DEFENCE R&D DÉFENSE

Logiciels libres et ouverts

Survol et guide préliminaire pour le gouvernement canadien

Robert Charpentier

Richard Carbone

Les auteurs accepteront gracieusement la rétroaction et les commentaires à:
FOSS@drdc-rddc.gc.ca

Distribution illimitée
RDDC ECR 2004-232

R & D pour la défense Canada–Valcartier
décembre 2004

Sommaire exécutif

Évolution des logiciels libres et ouverts Au cours des deux dernières décennies, le marché du logiciel a été dominé par les produits commerciaux (Commercial Off-the-Shelf (COTS)) tels que MS Windows et le système de gestion de base de données Oracle qui offrent une myriade de fonctionnalités à un prix raisonnable. Cependant, les limitations intrinsèques des logiciels commerciaux (p. ex. code source propriétaire, fidélisation forcée, mises à niveau coûteuses, brèches de sécurité, etc.) sont apparues avec le temps. Cela a entraîné le développement d'une 'économie' parallèle basée sur les logiciels libres et ouverts (FOSS). Les logiciels libres se définissent comme des programmes dont le code source est rendu disponible pour usage et modification sans les droits de licence dispendieux imposés par les éditeurs de logiciels commerciaux. Les logiciels libres sont développés soit par des volontaires, soit par le développement commandité par de grandes compagnies informatiques qui veulent inclure des logiciels de base afin de donner un avantage concurrentiel à leur matériel. Durant les dix dernières années, le phénomène des logiciels libres a connu un accroissement constant : des milliers de projets de logiciels libres réalisés grâce à la collaboration par Internet ; des centaines d'applications de grande qualité disponibles pour utilisation ou modification à un coût nul (ou faible) et des dizaines de logiciels libres maintenant considérés aussi matures et sécuritaires que leurs équivalents commerciaux.

La bonne réputation des logiciels libres et ouverts a attiré l'attention de plusieurs gouvernements dans le monde qui considèrent maintenant la migration systématique de leurs serveurs et leurs stations de travail vers les logiciels libres. Les chefs de file qui sont présentement en voie de migrer vers les logiciels libres sont le Royaume-Uni, l'Allemagne et la France, mais on estime que plus de 20 autres pays préparent une politique et un plan d'action afin d'adopter systématiquement les logiciels libres au sein de leur gouvernement et de leur système industriel. La justification stratégique de la migration vers les logiciels libres est typiquement liée à trois facteurs principaux : 1) l'espérance d'économies de coûts directs, 2) la réduction des pertes financières au niveau national dues aux importations de logiciels commerciaux et 3) l'espoir de mieux développer l'expertise nationale en TI grâce à l'accès au code source (et au développement d'éléments originaux) ce qui n'est pas vraiment possible avec les logiciels commerciaux.

Le Canada semble tirer de l'arrière dans l'adoption des logiciels libres. L'absence d'analyses de rentabilisation et la sous-estimation de la valeur stratégique des logiciels libres expliquent partiellement cette situation. Cependant, le Gouvernement du Canada (GdC) a récemment endossé une position proactive concernant les logiciels libres afin de s'assurer que le personnel du GdC est au courant des options disponibles et qu'aucune barrière à l'approvisionnement ne subsiste. Quelques initiatives bien structurées existent dans les secteurs de la santé et de l'éducation, et on remarque une prise de conscience accrue de la part du GdC, qui voit les logiciels libres comme une option de rechange viable aux logiciels commerciaux et au développement de code sur mesure coûteux.

Perspectives proposées pour le GdC Les logiciels libres ne constituent pas une panacée, mais ils offrent une possibilité technologique concrète et crédible. Le GdC pourrait profiter d'une plus grande diversité dans l'approvisionnement logiciel (code sur mesure vs libre vs commercial), d'une sécurité accrue par la vérification (et l'amélioration) du code source et d'une plus grande conformité aux normes et spécifications ouvertes qui contribuent à l'interopérabilité des systèmes.

Des mesures particulières sont proposées afin d'en améliorer la perception et l'utilisation au sein du GdC telles que : promouvoir les logiciels libres au moyen de publications, d'ateliers et de conférences ; tenir compte des solutions basées sur les logiciels libres lors de travaux contractuels lorsqu'elles sont concurentielles avec les autres stratégies de développement ; soutenir les ministères du GdC dans l'évaluation de cette technologie émergente. Ce rapport inclut aussi divers outils de navigation afin d'aider à identifier les logiciels libres appropriés, un chiffrier facilitant les comparaisons côte à côte des logiciels libres et commerciaux de même que quelques lignes directrices pour aider les gestionnaires de projets à déterminer la pertinence des logiciels libres dans le cadre de leurs projets particuliers.

Table des matières

1 Introduction

Après un début plutôt lent à la fin des années 1990, les logiciels libres ont connu une croissance constante et s'étendent maintenant à plusieurs architectures logicielles à travers le monde. Cette croissance impressionnante s'est appuyée sur de nombreux succès, la réputation de haute qualité des systèmes basés sur les logiciels libres et, bien sûr, l'espérance d'économies à la mise en oeuvre.

À l'automne 2003, Recherche & développement pour la défense Canada (RDDC) a entrepris une étude spéciale afin de déterminer le rôle des logiciels libres dans l'évolution de l'architecture de nos systèmes d'information. Celle-ci a ensuite été étendue au Gouvernement du Canada (GdC). Ce rapport résume nos conclusions en quatre principaux chapitres. Dans la première partie du rapport, nous offrons une introduction générale à cette technologie, suivie d'une perspective préliminaire pour le GdC (parties I et II). Nous tentons aussi d'identifier/catégoriser les logiciels libres par domaine d'application technique (partie III). Finalement, quelques lignes directrices sont proposées aux gestionnaires de projets du GdC pour l'évaluation de l'utilité des logiciels libres dans le cadre de leurs projets particuliers (partie IV).

2 Comment parcourir ce rapport

Il est recommandé aux lecteurs de débuter par le rapport de navigation. Si de plus amples informations sont nécessaires, des hyperliens peuvent être utilisés jusqu'aux références précises dans les bibliographies. Dans la plupart des cas, la référence complète peut être trouvée sur Internet en inscrivant le titre (et le nom de l'auteur principal) dans un moteur de recherche comme Google.

Les liens utilisent un code de couleurs. Un lien bleu pointe à l'intérieur même du rapport, tandis qu'un lien magenta pointe vers une page Web, indiquant que l'accès à Internet est requis. Pour revenir après avoir suivi un lien, utilisez la flèche 'Vue précédente' ('Go to previous view') ou les touches ALT-Flèche à gauche. Les flèches gauche et droite peuvent être utilisées pour se rendre à la page précédente/suivante. Il est aussi possible de faire un zoom avant/arrière en utilisant les raccourcis clavier suivants :

- Page entière (Fit in window) : CTRL-0
- Taille réelle (Actual size) : CTRL-1
- Pleine largeur (Fit width) : CTRL-2
- Contenu (Fit visible) : CTRL-3
- Plein écran (View in full-screen mode) : CTRL-L

En utilisant les signets de navigation (bookmarks) définis dans le format Portable Document Format (PDF), il est facile pour le lecteur d'obtenir une vue d'ensemble du rapport. Utilisez la touche de fonction F-5 du clavier pour faire apparaître ou cacher les signets (F-6 pour Acrobat 6).

3 Procédure de validation du rapport

Membres du groupe consultatif sur les logiciels libres
Cycle 1 – Valcartier

- *Robert Charpentier (RDDC, autorité scientifique)*
- *Richard Carbone (RDDC, éditeur technique)*
- *Paul-André Côté (RDDC, secrétaire)*
- *Martin Salois (RDDC, éditeur du rapport)*
- *David Demers (RDDC)*
- *Slt Stéphane Fortin (RDDC)*
- *Dr Denis Poussart (Université Laval)*
- *Max Blanchet (CGI)*
- *Bertrand Couture (DMR Conseil)*

Avec les précieux commentaires de :

- *Micheline Bélanger (RDDC)*
- *Yves van Chestein (RDDC)*
- *Julie Couët (RDDC)*
- *Louis Bastarache (IEEE Section - Québec)*

Cycle 2 – Révision à RDDC Bureau central

- *Gavin Hemphill (RDDC Atlantique)*
- *Bruce Skinner (RDDC Atlantique)*
- *Dave Hazen (RDDC Atlantique)*
- *Bill Page (RDDC Bureau central)*
- *Lawrence Kemdirim (RDDC Bureau central)*
- *Ingar Moen (RDDC Bureau central)*
- *Eric Fresque (RDDC Bureau central)*
- *Delmar Permann (RDDC Bureau central)*
- *Philip Staal (RDDC Bureau central)*
- *Dr Robert Walker (RDDC Bureau central)*
- *Mark Daniels (MDN DBGI (DIMR) 4-6)*
- *Ken Heaton (MDN J2 RS (SI))*
- *Karine Pellerin (MDN DIIRI (DDCEI) 3-5-3)*

Cycle 1 – Évaluation technique à Valcartier

En appui au processus d'écriture, un groupe d'experts en TI a été formé à RDDC Valcartier afin de réviser et valider le contenu de ce rapport (janvier/février 2004.) Voir la liste des noms ci-contre.

Cycle 2 – Révision à RDDC Bureau central

À la suite de cette première révision, une ébauche avancée a circulé au quartier général de RDDC pour permettre une revue critique du rapport dans une perspective corporative. Les auteurs tiennent à remercier particulièrement ceux qui ont proposé des commentaires dans ce second cycle de validation ainsi que M. Brian Cheers pour avoir patiemment révisé l'anglais de la version originale de ce rapport.

Cycle 3 – Révision au MDN/FC et dans les autres ministères

Au cours des mois de juin, juillet et août 2004, une ébauche avancée de ce rapport a circulé à l'intérieur du MDN et des FC ainsi que dans les autres ministères. Les commentaires et suggestions ont été ajoutés à la version mise à jour en septembre 2004. La version française de ce rapport a été produite en décembre 2004 grâce à l'appui financier du Secrétariat du Conseil du Trésor du Canada. Les auteurs tiennent à remercier Mme Caroline Lemelin pour son précieux travail de traduction.

Vous pouvez contribuer aussi !

Étant donné la rapidité de l'évolution des TI et la diversité des sujets couverts par ce rapport, toute contribution au contenu technique pour des éditions subséquentes serait grandement appréciée. Pour porter à notre attention des références pertinentes ou des commentaires généraux au sujet de ce rapport, les lecteurs sont invités à soumettre les références qui peuvent être citées à l'adresse suivante : FOSS@drdc-rddc.gc.ca

Partie I

Les logiciels libres (FOSS)
Introduction pour les gestionnaires

4 Définitions principales

Le terme 'logiciel libre' (Free Software (FS)) définit un concept de liberté, plus qu'un concept de gratuité

Selon la fondation FSF [1] de Richard Stallman, le mot anglais 'free' ne doit pas être pris au sens de gratuit (free-of-charge), mais plutôt au sens de **liberté** pour l'utilisateur (user's freedom) :

- **D'exécuter** le programme dans n'importe quel but.
- **D'étudier** le fonctionnement du programme et de l'**adapter** à des besoins particuliers.
- **De redistribuer** des copies originales ou modifiées du programme.

Un glossaire est disponible à la fin de ce rapport pour une définition plus formelle des termes reliés aux logiciels libres et une taxonomie graphique est disponible à la figure 1.

Le terme 'logiciel ouvert' (Open Source Software (OSS)) englobe plus que le simple accès au code source

L'Open Source Initiative (OSI) exige que le logiciel ouvert (OSS) se conforme aux critères suivants [2] :

- Distribution gratuite des programmes ou de leurs éléments.
- Le code source doit être inclus.
- Les travaux dérivés doivent être possibles et distribuables, au moins en fichiers de correctifs (patch files).
- La discrimination contre des personnes ou des groupes n'est pas permise.
- La discrimination contre un champ d'application n'est pas permise.
- Aucune licence additionnelle ne peut être imposée lors de la redistribution des logiciels ouverts (OSS).
- Les mêmes droits que pour la distribution initiale du logiciel devraient être octroyés.
- La licence ne doit pas restreindre d'autres logiciels.
- Aucune disposition de la licence ne peut être basée sur une technologie ou un style d'interface particuliers.

Quelques contributeurs aux logiciels libres trouvent ces exigences (FSF & OSI) difficiles à satisfaire dans le cadre concurrentiel de la production de logiciels, mais la plupart d'entre eux tentent d'adopter la philosophie de l'OSI, laquelle est considérée être plus pragmatique que "l'idéalisme de la fondation FSF."

Le terme 'code propriétaire' (Closed Source Software or Proprietary code) désigne les logiciels pour lesquels le code source n'est pas disponible

Les firmes commerciales cherchent à restreindre l'accès à leur code source dans le but de protéger leur propriété intellectuelle. Elles compilent leur code source propriétaire et distribuent le code exécutable (c.-à-d. la forme binaire du programme) qui ne peut être essentiellement compris que par l'unité centrale (Central Processing Unit (CPU)) de l'ordinateur. La plupart des logiciels commerciaux (COTS software) sont vendus selon cette stratégie (p. ex. MS Windows, MS Word, les détecteurs de virus de Symantec ou McAfee, etc.)

