. D'ailleurs parallèlement à la vente de distributions sous forme de packs (CD-roms d'installation, manuel et aide en ligne), ces sociétés permettent toujours de télécharger gratuitement leurs dernières versions. Ne possédant pas ce qu'elles vendent, elles n'ont pas à chercher à en limiter la distribution. Au contraire, leurs sources de revenus sont bien davantage basées sur la vente de services associés, où leur expertise peut réellement être monnayée. Cela peut aller du soutien offert aux acheteurs de packs logiciels (qui payent plus un service que les logiciels eux-mêmes), à la mise en place et au déploiement de solutions particulièrement volumineuses basées sur Linux dans les entreprises. Des acteurs comme SuSE, Red Hat ou Turbolinux ont conclu de nombreux accords de partenariats avec les plus grands fabricants de matériel informatique afin de permettre à ces derniers d'enrichir leur offre de solutions logicielles. Red Hat est par exemple lié à IBM, Compaq, Dell, HP, Intel et bien d'autres.

Quelques distributions commerciales de Linux :
RedHat, Mandrake, SuSE, Turbolinux et RedFlagLinux

Si les perspectives de profit restent moins importantes qu'avec le système de copyright des logiciels propriétaires, d'un autre coté toute société compétente peut proposer une offre complète de logiciels et de services identique. Le besoin de services sur lequel se base leur stratégie et leur revenus est d'ailleurs bien réel comme le rappelle Bob Young, cofondateur de RedHat :

« Il faut bien se rendre compte que Linux a commencé avec les « early adopters» , c'est à dire avec des étudiants ou des ingénieurs qui n'ont absolument pas besoin de support ou de services. C'est pourquoi beaucoup d'observateurs ont considéré que Linux était un marché où il n'y avait pas d'argent à gagner. Les choses changent aujourd'hui car ces «early adopters» ne sont que 10% des clients potentiels. Les autres 90% ont besoin de services et de support, ils le paieront si Linux répond à leurs besoins et permet d'améliorer le TCO (coût total de possession)»

(interview de **Bob Young**, co-fondateur de RedHat, au magazine Linux+, septembre 2000, p10-11)

- un enrichissement collectif

Les logiciels libres sont pour ces éditeurs un moyen de s'affranchir de la dépendance envers d'autres sociétés, et de pouvoir apporter librement leurs compétences. Aucun frein comme le paiement de royalties ou le manque d'informations sur le fonctionnement du système ne viennent les subordonner aux décisions conceptuelles ou stratégiques d'autres acteurs. Cela permet une compétition plus saine entre les éditeurs de logiciels, dont le succès ne dépend que de la valeur ajoutée qu'ils produisent puisque leurs améliorations iront ensuite enrichir la communauté du logiciel libre dont ils profiteront eux-mêmes en retour. Par exemple, Red Hat est à l'origine du « Red Hat Package Manager », qui facilite l'installation de logiciels compilés au format rpm. De nombreuses autres distributions ont repris cet outil à leur tour, popularisant le format développé par RedHat.

« On nous oppose également à Debian ou à Mandrake. Ce n'est pas plus adapté. Plus Debian sera fort, plus nous le serons et vice versa. N'oubliez pas que nous sommes dans l'Open Source où les uns voient et critiquent le travail des autres »

(idem)

- des perspectives très larges

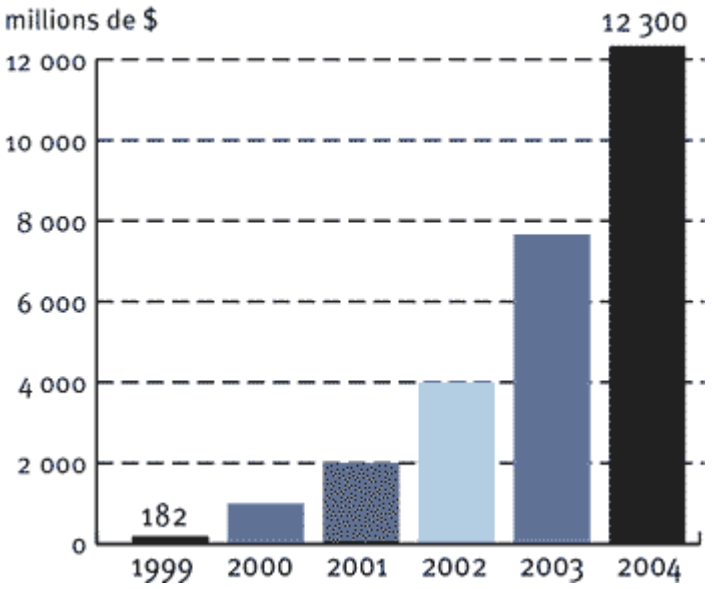
Ces sociétés cherchent aussi à se diversifier pour conquérir de nouveaux marchés dans lesquels le logiciel libre a un potentiel certain. Par exemple, les Internet Appliances (applications embarquées) et tous les types d'organiseurs personnels reliés à Internet, amenés à se développer fortement, pourront tirer avantage de Linux par ses faibles

besoins en ressources, sa forte adaptabilité et l'absence de coûts de licences dans le prix de vente de ces produits.
Les solutions commerciales basées sur les logiciels libres semblent donc tout à fait viables, bien qu'elles obéissent à un modèle économique différent.

Revenu mondial des solutions Linux (de 1999 à 2004)

Selon, IDC le marché de Linux atteindra 12,3 Milliards de dollars en 2004 contre à peine 182 Millions en 1999. Ce chiffre comprend les revenus pour le système d'exploitation mais aussi les applications, les utilitaires et les bases de données. Le système ne rapportant pas énormément, la grande partie de l'activité sera concentrée sur les applications et les services.

(Source IDC :
www.01net.com/rdn?oid=126342**)**



B) Les limites de l'approche propriétaire :

La production de logicielle du point de vue propriétaire obéit à des règles économiques très particulières, où les coûts se concentrent essentiellement sur la conception et ne concernent plus la production et la distribution que de façon marginale. Cela conduit à des phénomènes de concentration très importants, et la tendance au monopole dans l'économie logicielle apparaît bien plus élevée qu'ailleurs. Ces phénomènes ont pour effets logiques d'accroître la dépendance des utilisateurs, mais aussi de ralentir le progrès technologique à long terme. Voyons en détail les règles et limites de la production logicielle propriétaire :

- l'importance majeure des investissements :

Une caractéristique importante des biens immatériels comme les logiciels est le rapport élevé entre les coûts fixes d'investissement (plus humains que matériels) pour créer le premier exemplaire d'un bien, et le coût marginal de production. Concevoir un logiciel complexe demandera d'énormes investissements humains et coûtera infiniment plus cher que de le diffuser par la suite. Avec d'une part la progression très importante du taux d'équipement informatique ces dernières années, et d'autre part le développement de l'Internet et des CD-ROM, le phénomène s'est exacerbé par la

possibilité de communications massives, ultra-rapides et même interactives, permettant une grande varié de protocoles et donc une diffusion au coût de plus en plus marginal. Par conséquent, pour beaucoup de biens immatériels, on ne peut même plus parler de coût de revient car le coût marginal de production et de distribution peut pratiquement être considéré comme nul (si l'on fait abstraction du service après-vente et des hotlines, qui peuvent être indépendants de la production et de la distribution).

Cette situation conduit dans bien des cas à des phénomènes de concentration allant jusqu'au monopole, qui entravent le bon fonctionnement des mécanismes du développement économique et technologique, et conduisent à une gestion purement financière des ressources scientifiques, technologiques ou culturelles, avec nombre d'effets néfastes. C'est ce qui motive le procès anti-trust contre Microsoft depuis 1998.

- **de la concentration au contrôle des standards**

Sans revenir sur les nombreux phénomènes qui soutiennent la tendance à la concentration et au monopole dans les secteurs industriels, l'un d'entre eux s'avère particulièrement influent dans l'industrie du logiciel en raison du rôle primordial qu'y jouent les mécanismes de modularité dans la constitution de systèmes très variables et très complexes. Il se fonde principalement sur un cercle vicieux, lié au contrôle des standards par un phénomène de plate-formes et de contenus.

En effet, dès qu'une plate-forme domine le marché, tout créateur commercial de produits (matériel ou logiciel) liés aux plates-formes s'adaptera de préférence à la plate-forme dominante, pour des raisons évidentes de rentabilité. C'est ainsi le cas des fournisseurs de composants pour PC qui fournissent systématiquement les pilotes pour les plates-formes logicielles Microsoft et plus rarement pour les autres (certains modems sont même conçus pour ne fonctionner que sous Windows). C'est encore beaucoup plus vrai pour les éditeurs de logiciels d'application qui, lorsque leur marché est encore concurrentiel, n'ont souvent pas les moyens (ou parfois même l'intérêt économique, comme dans le cas des jeux) de développer pour autre chose que la plate-forme la plus répandue du marché. Ceci renforce encore son caractère dominant puisque les autres plates-formes, possédant alors moins d'applications, ont moins d'attrait. Certains outils comme Direct 3D pour les jeux servent d'ailleurs à encourager le développement sur une seule plate-forme, bien que des alternatives facilement portables comme OpenGL existent.

Ce phénomène est de plus renforcé par la protection légale ou technique des interfaces qui rend difficile, par manque d'informatio, le développement par d'autres sociétés de logiciels équivalents compatibles sur les plates-formes ainsi négligées. Il est par exemple très difficile d'exploiter sur une plate-forme Unix une encyclopédie prévue pour Windows, alors que cela ne pose en principe aucun problème technique.

- **Les effets de cette situation**

De façon générale, cette situation entrave le progrès technologique à long terme. Une fois la concurrence disparue, le seul producteur restant n'a plus véritablement d'intérêt à investir pour améliorer ses produits autant que possible. Le contrôle d'une technologie par une seule société implique que seul un petit nombre de professionnels sera impliqué dans l'amélioration de cette technologie. La recherche universitaire et l'enseignement sont alors entravés ou contrôlés par la rétention de l'information. De plus, la diversité réduite des développements, du fait d'une unique plate forme, limite considérablement les possibilités de progrès par évolution concurrentielle comme entre les distributions de Linux, et augmente la vulnérabilité générale du tissu technologique aux agressions (l'année dernière, quelques virus célèbres ont par exemple pu infecter un nombre très élevé de machines en exploitant une faille du logiciel de messagerie Outlook).

Dans le cadre d'une utilisation en entreprise, les inconvénients sont nombreux. Ne pouvoir disposer que d'un fournisseur unique en matière de solutions logicielles crée une situation de dépendance pour les prix et les services. Il en va de même pour la stratégie à long terme de l'entreprise qui peut dépendre des décisions de son unique fournisseur. Techniquement, la non disponibilité des codes sources (ou leur prix excessif) limite fortement ou interdit aux sociétés clientes toute utilisation et tout service personnalisé, que cela concerne la maintenance, la sécurisation, le portage sur de nouvelles plates-formes ou l'adaptation à des besoins spécifiques. En fait, la société cliente contrôle mal la qualité et la pérennité de son investissement, voire de ses structures informationnelles.

