2005RP-05

L'économie du logiciel libre et ouvert
Recommandations en vue d'une politique gouvernementale à l'égard du logiciel libre (*open source software*)

Marcel Boyer, Jacques Robert

Rapport de projet
Project report

Ce document a été produit dans le cadre d'un projet de recherche financé par le Secrétariat du Conseil du trésor du Québec sur les standards ouverts et les logiciels libres

Montréal
Mai 2005

CIRANO
Centre interuniversitaire de recherche en analyse des organisations

CIRANO

Le CIRANO est un organisme sans but lucratif constitué en vertu de la Loi des compagnies du Québec. Le financement de son infrastructure et de ses activités de recherche provient des cotisations de ses organisations-membres, d'une subvention d'infrastructure du Ministère du Développement économique et régional et de la Recherche, de même que des subventions et mandats obtenus par ses équipes de recherche.

CIRANO is a private non-profit organization incorporated under the Québec Companies Act. Its infrastructure and research activities are funded through fees paid by member organizations, an infrastructure grant from the Ministère du Développement économique et régional et de la Recherche, and grants and research mandates obtained by its research teams.

Les organisations-partenaires / The Partner Organizations

PARTENAIRE MAJEUR
. Ministère du Développement économique, de Innovation et de l'Exportation

PARTENAIRES
. Alcan inc.
. Banque du Canada
. Banque Laurentienne du Canada
. Banque Nationale du Canada
. Banque Royale du Canada
. Bell Canada
. BMO Groupe financier
. Bombardier
. Bourse de Montréal
. Caisse de dépôt et placement du Québec
. Fédération des caisses Desjardins du Québec
. Gaz Métro
. Groupe financier Norshield
. Hydro-Québec
. Industrie Canada
. Ministère des Finances du Québec
. Pratt & Whitney Canada
. Raymond Chabot Grant Thornton
. Ville de Montréal

. École Polytechnique de Montréal
. HEC Montréal
. Université Concordia
. Université de Montréal
. Université du Québec
. Université du Québec à Montréal
. Université Laval
. Université McGill
. Université de Sherbrooke

ASSOCIÉ À :
. Institut de Finance Mathématique de Montréal (IFM^2)
. Laboratoires universitaires Bell Canada
. Réseau de calcul et de modélisation mathématique [RCM^2]
. Réseau de centres d'excellence MITACS (Les mathématiques des technologies de l'information et des systèmes complexes)

ISSN 1499-8610 (Version imprimée) / ISSN 1499-8629 (Version en ligne)

L'économie du logiciel libre et ouvert
Recommandations en vue d'une politique gouvernementale à l'égard du logiciel libre (*open source software*)[*]

Marcel Boyer[†], *Jacques Robert*[‡]

Résumé / *Abstract*

Ce texte vise à circonscrire les fondements d'une politique gouvernementale à l'égard du logiciel libre et ouvert (*open source software*). Nous caractérisons brièvement l'ampleur du phénomène du logiciel libre et nous en analysons les avantages et inconvénients pour le gouvernement tant comme moteur du développement économique que comme grand utilisateur de technologies d'information et de communications. Nous concluons par un ensemble de recommandations à l'intention du gouvernement, tant de nature « politique économique et industrielle » que de nature « grand utilisateur » face au développement du logiciel libre et ouvert.

Mots clés : logiciel libre, propriété intellectuelle, code source libre, code source ouvert, système d'exploitation libre, licence GPL, licence BSD, innovation, effet d'arborescence

[*] Nous remercions Mélanie Arcand pour son assistance de recherche et le CIRANO pour son support.
[†] Fellow CIRANO et titulaire de la Chaire Bell Canada en économie industrielle, Département de sciences économiques, Université de Montréal, C.P. 6128, succursale Centre-ville Montréal (QC) H3C 3J7, Canada, tél. : (514)-985-4002, courriel: marcel.boyer@umontreal.ca
[‡] Fellow CIRANO et Département des technologies de l'information, HEC Montréal, courriel : jacques.robert@hec.ca

Ce rapport ne cherche pas à faire une revue exhaustive de l'ensemble des discussions, analyses et problématiques associées au développement et à l'utilisation des logiciels libres et ouverts (LLO), logiciels dont le code source est libre d'accès et gratuit d'utilisation. Nombreuses sont les questions pertinentes qui dépassent nos compétences. Notre objectif est plutôt d'offrir une évaluation neutre et balancée des perspectives qu'offre l'utilisation des logiciels libres et ouverts dans les administrations publiques en s'appuyant sur l'analyse économique, plus précisément l'économie des organisations (motivation, incitation et coordination) et l'économie industrielle. Nous espérons, ce faisant, contribuer au débat de manière constructive et éclairante.

Le présent document est divisé en cinq sections. En introduction, nous présentons les objectifs généraux de l'étude et un bref historique du développement du logiciel libre et ouvert. Dans la section 2, nous présentons les critères qui font d'un logiciel un logiciel libre et ouvert et nous discutons brièvement des différents types de licences d'utilisation de ces logiciels. La section 3 est consacrée aux avantages et inconvénients liés à l'*utilisation* (demande) de logiciels libres et ouverts. Dans la section 4, nous nous intéressons à la problématique de l'*innovation* (offre) et nous montrons comment un régime basé sur la protection des logiciels propriétaires et un régime basé sur la reconnaissance des logiciels libres peuvent tous deux favoriser, chacun à sa façon, l'innovation et la création de valeur. La section 5 est consacrée aux principes de calcul économique particulièrement pertinents au choix entre les logiciels propriétaires et les logiciels libres que doit faire un gouvernement préoccupé tant par son intérêt comme utilisateur que par son devoir de contribuer au développement économique et au développement de l'industrie du logiciel et des services informatiques. Nous concluons en offrant un certain nombre de recommandations à l'égard de la politique gouvernementale vis-à-vis le logiciel propriétaire et le logiciel libre.

1. Introduction

Le logiciel libre et ouvert est un phénomène qui prend de plus en plus d'importance. Des initiatives majeures comme Linux et Apache[1], de même que OpenOffice, pour n'en nommer que quelques-unes, ont permis la création de logiciels dont le code est libre d'accès, gratuit et ouvert à l'analyse et aux contributions des pairs. Jusqu'à tout récemment, les logiciels libres n'intéressaient qu'une maigre communauté de programmeurs et de développeurs. Aujourd'hui, ces logiciels sont devenus une alternative sérieuse aux logiciels propriétaires.

Le simple fait que ces logiciels existent et même foisonnent remet en cause le principe selon lequel la création de valeur est maximisée par la protection de systèmes propriétaires, protégés par les régimes de propriété intellectuelle (copyrights), sous l'hypothèse que ces derniers seront nécessairement plus performants. Ce document vise à mieux comprendre le phénomène du logiciel libre et ouvert du point de vue de l'analyse économique et à déduire de cette analyse des recommandations à l'égard de la politique gouvernementale vis-à-vis le logiciel libre.

1.1 Objectifs généraux

En tant qu'utilisateurs de logiciels, les gouvernements sont de plus en plus interpellés par les initiatives de développement de logiciels libres dans la mesure où ceux-ci représentent de réelles alternatives aux solutions propriétaires. Parmi les avantages documentés des logiciels libres, on note qu'ils offrent des solutions moins chères, dans certains cas plus robustes et plus sécuritaires, et que les utilisateurs de logiciels libres acquièrent une plus grande liberté contractuelle vis-à-vis de leurs fournisseurs et une meilleure garantie de non-opportunisme de la part de ces derniers au moment du renouvellement des contrats de services ou des mises à jour. De plus, les utilisateurs gouvernementaux peuvent plus facilement choisir de s'appuyer davantage sur les ressources locales sans pour autant perdre les avantages en

[1] Le système d'exploitation libre Linux (du nom de son créateur le Finlandais Linus Torvalds et du système Unix qui est à son origine) et le serveur Web Apache (qui tire son nom de l'expression *a patchy server*) sont deux exemples de logiciels libres.

termes de coûts (économies d'échelle, d'envergure, d'arborescence et de réseaux) que pouvaient revendiquer, non sans raisons, certains fournisseurs de logiciels propriétaires.

Les logiciels libres soulèvent aussi des enjeux de politique économique. L'utilisation de logiciels libres par les gouvernements affecte et contribue à transformer la structure de l'industrie des logiciels et des services informatiques. On ne sera pas surpris d'apprendre que le développement des logiciels libres et ouverts s'est fait dans une large mesure en réaction au pouvoir de marché des grandes entreprises de logiciels propriétaires et vise précisément à modifier les rapports concurrentiels dans l'industrie du logiciel et à encourager l'industrie locale des services informatiques.

Faut-il pour autant encourager (ou décourager) de manière générale le développement de solutions en code ouvert? La réponse à cette question dépend des objectifs politiques poursuivis et des poids relatifs perçus des différents arguments. Les membres de la communauté du logiciel libre affirment que les logiciels libres doivent être appuyés, par principe, car leur développement procède de la recherche du bien commun plutôt que la poursuite (soi-disant « *vile et intéressée* »!) d'un profit corporatif. Les défenseurs des logiciels propriétaires, pour leur part, affirment et défendent le rôle crucial d'un régime de propriété intellectuelle clair et exécutoire comme facteur incitatif crucial de l'innovation dans nos sociétés.

De toute évidence, la production quantitative et qualitative d'œuvres de création intellectuelle nécessite une protection adéquate des droits d'auteurs. C'est là une condition importante pour motiver adéquatement les auteurs, acteurs et autres agents sociaux et économiques à investir temps et ressources à la production de telles œuvres. Le courant de pensée majoritaire repose sur un argumentaire général qui veut que la propriété intellectuelle privée, protégée par un régime efficace de reconnaissance du droit d'auteur, à l'instar d'un régime clair et exécutoire de propriété physique privée, permette de maximiser la création de richesse en donnant aux créateurs la capacité de capturer une part significative de la valeur créée.

Mais cette approche n'est pas la seule à laquelle nous pouvons avoir recours. Dans certains cas, on reconnaît l'importance d'assurer une libre circulation et réutilisation des idées et contributions intellectuelles. Dans une large mesure, les états modernes font la promotion de cette libre circulation et réutilisation des idées en encourageant par un recours direct aux ressources et pouvoirs fiscaux le développement des sciences et de la recherche. Collectivement, nous profitons tous des retombées (effets de débordement) de l'accroissement de cette connaissance, dite du domaine public. Ainsi s'opposent deux approches ou visions, à savoir celle où les contributions intellectuelles sont protégées par le droit d'auteur et celle où, tout en « reconnaissant » les contributions des auteurs et créateurs, on s'attend à ce que leurs créations soient versées au domaine public.

Les logiciels libres doivent être évalués à leur mérite au même titre que toutes les solutions organisationnelles alternatives. Ultimement, la solution technologique retenue doit faire l'objet d'une évaluation au cas par cas. Néanmoins, nous croyons qu'il existe des particularités propres aux logiciels libres qu'il faut mieux analyser et mieux comprendre.

L'importante réalité du logiciel libre soulève un incontournable paradoxe: comment expliquer que des développeurs et des entreprises privées investissent des ressources, souvent importantes, pour développer des logiciels distribués gratuitement? Parmi les explications que l'on peut retrouver dans la documentation spécialisée, on note en particulier la volonté des développeurs de se faire connaître et de promouvoir leur expertise complémentaire et la difficulté de développer des logiciels complexes en vase clos. En effet, on se rend compte que l'industrie du logiciel possède des caractéristiques qui lui sont propres, caractéristiques qui peuvent rendre le logiciel libre économiquement viable et avantageux tant pour les développeurs que pour les utilisateurs. En effet, cette industrie est tributaire d'importants effets de réseaux, de fortes économies d'échelle et d'envergure, d'effets d'arborescence importants, d'un insatiable désir et besoin d'innover, et de coûts de transition (switching costs) importants.