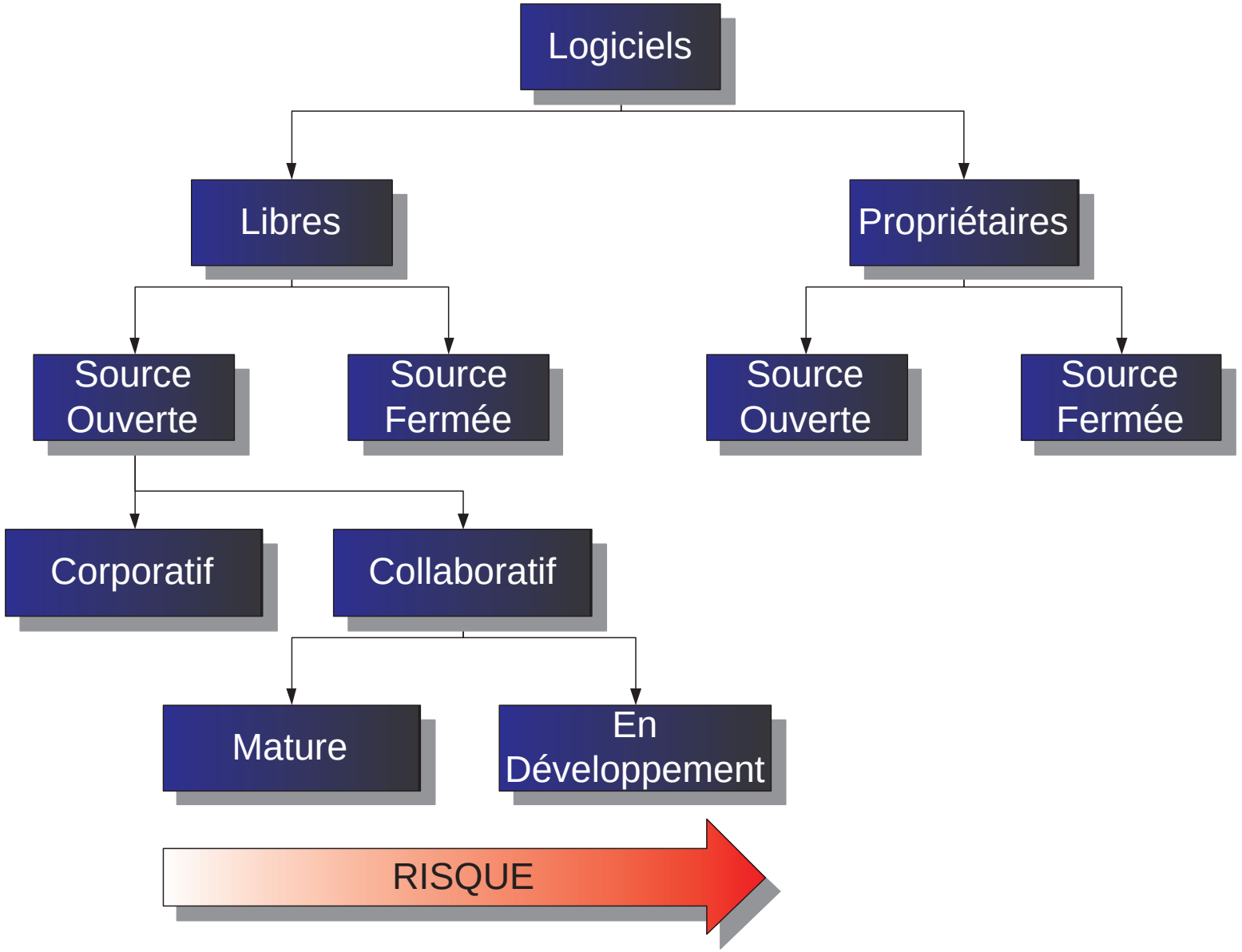

Figure 1: *Une taxonomie des logiciels*

5 Cadre juridique des logiciels libres

Les licences rattachées aux logiciels libres prévoient les droits fondamentaux pour le titulaire de licence et l'utilisateur, mais elles varient beaucoup selon les préférences du concepteur

Même si les utilisateurs de logiciels libres disposent de beaucoup de liberté, les programmeurs peuvent imposer certaines contraintes concernant l'exploitation des composants logiciels qu'ils rendent disponibles. Par exemple, une licence peut stipuler que le composant rendu disponible doit être intégré dans un système purement constitué de logiciels libres (c.-à-d. approche puriste) ou, alternativement, peut être assemblé avec des librairies propriétaires (approche plus pratique). Les architectes de systèmes doivent être attentifs aux modalités de la licence lors du choix d'un produit, afin d'éviter les problèmes juridiques. Des comparaisons de licences sont disponibles dans plusieurs références [3, 4, 5].

La licence la plus répandue est la GNU General Public License (GPL), laquelle est utilisée pour 65 % des logiciels libres

La licence GNU GPL a été adoptée par la plupart des programmeurs durant les 15 dernières années et est considérée comme le modèle de référence pour plusieurs autres contrats de licence. Ce modèle de licence exige que tout le développement complémentaire soit intégré seulement à des logiciels ouverts (open source) et publié sous une licence compatible avec la GPL. On s'attend cependant à ce que les modèles moins restrictifs (comme ceux de Mozilla et BSD) deviennent plus populaires dans le futur [4], car les systèmes hybrides, à la fois propriétaires et basés sur les logiciels libres, sont plus réalistes dans les infrastructures modernes.

La fondation FSF a proposé le terme 'copyleft' pour décrire le privilège d'utilisation libre des logiciels libres

S'opposant au concept de protection du droit d'auteur (copyright), un 'copyleft' décrit le cas ou le propriétaire abandonne la propriété intellectuelle et le droit de licence privé. Ce ne sont pas tous les logiciels libres qui adoptent le 'copyleft', mais la plupart de ceux qui sont sous licences GPL et Lesser General Public License (LGPL) l'adoptent. La compatibilité des divers modèles de licence entre eux et avec "l'idéal Copyleft" est discutée dans [3].

Quelques tactiques supplémentaires ont été développées afin de prévenir l'appropriation des logiciels libres par des firmes commerciales

Les logiciels libres sont intrinsèquement exposés au risque d'appropriation par les entreprises commerciales. Donc, en plus de l'octroi de licence, les développeurs peuvent se constituer légalement en corporation et/ou transférer leurs droits de propriété à une entreprise à but non lucratif et/ou déposer une marque de commerce sur leurs logiciels et logos, etc. [6].

Le brevetage d'algorithmes est intrinsèquement incompatible avec le développement de logiciels libres

Selon Richard Stallman, la pire menace à laquelle les logiciels libres sont confrontés provient des brevets d'inventions logicielles, lesquels peuvent interdire au logiciel libre l'usage d'algorithmes et de fonctionnalités pour une période allant jusqu'à vingt ans. La Free Software Foundation (FSF) fait circuler une pétition contre les brevets d'inventions logicielles [7]. Dans [8], l'auteur mentionne que Microsoft a augmenté sont taux de brevetage de plus de 30 % en 2002, ce qui semble confirmer que la menace est réelle. Des mesures pour empêcher les brevets d'interférer avec la liberté logicielle sont proposées dans [9].

---

6 Constats-clés sur l'évolution des logiciels libres

Avec les années, plusieurs logiciels utiles ont été distribués selon le paradigme des logiciels ouverts

Voici quelques exemples reconnus :

LaTeX Éditeur de texte (et composition) utilisé pour les publications scientifiques

Linux Système d'exploitation populaire basé sur Unix

Apache Serveur Web très fiable et sécuritaire

MySQL Base de données rapide, précise et polyvalente

Lors de la publication de ce rapport (mai 2004), on estimait que 115 logiciels libres [5] avaient atteint une maturité comparable ou supérieure à leur équivalent commercial.

Les logiciels libres ont aussi évolué vers un "processus de développement" très efficace

La force des logiciels libres provient de la capacité de recruter et motiver des programmeurs compétents, travaillant sur une base volontaire, afin de développer, déboguer et optimiser le code. La coordination est assumée par un leader délégué, qui est responsable de l'appréciation des diverses solutions offertes par les programmeurs et de l'intégration du meilleur code dans les prochaines mises à jour des logiciels libres, lesquelles sont rapidement mises en ligne [10].

Par sa simplicité et son efficacité, le modèle de développement des logiciels libres a régulièrement démontré plusieurs avantages

- Prodigieuse diversité de logiciels [11].
- Grande flexibilité et extensibilité (scalability) des solutions logicielles par l'édition de code source.
- Grande fiabilité et haute sécurité par la révision et la validation du code source [12].
- Mise en ligne de nouvelles versions à une vitesse d'un ordre de grandeur plus rapide que leurs équivalents commerciaux.
- Développement rapide de solutions sur mesure qui satisfont des exigences particulières grâce à la réutilisation et à l'ajout de code.
- Augmentation de la durée de vie des systèmes grâce aux mises à niveau du code source [13].
- Degré élevé de conformité aux normes ouvertes entraînant une plus grande interopérabilité des systèmes d'information.
- Des systèmes minimaux (en taille et en complexité) comparativement à leurs équivalents commerciaux qui souffrent souvent de ballonnement promotionnel.

Pour plus d'information sur les avantages des logiciels libres, les références suivantes sont recommandées [14, 8, 12].

---

7 Risques et inconvénients des logiciels libres

On retrouve quelques critiques dans la littérature technique

Au moment de la préparation de ce rapport (hiver 2004), on retrouvait les critiques suivantes dans la littérature technique :

- Le contrôle des versions peut être plus complexe avec les logiciels libres qu'avec les logiciels commerciaux (en évolution).
- Le maintien des systèmes requiert plus de ressources locales (discutable pour le long terme).
- Les administrateurs de systèmes doivent posséder des connaissances techniques supérieures.
- Les logiciels libres offrent parfois une moins bonne intégration aux suites d'applications, et une moins grande convivialité (en évolution).

Microsoft commandite plusieurs études contre les logiciels libres

On retrouve quelques autres critiques dans divers rapports, mais dans plusieurs cas, la perspective est clairement biaisée. Par exemple, le rapport de National Economic Research Associates (NERA) déclare qu'il n'y a rien d'incorrect concernant les logiciels propriétaires, car ils ont permis "une croissance très rapide de l'industrie du logiciel au cours des dernières décennies, procurant ainsi des logiciels plus puissants, faciles à utiliser pour plus d'usagers" [15]. Même si l'impartialité de NERA peut être mise en doute (car l'étude était commanditée par Microsoft), le rapport offre des contre-arguments très complets contre l'adoption des logiciels libres dans les gouvernements. Ce rapport n'est pas unique ! Un article récent de Todd Bishop indique que plusieurs études sur Linux réalisées par IDC, Giga et Meta Group étaient en fait commanditées par Microsoft. Les critiques remettent en question l'indépendance de ces analyses [16]. Quelques-unes de ces études sont énumérées ci-dessous :

- l'étude de Veritest comparant Windows 2003 et Red Hat Linux [17]
- l'étude du Meta Group sur un banc d'essai pour Linux [18]
- l'étude de Veritest comparant la performance des serveurs Windows 2003 et Linux [19]
- l'étude de Veritest comparant les serveurs Web de Windows 2003 et de Linux [20]
- l'étude de IDC sur Windows 2000 vs Linux dans l'informatique d'entreprise [21]
- l'étude de cas de 7-Eleven pour MS Windows 2003 au lieu de Linux [22]
- l'étude Total Economic Impact MS vs Linux [23]
- l'étude Counterpoint from Linux Insider comparant Windows vs Linux [24]

Fait intéressant, l'édition de janvier/février 2004 de IEEE Software inclut une série d'articles très positifs concernant les logiciels libres, lesquels sont précédés d'une introduction signée par deux directeurs scientifiques invités : Szyperski (Microsoft Research) et Spinellis (Athens University of Economics and Business) [11].

---

8 L'adoption des logiciels libres dans le monde

La communauté européenne adopte activement les logiciels libres, surtout dans les services publics

La Grande-Bretagne, la France et l'Allemagne sont les pays les plus avancés dans la migration des infrastructures des TI du secteur public vers des normes ouvertes et des logiciels libres. Le gouvernement de la Grande-Bretagne a adopté une politique afin de considérer les solutions ouvertes (open-source) par rapport aux solutions propriétaires lors de l'approvisionnement en Technologies de l'information (TI) [25]. À la fin de 2003, un guide de migration à été publié par NetProject [26]. Celui-ci fournit des lignes directrices très détaillées pour la migration vers les logiciels libres des postes de travail et des serveurs en général, et des principales applications (suite bureautique, courriel, bases de données, systèmes d'exploitation, etc.) en particulier. La KBSt (Allemagne) a aussi préparé un guide technique extrêmement détaillé qui facilite une stratégie de migration agressive vers les logiciels libres pour les serveurs et les postes de travail [27]. Les pays européens partagent ces connaissances au sein du projet Free/Libre and Open Source Software (FLOSS), qui est financé par la Commission européenne dans le cadre du programme des technologies de la société d'information. Leurs rapports couvrent, entre autres sujets : une politique pour l'Union européenne, des modèles de gestion des logiciels libres, une vue d'ensemble des développeurs et du code source [28, 29, 30, 31, 32, 33, 34, 35, 36]. L'Agence suédoise de gestion publique (Statskontoret) propose une excellente série de publications rigoureuses sur l'adoption des logiciels libres en Suède incluant [37, 38]. Lors d'une conférence tenue à Washington en juin 2003, on a estimé que vingt-quatre pays revoient présentement leur politique, incluant le Danemark [39], les Pays-Bas, l'Italie, la Norvège et la Suède [40, 41]. L'Open Source Observatory (OSO) maintient un site Web très instructif concernant l'évolution des logiciels libres en Europe [42].