- **le rôle économique des logiciels libres :**

En plus de réduire la dépendance stratégique des entreprises, les logiciels libres peuvent également avoir des effets importants sur l'économie et l'emploi, souvent bien supérieurs à ceux de logiciels contrôlés par des acteurs propriétaires. Comme dans le cas des distributeurs de solutions basées sur Linux, les logiciels libres sont créateurs d'emplois décentralisés (PME) de service, et de nombreuses petites sociétés se créent dans la maintenance ou le développement personnalisé. En supprimant le coût des licences, le logiciel libre autorise une marge supplémentaire en termes d'adaptation, de réactivité et de fiabilité. La simple économie en licences non payées peut servir à payer des ingénieurs pour adapter le logiciel libre aux besoins de l'entreprise et à former les utilisateurs. Ainsi le logiciel libre, adapté sur place, est créateur de plus d'emplois locaux que le logiciel propriétaire importé. Arkane Média, société prestataire de services en logiciels libres basée à Strasbourg, a par exemple reçu le prix de start-up de l'année lors des Trophées de la Nouvelle Economie (http://www.jdnet.fr/lille/010131arkane.shtml).

Ce schéma reste valable au niveau national pour des pays dont les ressources financières sont limitées, mais qui disposent d'une main d'œuvre bien formée ou pouvant l'être, comme nous le verrons en détail dans la partie III)3)B) p50.

Le logiciel libre semble donc tout à fait viable comme modèle économique pour des entreprises éditrices de logiciels. Voyons maintenant quel impact économique peuvent avoir les logiciels libres aux niveaux des utilisateurs professionnels.

2 – Au niveau des utilisateurs (entreprises)

A) Quelle compatibilité entre le monde du « libre » et celui de l'entreprise ?

A première vue, les mondes du logiciel libre et de l'entreprise sont opposés. Les entreprises ont généralement un point de vue négatif sur le logiciel libre (et Linux en particulier puisque c'est un des logiciels libres les plus connus), bien que l'on observe un changement sensible depuis peu. Ceci est lié à plusieurs raisons : la liberté en elle-même n'est pas toujours un concept bien accepté dans le monde de l'entreprise, le terme "logiciel libre" pouvant au premier abord faire penser à un certain côté "anarchiste". De plus, l'absence de structure régissant le monde du logiciel libre (d'ou son succès comme nous l'avons vu), mis à part des initiatives comme le projet GNU, peut freiner son adoption par l'apparente absence de garantie. Aucune société de taille respectable ne propose actuellement des solutions à base de logiciels libres sur une grande échelle, la société RedHat n'étant qu'un acteur mineur par rapport à des géants comme Microsoft, Sun ou Oracle. Le côté gratuit, couplé à une absence totale de marketing gène donc les décideurs, qui mettent les logiciels libres au même niveau que les logiciels gratuits (freeware), qui ne justifient souvent leur existence que par leur gratuité. Pourtant comme le rappelle la Free Software Foundation, la gratuité d'un logiciel libre n'est qu'un détail et absolument pas sa raison d'être.

Pour définir une politique logicielle, les entreprises se posent en général les questions suivantes avant de savoir si elle est compatible ou utile à la communauté économique :

- **quelle maintenance attendre sur un produit gratuit ?**
- **quelle pérennité du produit ?**
- **quelle qualité technique: fonctionnalité, performance, fiabilité ?**
- **quelle compatibilité avec les standards du marché ?**

- La maintenance

Contrairement à une idée reçue, mais fausse, la maintenance des logiciels libres est généralement mieux assurée que celle des logiciels commerciaux. La plupart, et surtout les plus importants d'entre eux ont une structure de maintenance organisée, qui peut d'ailleurs évoluer au cours du temps. Grâce au réseau, et à la multiplicité des intervenants, la réactivité aux problèmes qui surviennent est extrêmement grande. On cite souvent l'exemple d'un bogue des logiciels de connection à l'Internet, affectant l'ensemble des systèmes d'exploitation et qui permettait des attaques électroniques

contre les serveurs. Après identification du problème, le système libre Linux fut le premier à être corrigé, bien avant l'ensemble des systèmes commerciaux.

En outre, pour des exigences particulières, il reste toujours possible de recourir à une maintenance personnalisée, adaptée précisément à son besoin, et payante comme tout service, grâce à la disponibilité des codes sources. En fait, le développement de l'usage des logiciels libres remplace une activité commerciale centralisée (monopole) d'édition, dont la protection étouffe à terme le développement économique et technologique et qui est fort peu créatrice d'emplois, par une activité commerciale de services, plus créatrice d'emplois décentralisés et plus concurrentielle.

- la pérennité

Là aussi, la pérennité sur le long terme peut au moins être assurée par les entreprises elles-mêmes ou par un prestataire de leur choix, grâce à la disponibilité des sources. On peut cependant penser qu'une large communauté d'utilisateurs tendra davantage à pérenniser les produits, car elle contiendra toujours quelques éléments capable d'assurer le suivi technique, comme le montre la pérennisation par leur communauté de systèmes anciens, comme CP/M, qui ont perdu tout intérêt économique (système d'exploitation 8 bits antérieur à Dos, CP/M n'était pas un logiciel libre, mais il était assez simple pour que l'on puisse l'analyser et en faire des simulateurs). Cette pérennité se fonde donc sur une masse d'utilisateurs à l'expérience plus grande que celle assurée par une société commerciale (même importante) qui peut être amenée à abandonner des produits pour des raisons de stratégie industrielle, et à ne plus leur assurer qu'une maintenance dégradée. La disponibilité des sources est donc un gage de pérennité quelque soit le logiciel.

- les qualités techniques

En ce qui concerne des qualités plus immédiates telles que fonctionnalité, performance ou fiabilité, les expériences, les tests, les évaluations et les comparaisons publiés montrent que les grands logiciels libres font souvent au moins jeu égal avec leurs concurrents commerciaux et souvent les dépassent nettement. Une récente comparaison de serveurs de bases de données réalisée par IBM a par exemple montré un très net avantage en performances de la solution sous Linux RedHat à celle sous Windows 2000. (http://www.zdnet.com/filters/printerfriendly/0,6061,2760874-2,00.html et http://www.tpc.org/tpch/results/h-ttperf.idc)

- le respect des standards

Enfin, pour ce qui est de la compatibilité avec les standards du marché, les logiciels libres, fondés sur la coopération volontaire, intègrent naturellement le respect des normes les plus ouvertes et les plus répandues. En fait, certaines organisations fondées sur le modèle libre, comme l'Internet Society (www.isoc.org), sont même créatrices de standards (appelés RFC) maintenant universellement utilisés. À l'inverse, le développement économique du logiciel commercial se fonde très largement sur des guerres de standards (parfois jusque devant les tribunaux comme pour le langage Java), qui se traduisent pas une rétention d'information et une instabilité permanente et inutile des produits, dont Microsoft Word est un bon exemple. On peut d'ailleurs

craindre que le changement constant des formats de représentation, qui de plus ne sont pas documentés publiquement, n'entraîne à terme la perte d'une partie du patrimoine documentaire numérisé.

Ainsi le logiciel libre s'avère tout à fait compatible avec le monde de l'entreprise d'un point de vue stratégique. Voyons maintenant dans quelle mesure il peut s'avérer intéressant d'un point de vue économique face à une stratégie liée aux logiciels propriétaires.

B) Etude de trois scénarios de déploiement de logiciels en entreprise

- Analyse des coûts d'un point de vue théorique

Les trois scénarios suivants proviennent de l'étude de Jean-Paul Smets-Solanes, Docteur en sciences informatiques (Paris VI). Son étude détaillée des formules mathématiques à l'origine de la modélisation des coûts pour chacun des scénarios est disponible sur http://www.smets.com/it/tco/economie_libre.html . Je reprends ici les conclusions que l'on peut en tirer suite au modèle de calcul des coûts interactif que l'on trouve sur http://www.mmedium.com/dossiers/libre/model.html

Dans le cadre du déploiement d'un logiciel en entreprise, on peut isoler les trois cas suivants, dont les coûts peuvent être évalués selon ce modèle simple.

- **1er cas : l'entreprise achète un logiciel commercial et paiera des ingénieurs chargés de l'adapter à ses besoins**
- **2ème cas : cette entreprise télécharge un logiciel libre et là aussi paie pour l'adapter à ses besoins**
- **3ème cas : l'entreprise développe elle-même ce logiciel, qu'elle peut placer sous licence libre ou non**

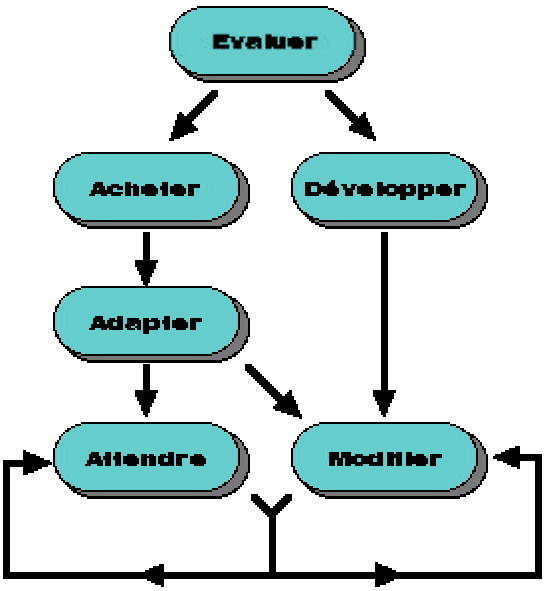
Il faut tenir compte du fait qu'une fois le logiciel déployé, des bogues vont probablement être découverts et des fonctionnalités manquantes deviendront nécessaires. Ces défauts, qui peuvent survenir plusieurs fois par an, coûtent chaque jour à l'entreprise en perte de temps des utilisateurs qui doivent redémarrer leur machine, modifier des paramètres, saisir à nouveau des informations... Afin d'en minimiser les coûts. elle peut :

- soit attendre une mise à jour du logiciel
- soit payer des ingénieurs pour modifier le logiciel et corriger les bogues le plus rapidement possible

Dans le cas d'un logiciel commercial, il faudra acheter une mise à jour si l'on choisit d'attendre; ou bien acquérir une licence d'accès au code source si l'on choisit de corriger les bogues soi-même. Dans le cas d'un logiciel libre, les mises à jour pourront

être effectuées par la communauté des utilisateurs actifs, ce qui signifie aussi que le coût de modification du logiciel sera partagé par l'ensemble des utilisateurs actifs.

Acheter ou développer ?



Quelle conclusions peut t-on tirer de la comparaison de ces approches selon ce modèle d'évaluation des coûts?

- Développer peut être rentable

La première conclusion est que développer un logiciel – libre ou non – peut être rentable dès que:

- le coût d'acquisition des licences est important (par exemple s'il faut équiper plusieurs milliers de postes, ex : administrations, grands comptes…);
- le fournisseur met du temps à corriger les bogues;
- la technologie des logiciels existants entraîne des coûts importants d'adaptation;
- ou lorsque les logiciels commerciaux ne sont pas fiables et présentent de coûteux défauts (ex : serveur web pour une entreprise de commerce électronique).

Rendre ensuite ce logiciel libre permet de bénéficier de la réactivité des utilisateurs passifs qui, en signalant les bogues à l'avance, permettent de corriger celles-ci avant qu'elles n'aient de conséquences. L'émergence d'une communauté d'utilisateurs actifs permettra enfin de partager avec eux le coût des modifications et de bénéficier de la réactivité de l'ensemble des autres utilisateurs actifs.

Autrement dit:

Lorsque l'offre de logiciel (libre on non) est insuffisante ou chère, il vaut mieux s'associer pour un créer un nouveau logiciel libre.