En somme, le logiciel est un produit hautement complexe et en constante mutation devant répondre à des standards de compatibilité élevés. Développer et imposer des solutions

propriétaires majeures est hors de portée d'une firme individuelle, à l'exception des plus grosses, voire même *la* plus grosse. C'est pourquoi des entreprises comme HP, IBM et SUN investissent des sommes importantes pour développer des initiatives « open source » de grande ampleur.

1.2 Un bref historique du phénomène des logiciels libres et ouverts

Un logiciel peut être transmis soit en distribuant le code source ou le code binaire. Le code source peut être interprété par un analyste et modifié, le code binaire correspond à une séquence de 0 et de 1 que l'analyste peut très difficilement interpréter et modifier. Ainsi, il est possible de protéger un logiciel par le secret professionnel en cachant le code source. La plupart des logiciels commerciaux sont distribués uniquement par le code binaire; lorsque le code source est distribué, il est accompagné par des licences très restrictives d'utilisation.

Ceci n'a pas toujours été le cas. Au départ, les systèmes d'exploitation des systèmes d'information étaient développés au sein des institutions universitaires tel que Berkeley ou le MIT ou dans les grands centres de recherche privés (Bell Labs et le Xerox's Palo Alto Research Center). Les programmeurs de ces centres de recherche se partageaient volontiers les codes sources qu'ils développaient.

En parallèle, on a vu l'émergence de l'industrie des ordinateurs personnels. Concevoir et commercialiser des logiciels devenait une entreprise avantageuse et de grandes sociétés ont vu le jour et on fait fortune en vendant des logiciels propriétaires. Lorsque au début des années 80, AT&T commence à exercer son droit de propriété intellectuelle sur le système d'exploitation UNIX, une résistance s'est organisée.

La Free Software Foundation (FSF)[2] est fondée par Richard Stallman du MIT Artificial Intelligence Laboratory. La fondation vise à promouvoir le développement et la dissémination de logiciels gratuits. La FSF, tel que mentionné sur son site WEB, "*is*

[2] Free Software Foundation : http://www.fsf.org

dedicated to promoting computer users' rights to use, study, copy, modify, and redistribute computer programs." La fondation propose la licence GPL (pour « *General Public License* ») et un système d'opération appelé GNU[3]. L'objectif de la licence GPL est d'exiger que (a) le code source du logiciel soit distribué gratuitement, et que (b) tout utilisateur qui reprend et modifie le code doit, à son tour, rendre ses modifications et son code source disponibles gratuitement. Bien que cette vision des choses soit difficilement conciliable avec le développement, la dissémination et la commercialisation des logiciels, l'idée de développer des logiciels et de les faire évoluer sur une base de coopération et de communautarisme s'est avérée de plus en plus populaire.

Parallèlement au développement de la licence GPL et du système d'opération GNU, on a vu l'apparition de diverses formes de logiciels gratuits. Un « shareware », par exemple, est un logiciel distribué gratuitement (éventuellement pour une période de test seulement)[4]. Contrairement au logiciel libre, le code source des sharewares n'est pas en général distribué. La licence BSD (Berkeley Software Distribution) offre une approche alternative à la licence GPL. La licence BSD permet à quiconque de copier et modifier le code source mais permet aussi à ceux qui le désirent de s'approprier les modifications et améliorations qu'ils ont apportées et donc de ne pas les rendre publiques et ce, afin de les commercialiser.

Le début des années 90 marque l'apparition du Web et un regain des activités dans le logiciel libre. En 1991, le Finlandais Linus Torvalds se propose de se créer un système d'exploitation qui remplace à UNIX. Après quelques travaux en solitaire, il rend son code source disponible et invite quiconque le désire à suggérer des ajouts et des améliorations.[5] À sa grande surprise,

---

[3] GNU est l'acronyme de « *GNU's Not Unix* ».

[4] Selon l'Office québécois de la langue française : « Il ne faut pas confondre, malgré certaines similitudes, le logiciel libre (*free software*) avec le gratuiciel (*freeware*), le partagiciel (*shareware*) et le publiciel (*public domain software*). Ici, *free* signifie « libre » (« sans contrainte ») et non « gratuit », même si, à l'heure actuelle, la plupart des logiciels libres sont distribués gratuitement ou à un prix dérisoire. Le logiciel libre est également protégé par des droits d'auteur. On emploie également le terme *logiciel ouvert*, parce que le code source est rendu public. » (http://www.granddictionnaire.com)

[5] 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

nombreux sont ceux qui s'y intéressent. Nous sommes, rappelons-le, alors au début du Web. Le succès de Linux est phénoménal. Le Linux est devenu un des principaux systèmes d'exploitation pour les serveurs et le seul système pouvant concurrencer Windows sur les ordinateurs personnels. Aujourd'hui, le développement de Linux est soutenu entre autre par OSDL (Open Source Development Labs), un consortium sans but lucratif. La liste des membres de OSDL comprend des grandes firmes de haute technologie comme IBM, Cisco, HP, Sun Microsystems, Novell, Intel, RedHat et Google.

Depuis 2000, c'est l'entrée massive des grandes corporations des technologies de l'information dans le soutien aux logiciels libres qui demeure le fait marquant. Il existe aujourd'hui peu d'applications logiciels pour lesquels il n'existe pas d'initiative en logiciel libre. Alors que Linux offre une alternative crédible à Windows, OpenOffice se veut une alternative à WindowsOffice. FireFox de Mozilla est un fureteur web qui a pris rapidement une large part de marché à Internet Explorer. Et la liste continue : MySQL (gestion de base de données), Thunderbird (courriel), Spybot (anti-mouchard), Apache (système d'exploitation de server web), Compiere (système ERP pour PME), etc.

Aujourd'hui, le monde du libre est plus dynamique que jamais. Le site SourceForge.net, le plus grand dépôt de codes sources ouverts sur l'Internet, répertorie jusqu'à près de 100 000 projets et plus de 1 million de membres enregistrés. Certes tous ces projets ne sont pas de l'ampleur de Linux, mais un grand nombre de ceux-ci ont atteint une maturité suffisante pour leur permettre de générer des logiciels de haute qualité. L'émergence du logiciel libre n'est plus simplement un épiphénomène marginal et paradoxal mais bien une réalité déjà bien établie, incontournable et en forte croissance.

2. Définition et critères des LLO et types de licences

---

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"

Sur son site Web, le OpenSource Initiative présente une définition exhaustive du logiciel libre (FOSS pour Free OpenSource Software). Il ne suffit pas que le code source soit ouvert, d'autres critères sont aussi importants. Le Open Source Initiative présente une liste de 10 critères :

TABLEAU 1: Les 10 critères de l'Open Source Initiative

(Source: http://www.opensource.org/docs/definition.php)

1. Free Redistribution – (…) The license shall not require a royalty or other fee for such sale.
2. Source Code- The program must include source code, and must allow distribution in source code as well as compiled form. (…)
3. Derived Works- The license must allow modifications and derived works, and must allow them to be distributed under the same terms as the license of the original software. (…)
4. Integrity of The Author's Source Code – (…) The license may require derived works to carry a different name or version number from the original software.
5. No Discrimination Against Persons or Groups - The license must not discriminate against any person or group of persons. (…)
6. No Discrimination Against Fields of Endeavor - The license must not restrict anyone from making use of the program in a specific field of endeavor. (…)
7. Distribution of License - The rights attached to the program must apply to all to whom the program is redistributed without the need for execution of an additional license by those parties. (…)
8. License Must Not Be Specific to a Product - The rights attached to the program must not depend on the program's being part of a particular software distribution. (…)
9. License Must Not Restrict Other Software - The license must not place restrictions on other software that is distributed along with the licensed software. (…)
10. License Must Be Technology-Neutral - No provision of the license may be predicated on any individual technology or style of interface. (…)"

Il existe différentes licences qui respectent les 10 critères présentés au Tableau 1. Chaque licence a ses particularités propres. Notre objectif ici n'est pas de faire une description exhaustive de chacune de ces licences mais plutôt de présenter succinctement les différences importantes. Le Tableau ci-dessous, tiré de l'étude de MITRE (2003), compare les droits et obligations des différentes licences pour les logiciels libres.

Table 1. A Comparison of FOSS and Related Licenses

Property / License:	GPL	LGPL	BSD & MIT	Apache	Public Domain	Microsoft MIT[4] EULA
a. Can be stored on disk with other license types	✓	✓	✓	✓	✓	(bans FOSS)[5]
b. Can be executed in parallel with other license types	✓	✓	✓	✓	✓	(bans FOSS)[5]
c. Can be executed on top of other license types	✓	✓	✓	✓	✓	(bans FOSS)[5]
d. Can be executed underneath other license types	✓[1]	✓	✓	✓	✓	(bans FOSS)[5]
e. Source can be integrated with other license types		✓	✓	✓	✓	(bans FOSS)[5]
f. User decides if and when to publish derived code	✓[2]	✓	✓	✓	✓	✓
g. Software can be sold for a profit	✓	✓	✓	✓	✓	✓
h. Binary code can be replicated by users as desired	✓	✓	✓	✓	✓	
i. Binary code can be redistributed as desired	✓[3]	✓	✓	✓	✓	
j. Binary code can be used as desired by users	✓	✓	✓	✓	✓	
k. New users always receive source code of derived works	✓	✓[6]				
l. New users receive full source modification rights for derived works	✓	✓[6]				
m. New users receive full redistribution rights for derived works	✓	✓[6]				
n. Binary code can be released without source code			✓	✓	✓	✓
o. Derived code can have a different type of license		[7]			✓	
p. Original source can be incorporated into closed source products					✓	

[1] *Provided that both programs are fully and independently usable in other unrelated contexts.*
[2] *Provided that the binary code has not been previously released to the public.*
[3] *Provided that source code is always redistributed along with the binary code.*
[4] *The proprietary Microsoft MIT EULA is not related to the similarly named MIT (X/MIT) license.*
[5] *Specifically bans use of: GPL, LGPL, Artistic, Perl, Mozilla, Netscape, Sun Community, and Sun Industry Standards.*
[6] *The rights granted by LGPL do not necessarily extend to the applications linked into an LGPL library.*
[7] *The LGPL does permit re-licensing under GPL as a special case, but not re-licensing under any other license type.*

License Acronyms:

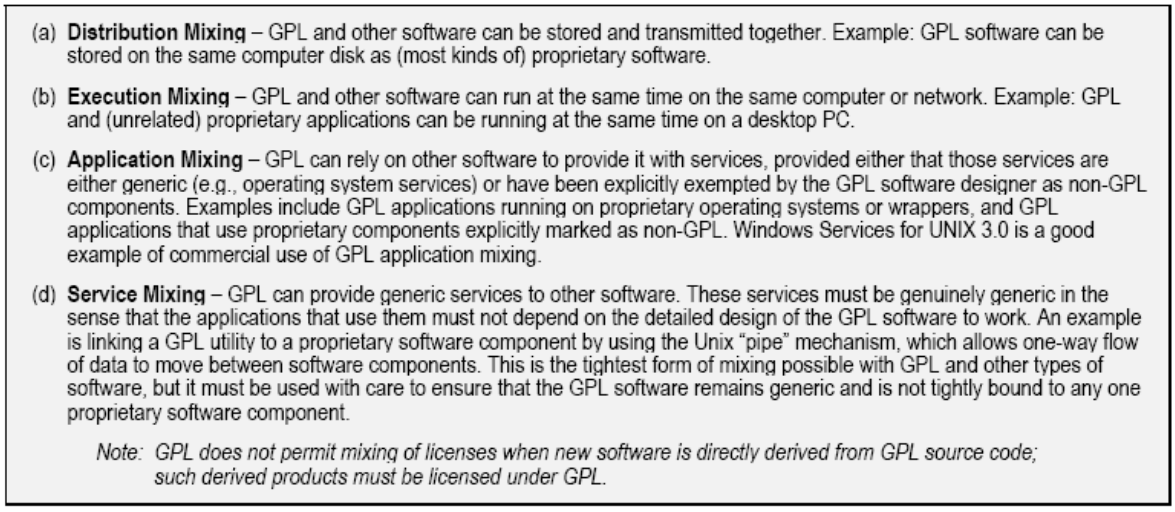
GPL – GNU General Public License
LGPL – GNU Lesser General Public License
BSD – Berkeley Software Distribution
MPL – Mozilla Public License
(Microsoft) MIT – Mobile Internet Toolkit
(X/MIT) MIT – Massachusetts Institute of Technology
EULA – End-User License Agreement
FOSS – Free and Open-Source Software

Ce qui distingue les différentes licences du monde du logiciel libre est la capacité de réutiliser le code source pour le transformer et éventuellement l'incorporer dans un nouveau

produit. La licence GPL interdit explicitement de redistribuer le logiciel ou des versions modifiées du logiciel sans le code source. On dit que la licence GPL est virale, puisqu'elle oblige que les modifications apportées au code soient aussi assujetties à la licence GPL. Les licences assujetties aux règles de type BSD ou Public Domain sont moins restrictives à cet égard.