Plusieurs pays d'Amérique latine, d'Afrique, d'Océanie et d'Asie s'orientent aussi, à divers degrés, vers les logiciels libres

Les raisons pour migrer vers les logiciels libres sont typiquement liées à trois facteurs principaux :

- L'espérance d'économies de coûts directs.
- La réduction des pertes économiques nationales causées par l'importation de logiciels commerciaux.
- L'espoir de mieux développer l'expertise nationale en TI via l'accès au code source (et le développement d'éléments originaux) ce qui n'est pas vraiment possible avec les logiciels commerciaux.

Un récent survol des politiques concernant les logiciels libres dans différents pays est disponible sur http://www.csis.org/tech/OpenSource/0408_ospolicies.pdf. Steven Weber (Université de Californie-Berkeley) offre une analyse intéressante des raisons pour lesquelles les pays en développement devraient migrer vers les logiciels libres [43]. Quelques annonces de décisions stratégiques ont été communiquées pendant notre étude par différents pays, comme Israël [44], le Japon, l'Afrique du Sud [45] et l'Australie [46, 47]. Le site Web de l'AFUL [48] offre aussi une liste, mise à jour périodiquement, des pays adoptant une politique ou des lois concernant les logiciels libres [49] et [50] est un site Web complémentaire intéressant. L'analyse de rentabilisation pour les pays émergents est revue dans [51].

9 Les logiciels libres aux États-Unis

Le mouvement pour les logiciels libres tire son origine principalement des États-Unis et y demeure très fort

Une pléthore de rapports abordent la croissance des logiciels libres dans différents secteurs de l'économie aux États-Unis. Une grande partie de cette information est incomplète ou biaisée – écrite afin d'appuyer une perspective donnée. Presque unanimement cependant, il est reconnu que les logiciels libres s'étendent rapidement dans la plupart des infrastructures des TI. Deux produits renommés, le système d'exploitation Linux et le serveur Web Apache, sont les plus souvent cités comme ayant une croissance rapide, en raison de leur maturité reconnue et de leurs qualités techniques lorsque comparées à celles de leurs équivalents commerciaux [52, 53].

Plusieurs grandes sociétés américaines contribuent au réseau des logiciels libres

En plus des logiciels développés par des groupes de volontaires, une contribution substantielle aux logiciels libres provient de grandes compagnies qui désirent expérimenter un modèle de gestion différent, basé sur le développement collaboratif. L'adoption de la stratégie ouverte par ***Netscape*** est un des cas de réussite les plus connus [54]. ***IBM, Hewlett-Packard, Sun Microsystems, Novell et Silicon Graphics*** ne sont que quelques-uns des chefs de file les mieux connus qui hébergent/contribuent/commanditent/soutiennent un grand nombre de projets sources ouvertes [55]. IBM s'est engagé formellement à accélérer le déploiement de Linux dans le secteur bancaire [56] et au gouvernement [57]. Dell Computer se tourne aussi vers les logiciels libres et s'attend à ce qu'ils soient largement adoptés [58].

Quelques initiatives du gouvernement des États-Unis contribuent aux logiciels libres

La commandite des logiciels libres par le gouvernement n'est pas fréquente, bien que quelques exemples soient rapportés dans [5] incluant le fameux projet Security Enhanced Linux (SELinux) qui peut être téléchargé directement à partir de la page Web de la National Security Agency (NSA) [59]. En géomatique, la National Technology Alliance (NTA) a commandité l'impressionnant projet Open Source Prototype Research, qui a eu un impact significatif sur les organisations du gouvernement américain utilisant de l'information géospatiale [60], incluant le département de la Défense (DoD). Plus récemment, un exemple de développement d'applications critiques au moyen de logiciels libres à été rapporté dans la revue IEEE Software [61] et décrit comment les logiciels libres ont été utilisés très efficacement dans un projet du JPL à la NASA.

L'adoption d'une politique ferme concernant les logiciels libres pourrait être problématique pour le gouvernement américain car l'industrie des logiciels propriétaires soutient grandement l'économie aux États-Unis

L'industrie du logiciel étant évaluée à 70 G$(US) [10], il n'est pas surprenant de constater une réaction vigoureuse de la part des éditeurs de logiciels commerciaux contre les logiciels libres [62]. Par exemple, la référence suivante [15] est une étude commanditée par Microsoft qui tente de contrecarrer le plan de mise en oeuvre des logiciels libres et un article plus formel [63] donne la perspective officielle de Microsoft sur les logiciels libres. Un essai nuancé sur l'économie des logiciels libres est proposé par David Adams dans [64].

---

10 Les logiciels libres au Canada

Le Canada semble tirer de l'arrière dans l'adoption des logiciels libres, particulièrement dans le secteur public

L'absence d'analyses de rentabilisation et la sous-estimation de la valeur stratégique des logiciels libres expliquent partiellement la situation [65]. Une excellente évaluation de la place des logiciels ouverts au Canada a été menée par e-Cology pour Industrie Canada [40] en 2003. Le rapport présente aussi le profil de 17 compagnies canadiennes exploitant les logiciels libres à différents degrés et offre une discussion révélatrice des stratégies de rentabilisation que ces firmes utilisent afin de maximiser le retour sur leur investissement.

Une politique sur les logiciels libres au Gouvernement du Canada (GdC) a récemment été entérinée

En juin 2004, le GdC a annoncé une nouvelle position concernant les logiciels libres. Elle se base sur une approche équilibrée afin de s'assurer que la politique et les lignes directrices gouvernementales ne favorisent pas un modèle de gestion logicielle plus qu'un autre (logiciels libres vs commerciaux vs code sur mesure). Quelques ministères ont reçu le mandat de soutenir la politique nationale sur les logiciels libres par des actions déterminées. Pour plus d'information consultez : `http://www.cio-dpi.gc.ca/fap-paf/oss-ll/oss-ll_f.asp`.

Les outils de développement logiciel, Apache et Linux sont les logiciels libres dont l'usage est le plus répandu au Canada

Lors de la rédaction de ce rapport, les logiciels libres au Canada étaient surtout utilisés en développement logiciel et dans les services de soutien (c.-à-d. serveurs et gestion de réseau). On s'attend à ce que cette tendance demeure dominante d'ici les prochaines années [65]. Les analystes décrivent souvent ce phénomène comme étant la pénétration horizontale du marché par les logiciels libres (c.-à-d. une couche de service offerte par les logiciels libres).

Quelques initiatives bien conçues existent dans les secteurs de la santé et de l'éducation

En Colombie-Britannique et au Québec, de vigoureux projets de logiciels libres tentent d'intégrer une suite logicielle complète pour les écoles et à d'autres fins éducatives [40, 66]. L'université McMaster et le Département de médecine familiale collaborent au Electronic Health Record (EHR) pour les médecins de famille [40]. La plupart des analystes considèrent qu'une telle pénétration verticale des logiciels libres (c.-à-d. à travers les multiples couches d'un domaine d'application donné) est requise pour confirmer une plus grande pénétration des technologies basées sur les logiciels libres.

---

11 Les logiciels libres et la sécurité logicielle

L'accès au code source facilite grandement la mise en oeuvre de la sécurité

Lorsqu'un logiciel est créé, son niveau de qualité dépend directement des compétences et de l'expérience du programmeur, ainsi que de sa méthodologie professionnelle. Afin d'accroître la fiabilité et la sécurité du code, il est essentiel d'utiliser quelques mécanismes complémentaires tels que la revue par les pairs, les tests, les audits de qualité, le contrôle les versions préliminaires d'évaluation (alpha et beta), etc. Les logiciels libres et les logiciels propriétaires s'appuient essentiellement sur les mêmes méthodes (probablement à des niveaux similaires) durant le développement principal. Cependant, après le premier lancement public, les logiciels libres offrent l'avantage très significatif de permettre l'accès au code source. Ceci permet d'effectuer plus de revues par les pairs, de tests, d'audits de qualité, et ce par une communauté beaucoup plus grande d'utilisateurs/développeurs que ce qui est possible avec le code propriétaire. Dans le cas du code propriétaire, les failles et les défectuosités du code sont souvent découvertes par des exploitations subversives qui peuvent entraîner des déstabilisations (c.-à-d. application de correctifs et réparations) dans les grandes sociétés qui dépendent des logiciels commerciaux. Paradoxalement, la confiance accordée aux logiciels libres peut croître plus rapidement et éventuellement atteindre un plus haut degré que son équivalent propriétaire [67, 68]. On trouvera une myriade de statistiques sur les vulnérabilités des logiciels au chapitre 6 de la référence [8]. Cela semble confirmer la perception générale qui veut que les logiciels ouverts soient souvent supérieurs au code propriétaire. Cette autre référence [69] donne une comparaison des vulnérabilités de Red Hat Linux (160) et Windows NT (1200) qui semble être plus scientifique, mais il faut être prudent afin d'éviter d'extrapoler au-delà de la portée originale de cette étude. En résumé, les logiciels libres ne sont pas intrinsèquement plus sécuritaires que les logiciels commerciaux, mais l'ouverture du code source rend l'application de la sécurité plus systématique et moins perturbatrice.

Le dilemme sur la sécurité dans l'obscurité vs l'ouverture a été le sujet d'un débat animé dans la communauté de la cryptographie dans les années 80. La décision finale a été de rendre les algorithmes de cryptographie généralement disponibles afin de s'assurer que la plus vaste communauté scientifique possible puisse procéder à l'évaluation et la validation de la sécurité. Whitfield Diffie, inventeur de la cryptographie à clé publique et maintenant officier de sécurité en chef chez Sun Microsystems, a souvent répété que "l'ouverture est essentielle à la confiance" en développement logiciel comme pour les protocoles de cryptographie d'il y a vingt ans [70, 71]. "Le soleil tue les bactéries" [67].

Les logiciels libres offrent aussi trois autres avantages clés

D'autres avantages des logiciels libres incluent :

1. Des systèmes moins lourds que leurs équivalents commerciaux qui souffrent souvent de ballonnement promotionnel. Comme ils sont plus petits, les systèmes sources ouvertes devraient laisser moins de possibilités à l'exploitation de failles.
2. Le code source peut être enrichi d'assertions, de contrôles de sécurité complémentaires, etc.
3. Une plus grande diversité de code dans "l'écosystème logiciel" qui pourrait réduire la vitesse et la prolifération des cyberattaques.

Quelques risques accrus à gérer

Les logiciels libres sont souvent perçus comme un retour en arrière vers la responsabilisation des ressources internes pour le développement et la maintenance des systèmes. Pour l'application de la sécurité, l'expertise qualifiée est rare et a souvent besoin d'être développée afin de faire face adéquatement aux responsabilités accrues requises par les systèmes basés sur les logiciels libres. Les logiciels commerciaux ont un avantage significatif sur les logiciels libres en raison de l'imputabilité intrinsèque au monde commercial (souvent grandement restreinte dans les licences d'utilisation du logiciel !) L'accès au code source peut aussi être un avantage pour l'agresseur qui peut tenter d'élaborer des attaques plus circonscrites sur le code source ouvert [63]. Quelques auteurs semblent aussi s'inquiéter des possibilités d'infiltration dans les projets de développement collaboratif par des programmeurs malicieux qui pourraient installer des portes dérobées (backdoors) ou d'autres fonctionnalités indésirables [72]. Quoi qu'il en soit, ni les logiciels commerciaux ni les logiciels sur mesure ne sont immunisés contre le code malicieux ou les défauts de programmation d'où résultent les vulnérabilités des systèmes d'information. Les adeptes des logiciels libres considèrent que ces menaces sont exagérées [73, 74, 75]. Comme on le notera plus loin dans ce rapport, les avantages et désavantages ne peuvent être évalués que dans le cadre précis d'un projet.

---

12 Synthèse des auteurs

Les logiciels libres ne devraient pas être considérés comme une panacée ni être ignorés comme un phénomène marginal sans pertinence

Les principales firmes de prévision technologique s'accordent à dire que les systèmes basés sur les logiciels libres continueront de se répandre au détriment de leurs équivalents commerciaux. Il semble évident que les avantages l'emportent sur les désavantages dans plusieurs cadres d'application. De nombreux logiciels libres ont atteint un niveau de maturité et de reconnaissance qui les placent dans une position de supériorité par rapport à leurs équivalents commerciaux. Compte tenu de la migration de plusieurs gouvernements autour du monde, on s'attend à ce que la qualité et la diversité des logiciels libres continue de s'améliorer.

Les limitations intrinsèques du code source fermé pourraient être trop contraignantes pour plusieurs systèmes du GdC dans le futur

Même si la stratégie du code fermé semble appropriée pour un marché de masse (p. ex. utilisation domestique ou personnelle avec peu ou pas de compétences en programmation), pour les systèmes militaires et pour l'informatique du gouvernement en général, l'accès au code source et l'adoption de normes ouvertes sont des avantages évidents. Le besoin d'une plus grande fiabilité et sécurité, de plus de flexibilité et d'extensibilité, de plus de compétition dans l'approvisionnement logiciel et finalement les économies de coûts directs vont toujours avoir tendance à justifier l'utilisation des logiciels libres durant la prochaine décennie.