- L'indépendance a un prix

La deuxième conclusion que l'on peut tirer ce modèle est que si une panne coûte très cher à l'entreprise et si l'éditeur de logiciel réagit lentement, il vaut mieux (si possible) acquérir le code source du logiciel et corriger soi-même les bogues. Dans ce cas, le logiciel libre dispose d'un avantage financier potentiel puisque non seulement son code source est gratuit mais que le coût des modifications peut être partagé entre plusieurs utilisateurs actifs. De plus certains éditeurs refuseront certainement de divulguer ce qu'ils considèrent comme des informations stratégiques, il suffit de voir la réticence qu'ont certains à seulement diffuser les spécifications nécessaires à la créations de pilotes par les fabricants de hardware.

Autrement dit:

Il vaut mieux aider une petite société à développer un logiciel libre plutôt que de lui acheter un logiciel commercial et de le mettre à jour régulièrement.

- Les grands utilisateurs ont intérêt à corriger les défauts eux-mêmes

Lorsque le délai de réaction de la communauté des utilisateurs actifs d'un logiciel libre coûte trop en dysfonctionnements à certains utilisateurs, ces derniers ont intérêt à modifier eux-mêmes le logiciel et à publier la modification. Ainsi, le nombre d'utilisateurs actifs croît, ce qui fait automatiquement diminuer le délai moyen de réaction jusqu'à ce que l'on atteigne un état d'équilibre : chaque utilisateur corrige le bogue qui compte le plus pour lui. Notons que, selon de nombreuses études, les logiciels libres ainsi maintenus sont aujourd'hui les moins bogués du marché.

Autrement dit:

Il existe un équilibre économique garantissant un nombre d'utilisateurs actifs suffisant pour satisfaire les utilisateurs les plus exigeants.

- du point de vue d'un utilisateur passif

Toutes les entreprises utilisatrices ne choisiront pas de modifier elles-mêmes le logiciel, mais le fait de pouvoir sous-traiter cette activité est très important On constate immédiatement que le logiciel libre, en supprimant le coût des licences, autorise une marge appréciable en termes d'adaptation, de réactivité et de fiabilité. Si l'on omet les paramètres réactivité et fiabilité, ce que l'on gagne en licences non payées peut servir à payer des ingénieurs pour adapter le logiciel libre aux besoins de l'entreprise et à former les utilisateurs.

Ainsi le logiciel libre, adapté sur place, est créateur de plus d'emplois locaux que le logiciel commercial importé.

- **Quelques exemples concrets suivant ces trois scénarios**

Exemple 1 : un serveur Web

Si l'entreprise souhaite installer un serveur Web, les choix possibles sont par exemple le serveur Web Microsoft sous Windows NT ou 2000 et le serveur Apache sous RedHat Linux. Ces deux serveurs sont similaires du point de vue des fonctionnalités mais Apache est réputé plus fiable et ses bogues sont corrigées plus rapidement. À moins d'une bonne raison, il vaut donc mieux choisir Apache puisqu'il coûte moins cher au départ et moins cher ensuite chaque année.

Exemple 2 : une base de données

Comparons maintenant deux petits serveurs de base de données ; considérons par exemple le logiciel commercial FileMaker Pro (MacOS) et le logiciel libre MySQL (Linux). Adapter FileMaker Pro pour définir une application d'entreprise demande moins d'effort qu'avec MySQL. En revanche, MySQL est plus éprouvé et son éditeur, TCX, corrige les bogues plus rapidement que l'éditeur de FileMaker..

Le choix du logiciel libre est le meilleur si le nombre de postes à équiper est important ou si l'application est stratégique. Le surcoût d'adaptation du logiciel libre par rapport au logiciel commercial sera probablement compensé par l'absence de licence et, le cas échéant, rentabilisé sur le long terme grâce à une meilleure fiabilité et une meilleur réactivité. Un exemple intéressant est celui du moteur de recherche Google, qui utilise plus de 4000 serveurs à base de PC bon marché, fonctionnant en réseau sous Linux. Ce fut pour cette société un moyen d'obtenir une puissante base de données à faible coût et dont la puissance peut-être facilement adaptée en augmentant le nombre de postes.

Google, le plus puissant des moteurs de recherche fonctionne à partir de 4000 systèmes Linux en réseau.

En revanche, si l'objectif est de monter très rapidement et en un seul exemplaire une base de données moyennement fiable, mieux vaut choisir le logiciel commercial car il reviendra moins cher à adapter. A petite échelle les solutions commerciales peuvent s'avérer plus intéressantes.

Exemple 3 : un logiciel de mise en page de documents

Comparons maintenant FrameMaker, un logiciel commercial de mise en page scientifique pour MacOS, et KLyX, un logiciel libre de publication scientifique pour Linux.

FrameMaker permet de réaliser des mises en pages complexes plus rapidement que KLyX grâce à une interface élégante qui gère les modèles de mise en page, les feuilles de styles, le placement des images, les liens hypertexte, la séparation quadrichromique, la conversion HTML, etc. Mais KLyX offre des avantages par rapport à FrameMaker pour l'automatisation du processus de rédaction et la gestion des formules mathématiques complexes.

FrameMaker est un produit stable. KLyX est un logiciel en développement qui fonctionne bien pour des tâches simples mais doit encore évoluer pour offrir que même degré de convivialité que FrameMaker pour les tâches complexes.

Dans une logique de court et moyen terme, le logiciel libre KLyX ne sera choisi que si le nombre de postes devant être équipé est élevé et si le coût des bogues est bas. Autrement dit, pour faire taper des documents simples à de nombreux employés peu payés, l'utilisation d'un logiciel libre de moins bonne qualité qu'un logiciel commercial peut être rentable.

Exemple 4 : développement d'un produit industriel

On peut ici reprendre l'exemple de la firme Lectra qui a envisagé il y a 4 ans deux systèmes d'exploitation pour développer son système de CAO textile : Windows et Linux.

- D'un point de vue économique, c'est Linux qu'il fallait choisir en raison de sa fiabilité.
- D'un point de vue commercial, c'est Windows qu'il fallait choisir en raison de sa notoriété.
- D'un point de vue stratégique, c'est Linux qu'il fallait choisir car l'accès au code source est absolument nécessaire pour pouvoir adapter, le cas échéant, le logiciel système aux contraintes très particulières d'un produit utilisé en milieu industriel où le coût des bogues est très élevé.

C'est donc Linux qui a été choisi. Cependant , ce choix pourrait être remis en question si Windows était plus fiable et si son code source était disponible.

Exemple 5 : production multimédia

Dans le domaine de la production multimédia, il n'existe pas encore de solution intégrée et stable à base de logiciels libres. Ce type de solution est coûteux à développer ou à adapter alors que de nombreux produits commerciaux concurrents existent déjà et ne coûtent que quelques milliers de francs, c'est à dire relativement peu pour un usage professionnel. Le secteur du multimédia reste un cas particulier, les salaires des producteurs multimédias étant élevés – donc le coût des bogues aussi – et les produits commerciaux très fiables. Comme la production multimédia intéresse essentiellement des professionnels pressés et peu habitués à développer eux-mêmes des logiciels complexes, il est peu probable que se développe une communauté d'utilisateurs actifs de logiciel libre pour la production multimédia en dehors de certaines niches.

- **la part de marché du libre**

De façon concrète, de plus en plus d'entreprises commerciales utilisent quotidiennement des applications basées sur des logiciels libres.

Une liste des plus grands noms se trouve sur http://www.linux-france.org/article/lbiz-fr parmi lesquels on trouve de nombreuses multinationales et services gouvernementaux.

Si la part de marché des logiciels libres est difficilement estimable, elle reste plus facile à évaluer sur les serveurs que sur les postes de travail. Et la part de marché de Linux, le logiciel libre le plus célèbre, est un bon estimateur de l'importance du phénomène des logiciels libres en entreprise :

OS – Part de marché	1998/1999	1999/2000
Windows NT	38	38
Linux	16	25
Netware	23	19
Unix	19	15
Autres	4	3

Source: IDC

Cette évolution fait dire à l'analyste Dan Kuznetsky, de la société IDC, que *« la part de marché de Linux croît beaucoup plus vite que nous ne l'avions prévu. Nous projetions qu'il serait numéro 2 en 2002 ou 2003. Et c'est arrivé en 1999 »*. En 1998, la part de marché de Linux a augmenté de 212%, et en 1999 de 156%, tandis que le marché des systèmes d'exploitation des serveurs dans sa globalité ne croissait pour sa part que de 23%. Cela tend à prouver l'intérêt économique de Linux dans un environnement professionnel.

Cependant, ce modèle ne serait pas complet sans souligner les limites que peut aussi avoir l'approche en faveur des logiciels libres dans le cadre d'un usage en entreprise.

C) Les limites de l'approche « logiciels libres » en entreprise :

Le modèle du « logiciel libre » présenté ci-dessus offre pour l'utilisateur de tels avantages en termes de coûts, de souplesse, de fiabilité et de sécurité qu'on pourrait être tenté d'exclure toute autre approche. Ce serait oublier les coûts et contraintes cachés qui n'apparaissent pas dans ce modèle de prise de décision.

- Un coût de migration important

Ce modèle néglige en effet un coût important : le coût de migration d'un système vers un autre, appelé aussi coût de sortie. Il ne suffit pas de savoir que le coût d'entrée et le coût annuel d'une solution à base de logiciels libres est le plus bas si l'on doit par ailleurs investir des sommes considérables pour sortir d'une solution propriétaire et migrer vers celle des logiciels libres. Il convient donc d'évaluer si les gains de fiabilité et de réactivité de la nouvelle solution, ainsi que l'économie des renouvellement des licences, permettent de rentabiliser le coût de sortie.

Les grands éditeurs ont bien compris l'intérêt pour eux du coût de sortie. Une application de *WorkFlow* réalisée dans l'environnement Lotus Notes est par exemple très coûteuse à porter dans un autre environnement en raison de caractère propriétaire

du langage Lotus Script utilisé pour définir les flux d'information. Ce langage n'est en effet disponible que dans l'environnement Lotus. Il existe bien des solutions équivalentes sous Linux (Perl, TCL etc.) mais elles ne sont pas compatibles avec Lotus Script et nécessiteraient de redéfinir les flux d'information dans un autre langage.

De même, une solution de base de données réalisée avec Microsoft Access et Visual Basic nécessite pour fonctionner la quasi-totalité de l'offre bureautique Microsoft ce qui rend impossible le portage vers une solution à base de logiciel libre à moins de consentir un effort important de développement. L'encouragement de formats de données publics (HTTP, XML, MIME…) et de protocoles de communications ouverts (SMTP, IMAP…) est d'ailleurs une condition importante pour éviter que des phénomènes d'incompatibilités ne rendent impossible la sortie du modèle propriétaire.

- Le coût des licences n'est pas linéaire

Contrairement à ce modèle, les éditeurs de logiciels commerciaux ne vendent pas les logiciels à un prix proportionnel aux nombres de postes. En fait, il existe des ristournes, allant jusqu'à la gratuité, et des prix de gros qui sont destinés à dissuader l'emploi de logiciels gratuits, du moins quand les menaces sont suffisantes. C'est pourquoi, le fait de faire appel à un bureau d'évaluation pour comparer logiciels libres et logiciels commerciaux conduira l'éditeur à baisser ses prix. Le phénomène de concurrence entre éditeurs permet de remettre en cause ce modèle

- Logiciel libre et multimédia

Dans le domaine de la production multimédia, il n'existe pas de solution intégrée et stable à base de logiciels libres. S'il existe en effet de nombreux logiciels libres pour le multimédia comme Gimp (retouche d'images), Mesa (affichage 3D), POV (rendu d'images de synthèse), Moonlight (modélisation 3D), Multitrack (enregistrement audionumérique multipiste), XAnim (lecture audio et vidéo)…, ces logiciels ne gèrent pas certaines fonctionnalités avancées nécessaires dans un cadre professionnel.