En effet, une question importante en pratique est à savoir si un logiciel libre peut être associé et joint à un système propriétaire. De manière générale, la réponse est oui, une entreprise privée peut reprendre le code source généré dans le domaine du libre pour créer un nouveau produit. Certaines limitations existent selon le type de licence utilisée. La Figure suivante, tirée de MITRE (2003), montre ce qu'une firme peut faire pour associer du code GPL ouvert à du code propriétaire.

Figure 1. Strategies for Mixing GPL and Proprietary Software



(a) **Distribution Mixing** – GPL and other software can be stored and transmitted together. Example: GPL software can be stored on the same computer disk as (most kinds of) proprietary software.

(b) **Execution Mixing** – GPL and other software can run at the same time on the same computer or network. Example: GPL and (unrelated) proprietary applications can be running at the same time on a desktop PC.

(c) **Application Mixing** – GPL can rely on other software to provide it with services, provided either that those services are either generic (e.g., operating system services) or have been explicitly exempted by the GPL software designer as non-GPL components. Examples include GPL applications running on proprietary operating systems or wrappers, and GPL applications that use proprietary components explicitly marked as non-GPL. Windows Services for UNIX 3.0 is a good example of commercial use of GPL application mixing.

(d) **Service Mixing** – GPL can provide generic services to other software. These services must be genuinely generic in the sense that the applications that use them must not depend on the detailed design of the GPL software to work. An example is linking a GPL utility to a proprietary software component by using the Unix "pipe" mechanism, which allows one-way flow of data to move between software components. This is the tightest form of mixing possible with GPL and other types of software, but it must be used with care to ensure that the GPL software remains generic and is not tightly bound to any one proprietary software component.

Note: GPL does not permit mixing of licenses when new software is directly derived from GPL source code; such derived products must be licensed under GPL.

D'un point de vue économique, cette question est importante. Les licences attachées aux logiciels libres ne doivent pas tuer toutes formes d'initiatives privées de valorisation. En pratique, créer, améliorer et supporter une application logicielle basée sur des logiciels libres doit être une proposition rentable. De nombreuses entreprises ont en fait réussi à relever de

défi et ont développé des modèles d'affaires rentables en s'associant à des logiciels libres. Red Hat vend du support et de la documentation pour Linux; MySQL AB est une entreprise privée qui offre des services de formation, de support et de consultation autour du système de base de données libre MySQL; nombreuses sont les entreprises, notamment Google, qui utilisent des logiciels libres pour distribuer des services avancés sur le Web.

3. Avantages et inconvénients des LLO

Un des avantages des logiciels libres est que leur développement s'appuie généralement sur des standards ouverts. Un standard ouvert est un protocole qui prescrit une manière de faire permettant d'interfacer des logiciels, des routines et sous-routines (des bouts ou morceaux de logiciels) pour former un tout cohérent et opérationnel. La présence de standards ouverts permet à quiconque, sans restriction, de développer du code capable de s'intégrer au code existant. L'utilisation des standards ouverts est un des éléments importants d'une stratégie à long terme. Dans un rapport déposé au gouvernement danois, le Danish Board of Technology a écrit en octobre 2002 :

> "*A strategy for e-government should not be based on a closed, proprietary standard in a key technology. The first reason for this is that it is unacceptable as a matter of principle for enterprises and citizens not to be able to choose between different suppliers of the software that is necessary to use the services of public authorities that are offered in the form of e-government. The second is that it is vital to the socio-economic cost-effectiveness of far-reaching e-government that a competitive situation can be established that ensures the presence of competing products. A condition that must be met for this to be achieved is that open standards are used*"

Le développement et la maintenance d'un logiciel propriétaire se font (exclusivement) par le propriétaire du code ou son agent licencié alors que dans le cas de logiciel libre, le code se développe librement selon les besoins de la communauté d'utilisation s'appuyant sur des standards ouverts. L'approche décentralisée de création d'un logiciel libre repose sur et influence les modalités et caractéristiques suivantes:

(i) les logiciels libres sont modulaires;

(ii) leur développement s'appuie sur les standards ouverts;

(iii) la grande modularité des logiciels libres permet l'installation partielle des logiciels selon les besoins;

(iv) l'utilisation de standards ouverts permet de les utiliser dans une vaste gamme de logiciels complémentaires qui utilisent ces standards;

(v) finalement, les logiciels libres évoluent constamment.

a) Coût

Le premier avantage des logiciels de sources libres est qu'ils ont un coût faible. On peut souvent les obtenir à coût nul puisqu'ils peuvent être téléchargés gratuitement de l'Internet. Lerner et Tirole (2000) soulignent que l'utilisation des logiciels de sources libres peut aussi protéger l'entreprise contre les comportements de monopole et la surenchère des coûts des mises à jour. Dale et alii (2004) mentionne que les logiciels de sources libres ont relativement moins de lignes de codes et qu'ils sont modulaires; ils peuvent ainsi fonctionner à partir d'ordinateurs moins puissants, contribuant ainsi à la baisse des coûts des systèmes. Kenwood (2001) mentionne que les logiciels libres apportent aux entreprises utilisatrices un meilleur ratio prix performance. Il est d'avis que les logiciels libres permettent de réduire les coûts pour les opérations routinières et administratives. Ils permettent de plus le support, l'installation, l'intégration, bref l'implémentation de systèmes à des coûts plus faibles et en facilitent les modifications et mises à jour. Le code source de ces logiciels est en général moins volumineux et ils nécessitent moins de ressources pour fonctionner efficacement.

Une étude menée par la MITRE Corporation (MITRE 2003)[6] sur l'utilisation des logiciels de sources libres au département de la défense américaine avait pour but d'analyser l'impact du bannissement des logiciels de sources libres pour le département. Les analystes en sont venus à la conclusion que les logiciels de sources libres jouent un rôle important au sein du département à plusieurs niveaux. Il existe sur le marché du logiciel des logiciels

6 Voir aussi la critique de ce rapport par l'ISC (Initiative for Software Choice) : http://softwarechoice.org/download_files/MITRE.Final.Web.pdf

commerciaux offrant des caractéristiques sensiblement équivalentes aux logiciels de sources libres. Le bannissement de ces derniers logiciels résulterait en une augmentation soudaine des coûts à court terme dû au remplacement des logiciels de sources libres des réseaux et des applications Web par des logiciels propriétaires équivalents. La recherche serait aussi sévèrement affectée par le bannissement des logiciels de sources libres. Dû aux budgets limités, les chercheurs du département utilisent les logiciels de sources libres. Ceci leur permet de se concentrer plus rapidement sur le centre principal de leurs recherches.

b) Effets d'arborescence

Un des avantages les plus importants mais encore bien mal connu est l'effet d'arborescence. Cet effet mesure la valeur pour une entreprise d'orienter le développement d'un logiciel libre et ouvert en y ajoutant une arête ou sous-routine (sentier) qui complète le logiciel actuel et invite les développeurs à améliorer ce nouveau sentier. Une fois le nouveau sentier amélioré par des développeurs exogènes à l'entreprise, cette dernière peut profiter des mises à jour en incorporant au moment voulu ces développements et améliorations dans sa propre adaptation du logiciel. Cet effet d'arborescence s'apparente à une option réelle que crée l'entreprise et qui pourrait s'avérer être fort rentable dans le futur. Dans son compte-rendu du rapport d'IDC cité précédemment, Techworld mentionne ce qui suit:

> « *Another surprise was that many companies said the ability to customise open-source software was important. IDC didn't suggest this as one of the standard multiple-choice answers. Instead, many companies added it in the "comments" section of the survey. Vendors of pre-packaged, proprietary software routinely downplay the customisability of open-source, arguing customers are not interested in extending software themselves. "These companies don't want to start building an application from scratch, but they can build their own additions to an already-complete application," [analyst Bo Lykkegaard] said. "Because they are part of an open-source community, they can feed this back into the software and it can become a part of the next release, meaning people are helping you to maintain your customisations." He said many companies turn to customisations when they can't find commercial software that meets their needs*. »

L'effet d'arborescence vient compléter les économies d'échelle, d'envergure et de réseau plus traditionnellement associées au développement de logiciels. Ces économies sont présentes dans l'ensemble de l'industrie des logiciels, qu'ils soient propriétaires ou libres.

c) Flexibilité et modularité

Un des principaux avantages des logiciels libres est le tandem flexibilité et modularité. La flexibilité revêt différentes formes : flexibilité d'adaptation, flexibilité dans le partage des systèmes, interopérabilité entre les systèmes, choix d'un fournisseur de services d'intégration. Quant à la modularité, elle permet à un utilisateur sophistiqué d'ajouter des modules à ceux déjà développés et adaptés aux besoins et de remettre ces modules dans le domaine public afin de favoriser des développements ultérieurs (arborescence) dont il pourrait profiter au moment opportun.

L'accès au code source permet de les modifier, de les améliorer et de les personnaliser (customisation). De plus en plus d'entreprises affirment qu'elles choisissent des logiciels de sources libres moins pour leur coût avantageux que pour leur vecteur de caractéristiques offrant une meilleure performance et une meilleure qualité. Un récent sondage de IDC[7] sur la pénétration des logiciels libres dans les entreprises européennes montre que les entreprises qui considèrent que la qualité de leurs logiciels est le facteur déterminant de leur compétitivité – telles les entreprises de télécommunications, celles des services financiers et celles des services d'affaire – optent davantage que les autres entreprises pour des logiciels libres pour des raisons liées à la qualité et à la flexibilité plus que pour de simples raisons de coût. En réalité, les entreprises se créent ainsi un portefeuille d'options réelles[8] qui peut avoir une grande valeur en particulier dans un environnement caractérisé par l'importance de l'incertitude, de la volatilité et de l'ignorance face à l'avenir, par une irréversibilité des coûts significative (il est coûteux de faire marche arrière) et par de sérieuses difficultés (coûts) dans l'adaptation ou la modification de logiciels. En effet, la valeur des options réelles telles que

[7] IDC, *Western European End-User Survey: 2005 Spending Priorities, Outsourcing, Open Source, and Impact of Compliance,* quoted by Techworld (www.techworld.com). IDC, une filiale de IDG (International Data Group), est le premier groupe mondial de conseil et d'étude sur les marchés des technologies de l'information.
[8] Voir Boyer, Christofferson, Lasserre et Pavlov (2003) pour une introduction aux options réelles.

représentées et créées par l'arborescence, la flexibilité et la modularité associées aux logiciels libres est d'autant plus élevée que l'incertitude et l'irréversibilité sont significatives.