Les communautés de R&D devraient faire preuve de leadership dans l'adoption des logiciels libres

Il semble que la communauté de R&D ait une responsabilité importante dans l'activation de projets pouvant démontrer la valeur stratégique des logiciels libres qui permettraient de disposer d'analyses de rentabilisation plus claires pour le GdC. Une de ses principales responsabilités est d'accomplir des activités exploratoires qui pourraient mener à une réduction des risques dans l'évolution technologique de nos ministères respectifs.

Partie II

Principes directeurs proposés pour le Gouvernement du Canada (GdC)

13 Principes directeurs de la stratégie

Les logiciels libres offrent une possibilité concrète et crédible notamment pour la R&D

La rentabilité des logiciels libres dans les projets de R&D à été analysée par quelques laboratoires de recherche, incluant le centre de recherche AMES de la National Aeronautics and Space Administration (NASA)) [76, 77], la National Nuclear Security Administration [78] et par un certain nombre d'universités [79]. Ces études ont conclu que les logiciels libres offrent une troisième option attrayante au dilemme "développer ou acheter", avec de nets avantages en matière d'expertise en développement, de créativité et de productivité. Dans les projets de R&D, les désavantages traditionnels des logiciels libres (tels que la complexité technique du développement logiciel et la maintenance à long terme) sont moins importants, puisque l'expertise est habituellement disponible dans les laboratoires et plusieurs projets visent à construire des prototypes de démonstration.

La diversité dans les approvisionnements est préférable
(Logiciels sur mesure vs logiciels libres vs logiciels commerciaux)

Les logiciels libres permet de construire un système spécialisé selon un processus de développement accéléré qui est au moins aussi efficace que l'achat d'éléments commerciaux. Pour les projets de R&D, l'utilisation de logiciels libres peut assurer le développement rapide (c.-à-d. réutilisation du code et modification) d'applications de haute qualité (c.-à-d. bien déboguées), ce qui serait très difficile à réaliser avec du code sur mesure développé à partir de zéro. Dans certaines circonstances, le développement basé sur les logiciels libres est la seule option raisonnable lorsque les produits commerciaux ne sont pas disponibles (p. ex. calcul de haute performance [78]) et lorsque le développement sur mesure excède les budgets disponibles [61]. Les logiciels libres aident à éviter d'être enfermés dans les produits et services propriétaires et à réduire la dépendance aux technologies monopolistiques.

Les normes et spécifications ouvertes contribuent directement à l'interopérabilité des systèmes

Les logiciels libres implantent les normes et spécifications ouvertes qui sont partagées par les développeurs durant la conception, le codage et les tests. Cela est généralement considéré comme un avantage stratégique dans l'application d'une politique d'interopérabilité entre des systèmes développés indépendamment [80].

L'évaluation des logiciels libres doit être faite sur une base de cas par cas

Bien que très attrayants en général, les logiciels libres doivent être évalués dans le cadre de chaque projet sur une base de "cas par cas" afin de déterminer si les avantages surpassent les désavantages dans la pratique. Dans le cas du GdC, une attention spéciale doit être accordée à la protection des technologies classifiées, à la protection de la propriété intellectuelle et à la sélection d'une licence convenable pour l'activité donnée. Quelques lignes directrices préliminaires sont disponibles dans ce document (parties III et IV).

---

14 Stratégie proposée pour le GdC

Adopter les logiciels libres progressivement

L'adoption des méthodes de développement des logiciels libres peut avoir des conséquences fondamentales profondes sur les pratiques en ingénierie, particulièrement si l'objectif est de contribuer activement à un projet de code source ouvert. On recommande d'acquérir de l'expérience avec les logiciels libres à titre d'utilisateur passif d'abord, pour ensuite prendre une part de plus en plus active, en signalant les bogues, en suggérant de nouvelles fonctionnalités et en modifiant le code existant avant de s'engager activement dans le développement dans le cadre d'un projet collaboratif. La figure 2 illustre le schéma d'évolution d'un utilisateur/développeur de logiciels libres.

Envisager les solutions basées sur les logiciels libres pour les travaux contractuels lorsqu'elles sont techniquement concurentielles par rapport à d'autres stratégies de développement

Le GdC devrait envisager les solutions basées sur les logiciels libres parallèlement aux solutions propriétaires lors des acquisitions en TI, particulièrement dans les contrats de développement importants tels que le Programme de démonstration de technologies (PDT). Selon Industrie Canada [81], les contrats sont accordés sur la base de l'optimisation des ressources et Travaux publics et Services gouvernementaux Canada (TPSGC) n'a pas de règle qui restreigne l'usage de logiciels libres dans les contrats du gouvernement fédéral. Le Conseil du Trésor ne dispose pas non plus de règle qui restreigne l'utilisation des logiciels libres dans nos programmes internes. La position du Canada concernant les logiciels libres, endossée en juin 2004, confirme qu'aucune barrière à l'approvisionnement ne devrait être maintenue.

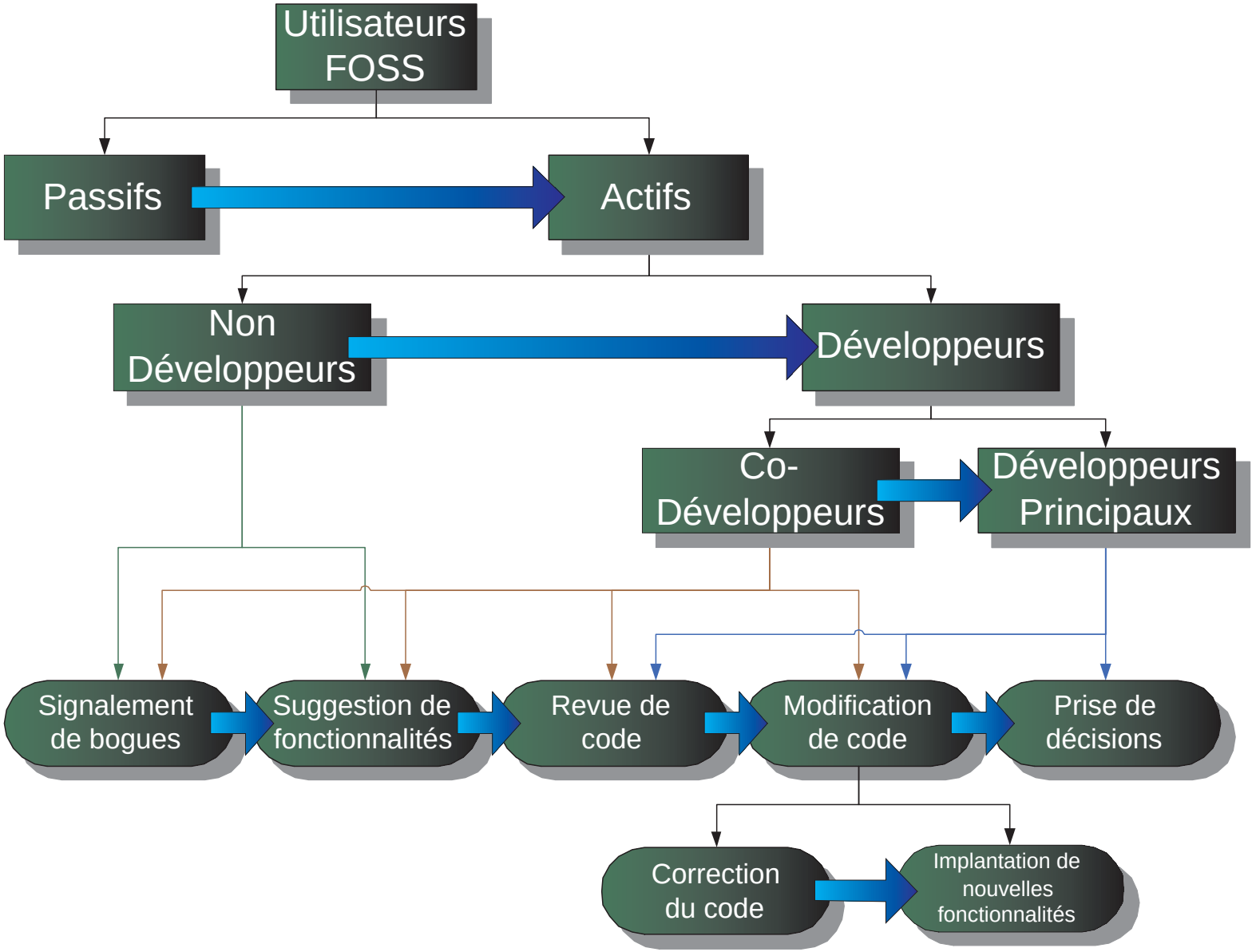

Figure 2: *Le schéma d'évolution des usagers/développeurs dans un projet de logiciel libre*

Partie III

Catalogue de logiciels libres sélectionnés pour utilisation dans les projets du GdC

15 Survol des logiciels libres disponibles

Sur la dizaine de milliers de projets de logiciels libres, seuls quelques centaines ont atteint assez de maturité pour être retenus aux fins d'inclusion dans les systèmes du GdC

Selon Spinellis et Szyperski [11], plus de 115 000 projets au code source ouvert étaient enregistrés à l'un des quatre principaux forums de code source ouvert (30 000 à `http://www.freshmeat.net`, 70 000 à `http://www.sourceforge.net`, 5 400 à `http://www.cpan.org` et 10 000 adaptations distribuées avec FreeBSD). Notez que quelques projets sont enregistrés deux fois et que de nombreux projets sont inactifs/morts. Afin de choisir le meilleur code, on recommande de se référer à des listes crédibles de logiciels libres matures ou sécuritaires telles que celles apparaissant ci-dessous. Ces listes fournissent des aides à la navigation pour identifier les logiciels libres appropriés à une application donnée, mais ne devraient pas être prises comme une position, une politique ou une décision officielle du gouvernement, sur la valeur de chacun des éléments logiciels particuliers.

MITRE a compilé une liste de 115 applications libres qui offrent un excellent point de départ pour identifier les logiciels de haute qualité

MITRE s'est appuyée sur la liste de logiciels libres de "Generally Recognized As Safe" (GRAS). Les logiciels libres de GRAS doivent répondre à ces critères : (a) jouir d'un soutien commercial, (b) être largement utilisés et acceptés, (c) avoir des états de service reconnus comme sécuritaires et fiables. La liste complète est disponible dans [5] et incluse dans la liste proposée à l'annexe A.

La liste "Generally Recognized As Mature" (GRAM) est une autre liste intéressante comprenant 39 logiciels libres

La liste GRAM, accessible dans [82], est maintenue à jour par David Wheeler, un gourou en sécurité informatique.

Le guide de migration de "Interchange of Data between Administrations"(IDA) énumère de nombreux logiciels libres de haute qualité

Les directives de migration d'IDA offrent une comparaison de plusieurs logiciels libres qui constituent une option de rechange aux équivalents commerciaux incluant : des systèmes d'exploitation, des suites bureautique, des serveurs de courrier, des logiciels de productivité de groupe, des services Web, des logiciels de gestion de documents et de bases de données, etc. [26].

---

16 Les logiciels libres pertinents pour le GdC

Les contributions aux logiciels libres provenant des grandes compagnies en TI se classent parmi les meilleurs logiciels disponibles

Plusieurs grandes compagnies de TI contribuent massivement aux logiciels libres. Dans [83], quelques 137 logiciels libres de haute qualité ont été identifiés provenant des fabricants suivants : Sun Microsystems (28), Silicon Graphics (15), Hewlett-Packard (33), Red Hat (2), AT&T (24) et IBM (35).

RDDC a préparé une liste détaillée de logiciels libres qui inclut des logiciels scientifiques et d'usage général (environ 392 logiciels libres)

Basée sur les listes introduites plus tôt et sur des recherches indépendantes, une liste plus détaillée a été élaborée par les auteurs. Elle offre de l'encadrement dans la sélection de logiciels scientifiques en plus de ceux d'usage général. Notre objectif est d'offrir un tableau plus complet de la richesse et de la diversité des logiciels présentement disponibles.

L'annexe A fournit des aides à la navigation pour aider à identifier les logiciels libres appropriés

À l'annexe A, le lecteur trouvera une liste des logiciels libres et ouverts les plus représentatifs au moment où cette étude a été réalisée (hiver 2004). Cette liste pourrait faciliter l'identification des produits candidats pour l'évaluation au sein du GdC. Nombre d'entre eux ont été utilisés dans le passé dans certains projets de RDDC, mais aucun test systématique ni aucune évaluation rigoureuse n'ont été effectués sur les logiciels libres qui y figurent. Par conséquent, ils ne doivent pas être considérés comme étant "approuvés pour le gouvernement" ou officiellement recommandés par le GdC. Il est fortement recommandé que chaque élément logiciel soit évalué avant son intégration. On propose une méthodologie rigoureuse à la partie IV.

Partie IV

Lignes directrices dans l'évaluation des logiciels libres pour le GdC

17 Principes directeurs

Les logiciels libres et leurs équivalents commerciaux devraient être évalués côte à côte

Le processus d'évaluation des logiciels libres et commerciaux est essentiellement le même et la comparaison côte à côte demeure la meilleure approche pour identifier les pour et les contre de chaque option [84, 85, 38]. Le processus d'évaluation peut varier beaucoup en durée et en profondeur technique selon le cadre de l'application et les exigences du projet.