La calibration colorimétrique, par exemple, n'est pas gérée par les logiciels libres. Les formats vidéo les plus avancés et les séquences composites (mélange de son, texte, image, vidéo, 3D etc.) ne sont pas non plus supportés. Le copier-coller ne fonctionne pas pour les données multimédias et comme chaque application utilise ses propres bibliothèques de routines et ses propres formats, l'intégration de données disparates dans un même document est quasi impossible. Quant au développeur, il ne dispose pas d'une bibliothèque de routines libres lui permettant de développer rapidement une application multimédia capable de gérer à la fois la calibration colorimétrique, les séquences composites, l'accès aux périphériques, la synchronisation, la 3D, etc.

Au contraire, l'offre de logiciel commerciale dans le domaine du multimédia est très riche et la concurrence joue parfaitement son rôle de moteur de l'innovation et d'ajustement des prix. Le graphiste professionnel, qui est souvent free lance et n'a pas

l'habitude de développer ses propres outils, peut donc difficilement justifier économiquement l'utilisation d'un environnement de logiciels libres relativement pauvre alors qu'existe un foisonnement de solutions commerciales très accessibles qui maximisent sa créativité et sa productivité. Le développeur, qui doit produire rapidement une application utilisant toute la richesse du multimédia, choisira probablement la bibliothèque Quicktime car c'est la seule à couvrir tous ses besoins à la fois sous MacOS et sous Windows pour un coût de licence quasiment nul. Le développeur qui souhaite bénéficier d'une gestion de la couleur parfaite utilisera probablement ColorSync sous MacOS faute de concurrence sérieuse sous Windows.

Or, si ni les utilisateurs ni les développeurs d'applications multimédias n'ont d'intérêt à s'orienter vers une approche « logiciel libre », il est peu probable que de telles applications voient le jour à moins qu'un autre acteur ne favorise l'émergence de technologies libres pour la production multimédia et pour la calibration colorimétrique. Dans le cas du multimédia, cet acteur pourrait être un grand laboratoire public ou l'industrie de l'électronique grand-public, qui est aujourd'hui menacée par la domination sans cesse croissante d'Apple et de Microsoft sur les technologies de diffusion vidéo par Internet .

3 – Le logiciel libre, une opportunité politique ?

A) Le logiciel libre, un régulateur de l'économie de marché :

Ce qui est bon sur un marché parfait avec des rendements décroissants ne l'est pas forcément dans une industrie du logiciel dont les caractéristiques - présences d'oligopoles, rendements croissants et coûts marginaux nuls - conduisent parfois à des tarifications abusives et à des situations contraires à l'intérêt général. En effet, le cadre juridique du marché du logiciel facilite l'exécution de stratégies de domination et de rachats particulièrement efficaces, fondées sur l'appropriation des normes de communication au profit d'un seul acteur.

Les logiciels libres permettent de corriger ces dérives, comme l'illustre le serveur Web Apache. En effet, lorsque le standard du marché est un logiciel libre, les stratégies d'appropriation des normes de communication deviennent inopérantes. Les logiciels libres agissent alors comme régulateurs du marché et favorisent l'émergence d'une offre diversifiée de logiciels commerciaux compatibles.

Les logiciels libres agissent donc comme un régulateur naturel du marché dans une industrie secouée de façon récurrente par des crises monopolistiques. En garantissant l'existence de normes de communication publiques largement adoptées, ils favorisent l'interopérabilité entre produits concurrents, stimulent la diversité de l'offre, modèrent les prix et définissent des standards de qualité.

B) Une opportunité pour développer des industries logicielles indépendantes

- en Europe

Les logiciels libres peuvent aussi permettre à des pays de développer une industrie logicielle indépendante à moindre coût, en récupérant l'ensemble des connaissances ainsi partagées et le travail effectué. La possibilité pour l'Europe de définir le contenu des distributions Linux en combinant logiciels libres et logiciels propriétaires offre les moyens de s'affranchir de la domination commerciale américaine dans l'industrie du logiciel. Les éditeurs de distributions européens (SuSE, Mandrake, etc.) jouent d'une certaine façon le même rôle qu'Airbus dans leur capacité à rassembler des qualités qui, individuellement, n'ont pas assez de puissance pour s'imposer sur la scène mondiale.

Linux et les logiciels libres sont donc une véritable opportunité pour les européens. Ils permettent aux consommateurs de s'affranchir des standards propriétaires que certains grands éditeurs américains tentent d'imposer. Ils conduisent à des créations d'emplois plus nombreuses dans les services aux entreprises. Ils garantissent aux éditeurs européens de logiciel, qui sont souvent des PME, de pouvoir lutter à armes égales avec les géants du secteur.

En particulier dans le domaine de l'éducation, Linux est une option particulièrement intéressante pour améliorer l'informatique dans les écoles et permettre une réduction importante des coûts d'équipement des établissements. En effet, les coûts réduits en licences se combinent avec de faibles besoins en maintenance et en matériel, un système Linux pouvant fonctionner sur un ordinateur plus ancien (Intel 486) avec des performances très intéressantes. De plus l'ouverture du code source peut largement être profitable dans un cadre pédagogique et éviter la monoculture des utilisateurs habitués à une seule plate-forme.

Ainsi, grâce à la réutilisation de vieux ordinateurs et à la gratuité de plusieurs logiciels tournant sous Linux, il est possible à peu de frais pour les étudiants:

- d'avoir accès à l'ordinateur comme outil pour rédiger des rapports, pour faire des recherches sur l'Internet...
- d'apprendre une plus grande quantité de logiciels;
- d'utiliser l'ordinateur pour des cours de sciences, d'arts... (en plus des cours d'informatique pour lesquels il est particulièrement adapté).

et pour les enseignants:

- d'avoir leur propre ordinateur comme outil de travail et d'échange d'information
- de se monter à peu de frais un Intranet pour le partage avec leurs collègues d'information, d'imprimante, et d'accès Internet.

Pratiquement toutes les universités utilisent déjà Linux dans un ou plusieurs laboratoires d'enseignement et de recherche. Un grand nombre de collèges en font de même (comme celles recensées sur http://www.cam.org/~ycd/colleges.html) , ainsi que

des écoles primaires. Au Québec par exemple, plusieurs écoles primaires et secondaires ont installé ou vont bientôt installer Linux dans des salles de classe et des laboratoires; réutilisant ainsi du matériel devenu obsolète. D'autres initiatives dans l'éducation sont aussi décrites par l'association française des utilisateurs de Linux (http://www.aful.org/education).

- dans les pays en voie de développement

Pour les pays en voie de développement, les logiciels libres sont encore davantage un moyen de promouvoir massivement un équipement logiciel de qualité et à faible coût et de développer une industrie logicielle indépendante. Là où les coûts de licences peuvent aller jusqu'à plus de 20% du coût global de l'équipement, les logiciels libres permettent d'une part d'en faire l'économie tout en réinvestissant l'argent gagné dans davantage de matériel, et d'autre part de réduire l'obsolescence du matériel existant, vu que les faibles ressources demandées par des systèmes comme Linux et le respect de standards généraux allongent la durée de vie utile du matériel. Il existe déjà quelques projets totalement indépendants visant à recycler dans le Tiers Monde le matériel mis au rebut par les entreprises des pays développés, par exemple pour l'accès à Internet dans des régions rurales d'Afrique ou l'équipement d'écoles. Bien que de nombreuses tracasseries administratives puissent exister, le potentiel d'équipement est réellement favorisé par les logiciels libres.

Deux pays sont particulièrement représentatifs des gains que peuvent apporter les logiciels libres dans les pays en développement, que ce soit pour favoriser le développement des technologies de l'information comme en Inde avec le **Simputer**, ou pour aider à la création d'une industrie logicielle nationale comme en Chine avec **RedFlag Linux**.

*** l'Inde et le Simputer**

Pour démocratiser l'accès à l'information et aux nouvelles technologies dans un pays où 80% de la population est analphabète, un projet open source basé sur des technologies ouvertes tente de réaliser au plus faible coût un ordinateur qui soit adapté aux besoins du plus grand nombre. Le projet SIMPUTER (pour Simple, Inexpensive and Multilingual comPUTER) a été initié par un groupe d'étudiants de l'Indian Institute of Science (IISc) de Bangalore depuis 1999. L'objectif est de proposer une machine simple et économique d'accès à l'Internet pour des personnes illettrées, qui puisse permettre un grand nombre d'utilisations comme le courrier électronique par reconnaissance vocale, la réalisation de micro paiements ou l'accès à une information réellement utile pour ses utilisateurs. De nombreux détails ont été étudiés pour que personne ne soit vraiment exclu de son utilisation, comme l'absence de clavier remplacé par un écran tactile, son coût équivalent à celui d'un petit téléviseur, son interface graphique très simplifiée ou l'utilisation de « smart cards » personnelles afin que chacun aie le moyen de stocker ses données personnelles et de ne se servir du Simputer que comme d'une borne d'accès à l'information. Des versions wireless sont également prévues pour les nombreuses localités n'ayant pas accès au téléphone.

Un aspect intéressant de ce projet est son ouverture totale, tant logicielle que matérielle. En effet, son système d'exploitation basé sur Linux sera associé à une plate forme matérielle aux spécifications ouvertes, ne reprenant rien de propriétaire. Dans le but de réduire au maximum le prix de cet appareil, sa fabrication sera accordée sous licence par le Simputer Trust, organisation à but non lucratif, qui définira les orientations matériels du Simputer afin que celui ci garde son unité et sa compatibilité. De plus l'ouverture du système a pour but de permettre à des développeurs et fabricants d'apporter eux-mêmes certaines améliorations de manière transparente.

Ce projet ambitieux est pour l'Inde un des meilleurs moyens de réduire le fossé numérique et de permettre une véritable démocratisation de l'information dans un pays qui compte moins de 5 millions d'ordinateurs pour plus d'un milliard d'habitants. Les technologies Opensource sont un aspect déterminant de ce projet où les coûts sont la contrainte la plus importante. Pour l'instant le Simputer a déjà accordé plusieurs licences de fabrication et ses premiers prototypes sont sortis au mois d'Avril 2001.

Un des premiers prototypes du SIMPUTER

- **la Chine et Red Flag Linux**

De son coté la Chine a décidé de se servir des technologies Open sources et de Linux en particulier pour assurer son indépendance informatique et créer une véritable industrie nationale du logiciel à partir des ressources du logiciel libre.

Désirant s'affranchir des logiciels de Microsoft, soupçonnés de contenir des trous de sécurités intentionnels, la Chine a rendu obligatoire dans l'administration l'usage du système Red Flag Linux, dérivé de Red Hat et adapté à la langue chinoise. Dans ce cas précis, l'usage de logiciels libre est avant tout à but politique, la Chine ayant toujours préféré développer elle-même ses technologies. Red Hat Software est en effet fortement soutenu par le gouvernement, son PDG étant d'ailleurs le fils du président chinois. Linux a permis à la Chine d'utiliser un standard international, qui n'appartenait à aucune entreprise, ni à aucun pays. Dans ce pays au si fortes potentialités dans le domaine des nouvelles technologies, le recours au logiciel libre est un moyen

d'affirmer son indépendance et de renforcer sa puissance technologique. Le président de Red Flag soulignait récemment qu'il y a davantage d'étudiants en Chine que d'habitants aux Etats-Unis, d'où sa confiance dans le potentiel de la distribution de Linux chinoise. Son adoption a grande échelle, bien qu'imposée, permettra sans doute au pays de rattraper son retard logiciel bien plus vite et plus économiquement que par tout autre moyen, tout en assurant le développement d'un éditeur national très rentable et coté à la bourse de Hong-Kong.