Puisque les utilisateurs ne sont pas homogènes, cette flexibilité permet de « confectionner sur mesure » les logiciels en les adaptant aux besoins spécifiques du client [Lerner et Tirole 2000; GBdirect[9]]. Il est possible d'éliminer des caractéristiques non nécessaires pour l'utilisateur et d'ajouter des caractéristiques additionnelles. [Kenwood 2001, Dale et alii 2004, Kuan 2000].

Les manufacturiers de logiciel cherchent à relier leur produit à la gamme de leurs produits. Dans la mesure où les logiciels libres ne sont pas tributaires d'un seul manufacturier, il est possible d'intégrer les logiciels libres sur des plateformes technologiques non propriétaires. Les utilisateurs des logiciels de sources libres, après avoir adopté cette technologie, ne seront pas dépendants du manufacturier et ne seront pas contraints aux prix de ce dernier [Lerner et Tirole 2000]. En conséquence, l'utilisateur peut mettre en concurrence divers intégrateurs pour satisfaire ses besoins. Pour l'utilisateur, le logiciel libre permet d'éviter le « lock-in ». Un utilisateur sophistiqué, comme pourrait l'être une agence gouvernementale, peut ainsi concevoir une stratégie multi fournisseurs (multisourcing), tant internes qu'externes, afin d'augmenter sa flexibilité d'approvisionnement, réduire ses coûts et augmenter la qualité de ses logiciels et systèmes.

Un gouvernement responsable du développement industriel peut mettre à profit ces caractéristiques pour favoriser l'émergence d'une concurrence plus forte sur le marché du support informatique (services d'intégration et autres) sans pour autant renoncer aux économies d'échelle, d'envergure, d'arborescence et de réseaux propres à cette industrie. Dans certains cas, le logiciel libre et ouvert peut augmenter ces économies. La maturité et la fiabilité des logiciels libres ne peuvent qu'augmenter avec le développement du réseau d'utilisateurs. La communauté du logiciel libre est particulièrement propice au jeu et processus de la « sélection naturelle » permettant une amélioration continue des logiciels.

[9] Gbdirect (www.ebusiness.gbdirect.co.uk) est une entreprise de conseil en commerce électronique et technologie de l'information. Voir en particulier http://open-source.gbdirect.co.uk/migration/benefit.html .

d) Risques et gestion des risques

Dans le domaine du logiciel, la notion de risque est une notion à multiples facettes. Plusieurs sources de risque sont présentes, interagissent entre elles, se renforcent ou se neutralisent. Il est ainsi important d'avoir une vue globale ou intégrée de ces sources de risques et de leurs interactions. La notion de risque elle-même n'est pas toujours bien comprise. Il y a risque lorsque la performance du logiciel dépend de facteurs exogènes plus ou moins aléatoires et hors du contrôle du décideur responsable de l'adoption ou non du logiciel et du gestionnaire responsable de la performance du logiciel. Ces facteurs exogènes et aléatoires peuvent faire en sorte que la performance du logiciel sera inférieure ou supérieure à la performance anticipée ou moyenne. Cette volatilité devrait être comprise comme une caractéristique de la distribution de performance autour de la performance moyenne anticipée par le décideur ou utilisateur mais le risque est souvent perçu comme une mesure de la probabilité que la performance soit *inférieure* à un niveau de performance critique situé au-dessous du niveau de performance anticipée. En s'inspirant de la terminologie utilisée en finance, on pourrait alors parler de « performance à risque », i.e. le niveau de performance qui serait inférieur d'un ou deux écart-type à la performance anticipée. En d'autres termes, le niveau de performance critique par lequel le niveau de risque pourrait être mesuré correspondrait à ce niveau tel que la probabilité que le niveau réel soit inférieur au niveau critique serait disons de 5% ou de 10%. De toute évidence, la caractérisation de cette performance à risque ou de ce niveau critique de performance n'est pas une notion facile à quantifier. Mais elle a l'avantage de correspondre à un concept de risque bien connu.

Quels sont ces facteurs exogènes et aléatoires qui peuvent affecter la performance anticipée d'un logiciel. On peut mentionner les facteurs suivants (liste non exhaustive): la capacité informatique internes et les compétences de l'organisation dans la manipulation et l'intégration des logiciels; le niveau de développement et de concurrence dans le marché des fournisseurs de services; les garanties et les pénalités imposées au fournisseur ou les coûts qu'il doit encourir en cas de mauvaise performance; le stade de développement du produit ou du logiciel dans son cycle de vie. Ces facteurs peuvent être évalués avant qu'une décision

importante soit prise. Ils sont présents à la fois dans le logiciel propriétaire et dans le logiciel libre.

Les mesures que l'organisation peut prendre pour contrôler et réduire ces risques prennent diverses formes. On peut penser à des mesures *ex ante* (mesures d'autoprotection) visant à augmenter la probabilité que le logiciel livrera le niveau de performance promis et à des mesures *ex post* (mesures d'auto-assurance) visant à limiter les dégâts si jamais le logiciel implanté s'avère plus ou moins catastrophique. Ces mesures seront de toute évidence fonction de la nature 'propriétaire' ou 'libre' des logiciels considérés.

On affirme souvent que les gestionnaires et décideurs responsables de la mise en place et de la performance des logiciels libres se sentent souvent piégés par les mécanismes incitatifs internes (rémunérations, bonis et promotions) qui biaisent les décisions en faveur d'une prise de risque minimale et du *statu quo*, au désavantage des logiciels libres. En effet, si un logiciel propriétaire s'avère peu performant ou mal adapté aux tâches pour lesquelles il a été acquis, le gestionnaire responsable de la décision originale peut faire porter l'odieux de cette sous performance au fournisseur propriétaire du logiciel. Dans le cas du logiciel libre, cette possibilité de transférer le blâme n'est généralement pas disponible. Pour être efficace dans l'utilisation des meilleures sources de logiciels, propriétaires ou libres, doit résoudre ce phénomène d'incitations perverses.

e) Protection et sécurité

Dans l'étude MITRE (2002) pour le département de la défense des États-Unis, on fait remarquer que des éléments majeurs de la sécurité du département dépendent de l'utilisation de logiciels libres. Pour des raisons historiques, les premiers logiciels créés dans la communauté du libre ont été des logiciels de gestion de l'Internet, y compris des logiciels de sécurité. Les logiciels de sources libres ont été conçus pour être utilisés dans un environnement multi usagers et les caractéristiques de sécurité ont été adaptées à cette fonction [YLEM].

Le fait que le code soit ouvert comporte des avantages et des inconvénients. Le fait que le code soit disponible librement permet aux pirates informatiques de voir les faiblesses des logiciels; mais à l'inverse, les programmeurs peuvent identifier rapidement les problèmes et failles des logiciels. Lorsqu'un bogue est découvert, il est rapidement corrigé et une mise à jour du logiciel est effectuée très rapidement. Dans la mesure où les mises à jour sont gratuites, les utilisateurs peuvent utiliser les versions corrigées avant même qu'elles deviennent officielles ou attendre la version officielle. Les améliorations et les innovations sont toujours accumulées dans la version la plus récente du logiciel et tous peuvent l'utiliser.

En cas de besoin, un utilisateur peut d'appliquer des outils d'analyse statique afin de détecter la présence de code hostile [Kenwood 2001]. Ainsi, des outils automates peuvent être appliqués afin de réduire les efforts impliqués dans la recherche de vulnérabilités [Dale et alii 2004].

Tel que mentionné précédemment, un des résultats les plus surprenant de l'étude MITRE (2002) est le degré auquel la sécurité du département de la défense dépend des applications des logiciels de sources libres. Le bannissement de ces logiciels aurait donc des impacts négatifs immédiats, étendus et parfois très sévères sur la capacité du département à analyser et protéger ses propres réseaux contre les intrusions. Le bannissement supprimerait aussi la capacité unique de pouvoir modifier le code source des infrastructures rapidement en réponse aux nouveaux modes de cyberattaques.

La communauté entourant les logiciels de sources libres contribue aussi à la sécurité du département de deux façons additionnelles. Dans un premier temps, la communauté produit des logiciels comme «OpenBSD» comportant un taux de panne très bas combiné à une fermeture rapide des brèches de sécurité. Deuxièmement, la communauté a eu une fascination à long terme pour développer des applications plus sophistiquées afin d'identifier et d'analyser ces failles de sécurité dans les réseaux et les ordinateurs personnels.

Les logiciels permettent aussi une forme de publication active que les chercheurs utilisent pour partager non pas des résultats mais bien des logiciels qui peuvent être immédiatement

utilisés pour leurs travaux de recherche. Le bannissement des logiciels de sources libres affecterait ainsi à court et à long terme la capacité des systèmes et leur sécurité puisque les chercheurs perdraient accès à des ressources importantes.

Puisque la majeure partie des infrastructures d'Internet a été crée sous le modèle des logiciels de sources libres, ces infrastructures sont généralement plus âgées, plus matures au niveau du fonctionnement et plus stables que la plupart des logiciels propriétaires équivalents sur le marché.

f) Durée de vie, cycle de vie et stabilité

Les logiciels libres ne sont pas tous fiables et performants. Il existe de nombreux projets dans le libre qui disparaissent d'eux-mêmes sans susciter l'intérêt d'une communauté suffisamment large pour soutenir son développement. Néanmoins, lorsqu'une communauté suffisante accepte d'utiliser un logiciel libre, son évolution peut être soutenue par cette communauté sans que l'apport de son créateur initial ne soit nécessaire. Ce n'est pas habituellement le cas d'un logiciel propriétaire qui sera parfois abandonné pour permettre la commercialisation d'une nouvelle version. Dans le cas des logiciels libres, les codes sont et restent disponibles et les développeurs peuvent continuer de les améliorer; ainsi, les logiciels de sources libres auront souvent une durée de vie plus longue.

g) Support et documentation

Les logiciels commerciaux sont vendus avec des tutoriaux de qualité et les entreprises commerciales de logiciels offrent des services professionnels de support à la clientèle. Dans le domaine du logiciel libre, la qualité du support est plus inégale. Les logiciels de sources libres sont généralement développés pour des utilisateurs sophistiqués. Les interfaces ne sont donc pas toujours conviviales. Blanas (2003) affirme qu'il n'y a pas suffisamment de manuels de référence de bonne qualité pour supporter l'apprentissage des utilisateurs de logiciels de sources libres. L'utilisateur est souvent laissé à lui-même et doit décoder le code pour comprendre son fonctionnement.

Dans d'autres cas, il existe une pléiade d'ouvrages de référence. L'industrie du support et de la documentation est une industrie en soi et elle pourrait être très rentable. Dans certains cas, le modèle d'affaire de ceux qui donnent leurs logiciels consiste justement à fournir contre rémunération les services de documentation, de formation et de support. Ghosh (2002) soutient que les utilisateurs de logiciels libres sont généralement prêts à tolérer un manque de documentation et une interface moins sophistiquée en échange d'une réduction de coûts et d'une possibilité de modifier le code pour l'adapter à leurs besoins particuliers. Lorsqu'un support additionnel devient nécessaire, on peut faire affaire avec un fournisseur externe sans attache avec le développeur original du logiciel.

Il faut également mentionner que les communautés du logiciel libre sont fondées sur une certaine réciprocité. Des membres de la communauté peuvent solliciter de l'aide à d'autres qui acceptent de les aider. En contribuant à sa manière à l'avancement d'un projet, on attire la reconnaissance de ceux qui ont le projet à cœur et qui en retour accepteront de répondre aux demandes d'aide.