Les avantages/désavantages devraient être comparés dans le cadre précis de chaque projet

Il est important de noter que la plupart des logiciels commerciaux sont conçus pour une très vaste clientèle et incluent typiquement une énorme somme de fonctionnalités et de configurations potentielles. D'un autre côté, les logiciels libres ont tendance à être plus spécialisés, puisqu'ils sont souvent conçus pour satisfaire les exigences d'une communauté d'utilisateurs donnée. On recommande la comparaison directe entre les deux types de logiciels et un cadre d'application bien défini afin de déterminer la meilleure option. En résumé, les principales étapes d'une évaluation incluent :

1. Comprendre les besoins et le cadre de l'application
2. Établir les priorités dans les critères de sélection
3. Identifier les logiciels commerciaux et libres candidats
4. Comparer les meilleurs candidats
5. Analyser en profondeur les meilleurs produits (p. ex. performance, audit de sécurité, coût), si nécessaire
6. Rechercher l'approbation de la direction locale et du client pour le projet.
7. Documenter les leçons apprises

Une attention particulière doit être accordée au modèle d'octroi de licence

Pour l'instant, il ne semble pas approprié pour le GdC de choisir un modèle d'octroi de licence et de l'imposer à tous les projets. Il semble préférable d'identifier le modèle de licence le mieux adapté au cadre de chaque projet en tenant dûment compte de :

1. La protection de la propriété intellectuelle,
2. les contraintes relatives aux partenariats nationaux et internationaux, et
3. les préférences du client.

---

18 Étapes d'évaluation recommandées

Étape #1 – Définir le cadre de l'application

1.1. Préciser les objectifs et les attentes du client.

1.2. Documenter les contraintes du projet telles que le niveau de classification, les exigences des partenaires, la compatibilité avec les environnements de développement et d'exécution, la compatibilité avec les systèmes existants et avec les formats d'information existants, les normes obligatoires à satisfaire, etc.

1.3. Établir les priorités pour les critères d'évaluation servant à comparer les logiciels, incluant les fonctionnalités, le coût, le soutien et la maintenance requis, la fiabilité, la sécurité, la performance, la flexibilité, l'extensibilité, la convivialité, les questions d'ordre juridique et de licences ainsi que les autres questions relatives aux applications.

1.4. Évaluer les ressources internes (et externes) disponibles pour le projet (incluant le budget, le temps et l'expertise technique pour laquelle les besoins peuvent être plus grands dans le cas du développement de logiciels libres).

1.5. Rechercher l'appui d'un collègue expérimenté qui pourrait agir à titre de 'mentor' durant le processus d'évaluation et aider à éviter les embûches.

Étape #2 – Identifier les candidats

2.1. Voir l'annexe A de ce rapport pour votre domaine d'application.

2.2. Effectuer des recherches complémentaires sur Internet, incluant les sites spécialisés : `http://www.sourceforge.net`, `http://www.gnu.org/directory`, `http://www.freshmeat.net`, `http://www.debian.org`, `http://www.savannah.gnu.org`, `http://www.icewalkers.com`, `http://www.cpan.org`.

2.3. Recueillir les examens techniques et les comparaisons de produits.

Étape #3 – Comparer côte à côte les 3 ou 4 meilleures options

3.1. Consulter les listes connues de logiciels libres 'fiables' telles que Generally Recognized As Safe (GRAS), Generally Recognized As Mature (GRAM), et Interchange of Data between Administrations (IDA).

3.2. Lire/évaluer les examens techniques (logiciels libres et commerciaux). Demeurer vigilant concernant les évaluations excessivement biaisées (rencontrées à la fois pour les logiciels commerciaux et les logiciels libres !)

3.3. Tenir compte de la compatibilité du logiciel aux librairies existantes et les environnements de développement et d'exécution.

3.4. Déterminer la maturité et le risque technique par les "relevés de téléchargement" (et d'autres mesures de popularité), la longévité du produit (souvent révélatrice de la maturité) et la pénétration du marché.

3.5. Résumer la démarche en regroupant les résultats dans un chiffrier qui inclut les critères selon les priorités établies à l'étape 1.3. Chiffrier pour l'évaluation des logiciels libres.

Étape #4 – Si approprié, effectuer une analyse en profondeur du code

4.1. Si le temps le permet, télécharger les versions d'évaluation afin de confirmer la performance, la compatibilité, la convivialité, etc.

4.2. Clarifier les détails avec les fournisseurs/développeurs.

4.3. Faire l'examen des licences et au besoin recueillir des avis auprès du service juridique local au sujet de la protection de la propriété intellectuelle.

4.4. Si opportun, effectuer une analyse détaillée du code avec des outils d'analyse logicielle pour détecter les failles et autres types de défauts. Voir [86].

4.5. Si opportun, évaluer la possibilité d'ajouter de nouvelles fonctions.

Étape #5 – Rechercher l'approbation du client et de la direction locale

5.1. Même si les progiciels sont utilisés tels quels (pas de développement de code), on recommande d'informer la direction locale (et éventuellement le client) de l'utilisation de logiciels libres.

5.2. Si les logiciels libres sont utilisés pour bâtir un prototype de recherche comprenant du développement de code substantiel, obtenir l'approbation de la direction locale et du client (s'il y a lieu). Voir le conseil dans "Les licences dans le développement logiciel au GdC", un peu plus loin.

5.3. Si un projet de développement du GdC est envisagé pour distribution dans un des réseaux de logiciels libres, évaluer l'effort additionnel requis pour nettoyer le code, améliorer la documentation et soutenir la communauté de façon efficace lorsque le logiciel sera mis à la disposition d'un des réseaux de logiciels libres. Obtenir l'approbation de la direction locale et du client (s'il y a lieu). Voir le conseil dans "Les licences dans le développement logiciel au GdC", un peu plus loin.

5.4. Si un projet de développement du GdC doit être réalisé dans un paradigme de collaboration en source ouverte, il pourrait être nécessaire de procéder à une analyse détaillée pour justifier cette approche. Obtenir l'approbation de la direction locale et du client (s'il y a lieu). Voir le conseil dans "Les licences dans le développement logiciel au GdC", un peu plus loin. Des conseils se trouvent sur le site Web de la fondation FSF et dans ce livre de Jan Sandred [87].

Étape #6 – Documenter les leçons apprises

6.1. Résumer les leçons apprises durant l'évaluation dans une brève note technique pour partager votre expérience avec les communautés du GdC.

6.2. Garder la trace de toute utilisation des logiciels libres et de tous les changements apportés au logiciel original au moyen d'un logiciel de contrôle des révisions durant tout le processus de développement (Concurrent Versions System (CVS)). Les données de contrôle des révisions doivent demeurer disponibles pour la Couronne après la fin du développement. Sans l'historique complet du développement, le nouveau code pourrait se retrouver par défaut sous d'autres modèles de licences tels que la GPL.

---

19 Les licences dans le développement logiciel au GdC

Les licences peuvent être regroupées en deux principales catégories

Une analyse détaillée des licences des logiciels libres va au-delà de la portée du présent document. Il suffit de mentionner que les multiples licences actuellement enregistrées à l'OSI (environ 48) peuvent être regroupées en deux principales catégories : (1) celles inspirée par la GPL, qui exige que tout développement complémentaire soit intégré uniquement à du code source ouvert et publié sous une licence compatible avec la GPL (p. ex. LGPL, Zope, etc.) ; (2) celles qui permettent la combinaison de logiciels libres avec du code propriétaire (p. ex. Berkeley Software Distribution (BSD), Mozilla, etc.) [88].

L'utilisateur final préfère habituellement la stratégie GPL

MITRE a examiné les différentes licences selon la perspective d'un utilisateur final (dans le cas présent, le DoD aux États-Unis), et en a conclu que la licence GPL est la meilleure en vertu de la diversité de code source et de la visibilité 'totale' des implantations. Cela garantit une réponse plus rapide et autonome aux cybermenaces [5]. On peut s'attendre à ce que la plupart des utilisateurs finaux préfèrent le modèle GPL.

Les centres de recherche préfèrent généralement les modèles BSD ou Mozilla

Des options de rechange à "l'octroi de licence plutôt strict de GPL" ont été développées pour répondre aux besoins des organisations qui doivent intégrer des logiciels ouverts et quelques éléments propriétaires. C'est le cas de plusieurs centres de recherche qui veulent protéger leur propriété intellectuelle novatrice tout en démontrant l'efficacité grâce à un prototype basé sur les logiciels libres. La NASA a comparé différentes licences, puis choisi le modèle de Mozilla en 2002 [76] et s'oriente présentement vers le développement de sa propre licence [77] inspirée de ce modèle. L'analyse du Conseil national de recherches du Canada (CNRC) conclut que les licences de BSD et Academic Free License (AFL) sont souvent les modèles les plus appropriés pour eux [89]. Gartner croit que les licences inspirées par BSD et Mozilla vont avoir une popularité grandissante, puisqu'elles offrent plus de flexibilité [4]. Étant donné que la sélection d'une licence est complexe et que de multiples options sont disponibles, les chercheurs devraient demander l'assistance de leur service juridique local lorsqu'un doute subsiste concernant les implications juridiques.

Le cadre juridique est présentement contesté

On observe que les compagnies commerciales sont souvent sur une trajectoire de collision avec les logiciels libres qu'elles combattent sur l'aspect technique mais aussi par des moyens légaux. Ces compagnies misent également sur la peur, l'incertitude et le doute engendrés par une nouvelle technologie pour faire passer leur message. Le développeur de logiciels libres qui vise un déploiement important de son travail doit être bien conscient des pièges et des questions de droit qui peuvent se poser à la suite d'événements qui se produisent dans la communauté, dont certains sont perçus comme étant clairement abusifs par divers observateurs [90]. Les exemples incluent la bataille que se livrent présentement Santa Cruz Operation (SCO) et IBM. Pour être mieux informés de tels courants sous-jacents, voir par exemple `http://www.groklaw.net`, `http://www.osriskmanagement.com`, et `http://www.pubpat.org`. Le site Open Source Development Labs `http://www.osdl.org`, qui accueille Linus Torvalds, le créateur de Linux, est aussi une bonne source d'information sur les enjeux et les nouvelles relatifs à la gestion de la propriété intellectuelle par rapport aux logiciels libres.

Plusieurs bonnes références sur les licences des logiciels libres se trouvent sur Internet

La réalité des diverses licences est beaucoup plus complexe que le survol présenté ci-dessus. Une étude complémentaire sera requise pour offrir des opinions juridiques sur l'octroi de licence, les droits de propriété, les transferts de technologie et d'autres questions de droit. Pour les lecteurs intéressés à une comparaison plus détaillée des licences, nous recommandons les références suivantes :

- Free Software Foundation - "l'école de pensée" GPL (plutôt idéaliste) [91] .
- Open Source Initiative - multiples modèles de licence (certains sont plus pragmatiques) [1].
- Guide européen pour choisir les licences des logiciels libres [92].
- Analyse détaillée des licences des logiciels libres [93].
- Comparaison des licences des logiciels libres - perspective du DoD aux États-Unis [5].
- Comparaison des licences des logiciels libres - perspective d'un développeur de la NASA) [76].
- Un bon aperçu des licences des logiciels libres [94].
- Un coup d'oeil pratique sur l'intégration des logiciels libres dans les produits commerciaux [95].
- Un coup d'oeil plutôt théorique sur les licences, démontrant les tendances intéressantes [96].
- "GPL copylefting vs non-copylefting licenses" [97].
- Normes complémentaires et tactiques légales concernant l'octroi de licences pour les logiciels libres [6].
- Une revue des enjeux des brevets logiciels [98, 9].

---

20 La migration vers les logiciels libres

Dans certains projets, une migration complète aux logiciels libres peut être envisagée

Dans certains cas, il pourrait être approprié d'envisager la migration complète d'un projet du GdC vers les logiciels libres. Même si plusieurs expériences récentes ont été très réussies à la NASA [61], dans le milieu de la santé [99], dans l'industrie [95, 100] et dans quelques universités [101] , on doit comprendre que le processus collaboratif des logiciels libres est un nouveau paradigme de développement qui implique des changements culturels significatifs. Le gouvernement d'Afrique du Sud recommande dans [45] le modèle VCS (c.-à-d. demonstrating Value–building Capacity–mobilizing Support) qui semble être un excellent cadre de travail conceptuel pour les études de migration vers les logiciels libres. Hewlett-Packard a défini un paradigme de génie logiciel innovateur pour les grandes sociétés, appelé "Progressive Open Source", qui semble très bien fait [102]. On trouvera plus d'information à propos des changements culturels aux références suivantes :

- La performance des applications scientifiques dans une grappe de terminaux Linux (cluster) [103].
- La migration des services de soutien (back-office) appuie la lutte anti-terrorisme (dans OSO) [104].
- Comprendre les caractéristiques des logiciels libres [105].
- La gestion de la configuration des logiciels libres [106].
- Linux au gouvernement - Livre blanc [57].
- Bon article résumant la pénétration du marché par Linux [107].
- La technologie Linux sur les ordinateurs de bureau et un survol du marché [52].
- Livre blanc d'IDC sur le coût total de possession (CTP) (ou TCO, en anglais) de Linux (commandité par IBM) [108].
- Livre blanc d'IDC concernant l'expansion de Linux dans les entreprises (commandité par IBM) [109].
- Livre blanc d'IDC sur l'accélération de l'adoption de Linux (commandité par IBM) [110].
- Bloor – Recherche et évaluation de la maturité de Linux [111].
- L'impact de l'adoption des logiciels libres au gouvernement [112].
- Les bénéfices perçus pour les logiciels libres dans le secteur public [113].
- La popularité croissante de Linux à Wall Street [114].
- La recommandation de Gartner d'utiliser les logiciels libres au gouvernement [115].
- Un excellent guide de migration de QinetiQ (UK) [116].
- Directives de migration très détaillées par IDA [26].
- Un autre guide de migration très impressionnant du ministère de l'Intérieur allemand [27].