RedFlag, le Linux Chinois

Bilan : Le logiciel libre est donc tout à fait viable économiquement, autant pour les éditeurs de logiciels que pour les entreprises et même au niveau des Etats. S'il permet de faire jouer davantage certains aspects de l'économie de marché menacés par la tendance au monopole de l'industrie logicielle propriétaire, il est avant tout un moyen de s'assurer une plus grande maîtrise de son équipement logiciel et des coûts qu'il entraîne. Si les logiciels propriétaires conservent leur intérêt pour un certain nombre d'utilisations, l'influence grandissante du modèle économique libre est sans doute révélatrice de sa future importance, tant pour les entreprises que pour les pays.

IV) Interrogations et conjectures sur l'avenir du mouvement Open source :

1) Quel avenir économique pour les logiciels libres ?

Avec tous ces éléments, quel avenir peut-on donc raisonnablement envisager pour les logiciels libres ? Si leurs qualités techniques et leur viabilité économique sont de plus en plus évidentes, quels paramètres peuvent influencer leur évolution à moyen terme et celle-ci ne risque t-elle pas de dévier de ses origines avec la professionnalisation et la marchandisation qui vont de plus en plus être associées au logiciel libre ?

Tout d'abord on peut sans risque prévoir un essor très important du logiciel libre, d'abord en entreprise pour les nombreux avantages qu'il apporte au fur et à mesure de son développement et de sa diversification. Le logiciel libre grand public n'est peut-être pas amené à se développer dans un avenir très proche, du moins tant que la facilité d'utilisation et la logithèque associée ne seront pas supérieures au logiciels propriétaires. En effet de fortes résistances accompagnent toujours l'acceptation d'une nouvelle manière de penser et les inerties dans les comportements des consommateurs sont habituellement élevées, particulièrement dans un domaine comme l'informatique. Pour le grand public, l'apparente complexité d'utilisation de cet outil pousse au rejet de toute nouveauté par peur de ne pas la maîtriser suffisamment, et à se contenter de ce que l'on a. La séduction que peut véhiculer l'aspect « culture alternative » des logiciels libres ne touche qu'une minorité de passionnés, qui sont d'ailleurs souvent ceux qui font le plus avancer l'idée du logiciel libre. Cependant, la dynamique du développement Open source devrait à terme aboutir à la création de logiciels supérieurs en tous points au logiciels propriétaires, tant au niveau technique que pour la facilité d'utilisation. En effet, comme le montre le caractère darwinien de l'évolution des logiciels libres, la meilleure solution s'impose de façon bien plus rapide que partout ailleurs. Si la facilité d'utilisation n'est pas la préoccupation première des développeurs traditionnels, elle devient un aspect prioritaire pour les éditeurs commerciaux de distributions grand public, comme Mandrake ou Red Hat. Par conséquent, le logiciel libre a de bonnes chances d'acquérir une part significative du marché des logiciels, de l'ordre de 30% d'après Linus Torvalds lui-même qui n'en espère d'ailleurs pas plus.

De plus au niveau des éditeurs de logiciels, un soutien croissant des logiciels libres devrait accompagner cette montée en puissance, d'autant plus que les plus grands noms des fabricants de matériel apportent un support de plus en plus marqué des logiciels libres. En effet, ceux-ci sont pour eux un moyen de proposer leurs propres solutions logicielles sans passer par des coûts de développement non supportables. Les logiciels libres vont leur permettre de réellement valoriser leurs solutions matérielles et de contrôler davantage leur marché particulier, comme le rappelle Colin Tenwick, Vice-Président de Red-Hat Europe :

> *« Par ailleurs, des gens comme IBM ont bien compris que leur précédent schéma était voué à l'échec. On ne contrôle pas un marché avec une solution propriétaire, lourde à entretenir. Au*

contraire, ils apportent désormais énormément au monde des logiciels libres en affectant d'énormes ressources. Ils en retirent toute l'innovation qui est liée à ce monde-là. Pour nous, c'est une très bonne nouvelle qu'ils investissent dans ce domaine. Aujourd'hui, les contributions majeures viennent de boîtes comme IBM, pas de hackers au fond d'un garage. »

(**Colin Tenwick,** VP & General Manager Europe Middle East & Africa de Red Hat, http://www.transfert.net/fr/dossiers/article.cfm?idx_rub=87&idx_art=3873**)**

Ainsi, d'un point de vue économique, les logiciels libres devraient voir leur influence et leur support se généraliser, sans pour autant supplanter les logiciels propriétaires qui gardent certains atouts, notamment sur le marché grand public. D'une façon plus générale, le modèle de développement des logiciels libres apparaît bien plus viable dans une économie de plus en plus tournée vers l'internet. est bien davantage adapté à l'économie de plus en plus basée sur l'échange d'informations

"Microsoft's centrally planned economy simply cannot compete with Linux's decentralized capitalism in the Internet era."
(**Jeff Prothero** "Cynbe ru Taren" : http://muq.org/~cynbe/rants/lastdino.htm)

2) l'influence grandissante des multinationales dans le logiciel libre

Cependant, si l'implication de plus en plus forte des multinationales de l'informatique dans le logiciel libre peut s'avérer un élément essentiel de son développement, ne menace t-elle pas aussi la cohésion et l'idéal désintéressé de tous les programmeurs volontaires qui constituent les meilleurs spécialistes du logiciel libre ? En effet, chez IBM 2000 personnes vont désormais se consacrer à Linux, ce qui en fait le principal vecteur du succès commercial pour les solutions Linux et plus globalement OpenSource. On peut y voir une source de conflit entre la communauté de développeurs à but non lucratif et les objectifs commerciaux de ces nouveaux contributeurs, comme le remarque George Weiss, un analyste financier cité par crn.com :

"Au début du développement en open source, tout était fondé sur la gratuité. Mais si les investisseurs se sont impliqués dans le développement de Linux, c'est pour réaliser des profits. Ils y réfléchiront à deux fois avant de parler gratuité à leurs actionnaires. Il va y avoir de plus en plus de tensions entre l'idéalisme d'une communauté Linux attachée à l'OpenSource et les fortes perspectives de profits des grandes entreprises vendeuses de solutions autour de Linux."

(http://www.crn.com/Components/Search/Article.asp?ArticleID=23424)

Pourtant, un élément fondamental de stabilité de logiciels libres comme Linux, qui est en fait le plus concerné, reste la présence d'un gourou comme Linus Torvalds qui décide en dernier lieu des orientations et modifications à apporter au système. Ne peut-il se voir contester ce droit par les industriels de l'informatique, dont les intérêts peuvent s'avérer différents de ceux de la communauté ? Ce danger a été soulevé par un article de crn.com (http://www.crn.com/Components/Search/Article.asp?ArticleID=23368) , d'après lequel beaucoup d'acteurs se plaindraient du peu de disponibilité de Linus Torvalds et voudraient un interlocuteur présent à plein temps. Cependant dans une autre interview de crn.com, Linus Torvalds nie subir une quelconque pression de la part d'IBM ou d'autres industriels de l'informatique Entre parenthèses on peut noter un certain parti pris de ce magazine qui tente systématiquement chercher une contradiction dans les propos du coordinateur de Linux, ce qui peut faire penser en effet à une pression du lobby de l'informatique dans ce sens. Cependant Linux semble être protégé contre le contrôle du développement par d'autres que Linus Torvalds, ce dernier ayant déposé le nom Linux en tant que marque commerciale. De plus, Stéphane Fermigier, président de l'Association francophone des utilisateurs de Linux, ne voit pas de menaces planer au-dessus de Torvalds.

> *"Le noyau de Linux est suffisamment souple pour que les gros industriels puissent développer leurs propres applications sans nuire à l'organisation générale du système*"

S'il est difficile d'imaginer quels moyens de pression les industriels pourraient exercer sur un univers aussi hétéroclite et différent du leur, ces derniers semblent pourtant accepter de s'en remettre en partie à la communauté OpenSource pour ce qui est des orientations futures du développement de Linux. Ainsi, le mode de développement de Linux n'est à priori pas remis en cause par l'implication de l'industrie informatique, celle ci ayant apparemment intégré les avantages du mode de développement communautaire, pourtant à l'opposé de leur raisonnement concurrentiel traditionnel. Cela tient là aussi en grande partie aux qualités du coordinateur, Linus Torvalds apparaissant comme la personne la plus compétente pour s'occuper du développement futur de Linux, comme l'admet Dan Frye, directeur du Linux Technology Center d'IBM à Beaverton :

> *"It's not going to be what IBM or Transmeta wants, all of the time. And we accept that" (…) "Linus is the leader, but not a dictator. He has changed his mind repeatedly in the past when the community convinces him to move in a different direction, and a good indication of that is [improving] SMP support in the kernel."*

A priori, malgré les intérêt énormes en jeu, avec par exemple les investissements d'un milliard de dollars d'IBM dans Linux en 2001, la communauté du logiciel libre ne devrait donc pas être menacée par l'implication des industriels dans le développement de Linux, grâce à la dynamique du développement collaboratif et à l'ouverture d'esprit de Linus Torvalds. L'approche délibérément désorganisée est en fait une des conditions du développement équilibré et performant de l'open source, et revenir à un

mode d'organisation plus centralisé ne servirait que des intérêts à court terme, ce que semblent comprendre des entreprises comme IBM ou Compaq.

3) la menace des brevets logiciels :

Une autre menace pour le développement des logiciels libres est la polémique actuelle sur la nécessité de breveter les logiciels. Le droit européen, en excluant la brevetabilité des logiciels et en garantissant une protection élevée des consommateurs, définit un cadre juridique bien plus favorable au développement des logiciels libres qu'au Japon ou aux Etats-Unis. Mais, sous la pression américaine, l'Union Européenne envisage d'étendre la brevetabilité aux logiciels. Une telle mesure, en limitant considérablement le pouvoir de régulation économique des logiciels libres, irait dans un sens contraire à l'intérêt général des européens puisque la plupart des grands éditeurs de logiciels sont américains alors que les éditeurs de logiciels européens sont des PME trop petites pour bénéficier en pratique d'une quelconque protection par le brevet. La brevetabilité irait à aussi à l'encontre de l'idée de propriété collective associée aux logiciels libres en rendant possible l'appropriation individuelle de concepts simples comme des méthodes intellectuelles (par exemple le *1-Click* d'Amazon.com), des formats de fichiers ou encore des algorithmes, indispensables à la création de logiciels. En permettant de breveter tout cela, il deviendrait quasiment impossible d'écrire un programme informatique dont le contenu ne soit pas lié d'une multitude de brevets. « *On pourrait objecter que toutes les entreprises, petites ou grandes, sont logées à la même enseigne* », fait remarquer Eitel Dignatz, un consultant allemand spécialisé dans les nouvelles technologies. « *Mais ce serait ignorer le problème des licences croisées. Les grands éditeurs de logiciels disposent en effet d'une montagne de brevets. Et souvent, ils pratiquent entre eux un jeu d'échange* »

De plus, il existe une inconnue concernant la validité juridique de la licence GPL, base légale des logiciels libres, bien que celle-ci donne satisfaction depuis sa création.