On peut conclure sur ces aspects du support et de la documentation par la citation suivante tirée du site Web de Gbdirect[10] :

> « *Detractors of open-source software quite rightly point out that the free licence to use the software includes no support contract. But they neglect to mention the other side of that issue: many proprietary software licences have no support included either. Indeed, the majority of mass-market proprietary software support is aimed at hand-holding for inexperienced users. Just as proprietary vendors will sell support contracts with agreed service levels, suppliers and third parties will provide support for open-source software. An example of this is the Gnat translator for the Ada programming language, as reported in [CONECTA2000] and elsewhere.*
>
> > *"ACT Europe was founded in 1996 to provide support for commercial, industrial and military uses of the GNAT Professional Ada 95 28*

[10] http://open-source.gbdirect.co.uk/migration/benefit.html

development environment in Europe. It was founded jointly by Ada Core Technologies Inc (ACT), and by the European members of the GNAT Ada 95 project. ACT is a privately held corporation and was founded in August 1994 by the principal authors of the GNAT Ada 95 compiler system. ACT was founded without any initial capital other than some small loans from the principles, and has existed entirely from revenue from its inception. ACT claims to be currently profitable. The people involved in both ACT and ACT Europe (from now on, ACT) have been working with Ada for over twenty years, starting with the development of a working Ada compiler for preliminary Ada in 1979, and the first validated Ada 83 system in 1983. This work was done at New York University by a team dedicated to the technical success of the Ada language, which moved to ACT after its foundation. GNAT is the most widely used Ada 95 development system, available on many platforms, from workstations and PCs to bare boards. Ada solutions using GNAT encompass conventional, real-time, embedded, and distributed systems applications. The GNAT technology has always been based on free software, and more specifically on the GNU toolset. The GNAT compiler is integrated with the GCC (the GNU C compiler) back-end. The GNAT debugger is based on GDB (the GNU debugger) that has been adapted for Ada 95. GLIDE, the GNAT integrated development environment, is based on Emacs, which has been adapted and complemented to create a friendlier and complete environment. The GTKAda GUI technology leverages on the GTK graphical toolkit and provides an advanced GUI builder. These are a few but significant examples of the technological offering of ACT. All the products being distributed by ACT are free software, usually under a GPL or LGPL-like licence. Since its foundation, ACT has established strategic partnerships with hardware and software manufacturers providing Ada 95 capabilities. In the former category is ACT's relationship with Silicon Graphics, Inc., whose new SGI Ada 95 product is based on GNAT. As of 1999, SGI sold over a

billion dollars of Ada related software and equipment. ACT has relationships with several other hardware manufacturers such Compaq and Hewlett Packard. Commercial customers of ACT Europe include Aerospatiale, Alenia, BNP, Boeing, British Aerospace, Canal+, CASA, Dasa, Ericsson, Hughes, Lockheed, TRW, etc."

To add some context to that quote above, Ada is a language designed specifically *for military, industrial and aerospace mission-critical and safety-critical systems where human life is routinely risked if software systems do not function correctly*. »

h) Le défi du changement

Il reste que le passage d'un environnement de logiciels propriétaires à un environnement de logiciels libres impose des coûts aux entreprises et organisations. Néanmoins, plusieurs le font. Quelques exemples bien documentés parmi bien d'autres suffiront pour illustrer notre propos.

Le gouvernement local de la Ville de Munich[11] a changé, en 2003, tout le système de son réseau informatique passant de Microsoft Windows au système d'exploitation d'Open Source Linux. C'est donc 14 000 ordinateurs de l'administration publique qui ont changé de système d'exploitation et qui seront aussi équipés d'autres applications provenant de l'Open Source. L'objectif premier de ce changement était de déployer une technologie de l'information qui stimule davantage la flexibilité commerciale et technologique à faible coût. Les raisons évoquées concernant ce choix sont multiples :

(i) indépendance : assurer à la ville une plus grande indépendance aux niveaux des technologies de l'information en n'étant pas lié qu'à une seule solution d'un seul vendeur;

[11] Références: Blau John, « Munich Chooses "LiMux" Over Microsoft », 02/2005.
Silicon.com. « Norwegian city government switches to open source », 02/2005.
Desktoplinux.com. « Munich goes with Open Source Software », 02/2005.
CRN. « Munich Approves Changeover from Microsoft to Linux ».
USA Today. « Linux took on Microsoft, and won big in Munich »
Galli Peter. « Microsoft Loses Munich Deal to Linux ».

(ii) liberté : apporter une plus grande liberté de choix, laisser aux utilisateurs la possibilité de prendre eux même leur décision concernant le temps propice pour le renouvellement des logiciels;

(iii) réduction des coûts : faible coût de déploiement de Linux sans aucun coût de mise à jour et permet d'éviter les coûts de licence de Microsoft;

(iv) efficience : une plus grande efficience dans les opérations;

(v) consolidation de la charge de travail;

(vi) un meilleur rendement sur investissement sur une plus longue période.

En novembre 2004, la compagnie nationale allemande de chemins de fer[12] a équipé ses 300 serveurs d'une technologie de source libre. Cette migration se poursuit toujours avec l'annonce du remplacement de Lotus Notes par un système d'exploitation de source libre. Ce changement touchera au total 55000 utilisateurs et est le plus gros projet de migration du genre. D'ici la fin de l'année, le système d'exploitation de sources libres va être utilisé dans toutes les bases de données, serveurs d'applications, serveurs Web et les infrastructures de réseau. Par ce changement, l'entreprise souhaite obtenir une technologie de l'information qui soit efficiente et économique. Parmi les raisons invoquées pour justifier ce choix on retrouve :

(i) le coût, l'entreprise prévoyant épargner 50% de ses coûts en adoptant cette technologie de sources libres;

(ii) la flexibilité, Linux étant indépendant des manufacturiers ce qui le rend plus flexible;

(iii) la stratégie, Linux étant considéré comme une clé importante de la stratégie de l'entreprise.

Sherwin-Williams[13] a décidé de remplacer son système d'exploitation Unix par Linux. Le logiciel de source libre Linux sera ainsi installé dans les 2 500 magasins de l'entreprise aux

[12] Références: McCarthy K. « World's Largest Linux Migration Gets Major Boost »
IDABC. « German national railway moves 55 000 Notes users to Linux system »
Smith W. R. « German Federal Railways opt for Open Source to cut costs »
Millard E. « German Railways on Linux Track »

[13] Références: BusinessWeek Online. « Giant Steps for a Software Upstart »
eWEEK.com « Sherwin-Williams Picks an All-Blue Network »

Etats-Unis, au Canada et à Puerto Rico. L'entreprise prévoit épargner des millions de dollars en utilisant Linux qui lui permettra d'éviter les coûts de licence et d'abonnement pour chaque ordinateur. Elle pourra modifier le code source afin de l'ajuster à ses besoins. En effet, elle estimait avoir besoin de solutions flexibles afin de maintenir son leadership dans un environnement compétitif. De l'avis des analystes, Linux donne accès à une meilleure technologie à chaque point de vente et ultimement permet d'apporter de meilleurs services à leurs clients.

4. Innovation, par design ou par évolution

4.1 Cathédrales ou bazars

Une des questions importantes à débattre est la suivante : quelle forme de propriété intellectuelle est la plus susceptible de favoriser l'innovation? Tel que mentionné en introduction, deux visions s'opposent : la science ouverte où les idées circulent librement et l'innovation commerciale protégée par la propriété intellectuelle et/ou le secret commercial. E.S. Raymond contraste cette approche en parlant de « *Cathedral or bazaar* ». L'innovation dans la science ouverte se fait de manière évolutive, décentralisée, presque désordonnée. Dans un contexte privé, l'innovation se fait de manière planifiée et structurée avec un objectif précis. De manière générale, nous ne croyons pas qu'une approche soit meilleure que l'autre, nous croyons au contraire que les bénéfices relatifs de ces approches dépendent du contexte et que globalement les deux approches sont complémentaires et nécessaires. Nous examinons, dans cette section, les différences entre les processus d'innovation ouverts et propriétaires et nous discuterons comment ces deux processus peuvent se complémenter.

La protection accordée aux sociétés privées pour leurs innovations garantit que celles-ci continuent à être incitées à investir dans la recherche appliquée et dans la création de nouveaux produits. Les entreprises privées ont la mission d'identifier de nouveaux produits et services susceptibles de générer de la valeur pour les consommateurs et de créer et mettre

---

Network World Fusion. « Sherwin-Williams picks IBM and Linux for stores »

sur le marché ces nouveaux produits et services de la manière la plus efficace qui soit. Les entreprises privées doivent être encouragées à le faire et par conséquent doivent pouvoir capturer une partie de la valeur que leurs innovations génèrent.

Dans un contexte concurrentiel, la poursuite du profit assure la création constante de nouvelles innovations et un accroissement du bien-être des consommateurs. De manière générale, les gouvernements ne sont pas particulièrement compétents ou efficaces à identifier, aux fins d'une politique industrielle, les « gagnants » dans une industrie donnée. De manière générale, que le gouvernement cherche à financer l'innovation de type commercial nous apparaît être une mauvaise base de politique industrielle. Il vaut mieux laisser à l'entreprise privée le soin de créer et commercialiser de nouveaux produits pour satisfaire aux besoins des consommateurs.

Ceci étant dit, la science ouverte reste essentielle dans le développement des innovations. Le XXe siècle a vu l'émergence d'une véritable révolution scientifique. Le nombre de chercheurs, le nombre de revues scientifiques et d'articles scientifiques ont cru à un rythme exponentiel au cours des dernières décennies. Cette croissance a été accompagnée, grâce au Web, par une plus grande accessibilité des connaissances. Le concept de la « science ouverte » fait partie de nos vies, et reste un moteur important de développement économique.

Afin d'expliquer la croissance exponentielle des innovations scientifiques, des chercheurs ont proposé le concept « innovation combinatoire ». Les innovations ne procèdent pas uniquement de la création de nouvelles idées, mais aussi de la combinaison ou recombinaison d'idées ou innovations déjà existantes. Ces nouvelles combinaisons permettent de créer à leur tour de nouvelles combinaisons et innovations. L'objet de la recherche fondamentale consiste à créer de nouvelles méthodes ou concepts qui permettent de développer, en association avec des connaissances existantes, des nouvelles innovations jusqu'alors impossibles. Si Leonardo da Vinci, malgré tout son génie, n'a pas pu réaliser sa machine volante, ce n'était pas parce qu'il était moins génial que les frères Wright, c'est qu'il n'avait pas accès au moteur à combustion.

Ce principe de la fertilisation croisée des idées est au cœur du processus de développement des innovations en sciences pures et appliquées, tant naturelles que sociales. Le même principe est au cœur du développement industriel et commercial. Varian [2004] note que l'utilisation de standards et de pièces interchangeables ont contribué à accélérer le processus d'innovation aux États-Unis à partir de la fin du XIXe siècle[14]. Le système de brevets vise à encourager la divulgation des secrets commerciaux. Ceci permet à une entreprise de reprendre une innovation créée et divulguée afin de la combiner à d'autres innovations pour créer un nouveau produit.

Un système efficace de protection de la propriété intellectuelle et le secret commercial sont des mécanismes bien adaptés à la création de nouveaux systèmes propriétaires et à la création de produits commerciaux pour lesquels l'adaptation aux besoins des consommateurs et la commercialisation sont coûteuses. Par ailleurs, ce système est moins bien adapté au soutien des innovations qui nécessitent la combinaison de différentes innovations provenant de différentes sources. En effet, les systèmes propriétaires, en particulier dans le monde du logiciel, s'appuient souvent sur le secret commercial; ainsi beaucoup des innovations contenues dans ces logiciels ne peuvent pas être reprises et être incorporées dans d'autres produits. Cet effet est atténué lorsque l'entreprise innovante a recours à la protection des brevets.