- Les logiciel libres : Enquête et étude [28].
- Les logiciels libres dans les firmes et institutions publiques européennes [29].
- Les motivations commerciales et les implications politiques [30].
- Politique à l'intérieur de l'Union européenne [31].
- Le marché des logiciels libres et les méthodes de gestion [32].
- Enquête sur les développeurs de logiciels libres [33].
- Enquête sur le code source des logiciels [35].

De multiples outils existent pour effectuer une analyse des coûts

Plusieurs modèles de coûts ont été développés pour les systèmes logiciels. Quelques-uns sont plutôt simples et faciles à utiliser et pourraient convenir aux petits projets de migration partielle vers les logiciels libres :

- Notes concernant un chiffrier de comparaison de coûts [117].
- Chiffrier comparant les coûts : Un modèle pour la comparaison des coûts.
- Une introduction de base aux logiciels libres avec une évaluation des coûts [118].
- Coût total de possession (CTP) de Linux en entreprise [119].
- Toute une collection de statistiques sur le CTP au chapitre 7 [8].

Les scientifiques qui dirigent un projet majeur ou encore qui envisagent une migration complète vers les logiciels libres, pourraient préférer des modèles de coûts plus complexes tels que :

- Un article de Gartner donnant une vue d'ensemble qui mène à des analyses de CTP [120].
- Une analyse de rentabilisation de MITRE sur les logiciels libres (voir chapitre 3) [121].
- Un rapport de Forrester sur l'impact économique total de Microsoft vs Linux/Java 2 Enterprise Edition (J2EE) [23].
- Comparaison du CTP de Linux vs Windows [122].
- Modèle danois d'évaluation des ordinateurs de bureau pour la migration vers les logiciels libres [123].

Références

1. Stallman, Richard (2003). Basic Idea Behind Open Source. Paper. Open Source Organization,. http://www.gnu.org.
2. OpenSource.org. Web Site. Open Source Organization,. http://www.opensource.org.
3. Wu, Ming-Wei and Lin, Ying-Dar (2001). Open Source Software Development : An Overview. (Technical Report IEEE 0018-9162/01). National Chiao Tung University, Taiwan.
4. Driver, M. (2001). The Future of Open-Source Software. (Technical Report SPA-13-7536). Gartner.
5. Bollinger, Terry (2003). Use of Free and Open-Source Software (FOSS) in the U.S. Department of Defense. (Technical Report MP 02W0000101 v1.2.04). MITRE.
6. O'Mahony, Siobham (2003). Guarding the Commons : How Community Managed Software Projects Project Their Works. Paper. Harvard University Graduate School of Business Administration.
7. Casamento, Gregory. Petition Against Software Patents. Web Page,. http://www.petitiononline.com/pasp01/petition.html.
8. Wheeler, David A. (2003). Why Open Source Software / Free Software (OSS/FS) ? – Look at the Numbers !. Paper. Personal Web Page.
9. Rosen, Lawrence (2004). Patents in an Open Source World. *newsforge.com*.
10. United Nation Conference on Trade and Development Secretariat (2003). Free open-source software : Implications for ICT policy and development, Ch. 4. United Nation.
11. Spinellis, Diomidis and Szyperski, Clemens (2004). How Is Open Source Affecting Software Development ?. Paper. IEEE SOFTWARE.
12. James W. Paulson, Giancarlo Succi and Eberlein, Armin (2004). An Empirical Study of Open-Source and Close-Source Software Products. (Technical Report IEEE 0096-5589/04). General Dynamics Canada, University of Bozen, University of Sharjah.
13. Dowling, Ted (2000). Software COTS Components – Problems, And Solutions ?. In *RTO SCI Symposium on "Strategies to Mitigate Obsolescence in Defense Systems Using Commercial Component"*, pp. 28–1—28–8.
14. Calvin, James B. and Rodgers, Steven L. (2003). The Case for Open Source Tools is Compelling. *COTS Journal*, pp. 25–29.
15. Evans, David S. and Reddy, Bernard (2003). Government Preferences for Promoting Open-Sources Software : A Solution in search for a problem. (Technical Report 9 Mich. Telecomm. Tech. L. Rev. 313 (2003)). National Economics Research Associates.
16. Bishop, Todd (2004). Studies on Linux help their patron : Microsoft. *Seattle Post-Intelligencer*.

17. VeriTest (2003). Microsoft Windows Small Business Server 2003 vs. Red Hat Enterprise Linux ES 2.1 Deployment. Test Report. VeriTest.

18. Rubin, Howard (2003). Mainframe Linux Benchmark Project Audit Report. Audit Report. META Group, Inc.

19. VeriTest (2003). Microsoft Windows Server 2003 vs. Linux Competitive File Server Performance Comparison. Test Report. VeriTest.

20. VeriTest (2003). Microsoft Windows Server 2003 with Internet Information Services (IIS) 6.0 vs. Linux Competitive Web Server Performance Comparison. Test Report. VeriTest.

21. Bozman, Jean, Gillen, Al, Kolodgyand, Charles, and al (2002). Windows 2000 Versus Linux in Enterprise Computing. Test Report. IDC.

22. Morrow, Keith (2004). 7-Eleven Upgrades Its Strategic Retail System, Lowers Costs, and Improves Customer Service. Case study. Microsoft Corporation.

23. John R. Rymer and Bob Cormier (2003). The Total Economic Impact of Developing and Deploying Applications on Microsoft and J2EE/Linux Platforms. Technical Report. Forrester Research.

24. Murphy, Paul (2004). Getting the Facts About Windows and Linux. *LinuxInsider*.

25. Office of the e-envoy. Leading the drive to get the UK online. UK online,. http://www.e-envoy.gov.uk/Home/Homepage/fs/en.

26. netproject (2003). The IDA Open Source Migration Guide. (Technical Report OSPL/EEC-01.10). Interchange of Data between Administrations.

27. KBSt (2003). Migration Guide – A guide to migrating the basic software components on server and workstation computers. (Technical Report 1.0). Bundesministerium des Innern, Germany.

28. Ghosh, Rishab Aiyer, Krieger, Bernard, Glott, Ruediger, and Robles, Gregorio (2002). Free/Libre and Open Source Software : Survey and Study – Deliverable D18 : FINAL REPORT. (Technical Report Part 0 : Table of Contents and Executive Summary). International Institute of Infonomics, University of Maastricht.

29. Wichmann, Thorsten (2002). Free/Libre and Open Source Software : Survey and Study – FINAL REPORT. (Technical Report Part 1 : Use of Open Source Software in Firms and Public Institutions – Evidence from Germany, Sweden and UK). Berlecon Reasearch.

30. Wichmann, Thorsten (2002). Free/Libre and Open Source Software : Survey and Study – FINAL REPORT. (Technical Report Part 2 : Firms' Open Source Activities : Motivation and Policy Implications). Berlecon Reasearch.

31. Ghosh, Rishab Aiyer, Krieger, Bernard, Glott, Ruediger, and Robles, Gregorio (2002). Free/Libre and Open Source Software : Survey and Study – Deliverable D18 : FINAL REPORT. (Technical Report Part 2B : Open Source Software in Public Sector : Policy within the European Union). International Institute of Infonomics, University of Maastricht.

32. Spiller, Dorit and Wichmann, Thorsten (2002). Free/Libre and Open Source Software : Survey and Study – FINAL REPORT. (Technical Report Part 3 : Basics of Open Source Software Markets and Business Models). Berlecon Reasearch.

33. Ghosh, Rishab Aiyer, Robles, Gregorio, Krieger, Bernard, and Glott, Ruediger (2002). Free/Libre and Open Source Software : Survey and Study – Deliverable D18 : FINAL REPORT. (Technical Report Part 4 : Survey of Developers). International Institute of Infonomics, University of Maastricht.

34. Ghosh, Rishab Aiyer, Robles, Gregorio, Krieger, Bernard, and Glott, Ruediger (2002). Free/Libre and Open Source Software : Survey and Study – Deliverable D18 : FINAL REPORT. (Technical Report Part 4A : Survey of Developers – Annexure on validation and methodology). International Institute of Infonomics, University of Maastricht.

35. Ghosh, Rishab Aiyer, Robles, Gregorio, and Glott, Ruediger (2002). Free/Libre and Open Source Software : Survey and Study – Deliverable D18 : FINAL REPORT. (Technical Report Part 5 : Software Source Code Survey). International Institute of Infonomics, University of Maastricht.

36. Ghosh, Rishab Aiyer (2002). Free/Libre and Open Source Software : Survey and Study – Workshop on Advancing the Research Agenda on Free / Open Source Software. (Technical Report Workshop Report). International Institute of Infonomics, University of Maastricht.

37. The Swedish Agency for Public Management (2003). Free and Open Source Software.

38. The Swedish Agency for Public Management (2003). Free and Open Source Software – a feasibility study.

39. Ministry of Science (2003). The Danish Software Strategy. Software Strategy. Danish Ministry of Science, Technology and Innovation.

40. e-cology Corporation (2003). Open Source Software in Canada – A Collaborative Fact Finding Study. Technical Report. e-cology Corporation.

41. e-cology Corporation (2003). Open Source Software in Canada – Appendices. Technical Report. e-cology Corporation.

42. Open Source Observatory (OSO). Web Site. Open Source Observatory,. http://europa.eu.int/ISPO/ida/jsps/index.jsp?fuseAction=showChapter&chapterID=452&preChapterID=0.

43. Weber, Stephen (2002). Open Source Software in Developing Economies. Technical Report. University California, Berkeley.

44. Enav, Peters (2003). Israel stops buying Microsoft software. *The Associate Press*.

45. South African Government (2003). Using Open Source Software in the South African Government – A Proposed strategy compiled by the Governmen Information Technology Officers Council. (Technical Report Version 3.3). Government Information Technology Officers Council.

46. Riley, James (2004). Tax to Open Up Software Policy. *Australian IT*.

47. Varghese, Sam (2003). ACT set to adopt open source bill. *IDG News Service, West Africa bureau*.

48. AFUL's Site. French speaking Linux and Libre Software Users' Association. AFUL,. http://www.aful.org/index.html.

49. AFUL's Site on Peru OSS. French speaking Linux and Libre Software Users' Association. AFUL Web Page on Peru,. http://www.aful.org/politique/perou/english/referencias.html.

50. proposicion.org.ar. Government's documents about Free Software. Open Source Organization Web Page,. http://proposicion.org.ar/doc/gob/.

51. Logan, D., Arevolo, W., and Bittinger, S. (2004). Open-Source Software Means Benefits for Emerging Nations. (Paper SPA-19-4429). Gartner.

52. Decrem, Bart (2003). Desktop Linux Technology & Market Overview. Technical Report. Open Source Application Foundation.

53. Horstmann, Jutta and Muehlig, Jan (2003). Linux Usability Study. (Paper Version 1.01). Relevantive.

54. Hecker, Frank (2000). Setting up Shop : The Business of Open-Source Software. Paper. Netscape.

55. Fricke, Pierre (2003). Linux Strategies and Solutions 2003 : Linux Server Suppliers Contend for Leadership. Technical Report. D.H. Brown Associates, Inc.

56. Jacob, Bart, Janson, David, Mark, Oliver, and Marras, Fabio L (2002). Linux and Branch Banking. (Technical Report SG24-6909-00). ibm.com/redbooks.

57. Fisher, Mary Ann (2002). Linux in Government White Paper. Paper. IBM.

58. Lowery, J. Craig (2003). Dell's Open Source Software Philosophy. In *Open Standards/Open Source for National and Local eGovernment Programs in the U.S. and EU : Conference*, eGovOS.

59. National Security Agency (NSA). Security-Enhanced Linux. NSA Web Page,. http://www.nsa.gov/selinux/index.html.

60. TYBRIN (2002). Open Source Prototype Research – Open Source Process. Technical Report. NTA – NCAT.

61. Norris, Jeffrey S. (2004). Mission-Critical Development with Open Source Software : Lessons Learned. (Technical Report vol 21 no1). IEEE Software.

62. Frye, Emily (2003). Open-Source Software, Proprietary Software : Implications for National and Economic Security. Technical Report. THE CIP REPORT.

63. Mundie, Craig (2002). Security : Source Access and the Software Ecosystem. Technical Report. Microsoft Corporation.

64. David Adams (2004). Free Can Mean Big Money : The Open Source Economy. Technical Report. OSNews.

65. Patrick, Ryan B. (2003). Linux deepening its Canadian foothold. *Computerworld*.

66. Dumais, Michel (2004). Le logiciel libre en éducation : le projet MILLE. *Accération du CRIM*, **4**, 26–27.

67. Hissam, S.A., Plakosh, D., and Weinstock, C. (2002). Trust and Vulnerability in Open Source Software. (Technical Report 20020208). IEE Proceeding online.