Pourtant face aux lobbying des grands éditeurs propriétaires, il existe de nombreuses initiatives politiques en faveur des logiciels libres et contre les brevets, comme le rapport du Député du Tarn Thierry Carcenac au Premier Ministre (Avril 2001), qui recommande de *« préserver la pluralité et la complémentarité des approches»* et présente un certain nombre de mesures en faveur du développement de standards ouverts et de logiciels libres dans l'administration (http://www.premier-ministre.gouv.fr/ressources/fichiers/carcenac.pdf). Catherine Tasca, ministre de la culture et de la communication, a déclaré elle aussi lors de la Conférence internationale sur la gestion et l'utilisation légitime de la propriété industrielle son opposition aux brevets logiciels :

> *" L'œuvre de l'esprit, une idée, une formule mathématique, des codes logiciels, une expression formelle nouvelle, ne sauraient faire l'objet d'une brevetabilité sans précaution pour éviter le risque de tarissement de la création. "*
>
> http://www.pro-innovation.org/rapport_brevet/html/resume.html

Conclusion

Les logiciels libres sont donc un modèle économique tout à fait viable, tant pour les éditeurs de logiciels que pour leurs clients. S'ils présentent de plus en plus d'avantages économiques avec leur amélioration constante et le soutien croissant d'industriels de l'informatique, ils permettent surtout de choisir une voie stratégique vers davantage d'indépendance vis-à-vis des fournisseurs traditionnels en logiciel. Mais ils ont également nombre de conséquences positives pour l'économie en général, en s'avérant paradoxalement un régulateur de l'économie de marché, parfois étouffée par le caractère monopolistique de l'industrie propriétaire. Au niveau national, leur promotion croissante sous diverses formes montre bien l'intérêt pour ce type de logiciels ouverts, de propriété collective et à la haute qualité technique.

Il est également intéressant de noter à quel point la mise en réseau d'individus suivant des structures de production totalement déstructurées et décentralisées peut s'avérer le meilleur moyen de parvenir à la performance optimale dans la création d'un bien. Plus que l'intérêt économique de ces logiciels, je crois que c'est davantage leur mode de production et sa gestion des ressources humaines qui est à retenir comme modèle.

Bibliographie et références utilisées

Rapports et études :

Rapport gouvernemental Carcenac sur l'administration électronique (Avril 2001)
http://www.premier-ministre.gouv.fr/ressources/fichiers/carcenac.pdf

Bernard Lang (INRIA)
"Ressources Libres et Indépendance Technologique dans les Secteurs de l'Information" (1997)
http://pauillac.inria.fr/~lang/ecrits/hanoi/

Mémoire de Nicolas Leclercq
« Logiciel libre : une volonté de transparence »
http://www.linux-france.org/article/these/memoire-leclercq/memoire_leclercq_monoblock.html

Antoine Brisset, Rapport de fin d'études à l'IAE de Paris
« le marché du système d'exploitation GNU/Linux »
http://www.linux-france.org/article/these/memoire-brisset/brisset_monoblock.html

Mémoire de Mélanie Clément-Fontaine
« La licence publique générale » (étude juridique)
http://crao.net/gpl/gpl.pdf

Dossiers :

Multimédium :
« L'économie du logiciel libre » par Jean-Paul Smets-Solanes
http://www.mmedium.com/dossiers/libre/

« Gnu/Linux dans les écoles québecquoises, un choix de société »
par Joël Pomerleau (Mai 2000)
http://www.mmedium.com/dossiers/linux_ecoles/index.html

Essais :

Eben Moglen, Professeur de droit et d'histoire légale à l'école de droit de Columbia
« L'anarchisme triomphant : Le logiciel libre et la mort du copyright »
http://emoglen.law.columbia.edu/publications/anarchism-fr.html

Jeff Prothero "Cynbe ru Taren"
« Le Dernier Dinosaure et les Mares de Goudron du Destin »
http://www.muq.org/~cynbe/rants/lastdino-fr.html

Christopher B. Browne
« Linux et le développement décentralisé »
http://www.linux-france.org/article/these/lsf-fr/lsf-fr_monoblock.html

Kadda SAHNINE (1998)
« Un plaidoyer pour une Informatique Libre »
http://www.linux-france.org/article/these/sahnine/sahnine_libre_monoblock.html

Richard M. Stallman
« Le système d'exploitation du projet GNU et le mouvement du logiciel libre » (1998)
http://www.linux-france.org/article/these/gnuproject/fr-thegnuproject_monoblock.html

Le projet GNU (1998)
http://www.fsf.org/gnu/thegnuproject.fr.html

Bruce Perens
« La définition de l'OpenSource » (1999)
http://www.linux-france.org/article/these/the_osd/fr-the_open_source_definition_monoblock.html

Eric S. Raymond
Qu'est-ce qui motive les développeurs de logiciels libres ?
« la cathédrale et le bazar » (1998)
http://www.linux-france.org/article/these/cathedrale-bazar/cathedrale-bazar_monoblock.html

---

Aspects juridiques

Qu'est ce qu'un logiciel libre ?
http://www.gnu.org/philosophy/free-sw.fr.html

Le logiciel et le droit
http://www.freepatents.org/liberty/droit.html

Logiciels libres et protection juridique
http://www.mediadev.fr/articles/partners/ionix/ionixlin.htm

Catégories de logiciels libres et non libres
http://www.gnu.org/philosophy/categories.fr.html
http://opensource.org/licenses/

Brevets logiciels
Freepatents
http://www.freepatents.org/liberty/

Alliance Eurolinux
http://www.eurolinux.org/about/indexfr.html

Jurisprudence en matière de protection du logiciel
http://www.legalis.net/legalnet/jurislog.htm

Article du Monde Interactif (Mars 1999)
(« Le piège des brevets informatiques »)
http://www.monde-diplomatique.fr/1999/03/RIVIERE/11769.html

Article du journal Le Soir (8 Décembre 2000)
(« Controverses passionnées sur un brevet européen pour les logiciels »)
http://dossiers.lesoir.be/laviedunet/decembre2000/A_00F62E.asp

Projet de loi en faveur des logiciels libres dans l'administration
(députés Jean-Yves LE DÉAUT, Christian PAUL, Pierre COHEN)
http://www.osslaw.org/

Articles de presse

Interviews de Linus Torvalds
* **crn.com**, Janvier 2001
  http://techweb.techreviews.com/sections/topics/article/TT20010209S0004/2

* **« Le manifeste de Linux »**
  http://www.linux-france.org/article/these/manifesto/index.html

* **« Qu'est-ce qui motive les développeurs de logiciels libres ? »** (1998)
  http://www.linux-france.org/article/these/interview/torvalds/fr-lt_first_monday.199803.html

- Le Monde Interactif
« Linux connaît une croissance spectaculaire » (18 Août 1999)
http://interactif.lemonde.fr/article/0,5611,2857--19160-0,FF.html

« Faut-il prendre Linux au sérieux ? » (19 Mai 1999)
http://interactif.lemonde.fr/article/0,5611,2857--4342-0,FF.html

« Un marché en pleine mutation » (31 janvier 2001)
http://interactif.lemonde.fr/article/0,5611,2857--141451-0,FF.html

« De l'infiltration à la consécration » (31 janvier 2001)
http://interactif.lemonde.fr/article/0,5611,2857--141452-0,FF.html

« La déstabilisation naturelle des lois de l'économie ? » (31 Janvier 2001)
http://interactif.lemonde.fr/article/0,5611,2858-5116-141447-0,FF.html

- 01net.com
« Linux progresse dans les entreprises » (Mai 2001)
http://www.01net.com/rdn?oid=146926

Interrogations sur l'avenir de Linux

« Operating without liability »
http://techweb.techreviews.com/sections/topics/article/TT20010209S0004

« Linus est-il bon pour Linux ? »
http://www.transfert.net/fr/dossiers/article.cfm?idx_rub=87&idx_art=3854

Interview de Jean-Marc Lange, responsable des ventes France et Colin Tenwick, vice-président Europe de RedHat
"Les mastodontes apportent beaucoup au monde du libre"
http://www.transfert.net/fr/dossiers/article.cfm?idx_rub=87&idx_art=3873

« L'open source n'est pas un business rentable »
http://www.transfert.net/fr/net_economie/article.cfm?idx_rub=86&idx_art=5088

Simputer

Simputer Trust
http://www.simputer.org

Article de Libération (30 Avril 2001)
(« Un ordinateur pour poches vides »)
http://www.liberation.fr/multi/actu/20010431/20010430lunz.html

Article de PCWorld (23 juin 2000)
http://www.pcworld.com/news/article.asp?aid=17401

Article de Indiaabroad.com
("Simputer could revolutionize IT in developing nations")
http://www.indiaabroaddaily.com/2001/03/13/13simputer.html

RedFlag Linux

Article de zdnet.fr (février 2001)
« La Chine cultive la révolution Linux »
http://www.zdnet.fr/prod/cgi-bin/affiche.pl?ID=18248

Articles de Libération (2 Novembre 2000):
« La Chine plante le drapeau rouge sur Linux »
« Le pingouin alternatif ravit les Etats »
http://www.liberation.fr/multi/actu/20001030/20001102jeuy.html

---

Autres sites internet :

Linux en entreprise
http://strasbourg.linuxfr.org/articles/linbusiness.html

Applications professionnelles de Linux
http://www.linux-france.org/article/lbiz-fr

etc, etc….

Autres ressources :

Magazines : Linux+, Maximum Linux, Linux Loader…

Annexes :

Traduction de la Licence GPL

Table des matières

This is an unofficial translation of the GNU General Public License into *french*. It was not published by the Free Software Foundation, and does not legally state the distribution terms for software that uses the GNU GPL--only the original English text of the GNU GPL does that. However, we hope that this translation will help *french* speakers understand the GNU GPL better.

Ceci est une traduction non officielle de la GNU General Public License en français. Elle n'a pas été publiée par la Free Software Foundation, et ne détermine pas les termes de distribution pour les logiciels qui utilisent la GNU GPL--seul le texte anglais original de la GNU GPL en a le droit. Cependant, nous espérons que cette traduction aidera les francophones à mieux comprendre la GPL.

Nous autorisons la FSF à apporter toute modification qu'elle jugera nécessaire pour rendre la traduction plus claire.

GNU GENERAL PUBLIC LICENSE

Version 2, juin 1991

Préambule

Les licences relatives à la plupart des logiciels sont destinées à supprimer votre liberté de les partager et de les modifier. Par contraste, la licence publique générale GNU General Public License veut garantir votre liberté de partager et de modifier les logiciels libres, pour qu'ils

soient vraiment libres pour tous leurs utilisateurs. La présente licence publique générale s'applique à la plupart des logiciels de la Free Software Foundation, ainsi qu'à tout autre programme dont les auteurs s'engagent à l'utiliser. (Certains autres logiciels sont couverts par la Licence Publique Générale pour Bibliothèques GNU à la place). Vous pouvez aussi l'appliquer à vos programmes.