Le rôle du système de brevets est précisément d'encourager la divulgation des secrets commerciaux afin de permettre la réutilisation par d'autres de l'innovation. Ceci permettrait à une entreprise de reprendre des innovations créées par d'autres pour les incorporer dans un nouveau produit ou service. Dans le cas des brevets, un nouveau problème apparaît, celui des « *anticommons* ». Lorsque plusieurs brevets appartenant à plusieurs firmes sont combinés pour créer un nouveau produit vraiment innovant, chacun des détenteurs de brevets peut revendiquer une part du surplus généré par la nouvelle innovation, ne laissant à la limite rien à la firme qui a innové en combinant ces brevets. En pratique, des brevets peuvent être

[14] « The attempts to develop interchangeable parts during the early nineteenth century are a good example of a technology revolution driven by combinatorial innovation. The standardization of design (at least in principle) of gears, pullies, chains, cams, and other mechanical devices led to the development of the so-called "American system of manufacture" which started in the weapons manufacturing plants of New England but eventually led to a thriving industry in domestic appliances. »

utilisés non pas pour encourager la création de nouvelles innovations mais plutôt pour bloquer l'entrée d'éventuels compétiteurs sur un marché. Dans ce cas, le système de brevet vient ralentir plutôt qu'accélérer l'innovation.

La science ouverte est idéale pour encourager la diffusion large des nouvelles idées, elle facilite l'assemblage et la combinaison de petites et diverses innovations, elle met en occurrence les chercheurs et assure un transfert et une transformation continue des connaissances et du savoir. La science ouverte s'appuie sur la motivation intrinsèque des innovateurs et sur leur volonté de contribuer à l'avancement des sciences et de promouvoir leur réputation dans leur communauté académique, intellectuelle et/ou scientifique. La science ouverte est particulièrement apte à résoudre des problèmes complexes et difficiles qui accordent à ceux qui réussissent à les résoudre gloire et prestige. Parce qu'il est difficile de financer des investissements dans de la recherche générique du domaine public, la science ouverte requiert l'assistance de fonds publics et c'est là un des rôles primordiaux de l'État.

Ainsi, les sciences « ouvertes » et « propriétaires » se complémentent, l'une et l'autre jouant un rôle dans la production de nouvelles idées, de nouveaux procédés et la création de nouveaux produits et services.

4.2 L'innovation dans le domaine du logiciel

Dans le domaine du logiciel, les deux modes d'innovation, ouvert et propriétaire, cohabitent. Le système propriétaire a permis la création de géants comme Microsoft ou SAP, le système ouvert a permis la création de système d'exploitation comme Apache, Linux, de logiciels de bureautique comme Open Office et le développement de standards ouverts comme HTML (Hyper Text Markup Language) et http (Hyper Text Transfer Protocol). Il ne faut pas être devin pour prévoir les prochaines années verront des innovations et des transformations importantes dans le domaine du logiciel et de l'Internet.

À première vue, on pourrait penser que le système d'innovation basé sur le logiciel propriétaire est plus adapté au développement de logiciels. Le développement de logiciels

n'est pas une activité de recherche fondamentale, précompétitive mais plutôt l'aboutissement d'une volonté de créer et diffuser un produit de nature commerciale. Cela relèverait donc du développement industriel. Dans ce contexte, l'approche basée sur le logiciel libre ne devrait pas pouvoir concurrencer le logiciel propriétaire. Or, voilà le paradoxe : cette approche semble fonctionner et être une manière efficace de créer de la valeur et de générer des innovations majeures dans le domaine du logiciel. On doit ce paradoxe aux caractéristiques propres au logiciel et à l'évolution récente des technologies de l'information et des communications.

L'avènement des technologies de l'Internet a contribué à accélérer le processus d'innovation, notamment de l'innovation s'appuyant sur le modèle de la science ouverte. L'Internet a permis une plus grande diffusion des idées. Mais aussi, les standards de l'Internet permettent de combiner et de réutiliser facilement différentes contributions et innovations pour créer de nouveaux services et produits. Le génie associé aux protocoles HTML et http est justement qu'il permet de relier et d'assembler des pages créées partout à travers le monde. Les blogs et les wikis sont des outils dont le rôle est de partager la connaissance. Commencée en 2001, l'encyclopédie libre Wikipedia est devenu une vaste encyclopédie en plus d'une trentaine de langues et regroupant, pour l'anglais seulement, près d'un demi million d'articles. Cette colossale ressource documentaire est le fruit de contributions volontaires et est gratuitement accessible sur le Web. Une telle réalisation a été possible parce que la technologie Web des wikis permet d'assembler, classer et gérer les contributions d'un grand nombre de contributeurs.

L'existence des blogs et des wikis reflète l'impact des nouvelles technologies du Web, mais aussi d'un phénomène social puissant qui est propre à la culture de l'Internet : l'effet Agora. L'Internet permet la création, souvent rapide et spontanée, de communautés de toutes sortes. Des groupes d'intérêt s'échangent des conseils ou des informations. Des jeux de rôles regroupant plus de 10,000 joueurs existent sur l'Internet, où chaque participant assume une identité dans le jeu virtuel. Les participants passent des heures à conquérir des nouveaux territoires, négocier des traités, faire des alliances et gravir l'échelle sociale. Les clients d'Amazon soumettent des commentaires sur les livres qu'ils ont lus. Des parents partagent

leurs expériences avec d'autres. L'Internet est devenu un véritable lieu d'échange et d'interaction sociale. Le développement du logiciel libre s'appuie en partie sur la formation spontanée de ces communautés. Le succès surprenant de Linus Torvalds, l'instigateur de Linux, est dû à la force de la communauté qu'il a réussi à créer autour de son projet. Aujourd'hui la communauté de support à Linux est considérable en ressources humaines et financières. L'effet Agora est un phénomène nouveau et puissant et certaines entreprises privées ont compris l'intérêt de chercher à l'exploiter.

Les récents développements en génie logiciel permettent de réutiliser et de recombiner des composantes logicielles existantes. La dynamique du monde des affaires requiert de développer toujours plus rapidement des applications informatiques, soit internes à l'entreprise, soit entre entreprises, pour des transactions de commerce électronique. Une des voies très prometteuses afin d'accroître la rapidité est celle de la réutilisation de composants prédéfinis, faciles à adapter à un contexte spécifique et à assembler. La réutilisation peut porter sur des éléments génériques, c'est-à-dire, communs dans plusieurs situations, ainsi que sur des patrons, c'est-à-dire, des solutions éprouvées dans des situations semblables. Pour que cette réutilisation soit efficace, elle doit permettre d'exprimer et d'exploiter la spécificité de chaque contexte.

Plusieurs technologies, fondées sur les technologies apparues dans les dernières années, permettent une réutilisation accrue du code source : UML (Unified Modeling Language), ebXML, (Electronic Business using eXtensible Markup Language), et MDA (Model Driven Architecture). Parallèlement, certains regroupements professionnels ont créé des catalogues de processus génériques propres à leur industrie, dans le but de favoriser le commerce entre entreprises. Ces développements récents permettent enfin de créer des patrons décrivant les processus d'affaires des outils informatiques adaptés et réutilisables. Les services Web s'appuient sur des standards et des technologies qui permettent aux ordinateurs d'assembler dynamiquement et au besoin des informations et des ressources disponibles sur le Web.

Ainsi, la nature même de ce qu'est un logiciel et les développements récents dans les technologies du Web et du génie logiciel tendent à favoriser un modèle d'innovation basé sur le code ouvert et le partage instantané des connaissances.

4.3 Pourquoi contribuer au projet de logiciels libres ?

Pour l'analyse économique, une des questions importantes et paradoxales est de comprendre pourquoi des individus et des sociétés investissent du temps et des ressources pour soutenir des projets en logiciels libres. Les contributions à un projet de logiciel libre peuvent venir de trois sources : des individus, c'est-à-dire des développeurs autonomes qui désirent contribuer volontairement à développer des solutions nouvelles et se faire une réputation dans la communauté; des sociétés de logiciels qui choisissent de développer leurs projets en mode libre parce que cela représente une stratégie avantageuse de développement; et les entreprises utilisatrices qui cherchent à améliorer le logiciel pour leurs propres besoins et jugent avantageux de partager leurs travaux avec la communauté, en partie pour capturer les effets d'arborescence mentionnés plus haut.

Lerner et Tirole [2000] tente d'expliquer pourquoi des individus acceptent de contribuer à des projets de logiciels libres. En participant à des projets de logiciels libres, un individu peut apprendre, se faire connaître dans la communauté et éventuellement vendre son expertise confirmée par l'intermédiaire de services d'intégration ou de customisation. Des entreprises géantes comme Sun, HP et IBM ont investi des ressources importantes dans le financement de projets de logiciels libres. En soit, ceci peut surprendre. Pourquoi des entreprises à but lucratif engageraient-elles des ressources financières et humaines dans la création de produits distribués gratuitement à la clientèle et ouvert à l'examen de la compétition sans la protection du secret commercial? Les entreprises qui supportent des projets de logiciels libres doivent évaluer qu'il est plus avantageux (moins coûteux et plus efficace) d'utiliser une approche ouverte pour créer et développer ces logiciels que d'utiliser une approche basée sur le secret commercial. De toute évidence, il doit y avoir un intérêt économique à le faire. Cet intérêt peut prendre diverses formes.

1) Les produits créés en mode « logiciel libre » sont complémentaires aux produits vendus en mode propriétaire par ces compagnies. Ainsi, en rendant accessibles sur le marché des solutions logicielles libres, ces entreprises augmentent la valeur de leurs autres produits et réussissent ainsi à tirer bénéfice de ces logiciels. Les firmes qui ont initié des projets de logiciels libres rentabilisent leur investissement en offrant du support, de la formation, etc. Ces services et opérations peuvent être très rentables.

2) Les logiciels sont devenus des objets très complexes. Prétendre que l'on peut tout planifier et contrôler est sans doute illusoire. Une approche plus évolutive, par essais et erreurs, est sans doute plus sage, moins risquée et plus rentable. En procédant par une approche ouverte, ces entreprises peuvent coopter l'apport – volontaire et gratuit – de nombreux collaborateurs externes et accéder à un pool plus vaste d'expertises. Ceci permet d'accélérer le processus d'innovation et d'identification des bogues et des besoins.

3) Une approche ouverte permet de mieux motiver les développeurs internes en les mettant en concurrence avec les développeurs externes et en leur permettant de se développer une notoriété externe. En rendant accessible le code source et en l'ouvrant à l'examen des autres développeurs, la firme peut rapidement savoir si le projet en logiciel libre suscite de l'intérêt auprès de la communauté et si la compétence des développeurs internes est reconnue par la communauté. Pour le développeur interne, un projet en logiciel libre peut lui permettre d'acquérir de la reconnaissance externe et faire augmenter sa valeur sur le marché des développeurs. Dans les deux cas, une plus grande transparence limite les problèmes de risque moral entre l'employeur et le développeur : le premier s'assurant d'une plus grande qualité et d'un plus grand effort au travail du second, alors que le travailleur s'assure que son employeur aura intérêt à le rémunérer à sa juste valeur sur le marché.

4) Dans le domaine des logiciels où les effets d'arborescence et de réseaux sont très importants, il est très difficile voire impossible de supplanter des logiciels propriétaires concurrents déjà largement utilisés. Seule une stratégie basée sur le logiciel libre semble pourvoir permettre à Sun, HP ou IBM d'offrir une réelle concurrence à Microsoft.