68. ITsecurity (2002). The Strengths and Weaknesses of Open Source Software – And Its Role in the Security Model. Paper. ITsecurity.com :.

69. Jiwnani, K. and Zelkowitz, M. (2002). Maintaining Software with a Security Perspective. (Technical Report IEEE). International Conference on Software Maintenance (ICSM'02).

70. Judge, Peter (2002). Diffie defends open-source security. *ZDNet, UK*.

71. Diffie, Whitfield (2003). Perspective : Decrypting the Secret to Strong Security. *news.com*.

72. Jones, A. Russell (2004). Open Source Is Fertile Ground for Foul Play. *DevX.com*.

73. Stone, Mark (2004). Is Open Source Secure ?. *www.devx.com*.

74. Anderson, Ross (2002). Security in Open versus Closed Systems – The Dance of Boltzmann, Coase and Moore. Paper. Cambridge University, England.

75. Angelius, Ladd (2004). Who's Guarding the Guards ? We Are.. Technical Report. DevX.com.

76. Moran, Patrick J. (2003). Developing An Open Source Option for NASA Software. (Technical Report NAS-03-009). NASA Ames Research Center.

77. Geurts, Bryan A. (2004). NASA's Open Source Licensing. *news.osdir.com*, Vol. 1.1.

78. Advanced Simulation and Computing (ASCI) (2003). Request for Information : Open Source Software Development Acceleration (OSSODA). (Technical Report UCRL-MI-153295). National Nuclear Security Administration (NNSA).

79. Dongarra, Jack, Malony, Allen, Hollingsworth, Jeffrey, and al. (2003). Response to the Request for Information : Open Source Software Development Acceleration. Technical Report. National Nuclear Security Administration (NNSA).

80. Office of Government Commerce (2002). Open Source Software – Guidance on Implementing UK Government Policy. Technical Report. Cabinet Office Minister of State.

81. Gagné, Claude (2003). Innovation in ICT Innovation in Canada. Presentation Power Point. Industry Canada.

82. Wheeler, David A. (2003). GRAM List. Paper. Personal Web Page. http://www.dwheeler.com/gram.html.

83. Carbone, Richard (2004). Contributions to FOSS from the Large IT Companies. Internal Document.

84. Wheeler, David A. (2003). How to Evaluate Open Source Software / Free Software (OSS/FS) Programs. Paper. Personal Web Page. http://www.dwheeler.com/oss_fs_eval.html.

85. Surman, Mark and Diceman, Jason (2003). Multimedia Training Kit – Choosing Open Source Software Handout. Paper. The Commons Group, for the Association for Progressive Communications (APC).

86. Crispin Cowan (2003). Software Security for Open-Source Systems. Technical Report. IEEE Computer Society.

87. Sandred, Jan (2001). Managing Open Source Projects, Robert Ipsen.

88. GNU Project. Various Licenses and Comments about Them. GNU Project web server,. http://www.fsf.org/licenses/license-list.html.

89. Désilets, Alain, Jenkins, Stephen, Kark, Anatol W., and al (2004). Open Source Software in NRC. Guidebook. National Research Council Canada – Information Management Services Branch.

90. Moglen, Eben (2004). Now They Own It, Now They Don't : SCO Sues Novell to Stay Afloat. *LinuxDevices.com*.

91. Ravella, John and Lum, Rosalyn (2004). Free as in Freedom. *Software Development Magazine*.

92. ATICA (2002). Guide to choosing and using free software licences for government and public sector entities. Technical Report. ATICA – Agency for Information and Communications Technologies in the Civil Service.

93. ATICA (2002). Guide to choosing and using free software licences for government and public sector entities, Appendix : Detailed analysis of licences. Technical Report. ATICA – Agency for Information and Communications Technologies in the Civil Service.

94. Webbink, Mark (2003). Understanding Open Source Software. *GROKLAW*.

95. Ruffin, Michel and Ebert, Christof (2004). Using Open Source Software in Product Development : A Primer. Study. IEEE Software.

96. Lerner, Josh and Tirole, Jean (2002). The Scope of Open Source Licensing. Paper. Harvard University and NBER ; University of Toulouse and MIT.

97. Aigrain, Philippe (2002). A Framework for Understanding the Impact of GPL Copylefting vs. non Copylefting Licenses. Paper. European Commission.

98. McOrmond, Russel (2003). A Review of Software Patent Issues. *Flora, Ca*.

99. Fitzgerald, Brian and Kenny, Tony (2004). Developing an Information Systems Infrastructure with Open Source Software. Study. IEEE Software.

100. Lussier, Stepane (2004). New Tricks : How Open Source Changed the Way My Team Works. Study. IEEE Software.

101. Serrano, Nicolas, Calzada, Sonia, Sarriegui, Jose Mari, and Ciordia, Ismael (2004). From Proprietary to Open Source Tools in Information Systems Development. Study. IEEE Software.

102. Dinlelacker, Jamie, Garg, Pankaj K., Miller, Rob, and Nelson, Dean (2001). Progressive Open Source. Study. Hewlett-Packard Compagny.

103. Dressler, Jean-Marc and Kandadai, Swamy N. (2001). Performance of Scientific Applications on Linux Clusters. Performance Technical Report. IBM.

104. Open Source Observatory (2003). Back-office migration supports anti-terrorism. Technical Report. Bundeskartellamt.

105. Scacchi, Walt (2001). Understanding the Requirements for Developing Open Source Software Systems. Paper. Institute for Software Research, University of California.

106. Asklund, Ulf and Bendix, Lars (2001). Configuration Management for Open Source Software. Paper. Lund Institute of Technology and Aalborg University.

107. Macvittie, Lori (2004). Inside Linux. Study. Network Computing.

108. Gillen, Al, Kusnetzky, Dan, McLarnon, Scott, and Perry, Randy (2003). Linux and Intel-Based Servers : A Powerful Combination to Reduce the Cost of Enterprise Computing. WhitePaper. IDC.

109. Gillen, Al, Kusnetzky, Dan, Melenovsky, Mark, and North, Bill (2003). Expanding Linux System Configurations for Enterprise Deployment. White Paper. IDC.

110. Gillen, Al, Kusnetzky, Dan, and Rosen, Michele (2003). Accelerating the Adoption of Enterprise Linux Through IBM Software Solutions. White Paper. IDC.

111. Claybrook, Bill (2002). Linux Is Ready Scalability, Reliability, Security, Flexibility, and Total Cost of Ownership Considerations. Technical Report. Bloor Research North America.

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

113.Taylor, Graham (2001). Open Source – Coming of Age. Paper. OpenForum Europe.

114.Shahrawat, Dushyant (2002). Wall Street Romances the Penguin : The Growing Popularity of Linux. Research Notes. TowerGroup.

115.Drakos, Nikos, Mai, Andrea Di, and Simpson, Robin (2003). Open-Source Software Running for Public Office. (Technical Report AV-19-5251). Gartner.

116.Briggs, Julie and Peck, Dr Matthew (2003). QinetiQ Analysis of Open Source Solution Implementation Methodologies – QOSSIModo. (Technical Report Version 1). QinetiQ.

117.netproject (2003). Notes on the use of the Cost Comparison Spreadsheet. (Technical Report version 1, see also Excell spreadsheet in References). Interchange of Data between Administrations.

118.Meng, Tan Tze (2003). The Case for Open Source : OSS vs Proprietary Software. (Technical Report Version : 1.2). MNCC OSSIG Awareness Sub-Group Paper.

119.Robert Frances Group (2002). Total Cost of Ownership for Linux in the Enterprise. Study. Robert Frances Group.

120.Smith, David Mitchell, Simpson, Robin, Silver, Michael A., and Fiering, Leslie (2003). Linux on the Desktop : The Whole Story. (Technical Report AV-20-6574). Gartner.

121.Kenwood, Carolyn A. (2001). A Business Case Study of Open Source Software. (Technical Report MP 01B0000048). MITRE.

122.Cybersource (2002). Linux vs. Windows – Total Cost of Ownership Comparison. (Technical Report Version 1.0.1). Cybersource.

123.Offentlig Information Online (OIO) (2003). Desktop Evaluation Model. (Technical Report 0.8). Danish Government.

Liste des acronymes et sigles

AFL	Academic Free License
AFPL	Aladdin Free Public License
AFUL	Association Francophone des Utilisateurs de Linux et des Logiciels Libres
AOL	America On Line
ANSI	American National Standards Institute
API	Application Programming Interface
ASCII	American Standard Code for Information Interchange
BBS	Bulletin Board System
BSD	Berkeley Software Distribution
CAD	Computer Aided Design
CGI	Common Gateway Interface
CORBA	Common Object Request Broker Architecture
COTS	Commercial Off-the-Shelf
CPAN	Comprehensive Perl Archive Network
CPU	Central Processing Unit
CTP	Coût total de possession
CVS	Concurrent Versions System
DBMS	Database management system
DLL	Dynamic Link Library
DND	Department of National Defence
DNS	Domain Name Server/Service
DoD	Département de la Défense
DRDC	Defence Research & Development Canada
EHR	Electronic Health Record
EJB	Enterprise JavaBeans
EPS	Encapsulated PostScript
ERP	Enterprise resource planning
FAQ	Frequently Asked Questions
FC	Forces Canadiennes
FLOSS	Free/Libre and Open Source Software
FOSS	Free and Open Source Software
FS	Free Software
FSF	Free Software Foundation
FTP	File Transfer Protocol
GIS	Geographic Information System
GNU	GNU’s Not Unix
GdC	Gouvernement du Canada
GPL	General Public License
GPS	Global Positioning System
GRAM	Generally Recognized As Mature

GRAS Generally Recognized As Safe

GRASS Geographic Resources Analysis Support System

GSF Generalized Satellite Format

GUI Graphical User Interface

HTML Hypertext Markup Language

HTTP Hypertext Transfer Protocol

HTTPS HTTP over SSL

IDA Interchange of Data between Administrations

IDE Integrated Development Environment

IDS Intrusion Detection System

IEEE Institute of Electrical and Electronics Engineers

IIS Internet Information Server

IT Information Technology

J2EE Java 2 Enterprise Edition

J2ME Java 2 Micro Edition

J2SE Java 2 Standard Edition

JDK Java Development Kit

JIT Just-in-time

JLS Java Language Specification

JPL Jet Propulsion Lab

JVM Java Virtual Machine

JVMS Java Virtual Machine Specification

KBSt Federal Government Co-ordination and Advisory Agency — Germany

KDE K Desktop Environment

LAN Local Area Network

LGPL Lesser General Public License

LPPL LaTeX Project Public License

MDN Ministère de la Défense nationale

MFC Microsoft Foundation Classes

MIT Massachusetts Institute of Technology

MPI Message Passing Interface

MPL Mozilla Public License

MS Microsoft

NASA National Aeronautics and Space Administration

NERA National Economic Research Associates

NIMA National Imagery and Mapping Agency

NIST National Institute of Standards and Technologies

CNRC Conseil national de recherches du Canada

NSA National Security Agency

NTA National Technology Alliance

ODBC Open Database Connectivity

OGSI Open Grid Services Infrastructure

ORB Object Request Broker

ORDBMS Object-Relational Database Management System

OS Operating System

OSI Open Source Initiative

OSO Open Source Observatory

OSPR Open Source Prototype Research

OSS Open Source Software

PDA Personal Digital Assistant

PDF Portable Document Format

PDT Programme de démonstration de technologies

PGP Pretty Good Privacy

PHP Hypertext Preprocessor

PIM Personal Information Manager

PKI Public Key Infrastructure

PS PostScript

PDF Portable Document Format

RBAC Role-Based Access Control

R&D Recherche et développement

RDDC Recherche & développement pour la défense Canada

RDBMS Relational Database Management System

RFC Request for Comments

RPC Remote Procedure Call

RTF Rich Text Format

SCO Santa Cruz Operation

SELinux Security Enhanced Linux

SNMP Simple Network Management Protocol

SSH Secure Shell

SSL Secure Socket Layer

SQL Structured Query Language

TCP Transmission Control Protocol

TCP/IP Transmission Control Protocol/Internet Protocol

TI Technologies de l'information

TPSGC Travaux publics et Services gouvernementaux Canada

UDP User Datagram Protocol

UK United Kingdom

UML Unified Modelling Language

URL Uniform Resource Locator

VCS demonstrating Value–building Capacity–mobilizing Support

VM Virtual Machine

VPN Virtual Private Network

VRML Virtual Reality Markup Language

WYSIWYG What-You-See-Is-What-You-Get

XML Extensible Markup Language

Glossaire des termes pertinents

(Extrait de [45])

Logiciel commercial Un logiciel est commercial si son développement fait partie d'une activité économique. Un logiciel commercial peut être gratuit ou non, selon sa licence. Dans la même veine, un programme développé par une école ou un individu peut être gratuit ou non, selon sa licence. Les deux questions, "quel type d'entité a développé le programme ?" et "quelle est la liberté dont ses usagers disposent ?", sont indépendantes. 'Commercial' et 'propriétaire' ne sont pas synonymes — la plupart des logiciels commerciaux sont propriétaires, mais il y a des logiciels commerciaux libres et il y a aussi des logiciels non commerciaux non libres.

Compatibilité Le terme compatibilité, dans un contexte logiciel, est lié de près à l'interopérabilité. Un produit est compatible avec une norme mais est interopérable avec les autres produits qui répondent à la même norme (ou atteint l'interopérabilité par l'entremise d'un courtier.)