Quand nous parlons de logiciels libres, nous parlons de liberté, non de gratuité. Nos licences publiques générales veulent vous garantir :

- que vous avez toute liberté de distribuer des copies des logiciels libres (et de facturer ce service, si vous le souhaitez) ;
- que vous recevez les codes sources ou pouvez les obtenir si vous le souhaitez ;
- que vous pouvez modifier les logiciels ou en utiliser des éléments dans de nouveaux programmes libres ;
- et que vous savez que vous pouvez le faire.

Pour protéger vos droits, nous devons apporter des restrictions, qui vont interdire à quiconque de vous dénier ces droits, ou de vous demandez de vous en désister. Ces restrictions se traduisent par certaines responsabilités pour ce qui vous concerne, si vous distribuez des copies de logiciels, ou si vous les modifiez.

Par exemple, si vous distribuez des copies d'un tel programme, gratuitement ou contre une rémunération, vous devez transférer aux destinataires tous les droits dont vous disposez. Vous devez vous garantir qu'eux-mêmes, par ailleurs, reçoivent ou peuvent recevoir le code source. Et vous demandez leur montrer les présentes dispositions, de façon qu'ils connaissent leurs droits.

Nous protégeons vos droits en deux étapes :

1. Nous assurons le droit d'auteur (copyright) du logiciel, et
2. Nous vous proposons cette licence, qui vous donne l'autorisation légale de dupliquer, distribuer et/ou modifier le logiciel.

De même, pour la protection de chacun des auteurs, et pour notre propre protection, nous souhaitons nous assurer que tout le monde comprenne qu'il n'y a aucune garantie portant sur ce logiciel libre. Si le logiciel est modifié par quelqu'un d'autre puis transmis à des tiers, nous souhaitons que les destinataires sachent que ce qu'ils possèdent n'est pas l'original, de façon que tous problèmes introduits par d'autres ne se traduisent pas par une répercussion négative sur la réputation de l'auteur original.

Enfin, tout programme libre est en permanence menacé par des brevets de logiciels. Nous souhaitons éviter le danger que des sous-distributeurs d'un programme libre obtiennent à titre individuel des licences de brevets, avec comme conséquence qu'ils ont un droit de propriété sur le programme. Pour éviter cette situation, nous avons fait tout ce qui est nécessaire pour que tous brevets doivent faire l'objet d'une concession de licence qui en permette l'utilisation libre par quiconque, ou bien qu'il ne soit pas concédé du tout.

Nous présentons ci-dessous les closes et dispositions concernant la duplication, la distribution et la modification.

Conditions d'exploitation portant sur la duplication, la distribution et la modification

1. Le présent contrat de licence s'applique à tout programme ou autre ouvrage contenant un avis, apposé par le détenteur du droit de propriété, disant qu'il peut être distribué au titre des dispositions de la présente Licence Publique Générale. Ci-après, le "Programme" désigne l'un quelconque de ces programmes ou ouvrages, et un "ouvrage fondé sur le programme" désigne soit le programme, soit un ouvrage qui en dérive au titre de la loi sur le droit d'auteur ; plus précisément, il s'agira d'un ouvrage contenant le programme ou une version de ce dernier, soit mot à mot, soit avec des modifications et/ou traduit en une autre langue (ci-après, le terme "modification" englobe, sans aucune limitation, les traductions qui en sont faites). Chaque titulaire de licence sera appelé "concessionnaire".

   Les activités autres que la duplication, la distribution et la modification ne sont pas couvertes par la présente licence ; elles n'entrent pas dans le cadre de cette dernière. L'exécution du programme n'est soumise à aucune restriction, et les résultats du programme ne sont couverts que si son contenu constitue un ouvrage fondé sur le programme (indépendamment du fait qu'il a été réalisé par exécution du programme). La véracité de ce qui précède dépend de ce que fait le programme.

2. Le concessionnaire peut dupliquer et distribuer des copies mot à mot du code source du programme tel qu'il les reçoit, et ce sur un support quelconque, du moment qu'il appose, d'une manière parfaitement visible et appropriée, sur chaque exemplaire, un avis approprié de droits d'auteur (Copyright) et de renonciation à garantie ; qu'il maintient intacts tous les avis qui se rapportent à la présente licence et à l'absence de toute garantie ; et qu'il transmet à tout destinataire du programme un exemplaire de la présente licence en même temps que le programme.

   Le concessionnaire peut facturer l'acte physique de transfert d'un exemplaire, et il peut, à sa discrétion, proposer en échange d'une rémunération une protection en garantie.

3. Le concessionnaire peut modifier son ou ses exemplaires du programme ou de toute portion de ce dernier, en formant ainsi un ouvrage fondé sur le programme, et dupliquer et distribuer ces modifications ou cet ouvrage selon les dispositions de la section 1 ci-dessus, du moment que le concessionnaire satisfait aussi à toutes ces conditions :

   a. Le concessionnaire doit faire en sorte que les fichiers modifiés portent un avis, parfaitement visible, disant que le concessionnaire a modifié les fichiers, avec la date de tout changement.

   b. Le concessionnaire doit faire en sorte que tout ouvrage qu'il distribue ou publie, et qui, en totalité ou en partie, contient le programme ou une partie quelconque de ce dernier ou en dérive, soit concédé en bloc, à titre gracieux, à tous tiers au titre des dispositions de la présente licence.

   c. Si le programme modifié lit normalement des instructions interactives lors de son exécution, le concessionnaire doit, quand il commence l'exécution du

programme pour une telle utilisation interactive de la manière la plus usuelle, faire en sorte que ce programme imprime ou affiche une annonce, comprenant un avis approprié de droits d'auteur, et un avis selon lequel il n'y a aucune garantie (ou autrement, que le concessionnaire fournit une garantie), et que les utilisateurs peuvent redistribuer le programme ou titre de ces dispositions, et disant à l'utilisateur comment visualiser une copie de cette licence (exception : si le programme par lui-même est interactif mais n'imprime normalement pas une telle annonce, l'ouvrage du concessionnaire se fondant sur le programme n'a pas besoin d'imprimer une annonce).

Les exigences ci-dessus s'appliquent à l'ouvrage modifié pris en bloc. Si des sections identifiables de cet ouvrage ne dérivent pas du programme et peuvent être considérées raisonnablement comme représentant des ouvrages indépendants et distincts par eux-mêmes, alors la présente licence, et ses dispositions, ne s'appliquent pas à ces sections quand le concessionnaire les distribue sous forme d'ouvrages distincts. Mais quand le concessionnaire distribue ces mêmes sections en tant qu'élément d'un tout qui représente un ouvrage se fondant sur le programme, la distribution de ce tout doit se faire conformément aux dispositions de la présente licence, dont les autorisations, portant sur d'autres concessionnaires, s'étendent à la totalité dont il est question, et ainsi à chacune de ces parties, indépendamment de celui qu'il a écrite.

Ainsi, cette section n'a pas pour but de revendiquer des droits ou de contester vos droits sur un ouvrage entièrement écrit par le concessionnaire ; bien plus, l'intention est d'exercer le droit de surveiller la distribution d'ouvrages dérivée ou collective se fondant sur le programme.

De plus, un simple assemblage d'un autre ouvrage ne se fondant pas sur le programme, avec le programme (ou avec un ouvrage se fondant sur le programme) sur un volume d'un support de stockage ou distribution, ne fait pas entrer l'autre ouvrage dans le cadre de la présente licence.

4. Le concessionnaire peut dupliquer et distribuer le programme (ou un ouvrage se fondant sur ce dernier, au titre de la Section 2), en code objet ou sous une forme exécutable, au titre des dispositions des Sections 1 et 2 ci-dessus, du moment que le concessionnaire effectue aussi l'une des opérations suivantes :

    a. Lui joindre le code source complet correspondant, exploitable par une machine, code qui doit être distribué au titre des Sections 1 et 2 ci-dessus sur un support couramment utilisé pour l'échange de logiciels ; ou bien

    b. Lui joindre une offre écrite, dont la validité se prolonge pendant au moins 3 ans, de transmettre à un tiers quelconque, pour un montant non supérieur au coût pour le concessionnaire, de réalisation physique de la distribution de la source, un exemplaire complet, exploitable par une machine, du code source correspondant, qui devra être distribué au titre des dispositions des Sections 1 et 2 ci-dessus sur un support couramment utilisé pour l'échange des logiciels ; ou bien

    c. Lui joindre les informations que le concessionnaire a reçues, pour proposer une distribution du code source correspondant (cette variante n'est autorisée que pour la distribution non commerciale, et seulement si le concessionnaire a reçu

le programme sous forme exécutable ou sous forme d'un code objet, avec une telle offre, conformément à l'alinéa b) ci-dessus).

Le code source d'un ouvrage représente la forme préférée de l'ouvrage pour y effectuer des modifications. Pour un ouvrage exécutable, le code source complet représente la totalité du code source pour tous les modules qu'il contient, plus tous fichiers de définitions d'interface associés, plus les informations en code machine pour commander la compilation et l'installation du programme exécutable. Cependant, à titre d'exceptions spéciales, le code source distribué n'a pas besoin de comprendre quoi que ce soit qui est normalement distribué (sous forme source ou sous forme binaire) avec les composants principaux (compilateur, noyau de système d'exploitation, etc.) du système d'exploitation sur lequel est exécuté le programme exécutable, à moins que le composant, par lui-même, soit joint au programme exécutable.

Si la distribution de lëexécutable ou du code objet est réalisée de telle sorte qu'elle offre d'accéder à une copie à partir d'un lieu désigné, alors le fait d'offrir un accès équivalent à la duplication du code source à partir de ce même lieu s'entend comme distribution du code source, même si des tiers ne sont pas contraints de dupliquer la source en même temps que le code objet.

5. Le concessionnaire ne peut dupliquer, modifier, concéder en sous-licence ou distribuer le programme, sauf si cela est expressément prévu par les dispositions de la présente licence. Toute tentative pour autrement dupliquer, modifier, concéder en sous-licence ou distribuer le programme est répétée nulle, et met automatiquement fin aux droits du concessionnaire au titre de la présente licence. Cependant, les parties qui ont reçu des copies, ou des droits, de la part du concessionnaire au titre de la présente licence, ne verront pas expirer leur contrat de licence, tant que ces parties agissent une manière parfaitement conforme.

6. Il n'est pas exigé du concessionnaire qu'il accepte la présente licence, car il ne l'a pas signée. Cependant, rien d'autre n'octroie au concessionnaire l'autorisation de modifier ou de distribuer le programme ou ses ouvrages dérivés. Ces actions sont interdites par la loi si le concessionnaire n'accepte pas la présente licence. En conséquence, par le fait de modifier ou de distribuer le programme (ou un ouvrage quelconque se fondant sur le programme), le concessionnaire indique qu'il accepte la présente licence, et qu'il a la volonté de se conformer à toutes les closes et dispositions concernant la duplication, la distribution ou la modification du programme ou d'ouvrages se fondant sur ce dernier.

7. Chaque fois que le concessionnaire redistribue le programme (ou tout ouvrage se fondant sur le programme), le destinataire reçoit automatiquement une licence de l'émetteur initial de la licence, pour dupliquer, distribuer ou modifier le programme, sous réserve des présentes closes et dispositions. Le concessionnaire ne peut imposer aucune restriction plus poussée sur l'exercice, par le destinataire, des droits octroyés au titre des présentes. Le concessionnaire n'a pas pour responsabilité d'exiger que des tiers se conforment à la présente licence.