Dans le cadre d'un projet de développement de logiciel libre, il peut être avantageux pour une entreprise utilisatrice qui a les compétences et les ressources nécessaires d'utiliser des logiciels libres, de contribuer à leur développement et de remettre à la communauté ses améliorations. Contrairement à ce que l'on peut penser, le logiciel libre n'est pas gratuit. La formule consacrée est que « le logiciel libre est gratuit seulement si votre temps est gratuit ». En effet, les solutions de logiciel libre sont souvent moins conviviales que les solutions propriétaires. Les développeurs du libre sont plus intéressés à résoudre des problèmes techniques qu'à documenter leurs travaux pour les profanes et à embellir leurs interfaces. Le support à la clientèle est souvent limité. De manière générale, les utilisateurs de logiciels libres doivent avoir une expertise minimale ou accepter de faire appel à de l'aide professionnelle. Néanmoins, si une entreprise a les ressources et l'expertise nécessaires, une solution en logiciel libre peut être plus intéressante. Le premier avantage est qu'un logiciel libre est bâti pour être évolutif, flexible, modulaire et facilement adaptable. À l'exception des logiciels de bureautique qui sont relativement standardisés, les logiciels d'applications d'affaires doivent être adaptés pour les besoins spécifiques de l'entreprise. Ainsi, une entreprise qui implante un système informatique doit investir des ressources pour adapter le système à ses besoins. Dans certains cas, elle devra faire des ajouts au code source pour faire cette adaptation.

5. Le calcul économique appliqué au logiciel libre et ouvert

Les gouvernements ont souvent de la difficulté à articuler une politique industrielle claire, parce que leurs objectifs sont souvent mal définis, trop nombreux et contradictoires. Les gouvernements veulent soutenir l'emploi, les régions et les grands centres, les entreprises

locales, créer des grappes industrielles, etc. Cette confusion dans les objectifs et la prolifération des programmes gouvernementaux qui les accompagnent réduisent la portée des efforts de définir une politique industrielle.

Dans un système économique, ce qui importe c'est la création de valeur. L'économie québécoise ou canadienne doit pouvoir créer de la valeur. Dans le domaine du logiciel, la création de valeur correspond à l'utilisation accrue de logiciels utiles, performants, adaptés au besoin et peu coûteux à produire. Ainsi, soutenir une industrie locale de logiciels qui produit des solutions coûteuses et peu performantes n'est pas une bonne politique; une telle politique dévie des ressources rares et précieuses à des fins peu ou pas productives.

5.1 Les retombées économiques

Les gouvernements sont souvent préoccupés par les retombées économiques, un concept généralement mal compris et mal utilisé. Le terme « retombées économiques » est souvent une manière déguisée de parler des coûts d'un projet, i.e. des dépenses accourues pour réaliser le projet et qui sont comptabilisées comme revenus par les fournisseurs de service.

L'objectif du calcul économique dans l'évaluation d'un projet ou d'une politique gouvernementale est de permettre de comptabiliser les véritables retombées économiques. Ce qui est pertinent, ce sont les bénéfices réels générés par un projet. Nous n'entendons pas par là des transferts de revenus mais une réelle création de valeur. Si on transfère une somme d'argent d'un agent économique à un autre, le gain de l'un est équivalent à la perte de l'autre et aucun gain net ne peut être comptabilisé. Mais si on rend disponible un produit ou un service peu coûteux à produire mais qui procure un grand bénéfice pour les utilisateurs, alors on aura créé de la valeur.

De manière générale, les mécanismes de marché concurrentiel permettent aux activités économiquement avantageuses de se réaliser; s'il est possible de faire un profit en vendant un service ou un produit désiré par les consommateurs, cette activité aura lieu. Il serait justifié pour l'État de financer une partie de cette activité si et seulement si elle génère des effets

externes, i.e. des bénéfices réels que le promoteur du projet ne peut exploiter et accaparer et qui ne sont pas comptabilisés par l'unité ou l'agent qui finance le projet.

Nous désirons, dans cette section, explorer les avantages et inconvénients d'utiliser un type ou l'autre de licences logiciel en s'appuyant sur les règles du calcul économique. Nous en tirerons un certain nombre de conclusions importantes.

5.2 Octroyer un droit exclusif de commercialisation d'un logiciel à une firme et lui permettre de bloquer l'accès au code source détruit de la valeur

Lorsqu'on octroie un droit exclusif de commercialisation d'un logiciel à une firme et qu'on lui permet de bloquer l'accès au code source, on transfère à cette entreprise de la valeur. Ce droit a une valeur économique correspondant à la rente économique que cette firme sera capable d'extraire sur le marché sous forme de profit de la vente des droits de licence et de service. Alors pourquoi dit-on que l'octroi d'un tel droit détruit de la valeur ? Parce que la rente économique qu'extrait la firme grâce à son droit exclusif provient du fait que le prix de la licence est supérieur au tarif qui aurait été appliqué si le logiciel avait été libre. Ainsi, le gain de la firme propriétaire du code est au mieux égal à la perte pour les utilisateurs et en général ce gain est bien inférieur à cette perte.

Lorsqu'un logiciel est développé et son code source est rendu libre, des effets bénéfiques importants sont réalisés. Le code source créé et partagé a une valeur que d'autres peuvent profiter. L'accès au code source permet de répondre à certains de leurs besoins. De plus, en accédant au code source, les développeurs internes ou externes peuvent connaître la structure du code, apprendre comment il fonctionne et comment des projets similaires peuvent être réalisés. Ces bénéfices vont en général dépasser les profits qu'une firme propriétaire du code pourrait aller chercher en détenant les droits de commercialisation exclusifs. Afin d'aller chercher cette rente, la firme tarife son produit au-dessus du coût marginal de production (souvent nul) ce qui entraîne une sous-exploitation du code. De plus, en limitant à cette seule entreprise le droit et le devoir de valoriser la propriété intellectuelle incorporée dans le code,

on élimine probablement des opportunités intéressantes. Rendre le code propriétaire détruit de la valeur.

5.3 Permettre à une entreprise qui le désire de reprendre un code ouvert afin de l'améliorer et éventuellement de le commercialiser sous licence propriétaire crée de la valeur.

La protection de la propriété intellectuelle est une nuisance nécessaire. Elle est une nuisance parce qu'elle confère à son détenteur un pouvoir de monopole pour l'exercice de ses droits de propriété. Comme nous l'avons expliqué précédemment, ceci génère des inefficacités économiques en limitant la concurrence et l'accessibilité au produit. Mais, c'est là un mal nécessaire parce que la protection de la propriété intellectuelle crée des incitations puissantes à la création de nouvelles idées.

Une entreprise peut vouloir reprendre un code source ouvert, y apporter des transformations et chercher à commercialiser le résultat. Donner la possibilité à cette firme de faire ces améliorations et de commercialiser le tout permet de créer de la valeur. Les adversaires d'une telle idée affirment que cela équivaut à permettre à des entreprises de faire des profits en s'accaparant un bien public. L'argument est faux. Le code source libre reste libre et accessible. Quiconque le désire peut réutiliser le code ouvert, l'adapter à ses besoins et redonner ou non les modifications ainsi apportées à la communauté. De plus, la rente économique que la firme saura extraire en commercialisant son code propriétaire dépendra de la valeur ajoutée qu'elle aura incorporée dans le code initial. Dans la mesure où les utilisateurs conservent toujours l'accès au code source initial, la firme ne pourra pas tarifer son produit au-dessus de la valeur incrémentale qu'elle aura apportée au code source.

Interdire à une entreprise de reprendre un code ouvert afin de l'améliorer et éventuellement le commercialiser sous licence propriétaire limite les opportunités d'innovations. Évidemment, on préférait que toutes les innovations apportées au code source ouvert reste dans la communauté du logiciel libre. Mais en pratique, certaines innovations peuvent être coûteuses à réaliser et certaines entreprises pourraient décider d'y investir des ressources seulement si

elles sont capables de commercialiser sous licence propriétaire le résultat de leurs investissements. Il faut laisser le libre choix aux intervenants et laisser la concurrence déterminer quel type de licence est le plus approprié pour stimuler l'innovation dans ce contexte.

5.4 Une politique qui viserait à appuyer systématiquement le logiciel propriétaire dans le but de soutenir l'industrie du logiciel ne se justifie pas sur la base du calcul économique.

Certains suggèrent que les utilisateurs de logiciels ont besoin d'une industrie de services informatiques prospère pour offrir le support et les services connexes nécessaires et que, sans leurs droits exclusifs de commercialisation, les firmes de logiciel ne peuvent être financièrement viables. C'est là un argument erroné. Si les utilisateurs de logiciels ont besoin de support et d'une offre de services complémentaires pour utiliser le code source libre, alors il existera une industrie économiquement viable pour fournir ces services. Comme le code source est ouvert, il n'existera pas de barrière artificielle à l'entrée dans cette industrie du service et la concurrence forcera cette industrie à être efficace.

Dans le cadre de projets fondés sur le logiciel libre, les dépenses en frais de licences sont réduites mais les dépenses en frais de service, d'adaptation et de mise à jour restent sans doute importantes. Un projet d'implantation d'un logiciel libre « fait vivre » l'industrie locale tout autant qu'un projet avec licence propriétaire, la différence est que les frais de licence sont moins élevés et les frais de consultation sont plus élevés.

Dans un système économique, ce qui importe c'est la création de valeur. L'économie québécoise doit pouvoir créer de la valeur. Dans le domaine du logiciel, la création de valeur correspond à l'utilisation accrue de logiciels utiles, performants, adaptés au besoin et peu coûteux à produire. Ainsi, soutenir une industrie locale de logiciels qui produit des solutions coûteuses et peu performantes n'est pas une bonne politique; une telle politique dévie des ressources rares et précieuses à des fins peu ou pas productives. Les centaines de millions de

dollars investis dans de grands projets informatiques qui se sont avérés des échecs auraient pu être mieux investis ailleurs.

Les politiques gouvernementales à l'égard de l'industrie du logiciel doivent soutenir et encourager d'abord et avant tout la concurrence et l'innovation. Ces deux objectifs sont liés : un des éléments importants d'une politique de soutien à l'innovation est la concurrence. Pour survivre dans un monde concurrentiel, les entreprises doivent innover et chercher constamment à satisfaire les besoins de la clientèle. La concurrence permet aussi aux consommateurs de capturer une partie de la valeur créée par les innovations.

Pour arriver à soutenir l'innovation, il faut dans la mesure du possible laisser jouer les forces du marché. Les gouvernements doivent rester neutres vis-à-vis l'utilisation ou non des logiciels libres dans les domaines qui dépassent leurs compétences et leurs propres besoins comme utilisateurs. Les gouvernements sont en général de mauvais juges pour identifier les solutions gagnantes.

Par contre, le gouvernement peut et doit comme utilisateur et consommateur de la technologie voir à ses propres intérêts et s'assurer que les solutions retenues sont celles qui répondre le plus adéquatement possible à ses propres besoins. Les gouvernements doivent éviter cependant de biaiser le jeu de la concurrence en soutenant une entreprise (locale ou étrangère) pour des raisons autres que le coût ou la qualité du produit. La libre concurrence doit récompenser les firmes les plus méritantes. De même, les contrats informatiques octroyés par l'État ne doivent pas être des subventions cachées à une industrie ou une entreprise locale ou étrangère.

Le gouvernement doit reconnaître aux entreprises innovantes le droit à la protection de leur propriété intellectuelle. Ceci ne signifie pas par ailleurs que le gouvernement doit accepter de payer plus cher pour un logiciel propriétaire que pour un logiciel libre de qualité équivalente sous prétexte qu'il faille encourager le développement de la propriété intellectuelle. Le gouvernement doit acheter le produit qui lui convient le mieux au meilleur prix.

5.5 Les critères utilisés dans le choix de solutions informatiques doivent inclure les effets externes et les effets à long terme.

Les décisions des administrations publiques sont grandement motivées par les contraintes budgétaires qui leur sont imposées. Trop souvent, les décisions prises le sont sans considérer les effets externes et les effets à long terme. Le résultat est que les décisions prises par certaines administrations publiques sont contraires à l'optimum économique.