Logiciel protégé par un 'copyleft' Le 'copyleft' (s'opposant au concept de protection du droit d'auteur, ou 'copyright' en anglais), c'est l'idée et la stipulation, lorsqu'un logiciel est distribué, que l'usager aura la possibilité de le copier librement, d'en examiner et modifier le code source et de redistribuer le logiciel à d'autres (gratuitement ou tarifé) aussi longtemps que le logiciel redistribué est passé au même titre avec la stipulation du copyleft. Le terme a été introduit par Richard Stallman et la Free Software Foundation (FSF). Un logiciel sous copyleft est un logiciel libre dont les modalités de la distribution ne permettent pas aux redistributeurs d'ajouter aucune autre restriction lorsqu'ils redistribuent ou modifient le logiciel. Cela signifie que chaque copie du logiciel, et ce même si elle a été modifiée, doit être un logiciel libre. Le copyleft est un concept général ; pour effectivement protéger un programme avec un copyleft, on doit utiliser un ensemble particulier de modalités relatives à la distribution (voir la référence plus bas). Référence : La définition de copyleft selon la FSF : http://www.gnu.org/copyleft/copyleft.html

Logiciel libre Un logiciel libre est un logiciel qui vient avec la permission pour n'importe qui de l'utiliser, le copier et le distribuer, soit mot à mot ou avec des modifications, soit gratuitement ou avec des frais. En particulier, cela signifie que le code source doit être disponible. En anglais, il faut se rappeler que le mot 'free' dans "Free software" réfère à la notion de liberté et non de gratuité. Dans ce contexte, 'free' doit être compris comme dans "free speech", et non comme dans "free beer". Les logiciels libres offrent aux usagers la liberté d'exécuter, copier, distribuer, étudier, changer et améliorer le logiciel. Plus précisément, il y a quatre types de liberté pour les utilisateurs de logiciels libres :

- Liberté 0 - La liberté d'exécuter le programme, dans n'importe quel but.
- Liberté 1 - La liberté d'étudier le fonctionnement du programme, et de l'adapter à ses besoins - l'accès au code source est un condition préalable.
- Liberté 2 - La liberté de redistribuer des copies afin d'aider son voisin.
- Liberté 3 - La liberté d'améliorer le programme et de rendre ces améliorations disponibles au public afin que toute la communauté en bénéficie - l'accès au code source est une condition préalable.

Référence :

La définition de logiciel libre selon la FSF : http://www.gnu.org/philosophy/free-sw.html

logiciel libre vs ouvert Il y a un important désaccord au sein de la communauté à propos de ces deux concepts (presque synonymes) - jusqu'à un certain point, le mouvement pour les logiciels libres et le mouvement pour les logiciels ouverts sont comme deux clans politiques à l'intérieur de la communauté des logiciels libres. La définition officielle de "logiciel ouvert", telle que publiée par la Open Source Initiative, est très proche de la définition de "logiciel libre" utilisée par la Free Software Foundation, quoiqu'elle soit un peu plus ''relâchée' à certains égards. Nous n'irons pas plus loin dans ce débat, sauf pour le reconnaître comme un sujet litigieux. Il y a plus d'information disponible à : http://www.gnu.org/philosophy/free-software-forfreedom.html.

Gratuitiel (Freeware) Il n'y a pas de consensus sur la définition du terme anglais 'freeware', mais il est couramment utilisé pour les progiciels qui permettent la redistribution sans modification (et leur code source n'est pas disponible). En français, il n'y a pas de confusion possible entre les termes "logiciel libre" et gratuitiel. Un gratuitiel est offert gratuitement, mais il est généralement protégé par un droit d'auteur. On ne peut donc pas l'incorporer en tout ou en partie avec ce que l'on développe.

Interopérabilité L'Institute of Electrical and Electronics Engineers (IEEE) définit l'interopérabilité comme l'habilité de deux ou plusieurs systèmes ou éléments à échanger de l'information et à utiliser l'information échangée. L'interopérabilité, c'est la capacité d'un système ou d'un produit de fonctionner avec d'autres systèmes ou produits sans effort particulier de la part du client. Le terme est largement utilisé dans les descriptions de mise en marché de produits. Les produits atteignent l'interopérabilité avec d'autres produits en utilisant l'une ou l'autre de ces deux approches, ou encore les deux :

- En adhérant aux normes d'interface publiées
- En faisant usage d'un 'courtier' de services qui peut convertir "à la volée" l'interface d'un produit vers l'interface de l'autre

Un bon exemple de la première approche est l'ensemble des normes qui ont été développées pour le World Wide Web. Ces normes incluent TCP/IP, HTTP, et HTML. Le second type d'approche pour atteindre l'interopérabilité est illustré par "Common Object Request Broker Architecture (CORBA)" et ses "Object Request Broker (ORB)".

Liteware Liteware est le terme utilisé pour un logiciel qui est distribué gratuitement dans une version ayant moins de fonctionnalités que la version complète sur le marché. Ces logiciels sont habituellement conçus dans le but d'offrir à un client potentiel un échantillon de l'aspect et de la convivialité d'un produit et un sous-ensemble de ses pleines capacités. Ils peuvent être considérés comme un type de logiciel contributif (shareware) (les logiciels contributifs incluent aussi des produits distribués gratuitement, habituellement dans le but d'être mis à l'essai et qui n'ont pas leur pleine capacité.)

Logiciel ouvert En général, le terme logiciel ouvert réfère à tout programme dont le code source est rendu disponible pour utilisation ou modification comme l'usager ou les autres développeurs le jugent approprié. Les logiciels ouverts sont habituellement développés à titre de collaboration publique et ensuite offerts gratuitement. Au sens le plus strict, le terme logiciel ouvert réfère au logiciel qui est conforme à la "définition de source ouverte selon l'OSI".

Normes ouvertes Les normes ouvertes sont caractérisées par le fait que les spécifications sur lesquelles elles sont basées sont la propriété d'une organisation non liée à un fournisseur plutôt que celle des développeurs initiaux. N'importe qui est libre de constituer un logiciel selon ces

spécifications sans violation des droits de propriété intellectuelle, bien qu'habituellement il y ait plusieurs implantations disponibles gratuitement (commerciales ou ouvertes). Leur véritable avantage, c'est qu'elles ont été adoptées par l'industrie et sont "à l'abri du vieillissement". Une norme ouverte est plus qu'une simple spécification. Les principes derrière la norme et la pratique d'offrir et d'utiliser la norme, sont ce qui rend la norme 'ouverte' :

Disponibilité Les normes ouvertes sont disponibles pour tous pour lecture et implantation.

Maximise le choix de l'utilisateur final Les normes ouvertes créent un marché juste et compétitif pour les implantations de la norme. Elles n'imposent pas au client un fournisseur ou un groupe particulier.

Pas de redevance Les normes ouvertes sont gratuites à implanter pour tous, sans redevance ni frais. La certification de conformité par les organisations responsables des normes peut occasionner des frais.

Pas de discrimination Les normes ouvertes et les organisations qui les administrent ne favorisent pas un produsteur plus qu'un autre pour quelque raison que ce soit, mis à part la conformité aux normes techniques de l'implantation d'un fournisseur. Les organisations responsables de la certification doivent fournir une piste pour la validation des implantations à coût faible ou nul, mais elles peuvent aussi fournir des services de certification plus complets.

Extension ou sous-ensemble L'implantation des normes ouvertes peut être étendue ou offerte sous forme de sous-ensemble. Cependant, les organisations responsables de la certification peuvent refuser de certifier les implantations de sous-ensembles et peuvent imposer des conditions sur les extensions (voir Pratiques prédatoires)

Pratiques prédatoires Les normes ouvertes peuvent employer des clauses de licence qui protègent contre la subversion de la norme par des tactiques du style "englobe et dépasse." Les licences rattachées à la norme peuvent exiger la publication d'information de référence pour les extensions et une licence pour tous les autres pour créer, distribuer et vendre des logiciels compatibles avec les extensions. Une norme ouverte ne peut pas interdire les extensions d'une autre façon.

Une motivation importante pour adhérer aux normes ouvertes est d'atteindre et de promouvoir l'interopérabilité.

Un second ensemble de normes ouvertes est généralement créé par un consortium de leaders de l'industrie (des institutions ou des individus) qui déterminent qu'il y a une spécification générale pour une norme précise. De plus, il est important de noter l'influence de l'acceptation générale des normes ouvertes. Si une norme n'est pas largement adoptée, son développement sera probablement interrompu et elle finira par ne supporter seulement qu'un nombre très limité d'interactions entre les produits propriétaires.

Référence : Principes et pratiques des normes ouvertes : http://perens.com/OpenStandards/Definition.html

La définition de source ouverte selon l'OSI (The Open Source Definition) "Open Source" est une marque de certification logicielle détenue par l'Open Source Initiative (OSI). Les développeurs de logiciels qui sont destinés à être partagés librement, possiblement améliorés, et redistribués par d'autres peuvent utiliser la marque de commerce "Open Source", pourvu que les modalités de leur distribution soient conformes à la définition de l'OSI. En résumé, le modèle de définition pour les modalités de distribution requiert que :

- Le logiciel distribué doit être redistribué à n'importe qui sans aucune restriction

- Le code source doit être disponible (de manière que le receveur puisse l'améliorer ou le modifier)
- La licence peut exiger que les versions améliorées du logiciel adoptent un nom ou une version différente du logiciel original

Référence : La définition de source ouverte selon l'OSI : http://www.opensource.org/docs/definition_plain.php

Logiciel utilisable avec retour de carte postale (Postcardware) Un logiciel utilisable avec retour de carte postale est un gratuitiel (logiciel sans frais qui est partagé gratuitement) qui requiert seulement, de la part de l'utilisateur, d'envoyer une carte postale au fournisseur du logiciel, en guise de paiement. L'idée est d'humaniser la transaction, de rappeler à l'utilisateur que quelqu'un d'autre a partagé quelque chose gratuitement, et d'informer le fournisseur que quelqu'un utilise effectivement sa création.

Logiciel propriétaire Le terme logiciel propriétaire décrit un logiciel dont les droits sont détenus exclusivement par une seule compagnie qui protège soigneusement les connaissances au sujet des technologies utilisées et le fonctionnement interne du logiciel. Quelques produits propriétaires ne peuvent fonctionner correctement que lorsqu'ils sont utilisés avec d'autres produits de la même compagnie. Les logiciels propriétaires ne sont ni libres, ni semi-libres. Leur utilisation, redistribution ou modification est interdite, ou tellement restreinte qu'on ne peut effectivement pas le faire librement.

Logiciel du domaine public Les programmes qui ne sont pas protégés par des droits d'auteur, parce que leurs auteurs prévoyaient les partager avec tout le monde, sont du domaine public. La communauté Unix a développé un bon nombre de programmes de ce genre au fil des ans. Les programmes du domaine public peuvent être utilisés sans restriction comme éléments d'autres programmes. La façon la plus simple de rendre un programme libre est de le déposer dans le domaine public, sans droit d'auteur. Ceci permet à des gens de partager le programme et leurs améliorations, s'ils en ont l'intention. Cependant, cela permet aussi à des gens de convertir le programme en un logiciel propriétaire. Ils peuvent apporter des changements, nombreux ou rares, et distribuer le résultat comme un produit propriétaire, retirant ainsi la liberté que l'auteur initial avait prévu. Les logiciels du domaine public sont des logiciels qui ne sont pas protégés pas des droits d'auteur. Si le code source est du domaine public, il s'agit d'un cas particulier de logiciel libre non protégé par un 'copyleft', ce qui signifie que certaines copies ou versions modifiées peuvent ne pas être libres du tout. Il est possible qu'un programme en version exécutable soit du domaine public, mais que son code source ne soit pas disponible. Ce n'est donc pas un logiciel libre, puisqu'un logiciel libre requiert l'accessibilité au code source.

Logiciel semi-libre Un logiciel semi-libre est un logiciel qui n'est pas libre, mais qui accorde aux individus les permissions d'utiliser, copier, modifier et distribuer (incluant la distribution de versions modifiées) à des fins non lucratives. Pretty Good Privacy (PGP) est un exemple de programme semi-libre.

Logiciel contributif (shareware) Un logiciel contributif est un logiciel qui est distribué gratuitement à des fins d'essai, en supposant que l'usager pourrait devoir ou vouloir le payer plus tard. Certains développeurs de logiciels offrent une version contributive de leur programme avec une date d'expiration intégrée (p. ex. après 30 jours, l'usager ne peut plus accéder au programme). D'autres logiciels contributifs (parfois appelés 'liteware') sont offerts avec certaines capacités désactivées comme une façon détournée de forcer l'achat de la version complète du programme. Un logiciel contributif accorde la permission aux utilisateurs d'en redistribuer des copies, mais quiconque qui continue d'utiliser une copie doit payer un droit de licence. Un logiciel contributif n'est pas libre, ni même semi-libre, pour deux raisons :

- Pour la plupart des logiciels contributifs, le code source n'est pas disponible ; on ne peut donc modifier le programme d'aucune façon.
- Un logiciel contributif n'offre pas la permission de faire une copie et de l'installer sans payer un droit de licence, pas même pour les individus l'utilisant à des fins non lucratives. (En pratique, les gens négligent souvent les modalités de distribution et l'utilisent quand même sans payer, mais les modalités ne le permettent pas.)