8. Si, en conséquence une décision de justice ou une allégation d'infraction au droit des brevets, ou pour toute autre raison (qui n'est pas limitée à des problèmes de propriétés industrielles), des conditions sont imposées au concessionnaire (par autorité de justice, par convention ou autrement), qui entrent en contradiction avec les dispositions de la présente licence, elles n'exemptent pas le concessionnaire de respecter les dispositions

de la présente licence. Si le concessionnaire ne peut procéder à la distribution de façon à satisfaire simultanément à ces obligations au titre de la présente licence et à toutes autres obligations pertinentes, alors, en conséquence de ce qui précède, le concessionnaire peut ne pas procéder du tout à la distribution du programme. Par exemple, si une licence de brevet ne permettait pas une redistribution du programme, sans redevances, par tous ceux qui reçoivent des copies directement ou indirectement par l'intermédiaire du concessionnaire, alors le seul moyen par lequel le concessionnaire pourrait satisfaire tant à cette licence de brevet qu'à la présente licence, consisterait à s'abstenir complètement de distribuer le programme.

Si une partie quelconque de cette section est considérée comme nulle ou non exécutoire dans certaines circonstances particulières, le reste de cette section est réputé s'appliquer, et la section dans son ensemble est considérée comme s'appliquant dans les autres circonstances.

La présente section n'a pas pour objet de pousser le concessionnaire à enfreindre tous brevets ou autres revendications à droit de propriété, ou encore à contester la validité de une ou plusieurs quelconques de ces revendications ; la présente section a pour objet unique de protéger l'intégrité du système de distribution des logiciels libres, système qui est mis en oeuvre par les pratiques liées aux licences publiques. De nombreuses personnes ont apporté une forte contribution à la gamme étendue des logiciels distribués par ce système, en comptant sur l'application systématique de ce système ; c'est à l'auteur/donateur de décider s'il a la volonté de distribuer le logiciel par un quelconque autre système, et un concessionnaire ne peut imposer ce choix.

La présente section veut rendre parfaitement claire ce que l'on pense être une conséquence du reste de la présente licence.

9. Si la distribution et/ou l'utilisation du Programme est restreinte dans certains pays, sous l'effet de brevets ou d'interfaces présentant un droit d'auteur, le détenteur du droit d'auteur original, qui soumet le Programme aux dispositions de la présente licence, pourra ajouter une limitation expresse de distribution géographique excluant ces pays, de façon que la distribution ne soit autorisée que dans les pays ou parmi les pays qui ne sont pas ainsi exclus. Dans ce cas, la limitation fait partie intégrante de la présente licence, comme si elle était écrite dans le corps de la présente licence.

   La Free Software Foundation peut, de temps à autre, publier des versions révisées et/ou nouvelles du General Public License. Ces nouvelles versions seront analogues, du point de vue de leur esprit, à la présente version, mais pourront en différer dans le détail, pour résoudre de nouveaux problèmes ou de nouvelles situations.

   Chaque version reçoit un numéro de version qui lui est propre. Si le programme spécifie un numéro de version de la présente licence, qui s'applique à cette dernier et "à toute autre version ultérieure", le concessionnaire a le choix de respecter les closes et dispositions de cette version, ou une quelconque version ultérieure publiée par la Free Software Foundation. Si le programme ne spécifie pas de numéro de version de la présente licence, le concessionnaire pourra choisir une version quelconque publiée à tout moment par la Free Software Foundation.

10. Si le concessionnaire souhaite incorporer des parties du programme dans d'autres programmes libres dont les conditions de distribution sont différentes, il devrait écrire à l'auteur pour demander son autorisation. Pour un logiciel soumis à droit d'auteur par

la Free Software Foundation, il devra écrire à la Free Software Foundation ; nous faisons quelquefois des exceptions à cette règle. Notre décision va être guidée par le double objectif de protéger le statut libre de tous les dérivés de nos logiciels libres, et de favoriser le partage et la réutilisation des logiciels en général.

ABSENCE DE GARANTIE

11. COMME LA LICENCE DU PROGRAMME EST CONCEDEE A TITRE GRATUIT, IL N'Y AUCUNE GARANTIE S'APPLIQUANT AU PROGRAMME, DANS LA MESURE AUTORISEE PAR LA LOI EN VIGUEUR. SAUF MENTION CONTRAIRE ECRITE, LES DETENTEURS DU DROIT D'AUTEUR ET/OU LES AUTRES PARTIES METTENT LE PROGRAMME A DISPOSITON "EN L'ETAT", SANS AUCUNE GARANTIE DE QUELQUE NATURE QUE CE SOIT, EXPRESSE OU IMPLICITE, Y COMPRIS, MAIS SANS LIMITATION, LES GARANTIES IMPLICITES DE COMMERCIALISATION ET DE L'APTITUDE A UN OBJET PARTICULIER. C'EST LE CONCESSIONNAIRE QUI PREND LA TOTALITE DU RISQUE QUANT A LA QUALITE ET AUX PERFORMANCES DU PROGRAMME. SI LE PROGRAMME SE REVELAIT DEFECTUEUX, C'EST LE CONCESSIONNAIRE QUI PRENDRAIT A SA CHARGE LE COUT DE L'ENSEMBLE DES OPERATIONS NECESSAIRES D'ENTRETIEN, REPARATION OU CORRECTION.

12. EN AUCUN CAS, SAUF SI LA LOI EN VIGUEUR L'EXIGE OU SI UNE CONVENTION ECRITE EXISTE A CE SUJET, AUCUN DETENTEUR DE DROITS D'AUTEUR, OU AUCUNE PARTIE AYANT LE POUVOIR DE MODIFIER ET/OU DE REDISTRIBUER LE PROGRAMME CONFORMEMENT AUX AUTORISATIONS CI-DESSUS, N'EST RESPONSABLE VIS-A-VIS DU CONCESSIONNAIRE POUR CE QUI EST DES DOMMAGES, Y COMPRIS TOUS DOMMAGES GENERAUX, SPECIAUX, ACCIDENTELS OU INDIRECTS, RESULTANT DE L'UTILISATION OU DU PROGRAMME OU DE L'IMPOSSIBILITE D'UTILISER LE PROGRAMME (Y COMPRIS, MAIS SANS LIMITATION, LA PERTE DE DONNEES, OU LE FAIT QUE DES DONNEES SONT RENDUES IMPRECISES, OU ENCORE LES PERTES EPROUVEES PAR LE CONCESSIONNAIRE OU PAR DES TIERS, OU ENCORE UN MANQUEMENT DU PROGRAMME A FONCTIONNER AVEC TOUS AUTRES PROGRAMMES), MEME SI CE DETENTEUR OU CETTE AUTRE PARTIE A ETE AVISE DE LA POSSIBILITE DE TELS DOMMAGES.

FIN DES CONDITIONS D'EXPLOITATION

Comment appliquer ces dispositions a vos nouveaux programmes?

Si le concessionnaire développe un nouveau programme, et s'il en souhaite l'utilisation la plus large possible dans le public, le meilleur moyen d'y arriver est d'en faire un logiciel libre, que tout le monde pourra redistribuer et modifier au titre des présentes dispositions.

Dans ce but, il convient de rattacher au programme les avis suivants. Le moyen le plus sûr consiste à les rattacher au début de chaque fichier source, pour avertir le plus efficacement possible de l'exclusion de garantie ; et chaque fichier doit comporter au moins la ligne "copyright", et un pointeur indiquant où est localisée la totalité de l'avis.

Une ligne pour donner le nom du programme et une idée de ce qu'il fait.
Copyright (C) 19*yy nom de l'auteur*
Ce programme est un logiciel libre ; vous pouvez le redistribuer et/ou le modifier conformément aux dispositions de la Licence Publique Générale GNU, telle que publiée par la Free Software Foundation ; version 2 de la licence, ou encore (à votre choix) toute version ultérieure.
Ce programme est distribué dans l'espoir qu'il sera utile, mais SANS AUCUNE GARANTIE ; sans même la garantie implicite de COMMERCIALISATION ou D'ADAPTATION A UN OBJET PARTICULIER. Pour plus de détail, voir la Licence Publique Générale GNU .
Vous devez avoir reçu un exemplaire de la Licence Publique Générale GNU en même temps que ce programme ; si ce n'est pas le cas, écrivez à la Free Software Foundation Inc., 675 Mass Ave, Cambridge, MA 02139, Etats-Unis.

Ajoutez aussi des informations sur le moyen permettant d'entrer en contact avec vous par courrier électronique (e-mail) et courrier normal.

Si le programme est interactif, prévoyez en sortie un court avis, tel que celui qui est présenté ci-dessous, lors du démarrage en mode interactif.

Gnomovision version 69, Copyright (C) 19 *yy nom de l'auteur*
Gnomovision est livré absolument SANS AUCUNE GARANTIE ; pour plus de détail, tapez "show w". Il s'agit d'un logiciel libre, et vous avez le droit de le redistribuer dans certaines conditions ; pour plus de détail, tapez "show c".

Les instructions hypothétiques "show w" et "show c" doivent présenter les parties appropriées de la Licence Publique Générale. Bien évidemment, les instructions que vous utilisez peuvent porter d'autres noms que "show w" et "show c" ; elles peuvent même correspondre à des clics de souris ou à des éléments d'un menu, selon ce qui convient à votre programme.

Si nécessaire, vous devrez aussi demander à votre employeur (si vous travaillez en tant que programmeur) ou à votre éventuelle école ou université, de signer une "renonciation à droit d'auteur" concernant le programme. En voici un échantillon (il suffit de modifier les noms) :

Yoyodyne, Inc., par la présente, renonce à tout intérêt de droits d'auteur dans le programme "Gnomovision" (qui fait des passages au niveau des compilateurs) écrit par James Hacker.
Signature de Ty Coon, 1er avril 1989
Ty Coon, President of Vice

La présente Licence Publique Générale n'autorise pas le concessionnaire à incorporer son programme dans des programmes propriétaires. Si votre programme est une bibliothèque de sous-programmes, vous pouvez considérer comme plus intéressant d'autoriser une édition de liens des applications propriétaires avec la bibliothèque. Si c'est ce que vous souhaitez, vous

devrez utiliser non pas la présente licence, mais la Licence Publique Générale pour Bibliothèques GNU.

Le début du développement coopératif :
le post de Linus Torvalds en 1991 à l'origine de Linux

(posté sur le newsgroup comp.os.minix)

> From: torvalds@klaava.Helsinki.FI (Linus Benedict Torvalds)
> Newsgroups: comp.os.minix
> Subject: What would you like to see most in minix?
> Summary: small poll for my new operating system
> Message-ID: <1991Aug25.205708.9541@klaava.Helsinki.FI>
> Date: 25 Aug 91 20:57:08 GMT
> Organization: University of Helsinki
>
>
> Hello everybody out there using minix -
>
> I'm doing a (free) operating system (just a hobby, won't be big and
> professional like gnu) for 386(486) AT clones. This has been brewing
> since april, and is starting to get ready. I'd like any feedback on
> things people like/dislike in minix, as my OS resembles it somewhat
> (same physical layout of the file-system (due to practical reasons)
> among other things).
>
> I've currently ported bash(1.08) and gcc(1.40), and things seem to
work. This implies that I'll get something practical within a few
months, and I'd like to know what features most people would want. Any
suggestions are welcome, but I won't promise I'll implement them :-)
>
>   Linus (torvalds@kruuna.helsinki.fi)
>
> PS. Yes - it's free of any minix code, and it has a multi-threaded
fs. It is NOT protable (uses 386 task switching etc), and it probably
never will support anything other than AT-harddisks, as that's all I
have :-(.