Une pratique courante dans les administrations publiques consiste à accorder aux firmes qui développent des applications pour le gouvernement toute la propriété intellectuelle sur le code créé. Cette pratique permet notamment de réduire les coûts d'acquisition du logiciel pour l'unité administrative. À notre avis, ceci est une mauvaise politique d'un point de vue économique. Les organismes publics désirent réduire leurs coûts d'achat de logiciel mais payer des entreprises en leur accordant des pouvoirs de monopole pour l'exploitation de travaux réalisés pour les besoins du gouvernement est une manière économiquement inefficace de financer ces contrats. En particulier, en accordant un pouvoir de monopole sur le code réalisé, le gouvernement, lui-même futur utilisateur des nouvelles versions du logiciel, limite la concurrence future et peut finir par payer plus cher pour les services de mises à jour ou d'adaptation du logiciel.

Un autre effet pervers de la gestion par enveloppe budgétaire annuelle dans les administrations publiques est que les décisions sont prises sur la base d'objectifs à très court terme. Parmi les critères que devraient utiliser les gouvernements pour évaluer une solution informatique, un critère primordial est la capacité de ces solutions à évoluer. Un logiciel est un produit bien particulier. Il doit pouvoir s'adapter aux nouvelles technologies et standards et évoluer pour satisfaire aux nouveaux besoins. Acheter un logiciel que l'on va devoir mettre à jour et adapter à grand frais dans un proche avenir est une proposition économiquement désastreuse. Le résultat, c'est l'achat de logiciels moins coûteux à court terme mais aussi moins adaptables. À moyen et long terme, ces pratiques peuvent s'avérer très coûteuses. Les administrations publiques ont malheureusement tendance à vivre avec des budgets et des objectifs à court terme.

Le niveau de flexibilité est un critère qui doit être explicitement inscrit dans les critères d'évaluation des logiciels lors du choix des solutions informatiques. En particulier, le gouvernement doit éviter les situations de « lock-in », où l'acheteur de services logiciels devient captif de son fournisseur. Le gouvernement doit éviter de signer des contrats de service qui donnent de facto à ses fournisseurs un pouvoir de marché pour la fourniture de services futurs.

Ainsi, un des éléments importants d'une stratégie gouvernementale devrait être le recours aux standards ouverts internationaux. En adoptant des standards ouverts, on s'assure que plusieurs fournisseurs pourront continuer à fournir le support nécessaire. Si tous les documents gouvernementaux respectaient les standards ouverts d'OASIS (Organization for the Advancement of Structured Information Standards)[15], les documents pourraient être lus, interprétés et modifiés par n'importe quel logiciel qui respecte les standards. Un des domaines d'application est le protocole de standards pour les documents (Open Document Format for Office Applications : OpenDocument)[16]. Sous ce standard ouvert, n'importe quelle entreprise de logiciels peut produire un utilitaire pour la création, la modification ou l'interprétation de ces documents. Ceci assure la concurrence entre les firmes de développement de logiciels de bureaucratique, permet aux différents utilisateurs (y compris les citoyens) de choisir les logiciels qui leur conviennent et encourage la concurrence, l'innovation et la création de nouveaux produits de la part de firmes locales.

[15] OASIS (http://www.oasis-open.org/who/) est un organisme international sans but lucratif qui vise à promouvoir le développement, la convergence et l'adoption des standards en affaires électroniques. OASIS est un consortium regroupant plus de 600 organisations et membres individuels de plus de 100 pays.
[16] Le standard OpenDocument vise à définir un standard ouvert pour les documents générés par les applications de bureautique.

6. Recommandations

Dans cette dernière section, nous résumons nos conclusions en offrant six recommandations.

1) **Le gouvernement devrait fortement encourager l'utilisation de standards ouverts, reconnus à l'échelle internationale. Il devrait éviter d'adopter des solutions propriétaires qui limitent la concurrence et le rendent dépendant d'un fournisseur unique.**

Un des défis majeurs des prochaines années sera le problème de l'interopérabilité des systèmes d'information. Une des solutions envisagées pour permettre à un grand nombre d'acteurs de partager de l'information est de construire des systèmes centralisés s'appuyant sur des banques de données communes. Cette approche s'est avérée excessivement coûteuse et maintes fois irréalisable. Une alternative beaucoup moins coûteuse consiste à permettre à plusieurs systèmes de s'échanger de l'information commune. Cette pratique requiert que des systèmes d'information différents puissent « se parler » et échanger automatiquement l'information requise. Pour se faire, les ordinateurs doivent se comprendre mutuellement et se partager des protocoles communs de communication et que les documents soient construits à partir de structures et sémantiques communes. Afin de répondre à ces exigences, des efforts considérables ont été entrepris pour créer des standards ouverts et des ressources très importantes ont été investies par de grands consortiums internationaux pour identifier de tels standards. Le gouvernement ne doit pas s'isoler de ces mouvements, au contraire il doit y participer.

2) **Le gouvernement devrait éviter dans la mesure du possible d'octroyer, sans motifs explicites, des droits exclusifs de commercialisation pour les logiciels développés pour les besoins gouvernementaux. Le gouvernement devrait encourager les licences de type ouvert.**

Octroyer un droit exclusif de commercialisation d'un logiciel à une firme et lui permettre de bloquer l'accès au code source détruit de la valeur. La protection de la propriété intellectuelle

doit être utilisée pour récompenser l'innovation et rien d'autre. Les logiciels développés et réalisés sur mesure pour des besoins définis par des ministères ne devraient pas être protégés au chapitre de la propriété intellectuelle. Accorder une propriété intellectuelle pour une réalisation qui nécessite peu ou pas d'innovation véritable ne sert pas les buts poursuivis par la protection de la propriété intellectuelle et ne contribue qu'à accorder des rentes et un pouvoir de marché à une firme et ce, sans raison valable. Le gouvernement devrait insister pour que les logiciels créés pour ses besoins s'appuient sur des standards ouverts et soient remis à la communauté du logiciel libre.

3) **Le gouvernement devrait de préférence utiliser des licences de type ouvert qui permettent néanmoins à des firmes de reprendre le code et en faire une exploitation commerciale (licences BSD).**

Permettre à une entreprise qui le désire de reprendre un code ouvert afin de l'améliorer et éventuellement de le commercialiser sous licence propriétaire crée de la valeur. Il faut reconnaître qu'ils existent plusieurs sources d'innovation dans le domaine du logiciel. Permettre qu'un code ouvert puisse être repris et transformé à des fins commerciales n'enlève rien au code source initial mais permet éventuellement de générer des innovations utiles. Ainsi, dans la mesure du possible, il faut encourager l'usage de licences qui permettent ce qui de réutilisation.

4) **Lorsque le gouvernement utilise et adapte un logiciel libre, il devrait redonner à la communauté du logiciel ouvert les améliorations et les adaptations qu'il a apportées.**

Il existe certains avantages à redonner à la communauté du logiciel libre les modifications faites par une agence gouvernementale. Cela permet principalement (i) de détecter les erreurs faites plus facilement et rapidement et (ii) d'éviter le phénomène de « code fork », selon lequel les adaptations faites deviennent incompatibles avec l'évolution du code source de base. Ainsi, sauf dans les cas où la sécurité du système et la protection des données confidentielles le justifient, le gouvernement devrait redonner à la communauté du libre ces contributions au code source libre.

5) **Le gouvernement devrait adapter ses pratiques administratives de manière à ce que les critères de choix de logiciels incorporent explicitement le calcul des effets externes et à long terme.**

Les pratiques administratives ne sont pas toujours adaptées à la complexité des problèmes de choix de logiciel. Un logiciel est un produit complexe qui doit évoluer. Il existe des effets de réseaux importants dans l'utilisation de logiciels. Si chaque unité administrative choisit ses solutions informatiques sans tenir compte des effets externes et des effets à long terme, le processus sera inefficace. Le défi d'assurer une prise en compte de ces effets externes et dynamiques est de taille. À cet égard, le gouvernement doit articuler une vision claire.

6) **Le gouvernement devrait soutenir le développement d'un réseau d'expertise en matière de logiciels libres, dont un des rôles sera de faire la promotion de logiciels libres.**

Les vendeurs de licences propriétaires ont intérêt à investir beaucoup de ressources dans la promotion de leur logiciel. Ainsi il est facile de trouver des experts capables de nous expliquer les qualités d'un logiciel propriétaire. Malheureusement, une telle contrepartie n'existe pas ou peu dans le domaine du libre. Il est donc difficile pour les administrations publiques d'avoir accès à une expertise en matière de logiciels libres capables de les aider dans leur sélection de solutions technologiques. Pour s'éclairer, le gouvernement devrait susciter la création d'une telle expertise.

Références

Australian Government Information Management Office (2005). "A Guide to Open Source Software for Australian Government Agencies. Developing and Executing an ICT Sourcing strategy".
http://www.sourceit.gov.au/__data/assets/pdf_file/42065/A_Guide_to_Open_Source_Software.pdf

Blanas, George (2003). "2SOS-ware DEVILS, [Strategic Open Software DEVelopment ILlnesseS]", Department of Project Management, TEI of Larissa, Greece.

Bessen, James (2004). "Open Source Software: Free Provision of Complex Public Goods", ResearchOnInnovation.org. http://www.researchoninnovation.org/opensrc.pdf

Dalle J-M, David P.A., Ghosh R.A. and W.E Steinmueller (2005). "Advancing Economic Research on the Free and Open Source Software Mode of Production", forthcoming in *How Open is the Future?*, Edited by Marleen Wynants and Jan Cornelis, Vrjie Universiteit Brussels (VUB) Press, Brussels.

Danish Board of Technology (2002). "Open-source software in e-government", Danish Board of Technology. http://www.tekno.dk/pdf/projekter/p03_opensource_paper_english.pdf

Ghosh, R.A. (2002), "Free/Libre and Open Source Software (FLOSS): Survey and Study", International Institute of Infonomics, University of Maastricht, The Netherlands; Berlecon Research GmbH, Berlin, Germany (http://flossproject.org/report/index.htm)

Hawkins, R.E. "The Economics of Open Source Software for a Competitive Firm, Why give it away for free?", *Netnomics* (forthcoming)

Johnson, J.P. (2002). "Open Source Software: Private Provision of a Public Good," *Journal of Economics and Management Strategy,* v. 11, no. 4, pp. 637-62.

Johnson, J.P. (2001). "Economics of Open Source",
http://opensource.mit.edu/papers/johnsonopensource.pdf

Kuan J. (2001). "Open Source Software as Consumer Integration into Production", Haas School of Business, University of California-Berkeley.
http://papers.ssrn.com/sol3/papers.cfm?abstract_id=259648

Kenwood, C.A. 2001. "A Business Case Study of Open Source Software," Report #MP01B0000048, MITRE Corporation: Bedford, MA.
http://www.mitre.org/support/papers/tech_papers_01/kenwood_software/

Lerner, J. and J. Tirole (2002). "Some Simple Economics of Open Source", *Journal of Industrial Economics*, Vol. 50, no. 2 , 197-234.

Lerner, J. and J. Tirole (2004). "The Economics of Technology Sharing: Open Source and Beyond", Working Paper 10956, NBER Working paper series, http://www.nber.org/papers/w10956.

MITRE Corporation (2003). "Use of Free and Open-Source Software (FOSS) in the U.S. Department of Defense", Prepared for The Defense Information Systems Agency (DISA). http://www.egovos.org/rawmedia_repository/588347ad_c97c_48b9_a63d_821cb0e8422d?/document.pdf

Raymond, E.S. (2001). "The Cathedral and the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary", O'Reilly & Associates, Farnham, UK Feb 2001, ISBN 0-596-00108-8.

Schmidt, K.M. and M. Schnitzer, (2002). "Public Subsidies for Open Source? Some Economic Policy Issues of the Software Market", University of Munich, CEPR and CESifo.